A CONTEXT-AWARE AUTHENTICATION METHOD USING BLOCKCHAIN FOR PERVASIVE COMPUTING

By

NG MIAO XUAN

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF INFORMATION TECHNOLOGY (HONOURS)
COMMUNICATIONS AND NETWORKING

Faculty of Information and Communication Technology
(Kampar Campus)

JANUARY 2021

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

Title: A CONTEXT-AWARE AUTHENTICATION METHOD USING BLOCKCHAIN FOR PERVASIVE COMPUTING

Academic Session: JAN 2021

I NG MIAO XUAN
(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

[illegible] [illegible]

(Author's signature) (Supervisor's signature)

Address:
Jalan Universiti,
Bandar Barat,
31900, Kampar, Perak.

Aun Yichiet
Supervisor's name

Date: 15 April 2021

Date: 15 April 2021

A CONTEXT-AWARE AUTHENTICATION METHOD USING BLOCKCHAIN FOR PERVASIVE COMPUTING

By

NG MIAO XUAN

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF INFORMATION TECHNOLOGY (HONOURS)
COMMUNICATIONS AND NETWORKING

Faculty of Information and Communication Technology
(Kampar Campus)

JANUARY 2021

DECLARATION OF ORIGINALITY

I declare that this report entitled "**A CONTEXT-AWARE AUTHENTICATION METHOD USING BLOCKCHAIN FOR PERVASIVE COMPUTING**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature	:	[illegible]
Name	:	Ng Miao Xuan
Date	:	15 April 2021

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisor, Dr. Aun Yichiet who has given me this bright opportunity to engage in a Raspberry Pi blockchain project. It is my first step to take part in the development of blockchain project. A million thanks to you.

I would also like to extend my thanks to the laboratory assistants of the FICT department for their help in offering me the resources in this project. Finally, I must say thanks to my friends and family for their love, support and continuous encouragement throughout the course.

ABSTRACT

802.11 networks are secured with authentication protocols like WEP or some variants of WPA. Currently, legit users must enter the correct Wi-Fi password for authentication in joining the Wi-Fi network. Although most Wi-Fi infrastructure these days can remember previously connected clients for faster re-authentication, it is time-consuming for users to manually key in long, complex passwords in joining any new Wi-Fi networks. Some existing Wi-Fi authentication is simplified by scanning QR codes or using password sharing for trusted devices. Replacing WPA keys with QR codes introduces additional security loopholes like QR code sharing to unauthorized users. Meanwhile, password sharing with trusted devices currently only works on Apple iOS devices and selected Android devices; while still requiring user interventions. In this project, a 'pervasive Wi-Fi authentication method' is proposed for automatic device authentication using Ethereum blockchain for faster, secure, and un-intrusive authentication. In this scheme, wireless access points (AP) are chained together using a shared ledger that contains a set of trusted devices Ethereum node ID(s). These trusted devices can easily hop to any previously unseen AP from a previous AP that shares the same ledger through blockchain verification. The proposed Ethereum-based authentication requires an average of 8.92ms compared to the standard WPA2 that requires 10.629ms for full Internet connectivity.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

IoT	Internet of Things
Wi-Fi	Wireless Fidelity
PoW	Proof-of-Work
PoS	Proof-of-Stake
SSI	Self-sovereign Identity
VNC	Virtual Network Computing
AP	Access Point

CHAPTER 1: INTRODUCTION

1.1 Problem Statement and Motivation

Authentication is the process to identify a person. However, due to the increase of security issues, authentication method is necessary to be strengthen to its best to avoid being easily exploited. It is not difficult for a hacker to authorise himself as the person he wants to be, if he takes hold of all the information of the intended victim. He can process the victim's fingerprint in biometric terms, while brute forcing his text-based password, or just easily phish his password using social engineering attack. Authentication methods all seemed so vulnerable when it comes to the victim being the chosen one. Cases related to identity theft is increasing day by day, mostly related to money such as credit card fraud. Without an authentication system which can only be uniquely verified by the owner itself, it is difficult to protect the credentials from getting hold by attackers. Another reason that facilitates the attack to happen is the fragility of the authentication method proposed on the nodes in the network. It creates a loophole for the attackers to find a chance to enter the network, and take gain of the important and sensitive data such as banking information. If the authentication technology of the network is strong enough, it might lower the possibility for the attacker to be able to intrude into the network.

Memory is also an issue in authentication. Not everyone can remember all their passwords ideally in their mind. There are too many areas that require passwords for authorization. To describe this situation with a scenario, a person goes to his friend's house and wants to connect to the Wi-Fi. However, his friend forgot the password to authenticate him. To ease authentication, blockchain technology can be used to enable the person to be added to the blockchain network by his friend to gain access without troublesome authentication methods.

This project proposes a Wi-Fi authentication system using blockchain technology, an emerging technology used for security in important transactions. This can improve the existing authentication methods for users to verify their identity before entering the network. Blockchain increases the security for authentication and increases the difficulty for a data breach to happen. It is known whereby the more users who join the chain, the more difficult for the block in the chain to be exploited.

1.2 Background Information

Authentication is the process of proving a claim, such as the identity of a user. Some users who do not belong to the identity tend to camouflage themselves to become the identity that they desire, and this is where authentication methods come in handy. Different authentication methods extract different factors or characteristics of a user to prove the uniqueness of the user. The most useful authentication methods are the ones that other users are unable to duplicate the patterns or the characteristics tested to verify the user, such as the biometric authentication method. There exists two-factor authentication, whereby user is granted access only after showing two factors that can identify themselves. It provides a layer of complexity to the most basic method used by basic users, which is the single factor authentication method. Multi-factor authentication is for two or more factors, which is obviously enhancing authentication security for users. There are three types of factors used to prove a user, which are knowledge, possession and inherence.

Knowledge-based authentication is something that a user knows. For instance, the most common username and password which are used widely in the current technology. Possession-based authentication is something that a user have. Connected tokens owned by the user are used to authenticate them by connecting them to the computer and automatically transmit authentication information to the user's computer. Inherent-based authentication is something that the user are. It is almost unique enough to represent a user, they are normally biometric characteristics of a user such as fingerprint. It is widely used in mobile phones for user to unlock their phone.

Blockchain, which is famous on the implementation of Bitcoin by Satoshi Nakomoto, is a chain of blocks storing information of transactions, the user performing it, and a hash that distinguishes the block from other blocks, along with the hash of the block before it. If a new block wants to enter the blockchain, it must be verified first by a process called mining, then the transactions will be bundled in a block. The header of the most recent block will be selected to insert it into the new block as a hash, then a devoted user in the chain uses enough computational power to solve the PoW problem. When the solution for it is found, then the block is lastly added to the end of the blockchain.

Pervasive computing, which is also known as ubiquitous computing, is a term that means computing appears anytime and anywhere, in any objects that a user can think of. The world is advancing whereby people are embedding computational ability into everyday objects to ease a person's life anytime and anywhere without needing them to turn on their laptops to perform the tasks (Rouse 2019). Leveraging on persasive computing, the proposed blockchain authentication enables seamless experience in connecting to multiple trusted networks.

1.3 Project Objectives

In the first stage, the Raspberry Pi will be configured using hostapd to allow it to act as an Access Point (AP) and authentication server. In the second stage, we will be learning the structure of Ethereum blockchain and add both the AP nodes and the end users into the same blockchain. They are able to discover each other and perform transactions to exchange data. Then, before end users connect to the AP nodes to get Internet access, an authentication process will take place to check whether they are in the same blockchain. A few objectives has been set and defined as following:

i. To develop a programmable wireless AP on Raspberry Pi for simplified dev/ops.

ii. To develop a method to chain a set of trusted networks on multiple AP nodes and end devices using Ethereum blockchain.

iii. To develop blockchain authentication function on wireless AP nodes to perform authentication on end users prior allowing them to get Internet connection.

1.4 Project Scope and Use Case

At the end of this project, an improvement plan on authentication methods in a network will be proposed. This project will be the development of Wi-Fi authentication system by connecting the wireless AP and end devices together using blockchain technology. If the users are inside the blockchain, then they are able to access the wireless AP and obtain Internet access. The user can be added into the blockchain by the wireless AP, whereby the wireless AP will need to know the IP address of the connected device and the blockchain node info.

In this project, it is scoped down to only using a few wireless APs to show that the blockchain technology can be implemented. To describe on how the Wi-Fi authentication system works, for instance, there are multiple wireless APs for multiple halls in a hotel. Once the user is connected into any of the wireless APs using first-time logon, he can communicate to the other APs in the same blockchain which is located at the other halls. Similar to mesh Wi-Fi, this type of connection management can also improve the Wi-Fi range for the user as movement will not be a restriction for the wireless connection as long as he is inside the range for the multiple wireless APs. If user is not inside the blockchain, then he will not be able to establish a connection to the wireless AP for security reasons.

The owner can add a new peer to the blockchain to add a new trusted device. In this project, there is no need for authentication servers such as Kerberos, where third party is completely eliminated. Trust will be distributed among blockchain members when verified by the blockchain database itself in the distributed ledger.

1.5 Impact, Significance and Contribution

For this proposed project, the user wanting to enter the private blockchain network will not need to undergo complicated authentication methods such as two-factor authentication. This project's contribution is to ease users' authenticating into a network efficiently, and the user's identity will be authentic. Only the user having the blockchain ID will be able to authenticate into the network. This will reduce users' risk of forgetting their authentication credentials as the network will be automatically connected once the user is added to the blockchain. The owner will add the user into the blockchain manually as it is a private blockchain.

Blockchain is capable of verifying data integrity in its own network chain, leading to the reason why blockchain can be a decentralised technology. By using blockchain, it allows processes to be more independent, efficient, secure and transparent. This project contributes to the networking architecture whereby it can improve the networking technology by implementing stronger method to which improves security. Blackhats will not be able to tamper data communication happening in the blockchain easily.

Using a decentralized authentication architecture, the proposed work eliminates a single point of failure. There is also no need for third party to maintain the network, instead, the blocks in the blockchain network is able to maintain and gain trust from each other. A set of trusted networks on AP nodes and end devices will be chained using blockchain.

1.6 Report Organization

In this report, there are 7 chapters. In chapter 1 which is introduction, we will study the background of the project, determine the scope and objectives and the contribution after completing the project. In chapter 2 which is literature review, we will discuss on the existing work that has been carried out by other researchers that are related to our current project. We try to solve the problem that we found out through their research and improve their system so that it is beneficial by developing this project. In chapter 3 which is system methodology, we will discuss the system development life cycle, system requirement and functional requirements. In chapter 4 which is system design, we will discuss on the network architecture, system flowchart and system module. In chapter 5 which is implementation, we will be discussing on the implementation of the system, on the steps used to compile the system. In chapter 6 which is results and discussion, we will be analysing the performance of our system, and perform the testing process to find out the bugs of our system. In chapter 7, we will summarise the project and highlight any future directions that could be achieved.

CHAPTER 2: LITERATURE REVIEW

2.1 Data Breaches Trends and Statistics

Data breaches happen in organisations, whereby authentication credentials are exposed to attackers that took control of the whole database. Normally the credentials exposed are email, username and password. Attackers mainly aim financial and medical sectors as their database information is more beneficial and valuable. There are organisations that store user authentication credentials poorly such as only using simple encryption methods to encrypt the password. Based on the 2019 Data Breach Investigations Report by Verizon, 22% of data breaches in 2017 involved the use of stolen credentials (Sobers 2020). Attacker gains the stolen credentials and further exploits until he get the desired information that he wanted.

According to HIBP by Troy Hunt, it is known that data breaches has been happening since long ago till now. The most recent huge data breach is known as the “Collection #1” breach which consists of 773 million records of user credentials. When attacker gains these information, they can perform credential stuffing which is the automated injection of breached username and password pairs to gain access to the accounts. The biggest data breach in history has impacted 3 billion user accounts which happens in the company Yahoo. If the current online security posture still maintains the same as it is now, user credentials will be vulnerable and exposed easily.

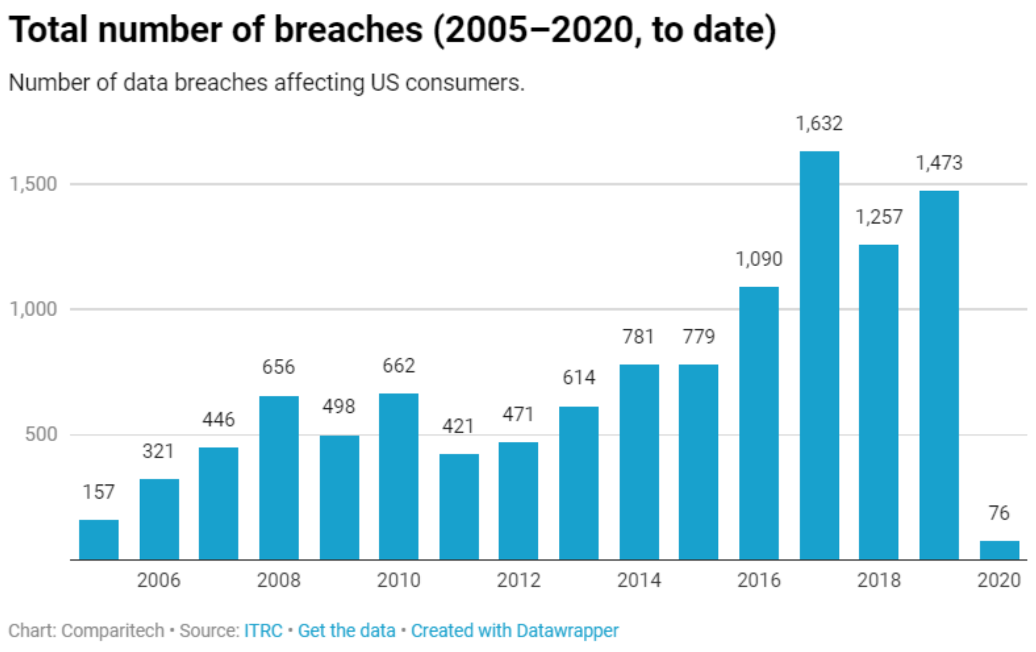


Figure 2.1 Total number of breaches from January 2005 to March 2020 (comparitech)

It is clearly seen that the trend for the data breaches has been increasing dramatically since 2005. Over the last 15 years, there are approximately more than 11,000 data breaches that are affecting US consumers. User credential is at risk for each data breach, some are exposed to the public and more people get hold of the credentials. This issue should be resolved by using a different authentication method, a simple username and password is tough to proof that the user is the exact user using the account to authenticate.

Similar to Wi-Fi authentication, the password can also be easily exploited since it is using plaintext and encryption methods to process it. Therefore, in this project, we propose a password-less authentication system, which is using blockchain as the authentication system. Once the user is added as a trusted peer in the blockchain with the AP, then the user will be able to connect to it to obtain Internet access.

2.2 Authentication Mechanisms

Knowledge-based authentication is something that a user knows. For example, the most basic yet vulnerable authentication method is username and password, to access the user’s account. User can choose to store their information in cognitive memory, which is the safest way, but a user does not only have an account, he might tend to forget his username or password. Studies have shown that user’s memory has been reduced due to large number of password used, and it leads to insecure work-practices such as writing their passwords down (Adams & Sasse 1999). On the other hand, users are also using the same password for almost every platform that needs an account for authentication for easy memorisation. However, there are risks for using only one same password for every platform. For example, if an attacker compromised one of the user’s account, he can try to use the same credentials to log in to other platforms such as online banking. Besides, the chance of the password being discovered will be higher as data breaches might happen in any platforms that the user registered his account on. Based on Troy Hunt, he created a free resource website for users to find out whether their online account are being compromised in a data breach. He wanted to highlight the seriousness of the risks for online attacks on the current internet (Hunt n.d.).

Possession-based authentication is something that a user have. Possession-based authentication significantly increases the communication security because users must have in possession the devices that their accounts are registered with (Rouse 2014). To visualise the usage, it is like owning a key to the security lock. Token-based authentication falls under the possession-based authentication, and it includes two forms, which are connected tokens and disconnected tokens. Connected tokens are physical devices such as card readers and USB tokens, which are used to authenticate the user by connecting them to the computer and automatically transmit authentication information to user's computer. Disconnected tokens does not require special input devices, instead they display the generated authentication data using the device's built-in screen and the user has to manually enter them in the user's computer to authenticate.

Inherent-based authentication is something that the user are. It is almost unique enough to represent a user, as they are normally biometric characteristics of a user such as fingerprint. Biometric authentication is widely used in mobile phones as a tool for users to unlock their phone. Based on studies, it also decreases the rate of identity theft and account takeover (Nicolls 2019). However, biometric authentication is not always safe; there is a report where a Chinese cybersecurity expert told people that making a V-sign in photos could expose a person's fingerprint data (Koetse 2019). There is a key disadvantage whereby biometric information cannot be easily changed. Once it is compromised, it cannot uniquely belong to the user anymore. Unlike passwords, user can easily change them if forgotten.

In this proposed project, we will be more focusing towards inherent-based authentication. If the user is inside the blockchain, then he will be connected to the blockchain. We will need to know the IP of the user which is unique to the user when connecting to the wireless AP. We will also need to know the blockchain node info of the user. By obtaining this two information, we can add user as a peer of the blockchain, and authenticate the user easily.

2.3 Blockchain Technology

Blockchain is an idea different from our current society organisation; it is decentralised as there is no central location to store the information. Instead, the information are spread across every block. In our current world, it is centralised as there are governments for every country, and organisations for every service provider such as Google. The core of blockchain transparency is that the transaction records are recorded in all of the blocks in the blockchain identically; one may not easily amend the records personally. According to (Reiff 2020), the ultimate goal of blockchain is to allow digital information to be distributed, recorded and not changed.

Blockchain does not only imply with digital currency, it is also used in different areas such as digital voting, IoT networks, and even online games such as Minecraft. By using blockchain with authentication, it can provide higher levels of security as blockchain is considered hacking-resistant. Hackers will no longer have a central location to perform an attack. Blockchain is a distributed ledger, it is secure whereby the information is spread across every block in the blockchain, allowing the information to be more difficult to manipulate (Reiff 2020). If one wants to do so, he needs to manipulate every copy of the block on the blockchain, which is time consuming and troublesome. However, there is an attack known as 51 percent attack. This attack is more susceptible for smaller blockchain because the attacker does not need to consume as much power as for attacking larger blockchains. 51 percent means that once the attacker succeed in manipulating over 50 percent of data in the network, and then the network will trust the manipulated data instead of the original data.

It is known that blockchain can be both private and also open for everyone (HBUS 2018). A user is unknown in public blockchain and free for anyone to join; but in private blockchain, it is a permissioned blockchain whereby it restricts on which node is allowed to communicate in the network. Blockchain uses public key and private key to process each transaction in the blocks. Public key is used to associate with the transaction and it remains open to everyone, while the encrypted private key is used to authorise the transaction and is only known by the owner of the block. By implementing blockchain authentication, user can register their identity on the blockchain using their public and private key pair. However, there is a disadvantage whereby once the private key is lost, the user can hardly retrieve it back as no one else is in charge of the whole blockchain. In networking aspects, blockchain is also similar to peer-to-peer network architecture as every node in the network has the same position and they do not have a centralised server to take care of them. Authentication with blockchain in networking focuses on how the new node joins the network using the blockchain idea and gain necessary trust from the nodes currently in the network.

2.4 Blockchain Two-factor Authentication using BitID

Based on the official Bitcoin website, in their news section, they stated that they are starting to implement BitID. Blockchain technology might be the future of two-factor authentication, as stated by the author (Buntinx 2015). BitID works by gathering already-verified identity that online services has created, such as bank or bitcoin exchange. Then, BitID creates an identity token for them, which consists of the actual Bitcoin address derived from the identity creation domain, and eight bytes of code to signify the ID documents that were used to verify it. After that, the token will be recorded on the blockchain.

BitID is working on a future project whereby users can use their unique Bitcoin wallet addresses and private keys to sign up to any service. This can be adopted to act as a two-factor authentication protocol on top of the Bitcoin blockchain. Any service or individual who ought to verify the identity can simply scan a QR code to check whether the identity token exists. When identity is verified, the user will also receive a link to the authority that created the identity token, then they can look up the initial vendor’s verification documents. BitID targets to make life easier, where it will be easy for user to authenticate. Besides user-friendly, BitID will also offer more security and privacy as it is like using blockchain on a blockchain, which is double the security of a sole blockchain. The verifier of the identity token does not need to see or save the data of the initial identity documents, which adds a layer of protection against hacking attacks and data theft (Southurst 2016).

Williams (2016) stated that things will become easier when users bring around a cryptographically provable identity with them, users can prove who they are easily. The strength for this solution is that collusion will not happen and the identity record is verified by consensus in the blockchain. It also does not tackle the issue of identity fraud. BitID establishes a profile for the users, while they provide the identity token available for users to use. The token will be rendered invalid if the blockchain recorded identity is found to be fraudulent (Southurst 2016). The weakness of the solution is that once the identity is modified by attacks such as the 51% attack, the identity will not be recovered back anymore. If the network is too small, the chance for 51% attack to happen will increase. Such an attack happens when one node by attacker manages to control more than 50% of the hashing power of network, which leads to the entity to disrupt the network.

To solve the weakness mentioned above, public blockchain is used and this technology can be broadcasted widely so that many users will be able to find out about this technology. The more users that participate, the harder for the 51% attack to happen since overly high computational power is needed and the data modified might not be able to cover the expenses needed. To describe how BitID works, it can be implemented based on public key cryptography (Larcheveque 2016).

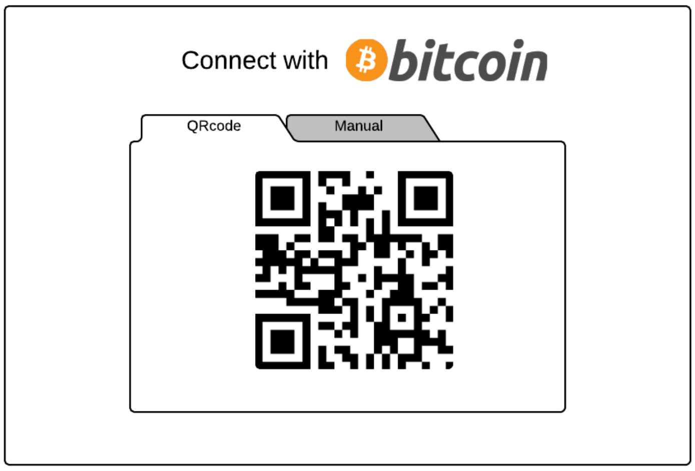


Figure 2.2 QR code that contains BitID

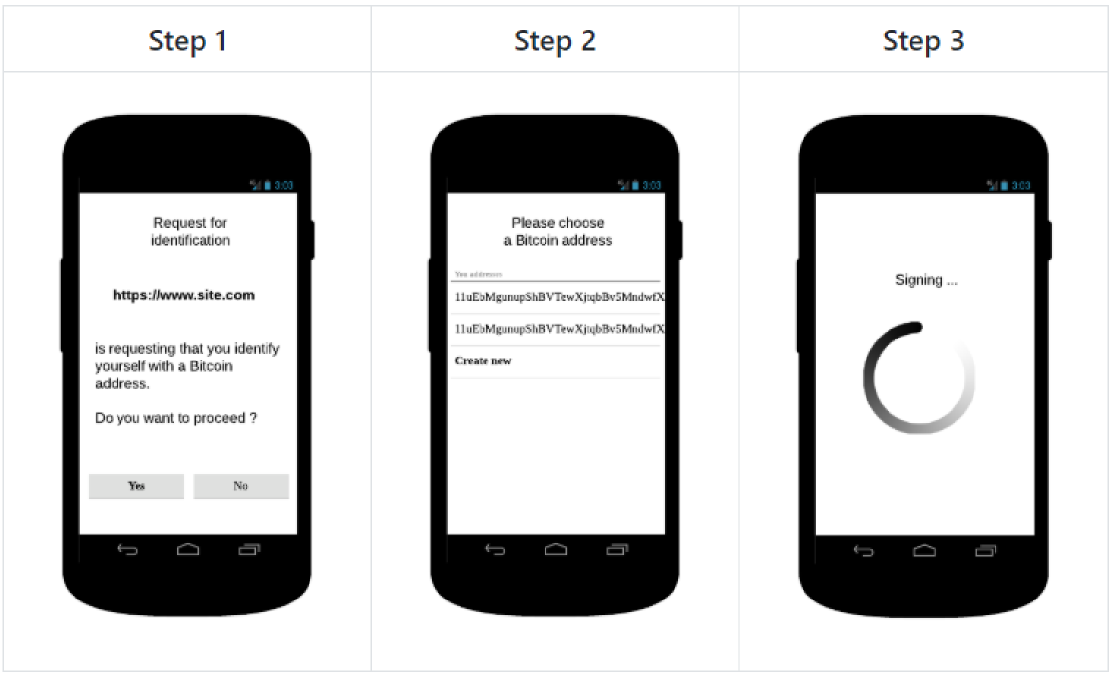

Step 1
Step 2
Step 3
Request for identification
https://www.site.com
is requesting that you identify yourself with a Bitcoin address.
Do you want to proceed ?
Yes
No
Please choose a Bitcoin address
11uEbMgunupShBVTewXjqbBv5MndwfX
11uEbMgunupShBVTewXjqbBv5MndwfX
Create new
Signing ...

Figure 2.3 User interface of mobile application for Bitcoin

Based on Figure 2.2, the user interface is shown to the user for the user to access a restricted area or authenticate himself against a given service. Once the user uses his phone to scan the QR code, the user will have to choose which Bitcoin private key will be used to sign the QR code contents (Figure 2.3). After selecting the desired Bitcoin address, the full BitID URI is signed with the Bitcoin address' private key. The user's public key will store in the server side, which is a secured proposed method whereby it can prevent authentication credentials getting stolen due to data breaches.

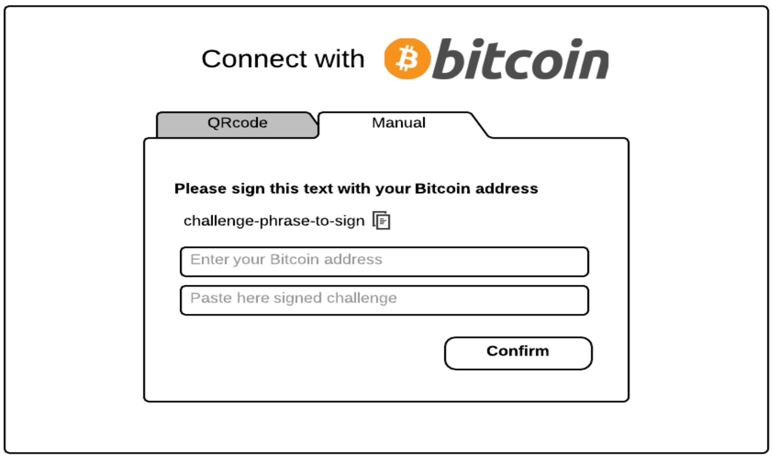

Connect with bitcoin
QRcode
Manual
Please sign this text with your Bitcoin address
challenge-phrase-to-sign
Enter your Bitcoin address
Paste here signed challenge
Confirm

Figure 2.4 Manual way to authenticate with BitID

Alternatively, user can also sign in manually by typing out all the necessary details related to the BitID protocol. This is prepared since not all wallets provide the support for the proposed BitID scheme using QR code. User will need to remember his bitcoin address to be able to proceed to connect to the network, which is a weakness itself since the problem for current authentication systems is the memory of users. It is unnecessary for users needing to remember their information in every platform, and this should be solved by using the proposed solution in this project.

Comparing this solution with the proposed solution in this project, the problem for needing to remember the Bitcoin address to be able to sign in to the system is solved. For this project, once the user is added into the blockchain for the network, it will be able to connect to the wireless APs. Once connected, the user can also communicate to the other wireless APs in the network to be able to improve the network coverage. This reduces the need for duplicated authentication methods needed in a single network.

2.5 Ethernet Level Authentication using Marconi Protocol

Based on the white paper by Marconi Foundation (2018), Marconi protocol is known as a networking and blockchain protocol that uses smart contracts for network packet. It is designed to meet the problems inherent with centralised systems. Marconi protocol is engineered down to the Ethernet level, it takes part in the layer 2 of the OSI model to work together with wired and wireless standards. It is responsible to virtualise and bind OSI layer 2 connections to allow smart logic for network packets. Marconi Foundation is convinced that by restructuring Ethernet, which is one of the Internet's core technologies, the new protocol Marconi protocol will be able to solve security and privacy problems, which are in a rise these days (Kline 2019).

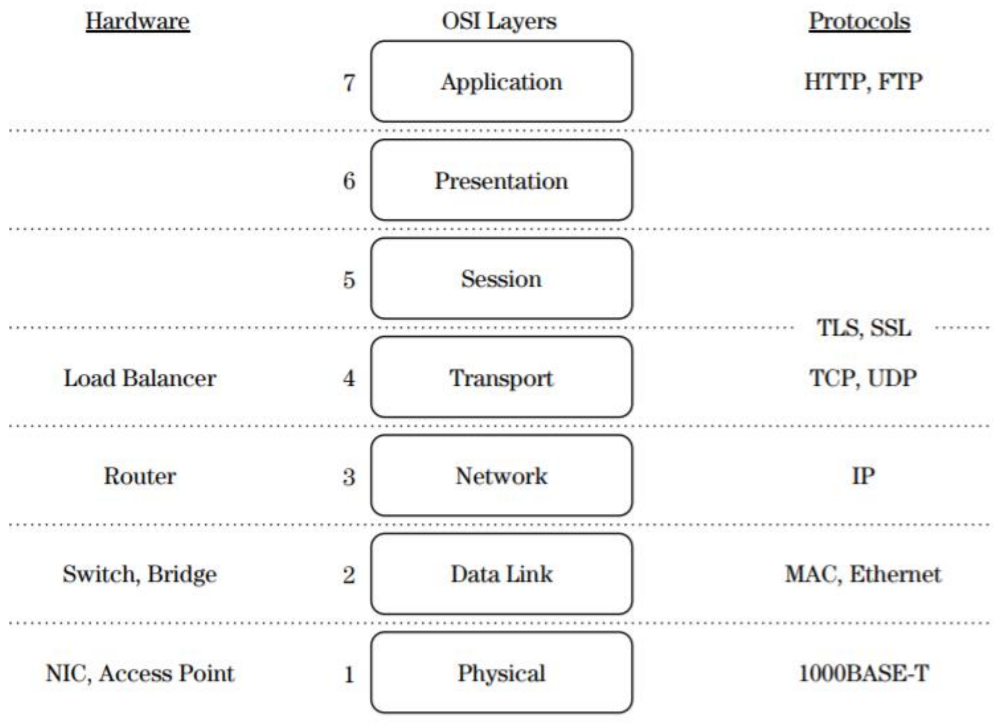


Figure 2.5 OSI model including relation with hardware and protocols – Marconi Protocol by Marconi Foundation, 2018

The network traffic between peers in the network will be routed via Marconi Pipes, which is also known as Marconi Link. It allows custom packet routing, processing and encryption by implementing the blockchain idea. The whole idea is applied on the layer 2 of the OSI model. It also enhances security via smart packet contracts, which are embedded into network packets. The Marconi network will be generated by multiple blockchains to keep track of nodes in the network and establishing a peer-to-peer network. The bandwidth capacity or contribution of each node can also be measured in the Marconi network. Branch chains will exist due to the usage of PoW consensus algorithm next to the global chain. Branch chains allows the new creation of blockchains, which runs in parallel to the global chain.

Marconi Protocol is proposed to allow organisations to build blockchain-backed networking applications. Challenges such as load balancing, inconsistent network topology and IoT device management can be overcome by using this protocol to create networking applications. With decentralised traffic auditing and metering, secure mesh networks can be formed. The ownership of network infrastructure can be decentralised, while each node in the network is authenticated with asymmetric keys. The traffic of the network is encrypted, hence, security is enabled automatically. There will be rules defined for peers in the network to securely connect and communicate with each other.

The other strengths of Marconi Protocol is that they are able to secure Ethernet with packet-level encryption. The challenges in today’s network infrastructure is security, whereby networks are insecure which allows attacker to penetrate into the network easily. By using this protocol which is blockchain-based, security can be highly enhanced by encrypting the packets in layer 2 of OSI model. Besides that, Marconi protocol enables dynamic network adjustment by enlarging existing network with programmable packets that uses the blockchain idea. This can solve the real world problem of network infrastructure inflexibility. By using Marconi protocol, network can be managed easier by being able to add new nodes into the network dynamically.

As stated in an ICO Review on Marconi Protocol by block42 Team on Medium (2018), the weaknesses of Marconi Protocol is that there might be scaling problems due to the PoW consensus. Scalability in blockchain is measured by how fast a transaction is processed, and the speed of the network. New blocks might take a long time to be created based on the number of blocks in the blockchain. The larger the size of the blockchain, the more computational power needed to perform the mining process to create a new block into the chain. This forms the scalability issue as for PoW, it costs time and effort to perform the mining process as compared to other protocols such as PoS protocol.

To resolve the weakness of scaling problems, PoS protocol can be used instead of PoW, as it requires lower computational power and have faster transaction speeds. PoW and PoS are both different consensus algorithms, whereby they are used by miners to reach consensus in the blockchain network. PoW is currently implemented in the world's most famous blockchain network which is Bitcoin. When consensus is achieved, a new block is able to be produced to the existing blockchain. A consensus mechanism is also able to prevent dispute between blocks, and guard against false transactions happening inside the blockchain network. To clearly describe each consensus algorithms, the table below describes the difference between PoS and PoW in a more distinct way.

Characteristics	**Proof-of-Work (PoW)**	**Proof-of-Stake (PoS)**
Transaction validation process	Mining	Forging
Method to reach consensus	Miners use computing power to solve equations to reach consensus	Stakers use stake (wealth) to reach consensus
Rewards given to contributed users	Miners are rewarded for being the first to solve the equation	Stakers are rewarded by being chosen based on their stake amount
Energy cost	Less efficient, high energy cost	More efficient, low energy cost
Decentralisation	More centralised for mining communities	More decentralised
Security	More secured	Less secured

Table 2.1 Comparison of PoW and PoS

In the proposed solution for this project, Ethereum will be used as the platform for PoS. The transaction speed for the network will be improved, and it will improve the reliability and connectivity of the end users towards the network.

2.6 Self-sovereign Identity (SSI)

Developed by the Sovrin Foundation, Sovrin Network is a public service utility which allows the SSI on the Internet. Self-sovereign means that the individual identity holder is able to control their credentials, to share their desired credentials without being forced (Sovrin Foundation 2017). Sovrin Network allows people, organisations and IoT devices to prove identities about themselves, and share necessary identities to desired parties without any intermediaries. They are able to communicate using data that other parties can automatically verify as being the authentic data using blockchain and cryptography technology. The means of authentication is possible to become an abstract as a pluggable type of verifiable credential (Ruff 2018).

Researchers in Sovrin Foundation proposed this solution to solve the problem of security in the authentication process. In the physical world, physical credentials are used to prove a person's identity, such as identification card and credit cards. However, on the Internet, users use very fragile authentication methods which can easily be exposed to attackers, such as username and passwords. Therefore, Sovrin Foundation proposes to invent an equivalent solution on the Internet to the physical world. They create the new standard for digital identity, to enable users to authenticate themselves to any parties that join the blockchain network. They aim to evolve the current system of siloed identities, and insecure databases in organisations. By using the blockchain technology, trust can be generated by parties as the information will not be easily tampered by attackers.

Revocation is also possible whereby the issuers can revoke digital credentials to undo a mistaken issuance. Without informing the issuer, the verifier has the ability to determine whether a submitted credential has been revoked (Ruff 2018). Using Sovrin-style revocation, the issuers can update revocation registries stored on the public ledger periodically, and the updates will become instantly available to verifiers in the blockchain.

The weakness of Sovrin's solution is that SSI do not seem to provide or require any verifiable guarantees as for the accurate functioning of agents in the network. They do not meet the property which is provable, whereby the identity credentials might be not accurately proven. There is no guarantee to fully prevent attacks from happening, in case an attack happens, the credentials will be a mess and things might not go the way as expected. The lack of centralised control also allows attackers to easily abuse the system.

This weakness can be resolved by building a private blockchain instead of a public blockchain. In a private blockchain, permissions can be set to participants in the blockchain. Although a private blockchain is not as decentralised as a public blockchain, however, it still serves the purpose of blockchain which is a secured distributed ledger. It is also decentralised in a sense that there is no need for a single authentication server needing to store all the authentication credentials. Instead, the information necessary will be stored in each node which is connected to the network.

In the proposed solution for this project, a private blockchain using Ethereum will be implemented on the Raspberry Pi and the end devices in the network. Raspberry Pi will act as a wireless AP, and user inside the same blockchain as the wireless AP will be able to connect to it and obtain Internet access. The Raspberry Pi wireless AP will be able to add the user as a peer, to prove that the user is a trusted device in the same blockchain network.

CHAPTER 3: SYSTEM METHODOLOGY

3.1 System Development Model

For this project, the waterfall model will be used as the development methodology. This development methodology shows a linear sequential flow, each phase must be completed before beginning the next phase. There is no overlapping for each phases and it happens sequentially.

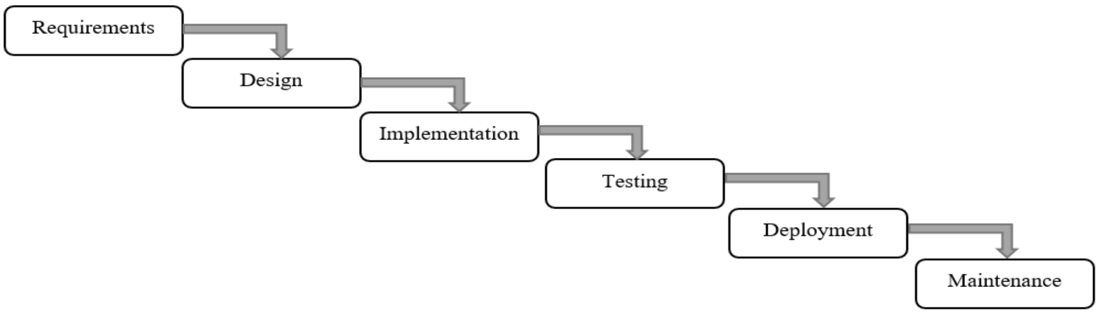


Figure 3.1 Waterfall Model

In the first phase known as the requirements gathering and analysis phase, all the requirements that are needed for the development of the system are gathered. The project documentation is created, whereby objectives and scope of the project are established. This phase is to understand the reason to create the system and the contribution that this system is able to provide. It is known that this project is able to provide a more secured Wi-Fi authentication system despite the current authentication methods in the real world right now. In the second phase known as the system design phase, the hardware and system requirements are specified to define the overall system architecture. The procedures and methods to achieve the goals of this project is also planned. In the third phase known as the implementation phase, the system is developed in small units of programs based on the system design plan. This is where the network can be configured to allow connectivity flow in the network. In the fourth phase known as the testing phase, the units of programs developed in the implementation phase are integrated and tested for its functionality. Testing plans will be done such as testing ping results and performing performance study. In the fifth phase known as the deployment phase, after the testing is done successfully, it will be deployed in the respective environment. For this project, it is only developed in the home environment. In the last phase known as the maintenance phase, once the system developed is ready to be used, new nodes can be continuously added to the blockchain to enable more nodes to connect to the network.

3.2 System Requirement

3.2.1 Software Tools and Libraries

1) Ethereum

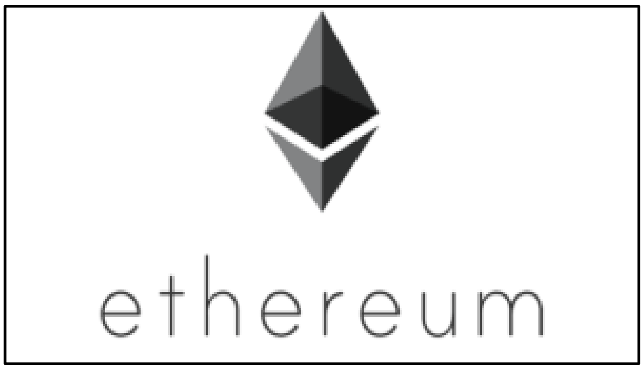


Figure 3.2 Ethereum

Ethereum, developed by Ethereum Foundation, is a global decentralised open source blockchain which includes the feature of programmable smart contracts. The tokens between nodes are exchanged via smart contracts. Ethereum uses Proof-of-Stake consensus mechanism instead of Proof-of-Work to improve the scalability and accessibility of the blockchain (Muzzy 2020). In this project, Ethereum is installed in Raspberry Pi.

2) Go Ethereum

Go Ethereum is also known as Geth. It is written in Go, which is one of the three original implementations of Ethereum protocol. The other two implementations are in C++ and Python respectively. Go Ethereum is cross platform, it can be installed as a standalone client on almost any operating system, or can be embedded as a library in projects.

3) Web3.js

Web.js is a collection of libraries which allows the interaction with local or remote Ethereum node using IPC, HTTP or WebSocket. It is a JavaScript API which can be easily applied in HTML files running JavaScript. In this project, it is used in the captive portal to interact with the Ethereum node to perform authentication.

4) Nodogsplash

Nodogsplash is an open source project which allows simple implementation of the captive portal. It is easy to be configured to self-preference and it incorporates an API that allows the creation of authentication modules.

5) VNC Viewer

VNC Viewer, developed by RealVNC Ltd, is used to remotely access to the Raspberry Pi to ease the process of configuring it.

6) Win32 Disk Imager

Win32 Disk Imager is a tool used to write and read images to the Raspberry Pi Micro SD card. It can be used to create a backup for the image in case any issue occurs.

7) Fing

Figure 3.3 Fing

This is a network tool used to scan the Wi-Fi and know the IP addresses of the devices connected to the Wi-Fi. It is cross platform, in this project, it is used in both the end device which are laptop and mobile phone.

3.2.2 Hardware Tools

1) Wireless N USB Adapter TP-Link TL-WN722N

Figure 3.4 Wireless N USB Adapter TP-Link TL-WN722N

This 150Mbps High Gain Wireless USB network adapter is used to monitor the network traffic between end devices and the three Raspberry Pi. The driver should be reinstalled using the manual found on Google to enable monitor mode.

2) Raspberry Pi 3 Model B, Raspberry Pi 3 Model B+

Figure 3.5 Raspberry Pi 3 Model B (left), Raspberry Pi 3 Model B+ (right)

Raspberry Pi 3 Model B is the first for the third generation Raspberry Pi. It comes with built-in wireless capabilities, which allows it to be configured to act as a wireless AP. Ethereum can also be installed into Raspberry Pi to allow the configuration for blockchain. Raspberry Pi 3 Model B+ is a better version of Raspberry Pi 3 Model B with more capabilities. It has the same built-in wireless capabilities as the Model B version, just that some capabilities were improved such as Bluetooth, Ethernet, and PoE. It does not affect much by using two different models as they still belong in the same generation. Both of the Rapsberry Pi model will not cause the difference in this project, therefore, it is categorized as the same hardware.

3) Micro SD Card

Micro SD card is needed to store and run the Operating System for Raspberry Pi which is known as Raspbian. In this project, two 16GB Micro SD card are used for both models.

4) Micro USB Power Connector

Micro USB Power Connector with output 2.5A is used to power up both models of Raspberry Pi respectively.

5) Micro SD Card Reader

Micro SD Card Reader is used to read and write the Micro SD Card that is needed to run the Raspbian. With a brand new Micro SD Card, the image of the Raspbian can be installed into the card by using a personal computer. Some configurations can also be tweaked in the computer before plugging the card into the Raspberry Pi.

6) Ethernet Cable

To connect the PC and the Raspberry Pi to configure the Raspberry Pi to become a wireless AP.

3.3 Functional Requirements

3.3.1 Raspberry Pi Obtain Internet Connectivity from Router

The three Raspberry Pi will be using the wlan0 interface to connect to the router. They will be using static IP address to connect to the router. The Internet access will be routed to wireless AP and allow users connecting to it to gain Internet access.

3.3.2 Configure Raspberry Pi as Wireless Access Point

The three Raspberry Pi will use host access point daemon (hostapd) to enable network interface cards to be turned into access point. The end devices will be able to connect to the ap@wlan0 interface of Raspberry Pi, and obtain Internet access.

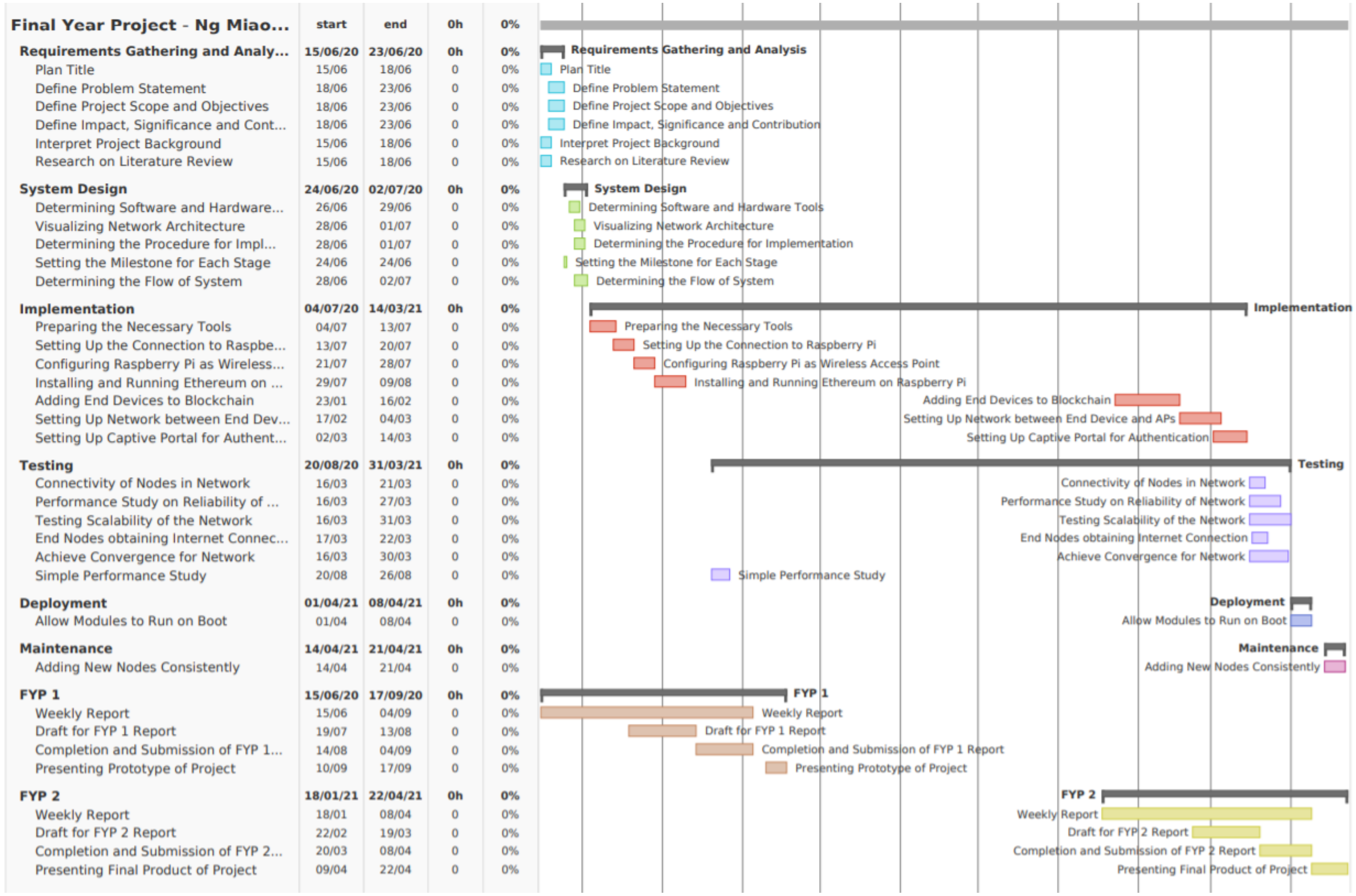
3.3.3 Implement Blockchain Network

Three of the Raspberry Pi will be configured with Ethereum blockchain. The first Raspberry Pi will be in a different blockchain with the second Raspberry Pi and the third Raspberry Pi. The blockchain in first Raspberry Pi will not have any peers, which mean that it will not have any trusted devices. The blockchain for second Raspberry Pi and the third Raspberry Pi will have peer which is the laptop end device. Therefore, the end device will also be running the Ethereum blockchain to be able to connect to the network of second Raspberry Pi and third Raspberry Pi. Other end devices which are not in any list of trusted devices of the Raspberry Pi using blockchain authentication system will not be able to obtain Internet access from the Raspberry Pi wireless AP.

3.3.4 Configure Captive Portal to Perform Authentication on End Device

The Raspberry Pi wireless AP will implement a captive portal to perform authentication, which is to check whether the end device connected to it is inside the same blockchain. If they are in the same blockchain, then the end device is able to obtain Internet connection from the wireless AP. If they are not in the same blockchain, then the authentication process will show failed to allow the end device to obtain Internet connection. Therefore, even if the device is connected to the wireless AP, he will not be able to enter the network as it is not able to obtain Internet connection.

3.4 Timeline



Final Year Project - Ng Miao...	start	end	0h	0%
Requirements Gathering and Analy...	**15/06/20**	**23/06/20**	**0h**	**0%**
Plan Title	15/06	18/06	0	0%
Define Problem Statement	18/06	23/06	0	0%
Define Project Scope and Objectives	18/06	23/06	0	0%
Define Impact, Significance and Cont...	18/06	23/06	0	0%
Interpret Project Background	15/06	18/06	0	0%
Research on Literature Review	15/06	18/06	0	0%
System Design	**24/06/20**	**02/07/20**	**0h**	**0%**
Determining Software and Hardware...	26/06	29/06	0	0%
Visualizing Network Architecture	28/06	01/07	0	0%
Determining the Procedure for Impl...	28/06	01/07	0	0%
Setting the Milestone for Each Stage	24/06	24/06	0	0%
Determining the Flow of System	28/06	02/07	0	0%
Implementation	**04/07/20**	**14/03/21**	**0h**	**0%**
Preparing the Necessary Tools	04/07	13/07	0	0%
Setting Up the Connection to Raspbe...	13/07	20/07	0	0%
Configuring Raspberry Pi as Wireless...	21/07	28/07	0	0%
Installing and Running Ethereum on ...	29/07	09/08	0	0%
Adding End Devices to Blockchain	23/01	16/02	0	0%
Setting Up Network between End Dev...	17/02	04/03	0	0%
Setting Up Captive Portal for Authent...	02/03	14/03	0	0%
Testing	**20/08/20**	**31/03/21**	**0h**	**0%**
Connectivity of Nodes in Network	16/03	21/03	0	0%
Performance Study on Reliability of ...	16/03	27/03	0	0%
Testing Scalability of the Network	16/03	31/03	0	0%
End Nodes obtaining Internet Connec...	17/03	22/03	0	0%
Achieve Convergence for Network	16/03	30/03	0	0%
Simple Performance Study	20/08	26/08	0	0%
Deployment	**01/04/21**	**08/04/21**	**0h**	**0%**
Allow Modules to Run on Boot	01/04	08/04	0	0%
Maintenance	**14/04/21**	**21/04/21**	**0h**	**0%**
Adding New Nodes Consistently	14/04	21/04	0	0%
FYP 1	**15/06/20**	**17/09/20**	**0h**	**0%**
Weekly Report	15/06	04/09	0	0%
Draft for FYP 1 Report	19/07	13/08	0	0%
Completion and Submission of FYP 1...	14/08	04/09	0	0%
Presenting Prototype of Project	10/09	17/09	0	0%
FYP 2	**18/01/21**	**22/04/21**	**0h**	**0%**
Weekly Report	18/01	08/04	0	0%
Draft for FYP 2 Report	22/02	19/03	0	0%
Completion and Submission of FYP 2...	20/03	08/04	0	0%
Presenting Final Product of Project	09/04	22/04	0	0%

Requirements Gathering and Analysis
Plan Title
Define Problem Statement
Define Project Scope and Objectives
Define Impact, Significance and Contribution
Interpret Project Background
Research on Literature Review
System Design
Determining Software and Hardware Tools
Visualizing Network Architecture
Determining the Procedure for Implementation
Setting the Milestone for Each Stage
Determining the Flow of System
Implementation
Preparing the Necessary Tools
Setting Up the Connection to Raspberry Pi
Configuring Raspberry Pi as Wireless Access Point
Installing and Running Ethereum on Raspberry Pi
Adding End Devices to Blockchain
Setting Up Network between End Device and APs
Setting Up Captive Portal for Authentication
Testing
Connectivity of Nodes in Network
Performance Study on Reliability of Network
Testing Scalability of the Network
End Nodes obtaining Internet Connection
Achieve Convergence for Network
Simple Performance Study
Deployment
Allow Modules to Run on Boot
Maintenance
Adding New Nodes Consistently
FYP 1
Weekly Report
Draft for FYP 1 Report
Completion and Submission of FYP 1 Report
Presenting Prototype of Project
FYP 2
Weekly Report
Draft for FYP 2 Report
Completion and Submission of FYP 2 Report
Presenting Final Product of Project

Figure 3.6 Gantt Chart for Project Planning

CHAPTER 4: SYSTEM DESIGN

In this chapter, the proposed pervasive blockchain based authentication method is demonstrated.

4.1 Pervasive Wi-Fi Authentication Framework

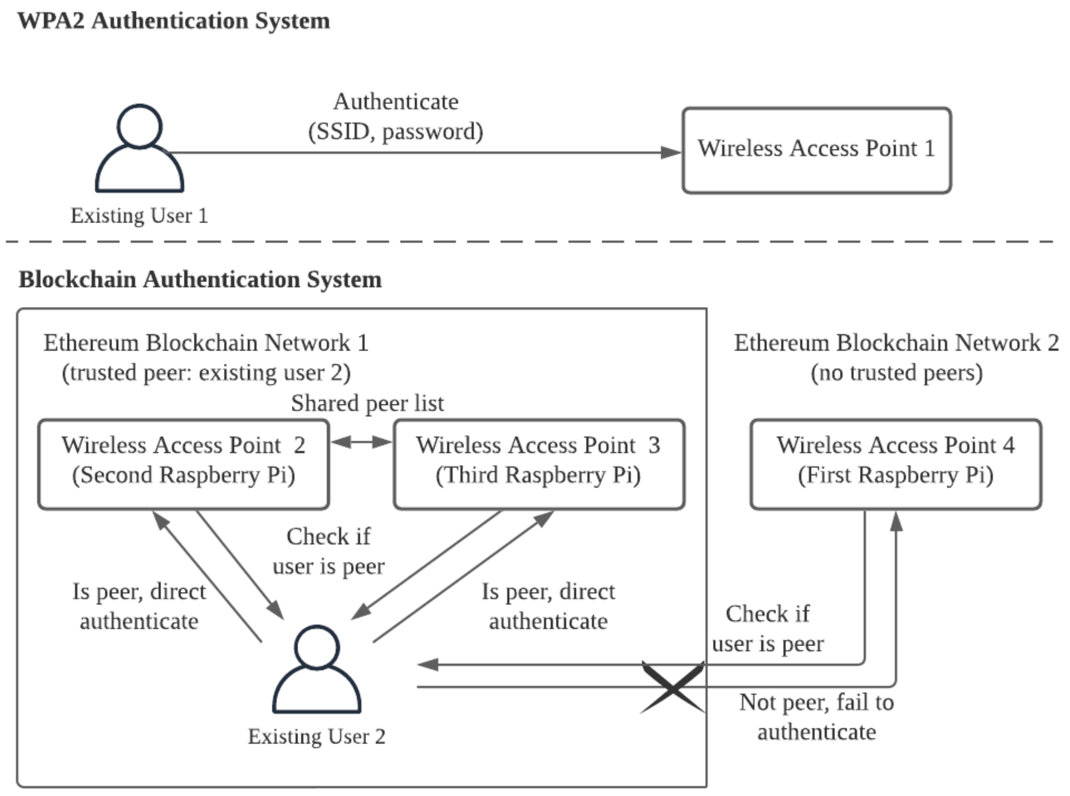

WPA2 Authentication System
Authenticate (SSID, password)
Wireless Access Point 1
Existing User 1
Blockchain Authentication System
Ethereum Blockchain Network 1 (trusted peer: existing user 2)
Shared peer list
Wireless Access Point 2 (Second Raspberry Pi)
Wireless Access Point 3 (Third Raspberry Pi)
Check if user is peer
Is peer, direct authenticate
Is peer, direct authenticate
Existing User 2
Ethereum Blockchain Network 2 (no trusted peers)
Wireless Access Point 4 (First Raspberry Pi)
Check if user is peer
Not peer, fail to authenticate

Figure 4.1 Ethereum Based Wi-Fi Authentication Diagram

The diagram above shows the Ethereum Based Wi-Fi authentication system. It is shown that existing user 1 for the WPA2 Authentication System is able to connect to the wireless AP 1 using WPA2. The existing user 1 will need to select the SSID of the wireless AP 1 and insert the password of the wireless AP 1. The wireless AP 1 will go through the WPA2-PSK encryption technique to authenticate the user.

For the blockchain authentication system, the existing user 2 is able to connect to the wireless AP 2 and wireless AP 3 in the same blockchain network as him. This is because they have added existing user 2 as a peer. The wireless AP 2 and wireless AP 3 will have the same SSID, therefore when the existing user 2 chooses to connect to the SSID, the nearest wireless AP will be chosen to be connected. The wireless AP 2 or wireless AP 3 will check whether the user is a peer, if he is then it will authenticate the user.

The existing user 2 is not able to connect to the wireless AP 4 as wireless AP 4 does not add existing user 2 as a peer. Wireless AP 4 is inside a blockchain different with wireless AP 2 and AP 3, and he has no peer in the blockchain. Therefore, wireless AP 4 will fail to authenticate the existing user 2.

Pervasive computing is implemented in this proposed system as the existing user 2 can direct authenticate to the wireless AP inside the same blockchain as him. This can happen only if the blockchain adds the existing user 2 as a trusted device. As for the existing network using WPA2, he will need to provide the password only to be able to successfully connect to the wireless AP.

4.2 Network Architecture

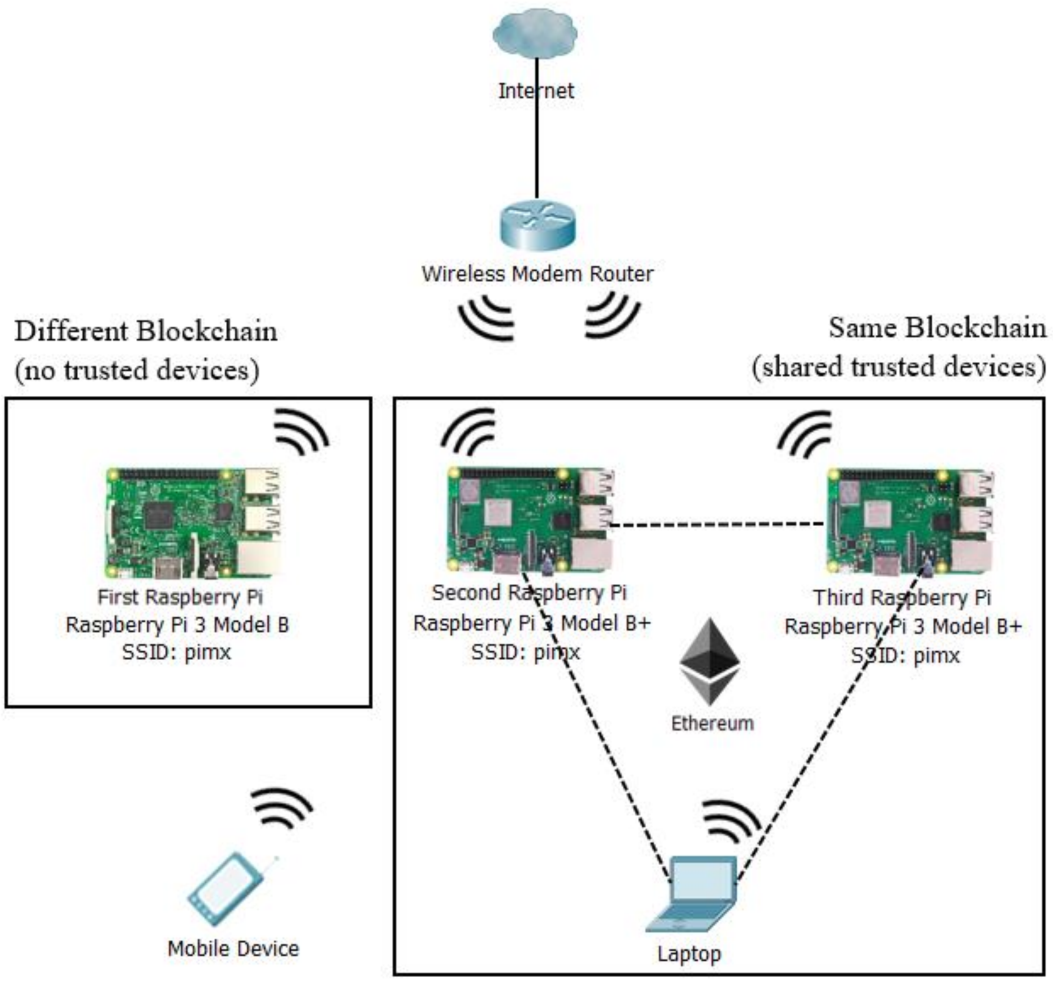


Figure 4.2 Network Architecture Diagram

Based on the network architecture diagram above, it is shown whereby three Raspberry Pi 3 boards will be used to be configured as wireless APs to provide internet connectivity to the end devices which are laptop and mobile device. For the Raspberry Pi 3 boards to get internet connectivity, they will connect to the wireless modem router wirelessly. The three Raspberry Pi will be using static IP address to connect to the wireless modem router, while end devices will get IP address from each Raspberry Pi wireless AP depending on which one they connect.

The first Raspberry Pi, second Raspberry Pi and third Raspberry Pi will be using open authentication with Ethereum blockchain configured as the authentication system. However, the first Raspberry Pi will not have any trusted devices. The second Raspberry Pi and the third Raspberry Pi will have one trusted devices in its list, which is the end device laptop using Windows 10. The second and third Raspberry Pi will share a trusted list of peer since they are in the same blockchain network with the same genesis block.

The end devices will connect to the nearest wireless AP. Configuring more wireless AP in a network allows for wider network coverage. The laptop end device connected to the first Raspberry Pi will not be able to access the Internet because he is not inside the list of trusted devices of the first Raspberry Pi. The end device connected to the second Raspberry Pi or third Raspberry Pi will be able to gain Internet access because he is inside the list of trusted devices of the wireless AP blockchain network.

The mobile device which is outside of the blockchain will not be able to obtain Internet connection by connecting to the wireless APs using the blockchain authentication system. He will only be able to connect to the wireless APs inside the same blockchain as him or with no blockchain.

4.3 Functional Modules

4.3.1 Wireless Access Point Module

The three Raspberry Pi will be configured as a wireless AP to provide Internet connectivity to end devices. It will be using the hostapd module which is used to convert network interface into access point interface mode. It is able to add a new interface which is in access point mode, to allow end devices to connect to it.

4.3.2 Blockchain Module

Go Ethereum (Geth) will be installed in all the Raspberry Pi and one end device. First and second Raspberry Pi together with an end device will form a blockchain network, this is performed by wireless AP Raspberry Pi nodes adding the end device as a trusted peer. The two Raspberry Pi will be using the same genesis node, so that they are able to exist inside the same blockchain and connect to each other. The first Raspberry Pi will install Geth but will not have any peers. Therefore, the end device will not be able to connect to it to gain Internet access.

4.3.3 Authentication Module

A captive portal will be configured in the all the Raspberry Pi wireless APs. Nodogsplash is an open source project that allows easy implementation of the captive portal. Web3.js will be used to interact with the blockchain module, to check if the end device is inside the blockchain same as the wireless AP, then allow Internet connectivity to the end device. It is using JavaScript which will be easily implemented with the captive portal.

4.4 The Intuition of Ethereum-based Authentication

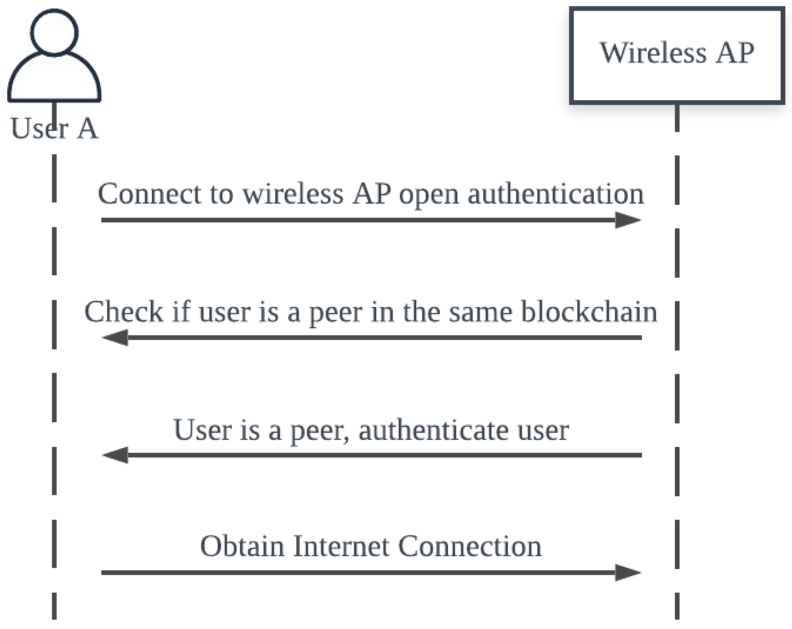

User A
Wireless AP
Connect to wireless AP open authentication
Check if user is a peer in the same blockchain
User is a peer, authenticate user
Obtain Internet Connection

Figure 4.3 Process flow of user authentication in the Ethereum-based network

When user A connects to wireless AP using open authentication, he will not be able to obtain Internet access at this step. After that, a captive portal will be shown to the user to authenticate the user. The wireless AP using Ethereum-based authentication system will check whether the user is in the same blockchain. If user is a peer, then authenticate the user. The user will be able to obtain Internet connection upon successful authentication.

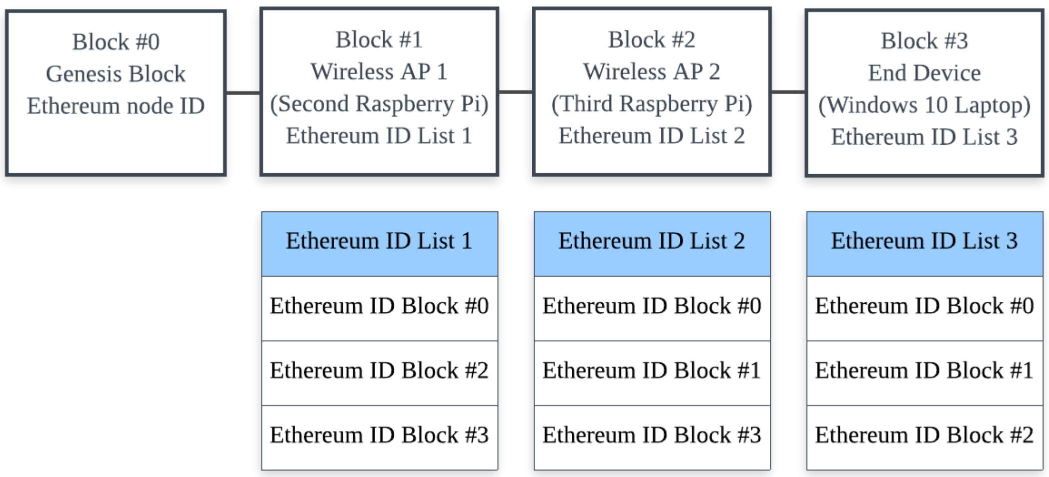


Figure 4.4 Ethereum Blockchain Details (with peers)

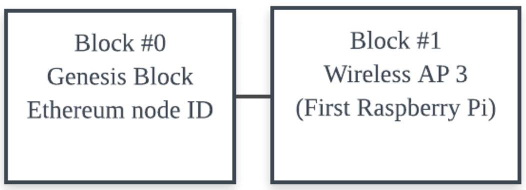


Figure 4.5 Ethereum Blockchain Details (with no peers)

Figure 4.4 shows the Ethereum Blockchain network details with peers. By using the same genesis block file, second Raspberry Pi is chained with third Raspberry Pi, with the shared peer which is the end device. End device is able to successfully being authenticated by second Raspberry Pi and third Raspberry Pi since he is inside the same blockchain as them.

Figure 4.5 shows another Ethereum Blockchain network details with no peers. The first Raspberry Pi is configured with no peers, therefore no end device is able to successfully undergo the blockchain authentication process since there are no peers inside the blockchain.

4.5 System Flow

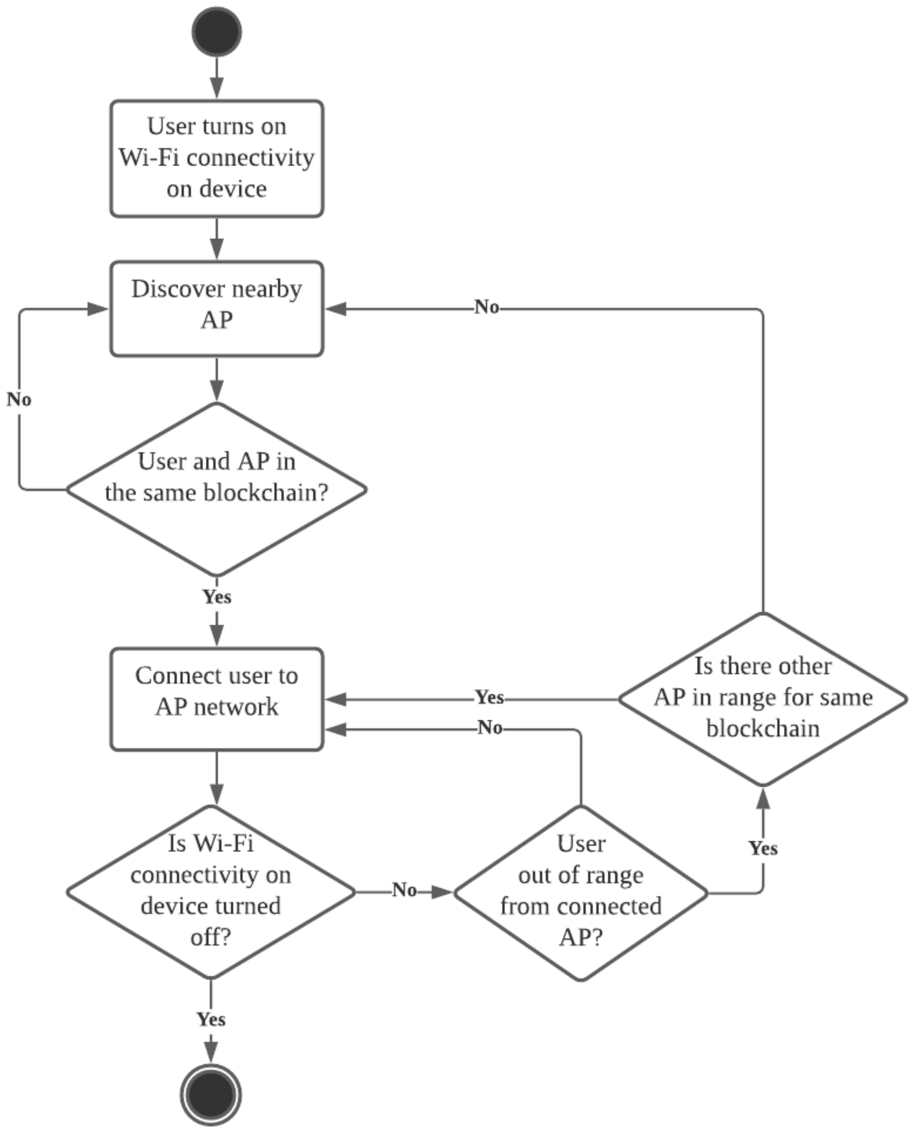

User turns on Wi-Fi connectivity on device
Discover nearby AP
No
No
User and AP in the same blockchain?
Yes
Connect user to AP network
Yes
No
Is there other AP in range for same blockchain
Is Wi-Fi connectivity on device turned off?
No
User out of range from connected AP?
Yes
Yes

Figure 4.6 Activity Diagram for Wi-Fi Authentication System

The above figure shows the overall flow of the Wi-Fi authentication system that will be done for this entire project. First, the user will turn on the Wi-Fi connectivity on his device, then the Wi-Fi will automatically discover for nearby AP. When the user and AP is in the same blockchain, then the user will be connected to the AP network in the blockchain. If user does not turn off his Wi-Fi connectivity, then he will remain connected to the network. However, if he is out of range, then it will check whether there is another AP in range for the same blockchain, if yes then he will connect to it easily. If no, then it will discover nearby AP for other blockchain network. A user can join multiple blockchain networks at a time. If user turns off his Wi-Fi connectivity, then the process ends.

CHAPTER 5: IMPLEMENTATION

5.1 Setting up the Connection to Raspberry Pi

Raspbian is installed in the Micro SD Card before inserting it in the Raspberry Pi. With only a laptop and a Raspberry Pi, after installing new fresh Raspbian image, an empty file named "ssh" needs to be added to the Micro SD Card to allow ssh to the Raspberry Pi. The "cmdline.txt" can also be edited to add a parameter to set static IP for the Raspberry Pi to enable the PC to discover the Raspberry Pi through Ethernet cable.

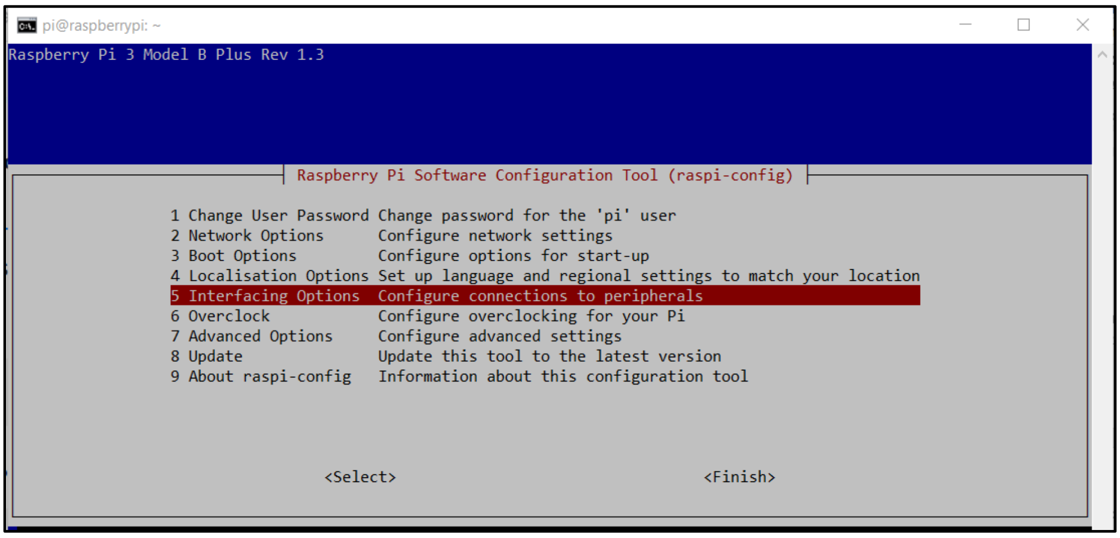
cmdline.txt file:

```
…...ip=192.168.137.13
```

Raspberry Pi is connected to the laptop using Ethernet cable. First, ssh into the Raspberry Pi by opening the Windows command line and type the following commands:

```
ssh pi@192.168.137.13
```

Then, use "sudo raspi-config" to edit the Interfacing Options to enable VNC so that VNC Viewer can be used to view the graphical interface of the Raspberry Pi to allow more flexibility:



```
pi@raspberrypi: ~
Raspberry Pi 3 Model B Plus Rev 1.3

            Raspberry Pi Software Configuration Tool (raspi-config)

 1 Change User Password Change password for the 'pi' user
 2 Network Options      Configure network settings
 3 Boot Options         Configure options for start-up
 4 Localisation Options Set up language and regional settings to match your location
 5 Interfacing Options  Configure connections to peripherals
 6 Overclock            Configure overclocking for your Pi
 7 Advanced Options     Configure advanced settings
 8 Update               Update this tool to the latest version
 9 About raspi-config   Information about this configuration tool

                <Select>                        <Finish>
```

Figure 5.1 Interfacing Options in "raspi-config"

After enabling VNC, next use VNC Viewer to be enable to connect to the Raspberry Pi and view the graphical interface to start configuring the Raspberry Pi. Input the same IP address that is performed in SSH in the VNC Viewer to start the configuration.

5.2 Configuring Raspberry Pi as Wireless Access Point

Due to the limitation of wireless distribution system on Raspberry Pi, to start configuring the AP connecting to the wireless modem router through wlan0 port, systemd-networkd will be used for general networking.

To start, this is the command to change current user to root:

```
$ sudo -Es
```

To uninstall classic networking, the classic networking packages are removed:

```
$ apt --autoremove purge ifupdown dhcpcd5 isc-dhcp-common isc-dhcp-client rsyslog
$ apt-mark hold ifupdown dhcpcd5 isc-dhcp-common isc-dhcp-client rsyslog
raspberrypi-net-mods openresolv
$ rm -r /etc/network /etc/dhcp
```

After removing the classic networking, enable systemd-networkd service:

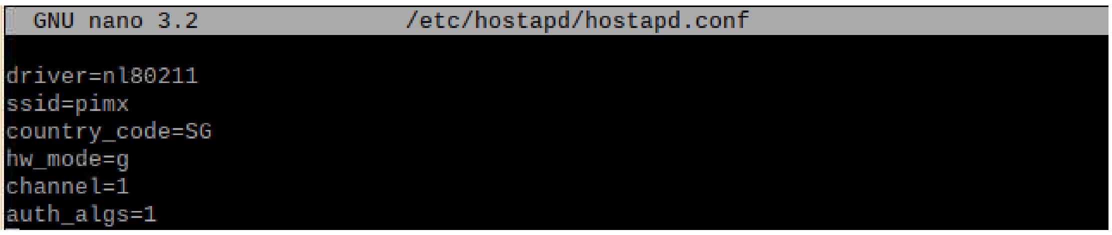
```
$ apt --autoremove purge avahi-daemon
$ apt-mark hold avahi-daemon libnss-mdns
$ apt install libnss-resolve
$ ln -sf /run/systemd/resolve/stub-resolv.conf /etc/resolv.conf
$ systemctl enable systemd-resolved.service systemd-networkd.service
```

Next, install hostapd package to create a wireless hotspot for Raspberry Pi:

```
$ apt-get install hostapd
```

Create a file using nano with details such as ssid (name of wireless hotspot), country_code (SG=Singapore). We will configure this access point as open authentication, therefore no WPA information needed:

```
$ nano /etc/hostapd/hostapd.conf
```



```
  GNU nano 3.2              /etc/hostapd/hostapd.conf

driver=nl80211
ssid=pimx
country_code=SG
hw_mode=g
channel=1
auth_algs=1
```

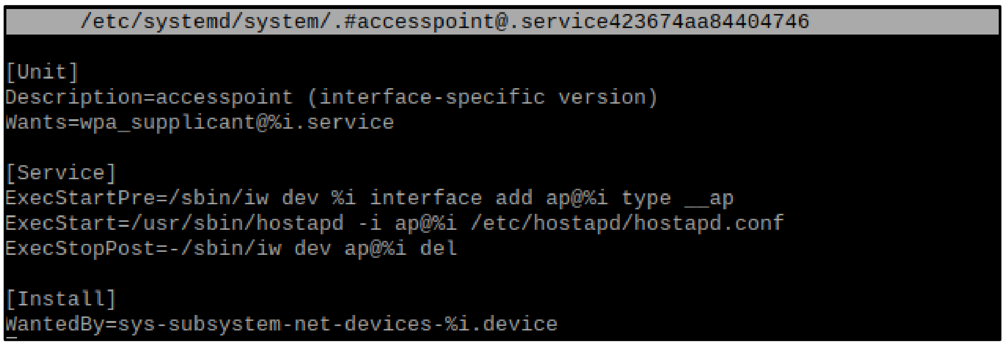
Figure 5.2 /etc/hostapd/hostapd.conf parameters (open authentication)

Edit permissions for the file created, 600 means owner is able to read and write but not execute, while group and others has no rights:

```
$ chmod 600 /etc/hostapd/hostapd.conf
```

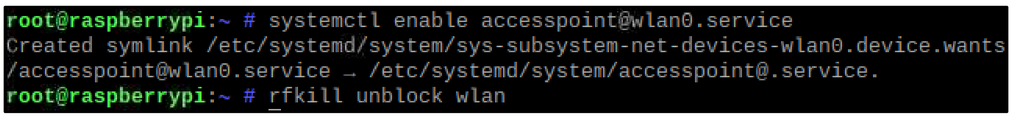
Creating a service for the access point using hostapd, and add the few statements shown in the diagram below:

```
$ systemctl edit --force --full accesspoint@.service
```



```
      /etc/systemd/system/.#accesspoint@.service423674aa84404746

[Unit]
Description=accesspoint (interface-specific version)
Wants=wpa_supplicant@%i.service

[Service]
ExecStartPre=/sbin/iw dev %i interface add ap@%i type __ap
ExecStart=/usr/sbin/hostapd -i ap@%i /etc/hostapd/hostapd.conf
ExecStopPost=-/sbin/iw dev ap@%i del

[Install]
WantedBy=sys-subsystem-net-devices-%i.device
```

Figure 5.3 accesspoint@.service parameters

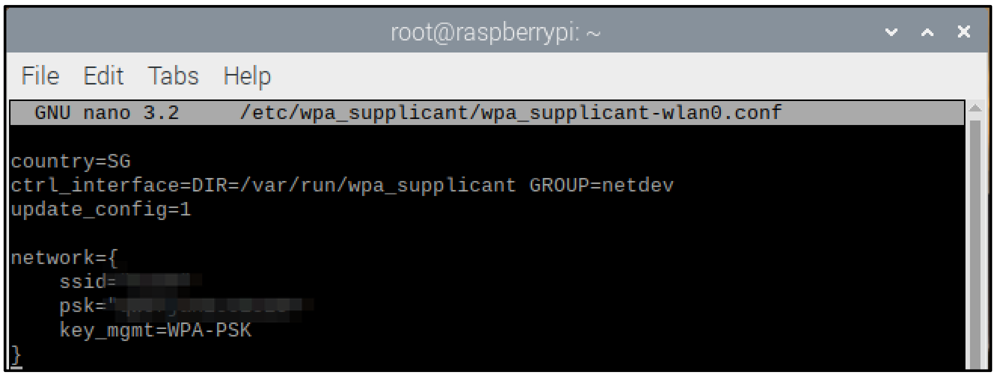


```
root@raspberrypi:~ # systemctl enable accesspoint@wlan0.service
Created symlink /etc/systemd/system/sys-subsystem-net-devices-wlan0.device.wants
/accesspoint@wlan0.service → /etc/systemd/system/accesspoint@.service.
root@raspberrypi:~ # rfkill unblock wlan
```

Figure 5.4 Enabling the service created using the wlan0 port and then unblock wlan

After that, set up wpa_supplicant with the below details to enable client connection to the wireless AP. The network ssid and psk is for the wireless modem router:

```
$ nano /etc/wpa_supplicant/wpa_supplicant-wlan0.conf
```



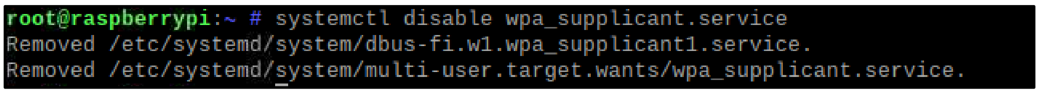
```
  GNU nano 3.2      /etc/wpa_supplicant/wpa_supplicant-wlan0.conf

country=SG
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1

network={
    ssid=[redacted]
    psk="[redacted]
    key_mgmt=WPA-PSK
}
```

Figure 5.5 /etc/wpa_supplicant/wpa_supplicant-wlan0.conf parameters

Edit permissions for the file created, 600 means owner is able to read and write but not execute, while group and others has no rights:

```
$ chmod 600 /etc/wpa_supplicant/wpa_supplicant-wlan0.conf
```

Then, disable the wpa_applicant service before editing it:

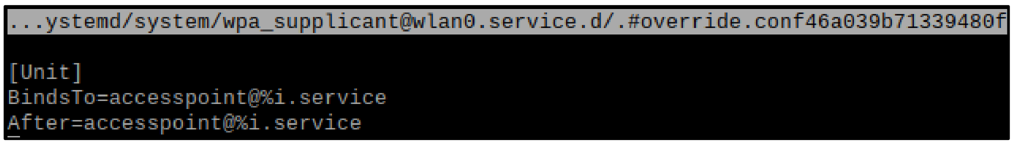


```
root@raspberrypi:~ # systemctl disable wpa_supplicant.service
Removed /etc/systemd/system/dbus-fi.w1.wpa_supplicant1.service.
Removed /etc/systemd/system/multi-user.target.wants/wpa_supplicant.service.
```

Figure 5.6 Disable wpa_applicant service

Now, edit the wpa_applicant service to bind wpa_supplicant to hostapd:

```
$ systemctl edit wpa_supplicant@wlan0.service
```

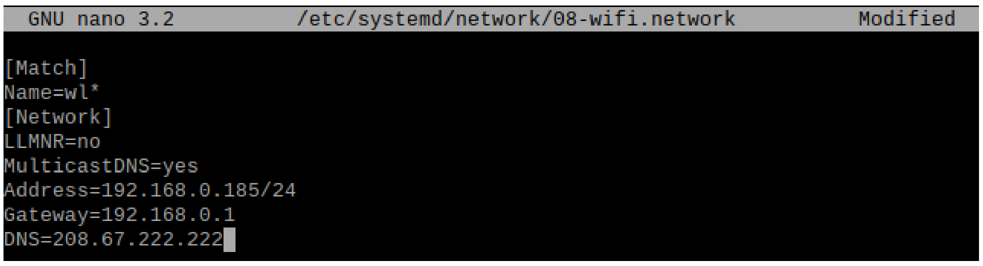


```
...ystemd/system/wpa_supplicant@wlan0.service.d/.#override.conf46a039b71339480f

[Unit]
BindsTo=accesspoint@%i.service
After=accesspoint@%i.service
```

Figure 5.7 wpa_supplicant@wlan0.service parameters

Static IP address is enabled for Raspberry Pi connecting to the wireless modem router. Three static IP address for three Raspberry Pi are 192.168.0.183, 192.168.0.184, and 192.168.0.185. DNS is also configured manually, for this project we use OpenDNS which is a public DNS server address:

```
$ nano /etc/systemd/network/08-wifi.network
```

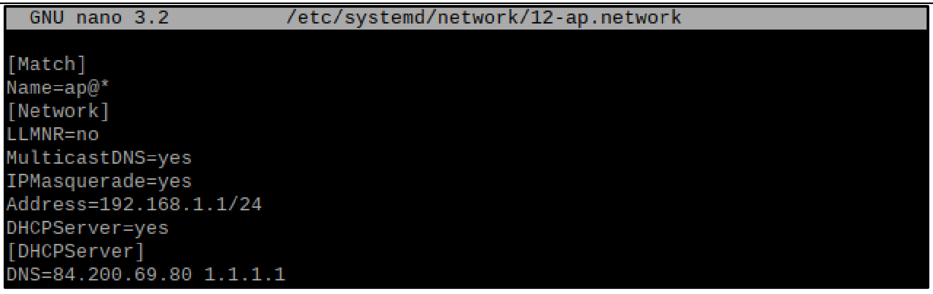


```
  GNU nano 3.2          /etc/systemd/network/08-wifi.network          Modified

[Match]
Name=wl*
[Network]
LLMNR=no
MulticastDNS=yes
Address=192.168.0.185/24
Gateway=192.168.0.1
DNS=208.67.222.222
```

Figure 5.8 /etc/systemd/network/08-wifi.network parameters

Static IP address for the Raspberry Pi as a wireless access point is set. This wireless AP provides DHCP service for devices connecting to it:

```
$ nano /etc/systemd/network/12-ap.network
```

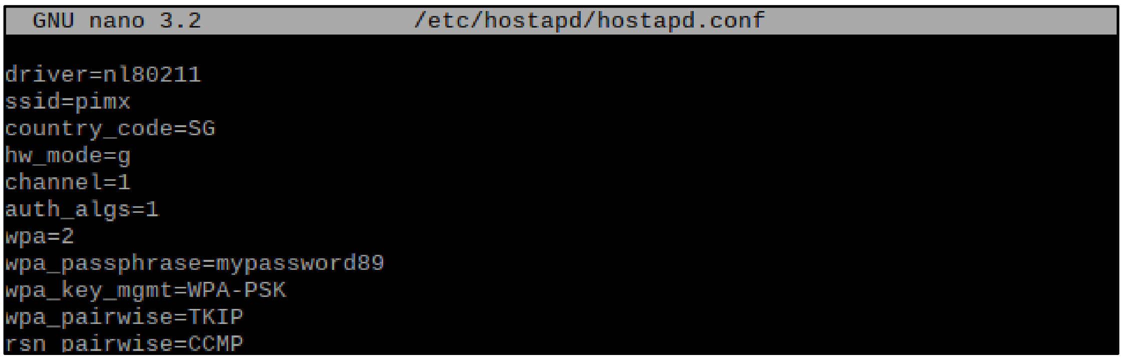


```
  GNU nano 3.2          /etc/systemd/network/12-ap.network

[Match]
Name=ap@*
[Network]
LLMNR=no
MulticastDNS=yes
IPMasquerade=yes
Address=192.168.1.1/24
DHCPServer=yes
[DHCPServer]
DNS=84.200.69.80 1.1.1.1
```

Figure 5.9 /etc/systemd/network/12-ap.network parameters

All the steps above are repeated for the other two Raspberry Pi respectively. To configure for the test bed, we will need to modify one step for the first Raspberry Pi. For the step where we configure the access point configurations, we will add WPA information such as wpa-passphrase (password for wireless hotspot, more than 8 characters):

```
$ nano /etc/hostapd/hostapd.conf
```



```
  GNU nano 3.2              /etc/hostapd/hostapd.conf

driver=nl80211
ssid=pimx
country_code=SG
hw_mode=g
channel=1
auth_algs=1
wpa=2
wpa_passphrase=mypassword89
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
```

Figure 5.10 /etc/hostapd/hostapd.conf parameters (WPA2)

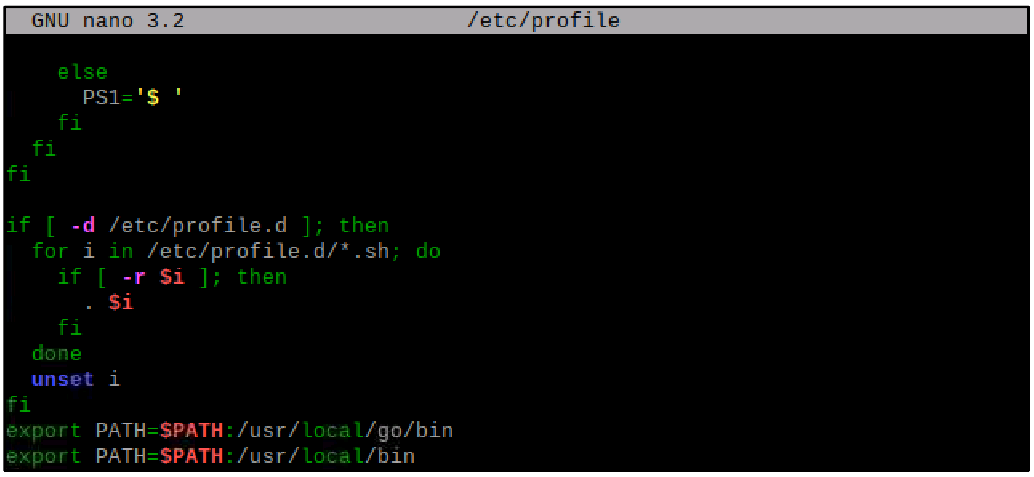
5.3 Installing and Running Ethereum on Raspberry Pi

Installing necessary packages for Ethereum to work on Raspberry Pi, such as Golang and go-ethereum:

```
$ sudo apt-get install git libgmp3-dev
$ git clone -b release/1.7 https://github.com/ethereum/go-ethereum.git
$ wget https://storage.googleapis.com/golang/go1.7.linux-armv6l.tar.gz
$ sudo tar -C /usr/local -xzf go1.7.linux-armv6l.tar.gz
$ cd go-ethereum
$ make
$ make geth
```

Then, copy the geth file to /usr/local/bin for easy access, and add the path for golang and geth to environmental variables at the end of the /etc/profile so that they can be called easily. (“export PATH=$PATH:/usr/local/go/bin” is for golang, “export PATH=$PATH:/usr/local/bin” is for geth):

```
$ sudo cp build/bin/geth /usr/local/bin/
$ sudo nano /etc/profile
```



```
  GNU nano 3.2                          /etc/profile

    else
      PS1='$ '
    fi
  fi
fi

if [ -d /etc/profile.d ]; then
  for i in /etc/profile.d/*.sh; do
    if [ -r $i ]; then
      . $i
    fi
  done
  unset i
fi
export PATH=$PATH:/usr/local/go/bin
export PATH=$PATH:/usr/local/bin
```

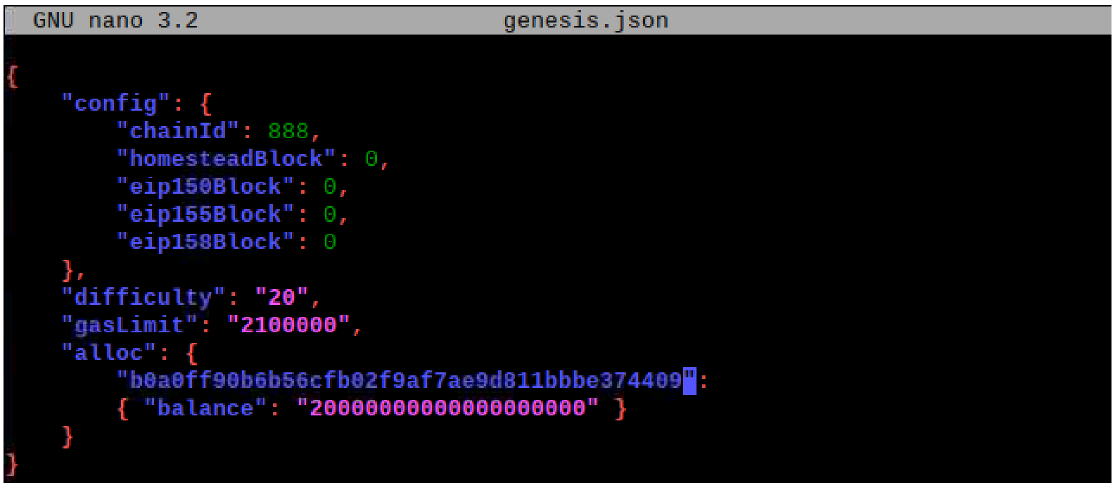
Figure 5.11 Adding environmental variables in /etc/profile

After all the packages necessary are downloaded and installed into the system, start to create a blockchain using Ethereum. A new account for the node is created to generate a hash value for the node:

```
$ sudo geth --datadir=./pimxblockchain account new
```

Next, generate a genesis.json file to initialise the blockchain. ChainId for all pi will be the same, in this project we set it as 888. For every pi, all the genesis.json which is genesis block must be same to enable them to add into the same blockchain. Under alloc, the desired node address is written to allocate some ether to the node address. In this project we allocate only the first Raspberry Pi to get the funds:

```
$ nano genesis.json
```

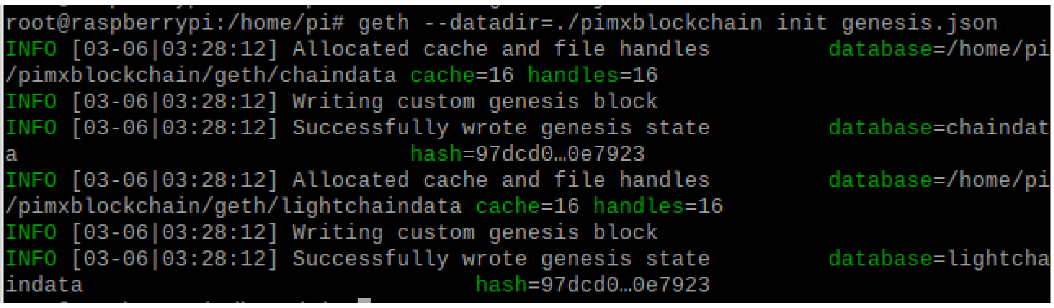


```
  GNU nano 3.2                          genesis.json

{
    "config": {
        "chainId": 888,
        "homesteadBlock": 0,
        "eip150Block": 0,
        "eip155Block": 0,
        "eip158Block": 0
    },
    "difficulty": "20",
    "gasLimit": "2100000",
    "alloc": {
        "b0a0ff90b6b56cfb02f9af7ae9d811bbbe374409":
        { "balance": "20000000000000000000" }
    }
}
```

Figure 5.12 genesis.json file

Now we initialize the blockchain using the genesis.json file created above using the following command:

```
$ geth --datadir=./pimxblockchain init genesis.json
```



```
root@raspberrypi:/home/pi# geth --datadir=./pimxblockchain init genesis.json
INFO [03-06|03:28:12] Allocated cache and file handles         database=/home/pi
/pimxblockchain/geth/chaindata cache=16 handles=16
INFO [03-06|03:28:12] Writing custom genesis block
INFO [03-06|03:28:12] Successfully wrote genesis state         database=chaindat
a                              hash=97dcd0…0e7923
INFO [03-06|03:28:12] Allocated cache and file handles         database=/home/pi
/pimxblockchain/geth/lightchaindata cache=16 handles=16
INFO [03-06|03:28:12] Writing custom genesis block
INFO [03-06|03:28:12] Successfully wrote genesis state         database=lightcha
indata                              hash=97dcd0…0e7923
```

Figure 5.13 Initialising the genesis block (genesis.json)

If the blockchain is unable to initialize, there is a possibility that there is a blockchain currently running, therefore this command can be used to remove it before initialising:

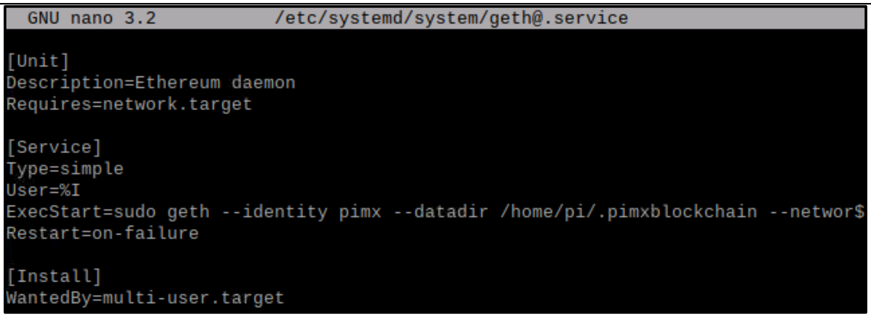
```
$ geth removedb --datadir=./pimxblockchain
```

After initialising the block, the blockchain can be run. Identity parameter sets the identity name for the current node, rpc is set to enable HTTP API to access at the desired port, rpccorsdomain allows Cross-Origin Resource Sharing (CORS) which is external RPC connections, datadir is to navigate to the directory of the blockchain, networkid has to be same as the chainId in genesis.json file:

```
$ geth --identity pimx --rpc --rpcport 8545 --rpccorsdomain "*" --datadir=./pimxblockchain --port 30303 --nodiscover --rpcapi "db,eth,net,web3" --networkid 888 console
```

To allow this code to run when Raspberry Pi starts up, the code can be configured as a service:

```
$ sudo nano /etc/systemd/system/geth@.service
```



```
  GNU nano 3.2              /etc/systemd/system/geth@.service

[Unit]
Description=Ethereum daemon
Requires=network.target

[Service]
Type=simple
User=%I
ExecStart=sudo geth --identity pimx --datadir /home/pi/.pimxblockchain --networ$
Restart=on-failure

[Install]
WantedBy=multi-user.target
```

Figure 5.14 /etc/systemd/system/geth@.service file

Enable this service when user starts up as user pi:

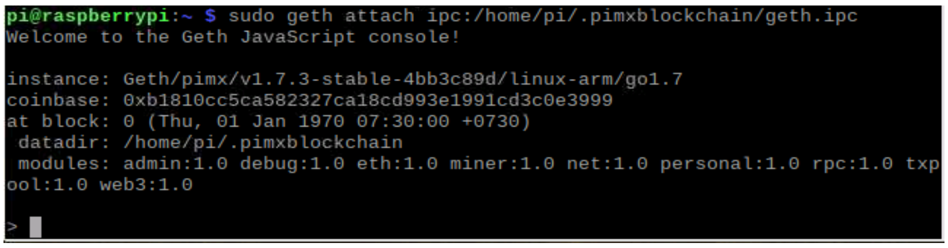
```
$ systemctl enable geth@pi.service
$ systemctl start geth@pi.service
```

To check whether the service is running after a reboot:

```
$ systemctl list-units --type=service
```

To attach to the running geth service for the blockchain to perform JavaScript code:

```
$ sudo geth attach ipc:pimxblockchain/geth.ipc
```

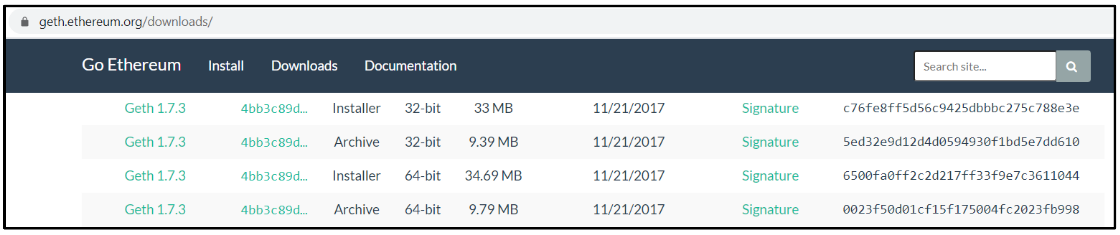


```
pi@raspberrypi:~ $ sudo geth attach ipc:/home/pi/.pimxblockchain/geth.ipc
Welcome to the Geth JavaScript console!

instance: Geth/pimx/v1.7.3-stable-4bb3c89d/linux-arm/go1.7
coinbase: 0xb1810cc5ca582327ca18cd993e1991cd3c0e3999
at block: 0 (Thu, 01 Jan 1970 07:30:00 +0730)
 datadir: /home/pi/.pimxblockchain
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txp
ool:1.0 web3:1.0

> 
```

Figure 5.15 Geth JavaScript console

5.4 Installing and Running Ethereum on End Device (Windows 10)

Navigate to https://geth.ethereum.org/downloads/ and search for the same version of Geth installed in Raspberry Pi (Geth 1.7.3). After downloading the installer, click on the exe file to complete the installation for Geth.

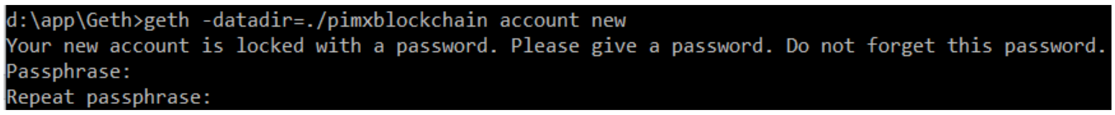


Geth 1.7.3	4bb3c89d...	Installer	32-bit	33 MB	11/21/2017	Signature	c76fe8ff5d56c9425dbbbc275c788e3e
Geth 1.7.3	4bb3c89d...	Archive	32-bit	9.39 MB	11/21/2017	Signature	5ed32e9d12d4d0594930f1bd5e7dd610
Geth 1.7.3	4bb3c89d...	Installer	64-bit	34.69 MB	11/21/2017	Signature	6500fa0ff2c2d217ff33f9e7c3611044
Geth 1.7.3	4bb3c89d...	Archive	64-bit	9.79 MB	11/21/2017	Signature	0023f50d01cf15f175004fc2023fb998

Figure 5.16 Go Ethereum official download page

Using command prompt in administrator mode, navigate to the location that Geth was installed In Windows version, the steps to download Go and make geth are skipped, compared to Raspberry Pi version. Directly, we can start to create a blockchain using Ethereum, which is the same method as in Raspberry Pi:

```
> geth –datadir=./pimxblockchain account new
```

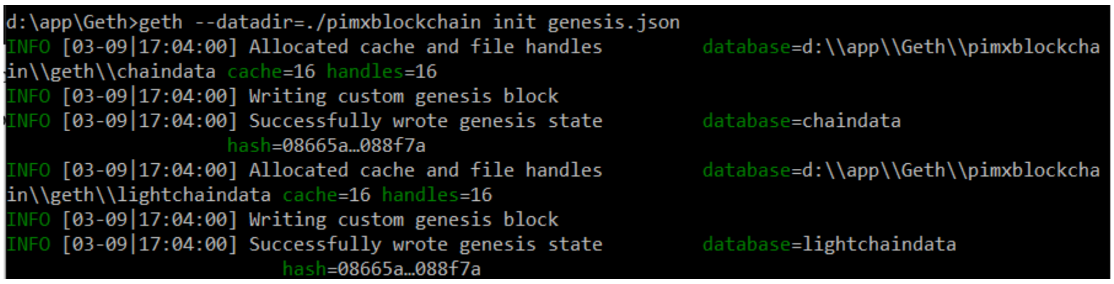


```
d:\app\Geth>geth -datadir=./pimxblockchain account new
Your new account is locked with a password. Please give a password. Do not forget this password.
Passphrase:
Repeat passphrase:
```

Figure 5.17 Creating a new blockchain

A genesis.json file is created using notepad in the same directory, then we use the same commands as in Raspberry Pi to initialise the blockchain using the genesis block file, then we run the blockchain using the command below:

```
> geth --datadir=./pimxblockchain init genesis.json
> geth --identity pimx --rpc --rpcport 8545 --rpccorsdomain "*" --datadir=./pimxblockchain --port 30303 --nodiscover --rpcapi "db,eth,net,web3" --networkid 888 console
```



```
d:\app\Geth>geth --datadir=./pimxblockchain init genesis.json
INFO [03-09|17:04:00] Allocated cache and file handles         database=d:\\app\\Geth\\pimxblockcha
in\\geth\\chaindata cache=16 handles=16
INFO [03-09|17:04:00] Writing custom genesis block
INFO [03-09|17:04:00] Successfully wrote genesis state         database=chaindata
                    hash=08665a…088f7a
INFO [03-09|17:04:00] Allocated cache and file handles         database=d:\\app\\Geth\\pimxblockcha
in\\geth\\lightchaindata cache=16 handles=16
INFO [03-09|17:04:00] Writing custom genesis block
INFO [03-09|17:04:00] Successfully wrote genesis state         database=lightchaindata
                         hash=08665a…088f7a
```

Figure 5.18 Initialising the genesis block (genesis.json)

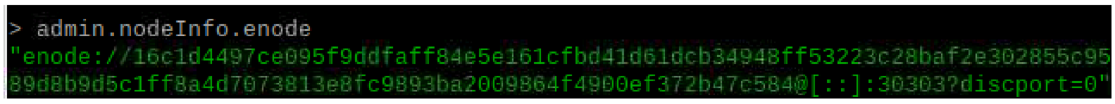
5.5 Adding End Device as the Blockchain Peer for Second and Third Raspberry Pi

After configuring and installing Ethereum in the selected devices, now we will start to build the connectivity between the end device and the wireless AP by using the function provided by Geth. The table below will show the details of each device:

Device	**First Raspberry Pi**	**Second Raspberry Pi**	**Third Raspberry Pi**	**End Device (Windows 10)**
Model	Raspberry Pi 3 Model B	Raspberry Pi 3 Model B+	Raspberry Pi 3 Model B+	Acer Aspire F15
Operating System	Raspberry Pi OS	Raspberry Pi OS	Raspberry Pi OS	Windows 10
Blockchain Address	{02b40215dee8a689d1f14b8002225eb8b31fd952}	{cc57afd075598687e44470d6081956e54b69535f}	{b0a0ff90b6b56cfb02f9af7ae9d811bbbe374409}	{ce0ecd9646574847abf2888801cac76a0812823a}
Password for address	mx	mx	mx	mx

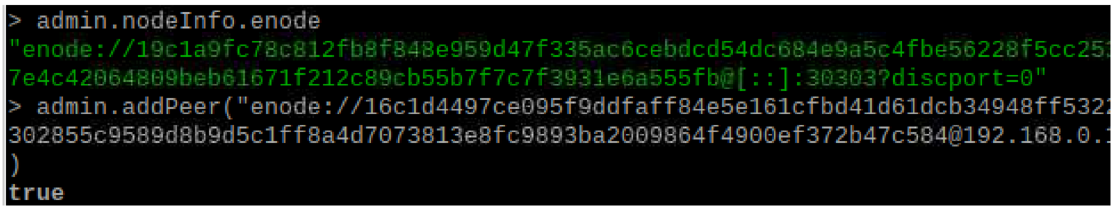
Table 5.1 Detailed Blockchain Information of Each Device

Now, we will link second Raspberry Pi with third Raspberry Pi using the addPeer method provided by Geth. The enode information can be discovered using the nodeInfo.enode method. The IP address of the Raspberry Pi will be using the IP that they connect to the wireless modem router. Now we will get the second Raspberry Pi node info and use the add peer function at the third Raspberry Pi side:



```
> admin.nodeInfo.enode
"enode://16c1d4497ce095f9ddfaff84e5e161cfbd41d61dcb34948ff53223c28baf2e302855c95
89d8b9d5c1ff8a4d7073813e8fc9893ba2009864f4900ef372b47c584@[::]:30303?discport=0"
```

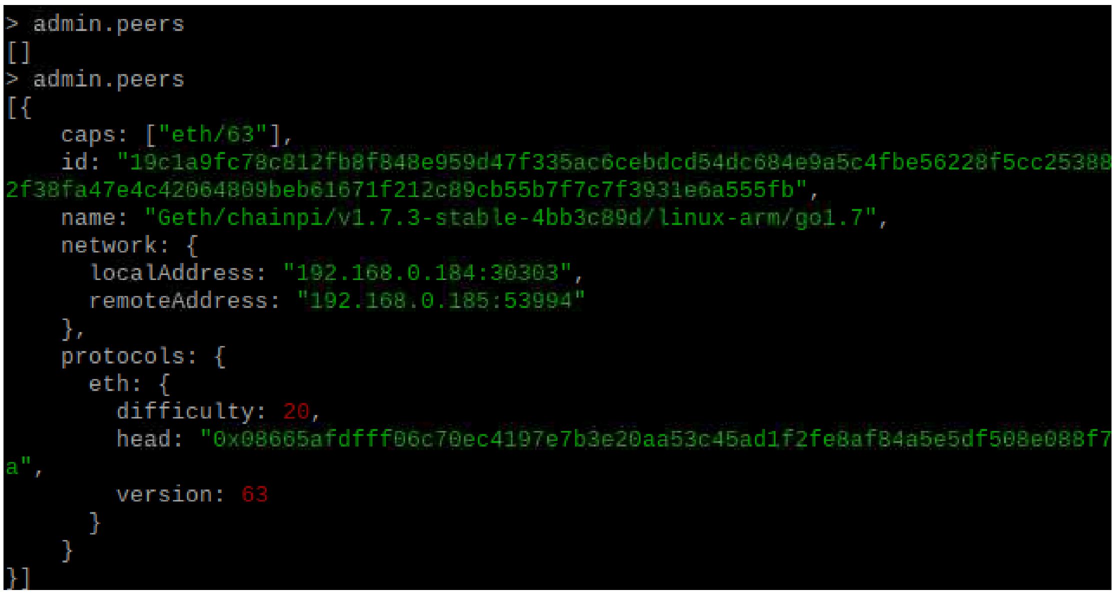
Figure 5.19 Second Raspberry Pi Geth Console (node info)



```
> admin.nodeInfo.enode
"enode://19c1a9fc78c812fb8f848e959d47f335ac6cebdcd54dc684e9a5c4fbe56228f5cc25
7e4c42064809beb61671f212c89cb55b7f7c7f3931e6a555fb@[::]:30303?discport=0"
> admin.addPeer("enode://16c1d4497ce095f9ddfaff84e5e161cfbd41d61dcb34948ff532
302855c9589d8b9d5c1ff8a4d7073813e8fc9893ba2009864f4900ef372b47c584@192.168.0.
)
true
```

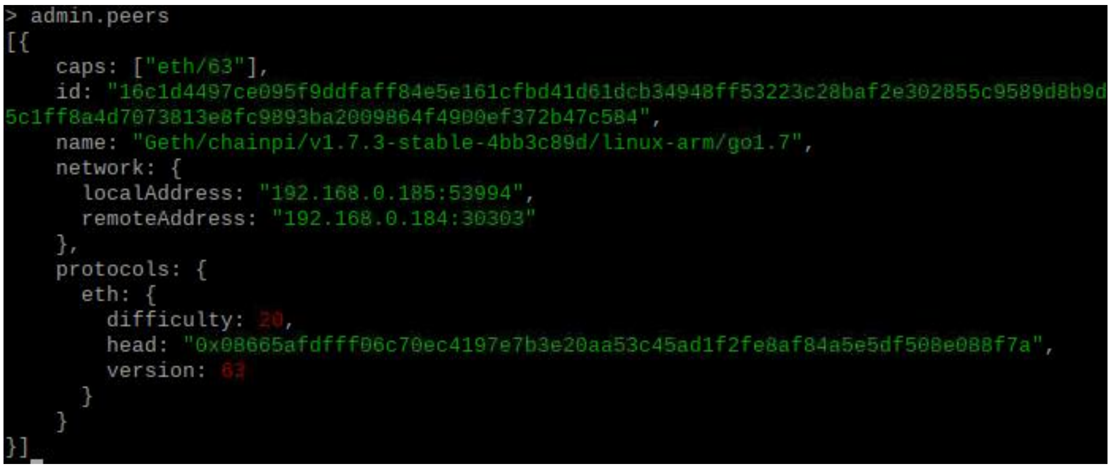
Figure 5.20 Third Raspberry Pi Geth Console (node info and add peer)

After adding the peer at only one side, at both sides, we can see that admin.peers is not empty anymore. They are now linked together as a peer. To briefly go through the information in the diagram below, the id is the node info of the end device, the name is the geth version that the end device is using. The local address is the second Raspberry Pi IP address, while the remote address is the third Raspberry Pi address. The head under protocols is the genesis block address:



```
> admin.peers
[]
> admin.peers
[{
    caps: ["eth/63"],
    id: "19c1a9fc78c812fb8f848e959d47f335ac6cebdcd54dc684e9a5c4fbe56228f5cc25388
2f38fa47e4c42064809beb61671f212c89cb55b7f7c7f3931e6a555fb",
    name: "Geth/chainpi/v1.7.3-stable-4bb3c89d/linux-arm/go1.7",
    network: {
      localAddress: "192.168.0.184:30303",
      remoteAddress: "192.168.0.185:53994"
    },
    protocols: {
      eth: {
        difficulty: 20,
        head: "0x08665afdfff06c70ec4197e7b3e20aa53c45ad1f2fe8af84a5e5df508e088f7
a",
        version: 63
      }
    }
}]
```

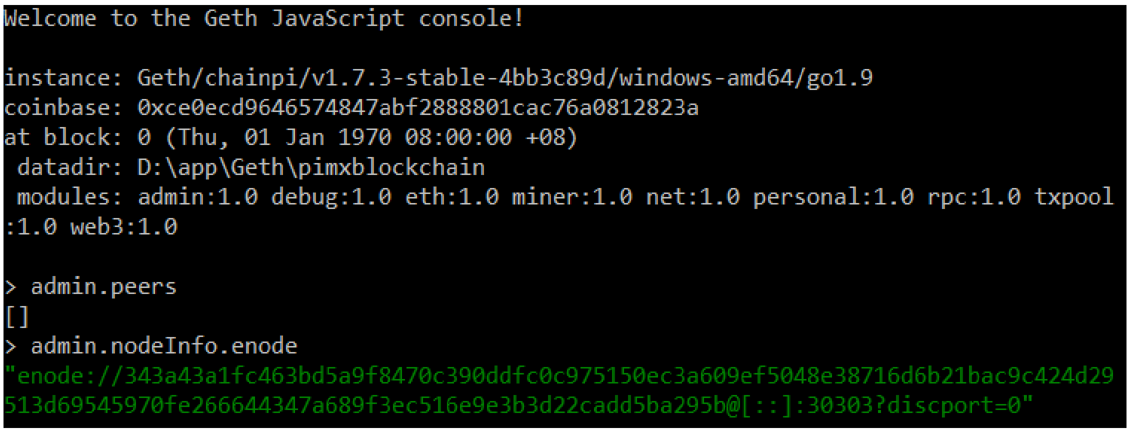
Figure 5.21 Second Raspberry Pi Geth Console (peer list)



```
> admin.peers
[{
    caps: ["eth/63"],
    id: "16c1d4497ce095f9ddfaff84e5e161cfbd41d61dcb34948ff53223c28baf2e302855c9589d8b9d
5c1ff8a4d7073813e8fc9893ba2009864f4900ef372b47c584",
    name: "Geth/chainpi/v1.7.3-stable-4bb3c89d/linux-arm/go1.7",
    network: {
      localAddress: "192.168.0.185:53994",
      remoteAddress: "192.168.0.184:30303"
    },
    protocols: {
      eth: {
        difficulty: 20,
        head: "0x08665afdfff06c70ec4197e7b3e20aa53c45ad1f2fe8af84a5e5df508e088f7a",
        version: 63
      }
    }
}]
```

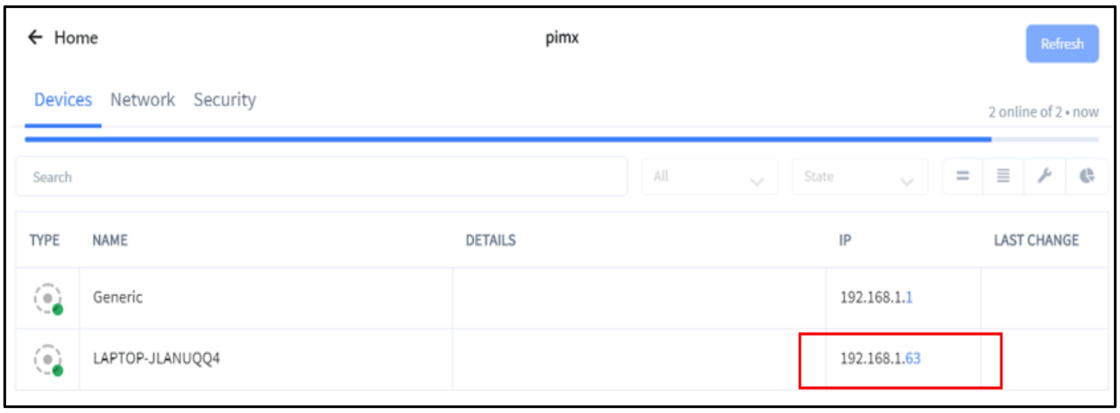
Figure 5.22 Third Raspberry Pi Geth Console (peer list)

Now, we will connect the end device into the blockchain using the same method. The enode information can be discovered using the nodeInfo.enode method. The IP address of the end device will be using the IP that the Raspberry Pi wireless AP provides upon connecting to it. These two information will be needed for the add peer function. By adding the end device as the peer of the wireless AP blockchain, it will work like a list of trusted devices in the wireless AP. Initially, the peer list at end device will be empty since the third Raspberry Pi has not added the end device as its peer yet. The end device must be connected to one of the Raspberry Pi to be able to add as a peer, since the IP address is obtained by connecting to one of the Raspberry Pi:



```
Welcome to the Geth JavaScript console!

instance: Geth/chainpi/v1.7.3-stable-4bb3c89d/windows-amd64/go1.9
coinbase: 0xce0ecd9646574847abf2888801cac76a0812823a
at block: 0 (Thu, 01 Jan 1970 08:00:00 +08)
 datadir: D:\app\Geth\pimxblockchain
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool
:1.0 web3:1.0

> admin.peers
[]
> admin.nodeInfo.enode
"enode://343a43a1fc463bd5a9f8470c390ddfc0c975150ec3a609ef5048e38716d6b21bac9c424d29
513d69545970fe266644347a689f3ec516e9e3b3d22cadd5ba295b@[::]:30303?discport=0"
```

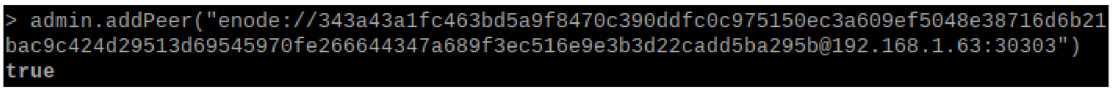
Figure 5.23 End Device Geth Console (empty peers and node info)



TYPE	NAME	DETAILS	IP	LAST CHANGE
	Generic		192.168.1.1	
	LAPTOP-JLANUQQ4		192.168.1.63	

Figure 5.24 End Device IP Address (Fing)

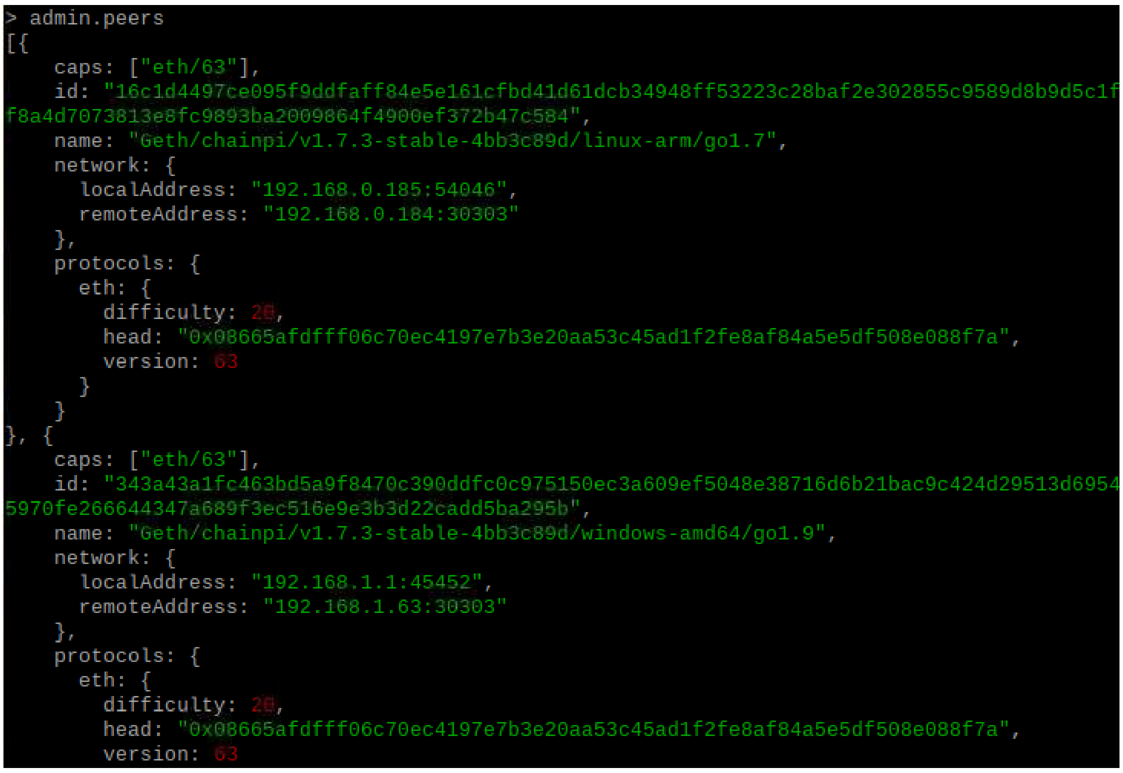
Now, we add the end device using the console of third Raspberry Pi. We use the addPeer function to add the end device by inserting the enode and IP of the device:



```
> admin.addPeer("enode://343a43a1fc463bd5a9f8470c390ddfc0c975150ec3a609ef5048e38716d6b21
bac9c424d29513d69545970fe266644347a689f3ec516e9e3b3d22cadd5ba295b@192.168.1.63:30303")
true
```

Figure 5.25 Third Raspberry Pi Geth Console (add peer)

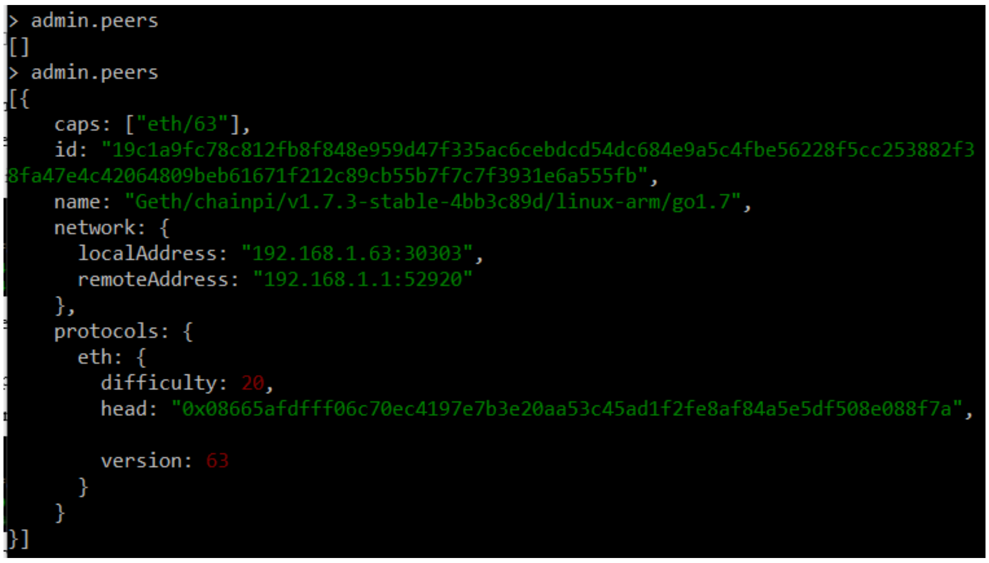
By checking the admin peers function, it is shown that there are end device peer information now. The third Raspberry Pi is now connected to the end device.



```
> admin.peers
[{
    caps: ["eth/63"],
    id: "16c1d4497ce095f9ddfaff84e5e161cfbd41d61dcb34948ff53223c28baf2e302855c9589d8b9d5c1f
f8a4d7073813e8fc9893ba2009864f4900ef372b47c584",
    name: "Geth/chainpi/v1.7.3-stable-4bb3c89d/linux-arm/go1.7",
    network: {
      localAddress: "192.168.0.185:54046",
      remoteAddress: "192.168.0.184:30303"
    },
    protocols: {
      eth: {
        difficulty: 20,
        head: "0x08665afdfff06c70ec4197e7b3e20aa53c45ad1f2fe8af84a5e5df508e088f7a",
        version: 63
      }
    }
}, {
    caps: ["eth/63"],
    id: "343a43a1fc463bd5a9f8470c390ddfc0c975150ec3a609ef5048e38716d6b21bac9c424d29513d6954
5970fe266644347a689f3ec516e9e3b3d22cadd5ba295b",
    name: "Geth/chainpi/v1.7.3-stable-4bb3c89d/windows-amd64/go1.9",
    network: {
      localAddress: "192.168.1.1:45452",
      remoteAddress: "192.168.1.63:30303"
    },
    protocols: {
      eth: {
        difficulty: 20,
        head: "0x08665afdfff06c70ec4197e7b3e20aa53c45ad1f2fe8af84a5e5df508e088f7a",
        version: 63
```

Figure 5.26 Third Raspberry Pi Geth Console (listed peers)

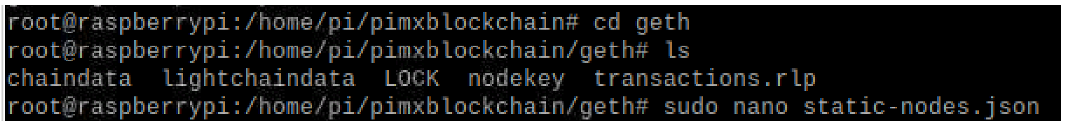
At the end device, initially we check that there are also no peers connected to the blockchain using admin.peers. After added by third Raspberry Pi, it is shown that the peer is connected.



```
> admin.peers
[]
> admin.peers
[{
    caps: ["eth/63"],
    id: "19c1a9fc78c812fb8f848e959d47f335ac6cebdcd54dc684e9a5c4fbe56228f5cc253882f3
8fa47e4c42064809beb61671f212c89cb55b7f7c7f3931e6a555fb",
    name: "Geth/chainpi/v1.7.3-stable-4bb3c89d/linux-arm/go1.7",
    network: {
      localAddress: "192.168.1.63:30303",
      remoteAddress: "192.168.1.1:52920"
    },
    protocols: {
      eth: {
        difficulty: 20,
        head: "0x08665afdfff06c70ec4197e7b3e20aa53c45ad1f2fe8af84a5e5df508e088f7a",

        version: 63
      }
    }
}]
```

Figure 5.27 End Device Geth Console (listed peers)

The addPeer function is volatile and will be needed to perform again on each Raspberry Pi boot. Therefore, to solve this problem, we can configure permanent static nodes by writing the following text into static-nodes.json:



```
root@raspberrypi:/home/pi/pimxblockchain# cd geth
root@raspberrypi:/home/pi/pimxblockchain/geth# ls
chaindata  lightchaindata  LOCK  nodekey  transactions.rlp
root@raspberrypi:/home/pi/pimxblockchain/geth# sudo nano static-nodes.json
```

Figure 5.28 Third Raspberry Pi Geth Console (move directory to add static-nodes.json)

The following is for Third Raspberry Pi file content. The first enode information is the second Raspberry Pi enode information, and the second enode information is for the end device, which is similar to the add peer function:

```
[
"enode://16c1d4497ce095f9ddfaff84e5e161cfbd41d61dcb34948ff53223c28baf2e302
855c9589d8b9d5c1ff8a4d7073813e8fc9893ba2009864f4900ef372b47c584@192.168
.0.184:30303",
"enode://343a43a1fc463bd5a9f8470c390ddfc0c975150ec3a609ef5048e38716d6b21b
ac9c424d29513d69545970fe266644347a689f3ec516e9e3b3d22cadd5ba295b@192.1
68.1.63:30303"
]
```

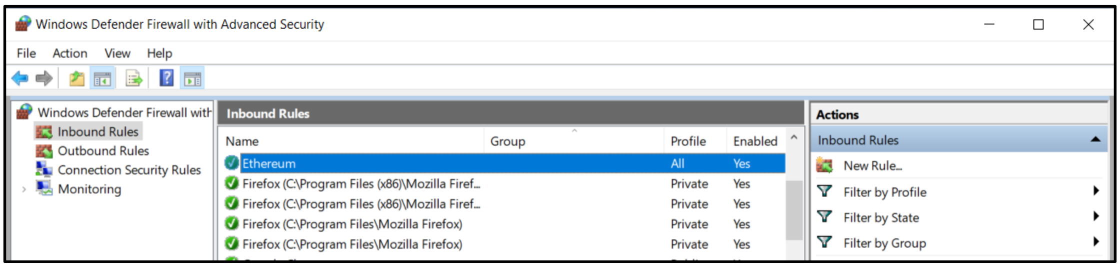
After adding the following information into the files, we can eliminate the use of addPeer function on runtime, instead, we can connect to the peers on each geth run. Until now, we have successfully added the end device as the list of trusted device for the third Raspberry Pi. Now, we will also add the end device as a trusted peer in second Raspberry Pi, similar to the previous step:

```
[
"enode://19c1a9fc78c812fb8f848e959d47f335ac6cebdcd54dc684e9a5c4fbe56228f5c
c253882f38fa47e4c42064809beb61671f212c89cb55b7f7c7f3931e6a555fb@192.168.
0.185:30303",
"enode://343a43a1fc463bd5a9f8470c390ddfc0c975150ec3a609ef5048e38716d6b21b
ac9c424d29513d69545970fe266644347a689f3ec516e9e3b3d22cadd5ba295b@192.1
68.1.63:30303"
]
```

Therefore, after all the steps above, the end device is now able to connect to the second and third Raspberry Pi because they add the end device under the trusted list of peer. The end device will be able to successfully undergo the blockchain authentication process.

5.6 Debugging Issues for Geth Windows Connection

If there are any issues for the connection on Windows Geth, the firewall inbound rule and outbound rule for TCP port 30303 will need to be configured:



Inbound Rules

Name	Group	Profile	Enabled
Ethereum		All	Yes
Firefox (C:\Program Files (x86)\Mozilla Firef...		Private	Yes
Firefox (C:\Program Files (x86)\Mozilla Firef...		Private	Yes
Firefox (C:\Program Files\Mozilla Firefox)		Private	Yes
Firefox (C:\Program Files\Mozilla Firefox)		Private	Yes

Figure 5.29 Windows Defender Firewall with Advanced Security

After configuring successfully, we can check that now the firewall status is allowed for geth.exe on port 30303:

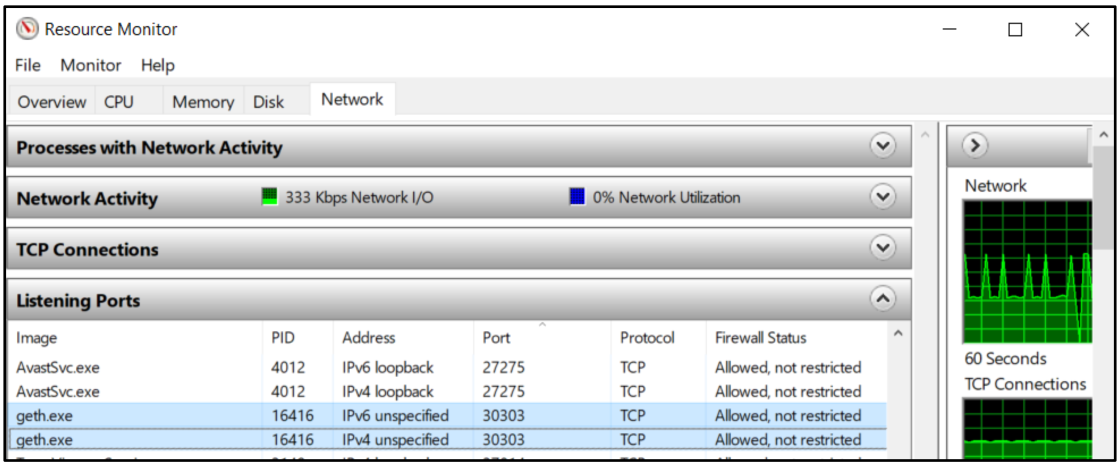


Image	PID	Address	Port	Protocol	Firewall Status
AvastSvc.exe	4012	IPv6 loopback	27275	TCP	Allowed, not restricted
AvastSvc.exe	4012	IPv4 loopback	27275	TCP	Allowed, not restricted
geth.exe	16416	IPv6 unspecified	30303	TCP	Allowed, not restricted
geth.exe	16416	IPv4 unspecified	30303	TCP	Allowed, not restricted

Figure 5.30 Resource Monitor

5.7 Set up a Captive Portal on the Wireless Access Point for Authentication

When a user connects to the Raspberry Pi Wireless AP, a captive portal will be shown to the user before granting the user internet connection. The captive portal will check whether the user is inside the Ethereum private blockchain, then it will grant internet access to the user. For this captive portal, we will use the Nodogsplash project which is easy to set up and configured. First, we will need to install the package that is needed to compile the nodogsplash git code:

```
$ sudo apt install libmicrohttpd-dev
```

Then, we will now clone the Nodogsplash project from github:

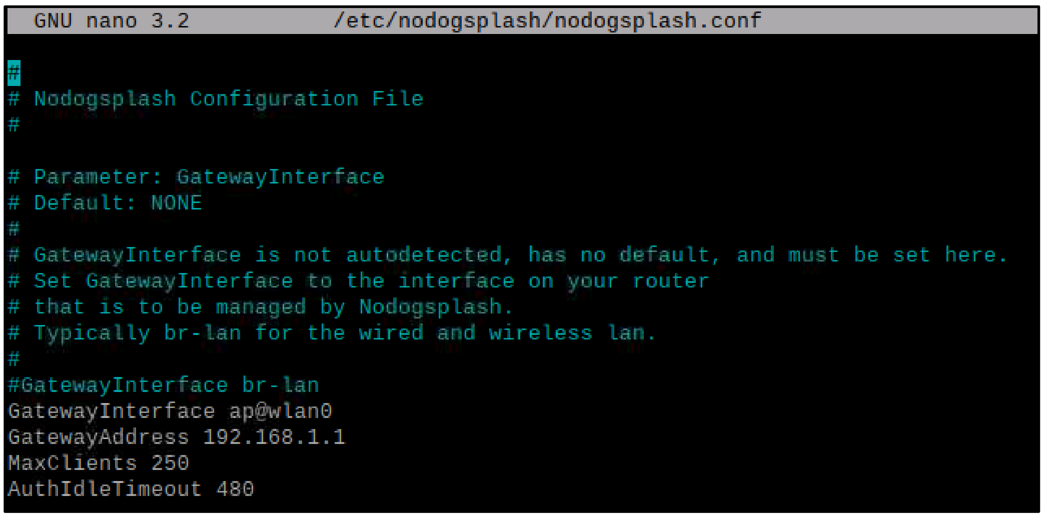
```
$ git clone https://github.com/nodogsplash/nodogsplash.git
```

Next, we can start to compile and install the Nodogsplash software:

```
$ cd nodogsplash
$ make
$ sudo make install
```

After installing it, we can now modify the configuration file, which is easy to understand. We will add some information into the configuration file. GatewayInterface is the interface that the nodogsplash software should be shown, GatewayAddress is the address of the access point:

```
$ sudo nano /etc/nodogsplash/nodogsplash.conf
```



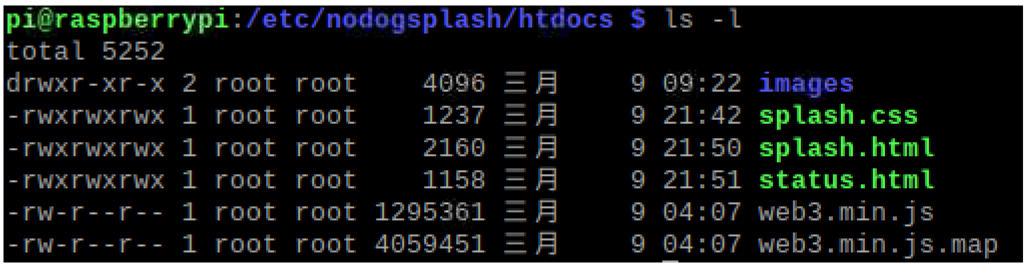
```
  GNU nano 3.2             /etc/nodogsplash/nodogsplash.conf

#
# Nodogsplash Configuration File
#

# Parameter: GatewayInterface
# Default: NONE
#
# GatewayInterface is not autodetected, has no default, and must be set here.
# Set GatewayInterface to the interface on your router
# that is to be managed by Nodogsplash.
# Typically br-lan for the wired and wireless lan.
#
#GatewayInterface br-lan
GatewayInterface ap@wlan0
GatewayAddress 192.168.1.1
MaxClients 250
AuthIdleTimeout 480
```

Figure 5.31 /etc/nodogsplash/nodogsplash.conf

To enable simple modification to the html files, instead of using nano, we will use Geany which has a simpler user interface. First, we will need to change the permission of the file:

```
$ sudo chmod -R 777 /etc/nodogsplash/htdocs
```

Now, we can modify the html to check whether user is in the blockchain same as the access point, then allow the user to access to the internet. The files are inside htdocs folder, specifically:



```
pi@raspberrypi:/etc/nodogsplash/htdocs $ ls -l
total 5252
drwxr-xr-x 2 root root    4096 三月  9 09:22 images
-rwxrwxrwx 1 root root    1237 三月  9 21:42 splash.css
-rwxrwxrwx 1 root root    2160 三月  9 21:50 splash.html
-rwxrwxrwx 1 root root    1158 三月  9 21:51 status.html
-rw-r--r-- 1 root root 1295361 三月  9 04:07 web3.min.js
-rw-r--r-- 1 root root 4059451 三月  9 04:07 web3.min.js.map
```

Figure 5.32 Files inside /etc/nodogsplash/htdocs folder

Download the required web3.js files using npm. Web3.js is an Ethereum JavaScript API, which is a collection of libraries to allow user to interact with a local or remote Ethereum node with HTTP or IPC connection:

```
$ sudo apt-get install nodejs
$ sudo apt-get install npm
$ npm install web3
```

Move the required external web3.js files from npm to the folder. The web3.min.js file will be inside /home/pi/node_modules/web3/dist:

```
$ sudo cp web3.min.js web3.min.js.map /etc/nodogsplash/htdocs
```

To run the software, just use a simple command:

```
$ sudo nodogsplash
```

Now, we use our end device which is connected to the blockchain, to connect to the wireless access point. A captive portal will be shown to the user.

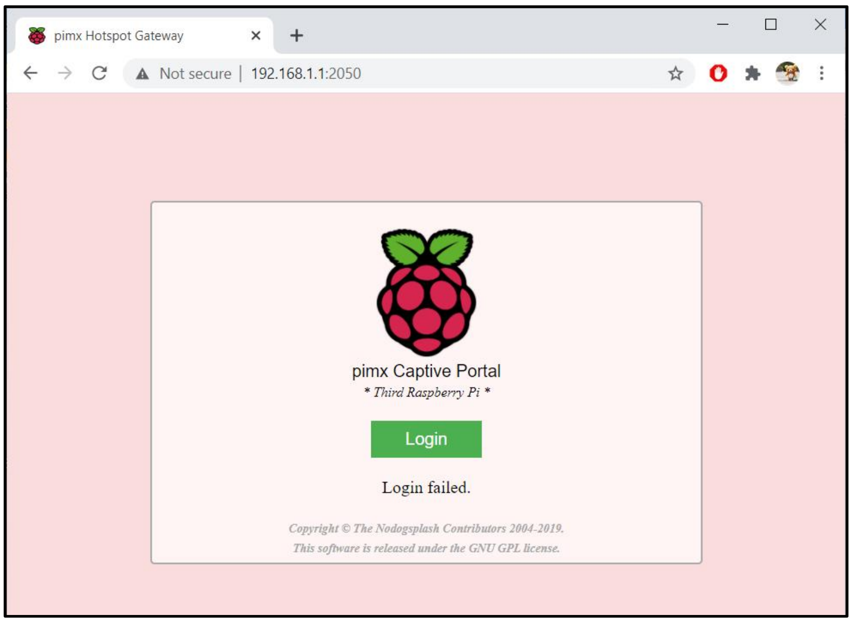
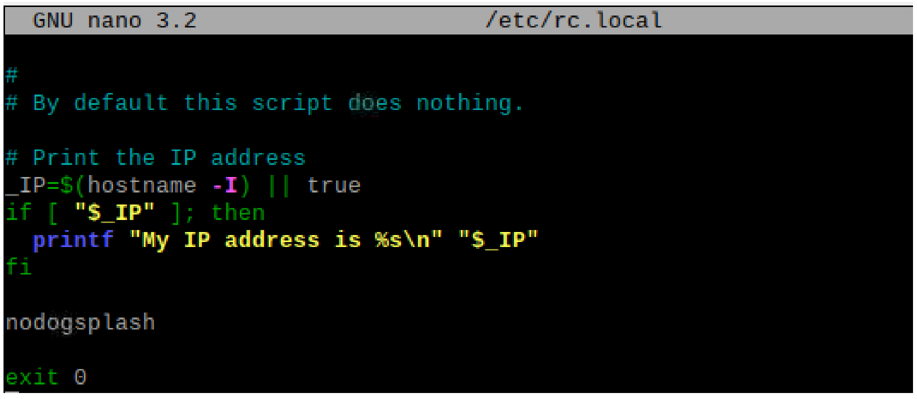

pimx Hotspot Gateway
Not secure | 192.168.1.1:2050
pimx Captive Portal
* Third Raspberry Pi *
Login
Login failed.
Copyright © The Nodogsplash Contributors 2004-2019.
This software is released under the GNU GPL license.

Figure 5.33 pimx Captive Portal homepage (initial look)

To allow nodogsplash to startup on launch, we can modify the rc.local file:

```
$ sudo nano /etc/rc.local
```

We add the command "nodogsplash" before the script exits:



```
  GNU nano 3.2                    /etc/rc.local

#
# By default this script does nothing.

# Print the IP address
_IP=$(hostname -I) || true
if [ "$_IP" ]; then
  printf "My IP address is %s\n" "$_IP"
fi

nodogsplash

exit 0
```

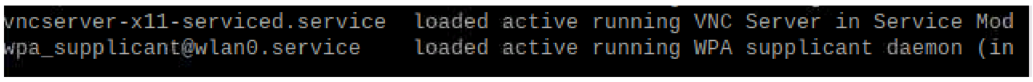
Figure 5.34 /etc/rc.local

CHAPTER 6: RESULTS AND DISCUSSIONS

6.1 Raspberry Pi Connectivity Check

To prove that Raspberry Pi is running as a wireless access point using the wlan0 service, the command below is to check what service is running in systemctl. Systemtcl is used to control the systemd system and service which runs when the Raspberry Pi boots up (wpa_supplicant@wlan0.service shows the service is active and running):

```
$ systemctl list-units --type=service
```



```
vncserver-x11-serviced.service   loaded active running VNC Server in Service Mod
wpa_supplicant@wlan0.service     loaded active running WPA supplicant daemon (in
```

Figure 6.1 “systemctl list-units --type=service” list

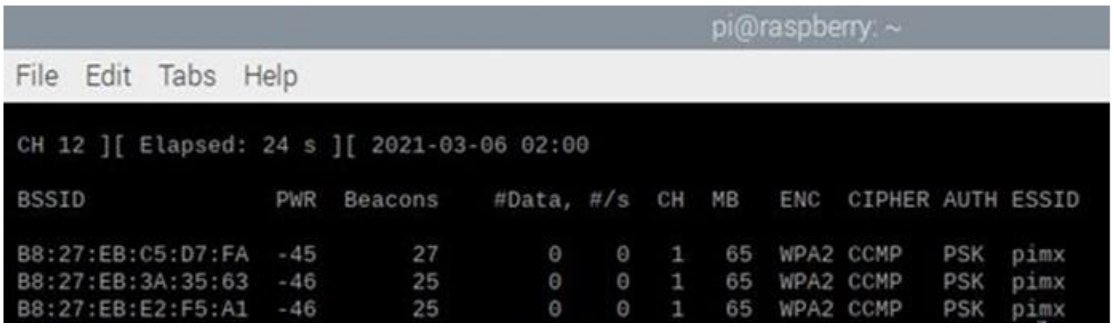
Now, we use a virtualbox machine to monitor the connection of the three Raspberry Pi wireless AP. We connect a wireless network adapter to the machine and the below commands are used to configure the wireless network adapter to monitor mode:

```
$ ifconfig wlan0 down
$ iwconfig wlan0 mode monitor
$ ifconfig wlan0 up
$ iwconfig
```

Then, we will run airodump which is a network monitoring software to check the three pi’s MAC address to differentiate them as they have the same SSID and not easily monitored without identifying their MAC address:

```
$ airodump-ng wlan0
```

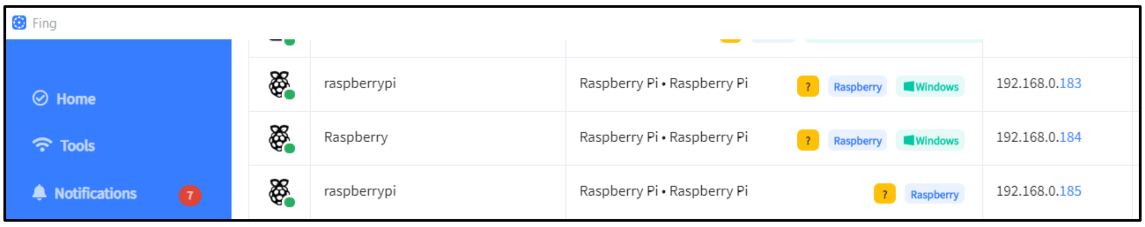
Below shows three pimx is near to the current wireless network adapter:



```
CH 12 ][ Elapsed: 24 s ][ 2021-03-06 02:00

BSSID              PWR  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

B8:27:EB:C5:D7:FA  -45       27        0    0   1   65  WPA2 CCMP   PSK  pimx
B8:27:EB:3A:35:63  -46       25        0    0   1   65  WPA2 CCMP   PSK  pimx
B8:27:EB:E2:F5:A1  -46       25        0    0   1   65  WPA2 CCMP   PSK  pimx
```

Figure 6.2 Virtualbox Machine Terminal (airodump-ng)

To prove that the Raspberry Pi is getting DHCP from wireless modem router to be able to access the Internet, Fing application for mobile device is used when connecting to the wireless modem router to check the IP addresses of the wireless AP. It is shown that all the three Raspberry Pi is connected to the wireless AP.



raspberrypi	Raspberry Pi • Raspberry Pi	? Raspberry Windows	192.168.0.183
Raspberry	Raspberry Pi • Raspberry Pi	? Raspberry Windows	192.168.0.184
raspberrypi	Raspberry Pi • Raspberry Pi	? Raspberry	192.168.0.185

Figure 6.3 Three Raspberry Pi obtaining DHCP from wireless modem router (Fing)

It is proven that end devices are able to discover the wireless APs and connect to them. This is shown using mobile device's built-in Wi-Fi function.

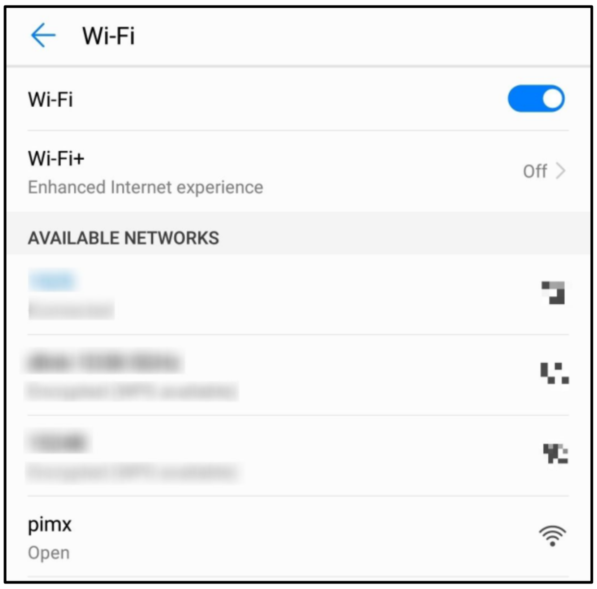
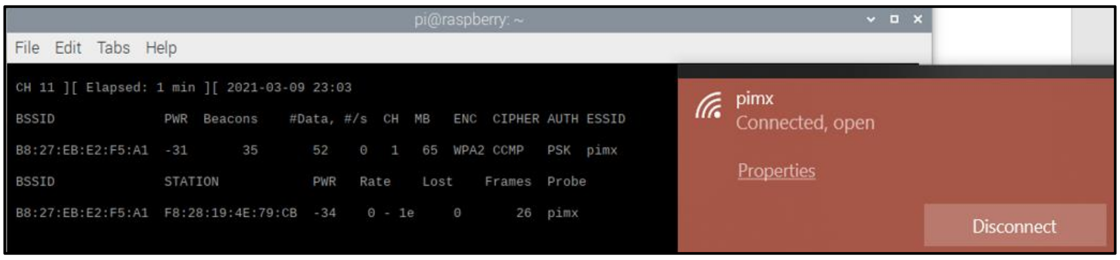


Figure 6.4 Android device able to discover the wireless AP

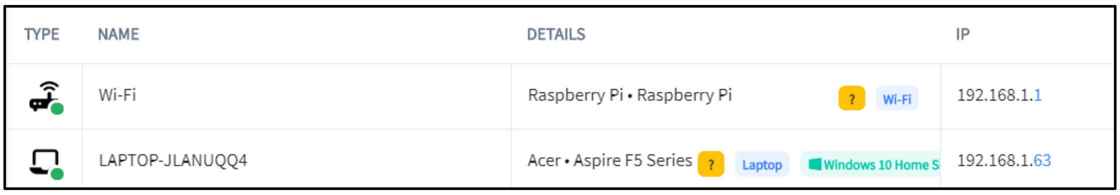
Windows 10 machine is also able to discover the wireless AP and connect to it. After being authorised in the captive portal, it shows that it has successfully connected to the wireless AP in the same blockchain. Based on the virtualbox machine, it is shown that the Windows 10 machine has been connected to the third Raspberry Pi:



```
CH 11 ][ Elapsed: 1 min ][ 2021-03-09 23:03

BSSID              PWR  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

B8:27:EB:E2:F5:A1  -31       35       52    0   1  65   WPA2 CCMP   PSK  pimx

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

B8:27:EB:E2:F5:A1  F8:28:19:4E:79:CB  -34    0 - 1e     0       26  pimx
```

Figure 6.5 Windows 10 machine able to discover the wireless AP

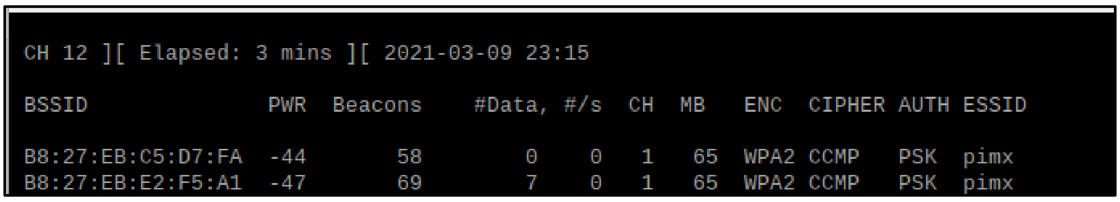
To find out the IP address of the wireless AP connected, Fing application is used. It is known that the IP of wireless AP (pimx) is 192.168.1.1, while the end device is given an IP address through DHCP:



TYPE	NAME	DETAILS	IP
	Wi-Fi	Raspberry Pi • Raspberry Pi ? Wi-Fi	192.168.1.1
	LAPTOP-JLANUQQ4	Acer • Aspire F5 Series ? Laptop Windows 10 Home S	192.168.1.63

Figure 6.6 IP address of wireless AP (Fing)

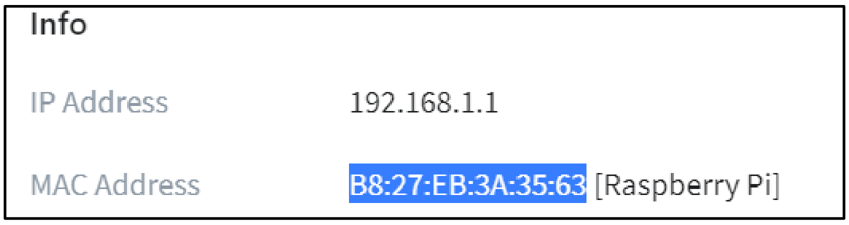
Now we move the second Raspberry Pi nearer to the host machine. It it shown that the second Raspberry Pi is now highest in the list, which means higher chance to get connected to:



```
CH 12 ][ Elapsed: 3 mins ][ 2021-03-09 23:15

BSSID              PWR  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

B8:27:EB:C5:D7:FA  -44       58        0    0   1  65   WPA2 CCMP   PSK  pimx
B8:27:EB:E2:F5:A1  -47       69        7    0   1  65   WPA2 CCMP   PSK  pimx
```

Figure 6.7 Second Raspberry Pi nearest to the host machine

Besides checking with airodump-ng, we can also check with Fing application, now when end device is connected to the wireless AP again, the MAC address is now belong to the second Raspberry Pi:



Info

IP Address	192.168.1.1
MAC Address	B8:27:EB:3A:35:63 [Raspberry Pi]

Figure 6.8 MAC address of wireless AP is second Raspberry Pi (Fing)

This shows that the nearer the device is with the wireless AP, the higher chance that the wireless AP will be chosen to be connected to.

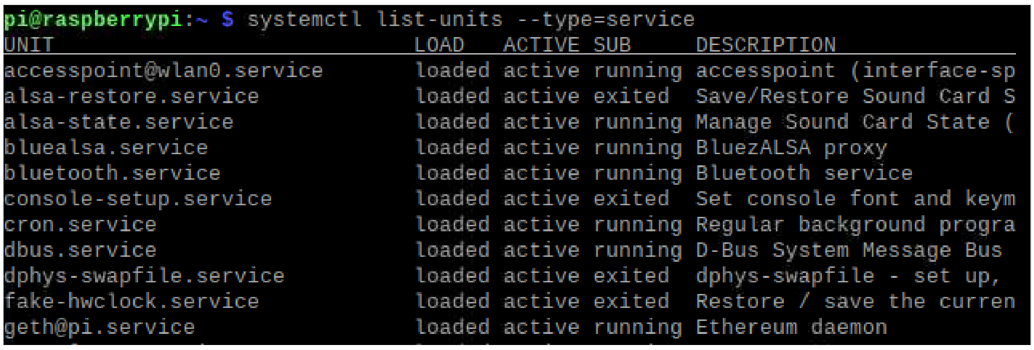
To conclude the findings, the table below has detailed network information of each of the Raspberry Pi.

Device	**First Raspberry Pi**	**Second Raspberry Pi**	**Third Raspberry Pi**
Static IP Address (wlan0)	192.168.0.183	192.168.0.184	192.168.0.185
Static IP Address (ap@wlan0)	192.168.1.1	192.168.1.1	192.168.1.1
MAC address	B8:27:EB:C5:D7:FA	B8:27:EB:3A:35:63	B8:27:EB:E2:F5:A1

Table 6.1 Network Information of Each Raspberry Pi

6.2 Ethereum Running on Raspberry Pi and End Device Windows 10

To prove that blockchain is running on boot, systemctl shows the system and service for geth@pi.service is running.

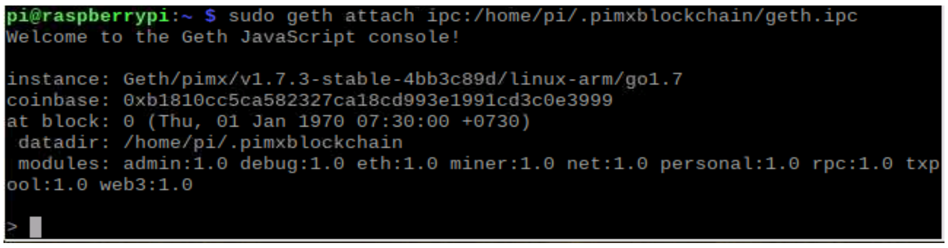


```
pi@raspberrypi:~ $ systemctl list-units --type=service
UNIT                            LOAD   ACTIVE SUB     DESCRIPTION
accesspoint@wlan0.service       loaded active running accesspoint (interface-sp
alsa-restore.service            loaded active exited  Save/Restore Sound Card S
alsa-state.service              loaded active running Manage Sound Card State (
bluealsa.service                loaded active running BluezALSA proxy
bluetooth.service               loaded active running Bluetooth service
console-setup.service           loaded active exited  Set console font and keym
cron.service                    loaded active running Regular background progra
dbus.service                    loaded active running D-Bus System Message Bus
dphys-swapfile.service          loaded active exited  dphys-swapfile - set up,
fake-hwclock.service            loaded active exited  Restore / save the curren
geth@pi.service                 loaded active running Ethereum daemon
```

Figure 6.9 “systemctl list-units –type=service” list

We can try to attach to the running geth service for the blockchain to perform JavaScript code. If it can successfully attach, means the blockchain is running successfully:

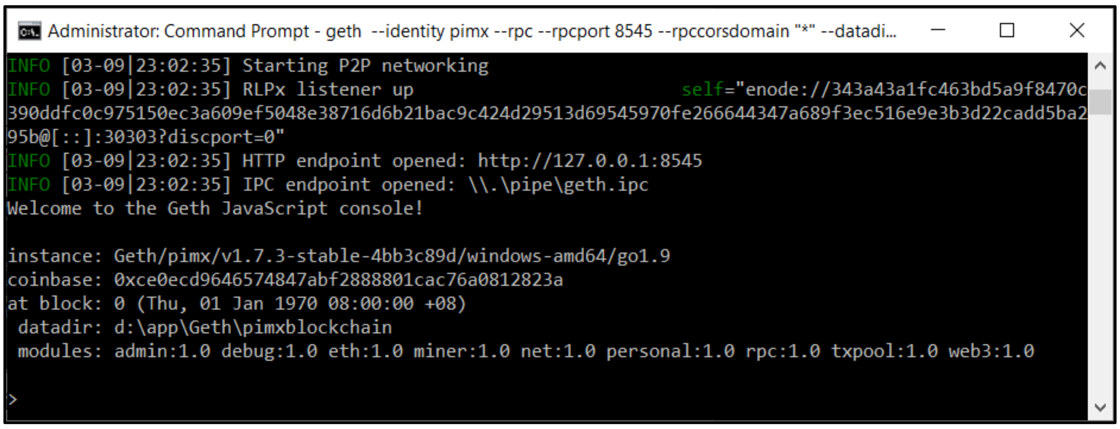
```
$ sudo geth attach ipc:pimxblockchain/geth.ipc
```



```
pi@raspberrypi:~ $ sudo geth attach ipc:/home/pi/.pimxblockchain/geth.ipc
Welcome to the Geth JavaScript console!

instance: Geth/pimx/v1.7.3-stable-4bb3c89d/linux-arm/go1.7
coinbase: 0xb1810cc5ca582327ca18cd993e1991cd3c0e3999
at block: 0 (Thu, 01 Jan 1970 07:30:00 +0730)
 datadir: /home/pi/.pimxblockchain
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txp
ool:1.0 web3:1.0

> 
```

Figure 6.10 Raspberry Pi Geth JavaScript console

In Windows 10, geth is also running successfully:



```
Administrator: Command Prompt - geth  --identity pimx --rpc --rpcport 8545 --rpccorsdomain "*" --datadi...
INFO [03-09|23:02:35] Starting P2P networking
INFO [03-09|23:02:35] RLPx listener up                          self="enode://343a43a1fc463bd5a9f8470c
390ddfc0c975150ec3a609ef5048e38716d6b21bac9c424d29513d69545970fe266644347a689f3ec516e9e3b3d22cadd5ba2
95b@[::]:30303?discport=0"
INFO [03-09|23:02:35] HTTP endpoint opened: http://127.0.0.1:8545
INFO [03-09|23:02:35] IPC endpoint opened: \\.\pipe\geth.ipc
Welcome to the Geth JavaScript console!

instance: Geth/pimx/v1.7.3-stable-4bb3c89d/windows-amd64/go1.9
coinbase: 0xce0ecd9646574847abf2888801cac76a0812823a
at block: 0 (Thu, 01 Jan 1970 08:00:00 +08)
 datadir: d:\app\Geth\pimxblockchain
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0 web3:1.0

>
```

Figure 6.11 Windows 10 Geth JavaScript console

6.3 Captive Portal Authentication Access

Now we connect our end device to the wireless AP. If it is the third Raspberry Pi, then authentication will be successful. If it is the second Raspberry Pi, then authentication will be unsuccessful as the end device is not a trusted device. The following demonstrates a successful process by connecting the end device (Windows 10) to the third Raspberry Pi as it is one of the peers for the third Raspberry Pi. Below shows the homepage of the pimx captive portal:

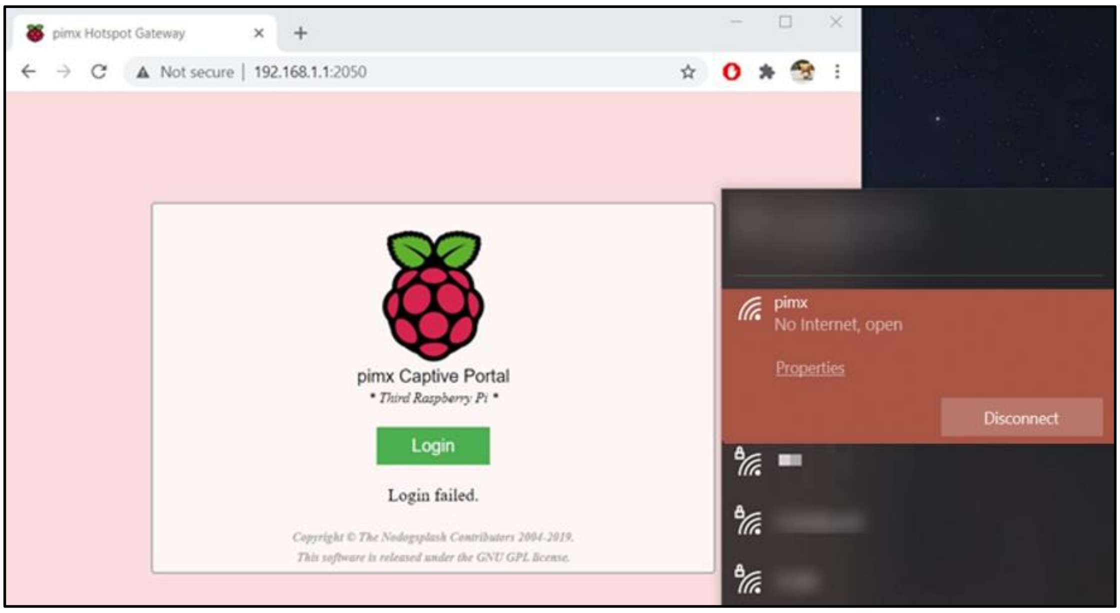
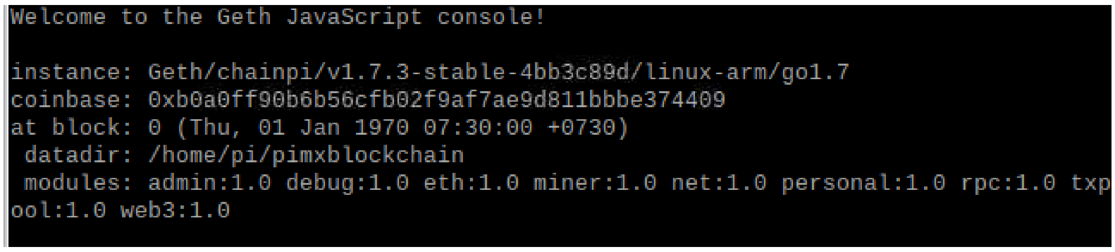

pimx Hotspot Gateway
Not secure | 192.168.1.1:2050
pimx Captive Portal
* Third Raspberry Pi *
Login
Login failed.
Copyright © The Nodogsplash Contributors 2004-2019.
This software is released under the GNU GPL license.
pimx
No Internet, open
Properties
Disconnect

Figure 6.12 Third Raspberry Pi pimx Captive Portal homepage (after connecting to wireless AP)

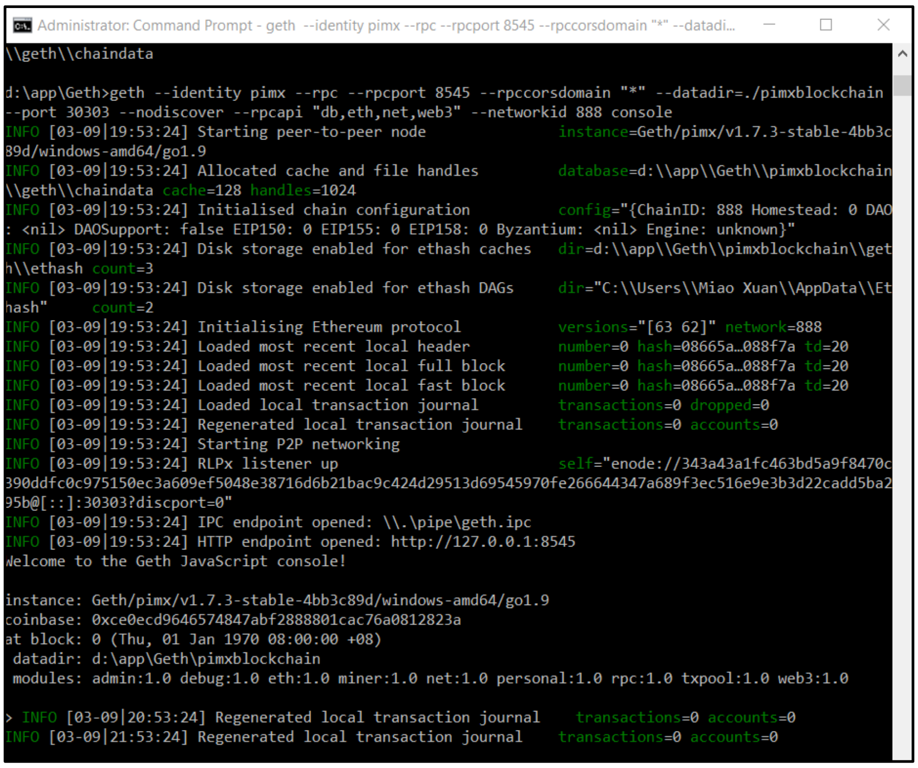
The third Raspberry Pi must be running Geth to enable end device to discover it as a peer:



```
Welcome to the Geth JavaScript console!

instance: Geth/chainpi/v1.7.3-stable-4bb3c89d/linux-arm/go1.7
coinbase: 0xb0a0ff90b6b56cfb02f9af7ae9d811bbbe374409
at block: 0 (Thu, 01 Jan 1970 07:30:00 +0730)
 datadir: /home/pi/pimxblockchain
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txp
ool:1.0 web3:1.0
```

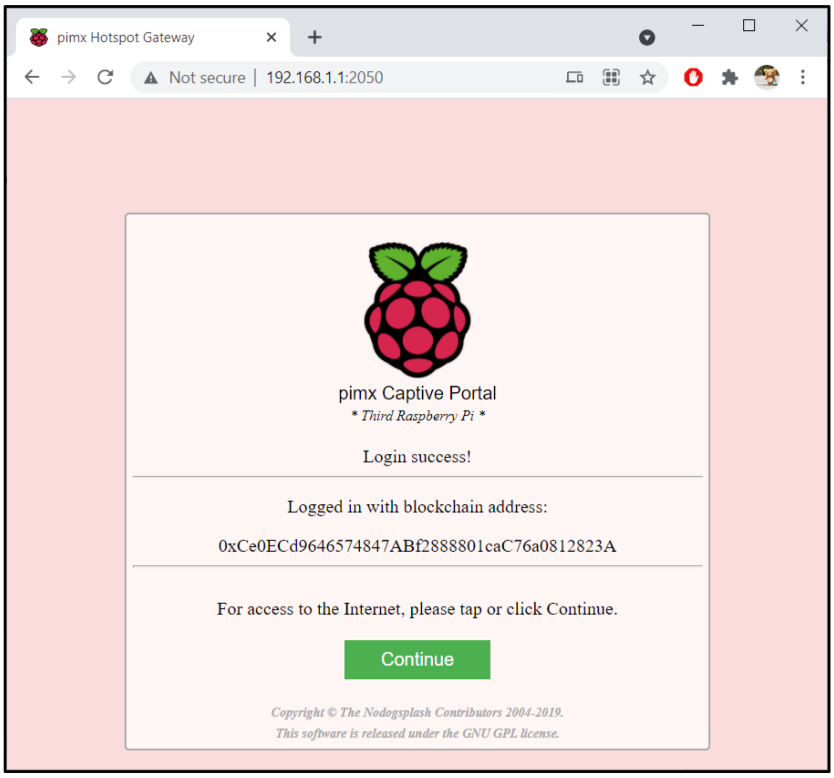
Figure 6.13 Third Raspberry Pi console (running Geth)

The end device must also be running Geth to connect to the third Raspberry Pi peer. The figure 6.14 below shows successful run of blockchain in the machine. The figure 6.15 below shows that the blockchain address is detected after clicking “Login” button:


Administrator: Command Prompt - geth --identity pimx --rpc --rpcport 8545 --rpccorsdomain "*" --datadi...

```
\\geth\\chaindata

d:\app\Geth>geth --identity pimx --rpc --rpcport 8545 --rpccorsdomain "*" --datadir=./pimxblockchain
--port 30303 --nodiscover --rpcapi "db,eth,net,web3" --networkid 888 console
INFO [03-09|19:53:24] Starting peer-to-peer node               instance=Geth/pimx/v1.7.3-stable-4bb3c
89d/windows-amd64/go1.9
INFO [03-09|19:53:24] Allocated cache and file handles         database=d:\\app\\Geth\\pimxblockchain
\\geth\\chaindata cache=128 handles=1024
INFO [03-09|19:53:24] Initialised chain configuration          config="{ChainID: 888 Homestead: 0 DAO
: <nil> DAOSupport: false EIP150: 0 EIP155: 0 EIP158: 0 Byzantium: <nil> Engine: unknown}"
INFO [03-09|19:53:24] Disk storage enabled for ethash caches   dir=d:\\app\\Geth\\pimxblockchain\\get
h\\ethash count=3
INFO [03-09|19:53:24] Disk storage enabled for ethash DAGs     dir="C:\\Users\\Miao Xuan\\AppData\\Et
hash"     count=2
INFO [03-09|19:53:24] Initialising Ethereum protocol           versions="[63 62]" network=888
INFO [03-09|19:53:24] Loaded most recent local header          number=0 hash=08665a…088f7a td=20
INFO [03-09|19:53:24] Loaded most recent local full block      number=0 hash=08665a…088f7a td=20
INFO [03-09|19:53:24] Loaded most recent local fast block      number=0 hash=08665a…088f7a td=20
INFO [03-09|19:53:24] Loaded local transaction journal         transactions=0 dropped=0
INFO [03-09|19:53:24] Regenerated local transaction journal    transactions=0 accounts=0
INFO [03-09|19:53:24] Starting P2P networking
INFO [03-09|19:53:24] RLPx listener up                         self="enode://343a43a1fc463bd5a9f8470c
390ddfc0c975150ec3a609ef5048e38716d6b21bac9c424d29513d69545970fe266644347a689f3ec516e9e3b3d22cadd5ba2
95b@[::]:30303?discport=0"
INFO [03-09|19:53:24] IPC endpoint opened: \\.\pipe\geth.ipc
INFO [03-09|19:53:24] HTTP endpoint opened: http://127.0.0.1:8545
Welcome to the Geth JavaScript console!

instance: Geth/pimx/v1.7.3-stable-4bb3c89d/windows-amd64/go1.9
coinbase: 0xce0ecd9646574847abf2888801cac76a0812823a
at block: 0 (Thu, 01 Jan 1970 08:00:00 +08)
 datadir: d:\app\Geth\pimxblockchain
 modules: admin:1.0 debug:1.0 eth:1.0 miner:1.0 net:1.0 personal:1.0 rpc:1.0 txpool:1.0 web3:1.0

> INFO [03-09|20:53:24] Regenerated local transaction journal    transactions=0 accounts=0
INFO [03-09|21:53:24] Regenerated local transaction journal    transactions=0 accounts=0
```

Figure 6.14 Blockchain running in End Device



pimx Captive Portal

* *Third Raspberry Pi* *

Login success!

Logged in with blockchain address:

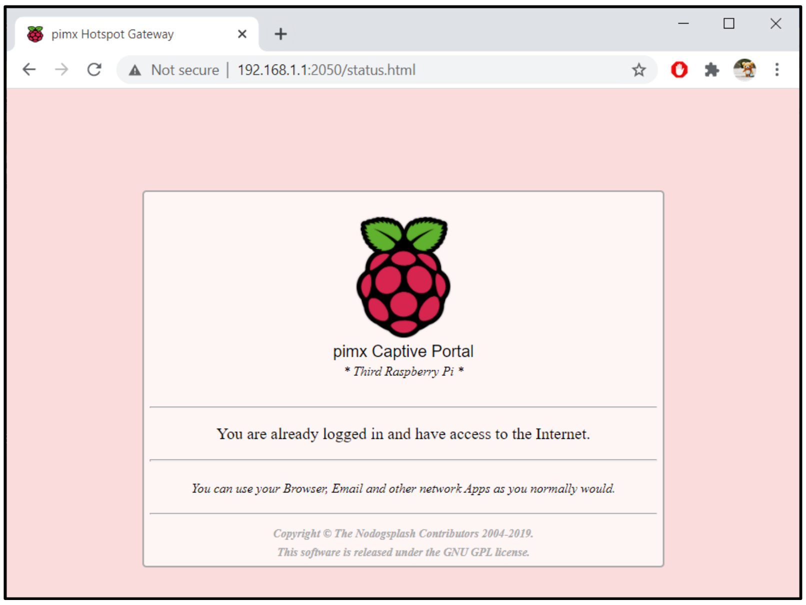
0xCe0ECd9646574847ABf2888801caC76a0812823A

For access to the Internet, please tap or click Continue.

Copyright © The Nodogsplash Contributors 2004-2019.
This software is released under the GNU GPL license.

Figure 6.15 Third Raspberry Pi pimx Captive Portal (successful login after verification of blockchain)

After granting internet access, this page will be shown and the device will be successfully connected to the internet.



pimx Captive Portal
** Third Raspberry Pi **

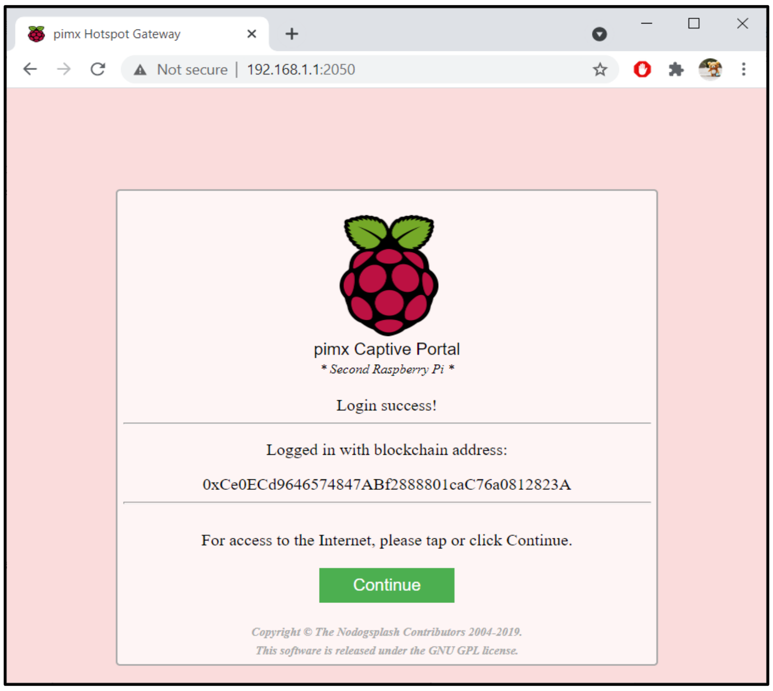
You are already logged in and have access to the Internet.

You can use your Browser, Email and other network Apps as you normally would.

Copyright © The Nodogsplash Contributors 2004-2019.
This software is released under the GNU GPL license.

Figure 6.16 Third Raspberry Pi pimx Captive Portal homepage (granting Internet access)

Since the end device is also a peer of second Raspberry Pi, therefore when near to the wireless AP, the end device is also able to successful authenticate:



pimx Captive Portal
** Second Raspberry Pi **

Login success!

Logged in with blockchain address:

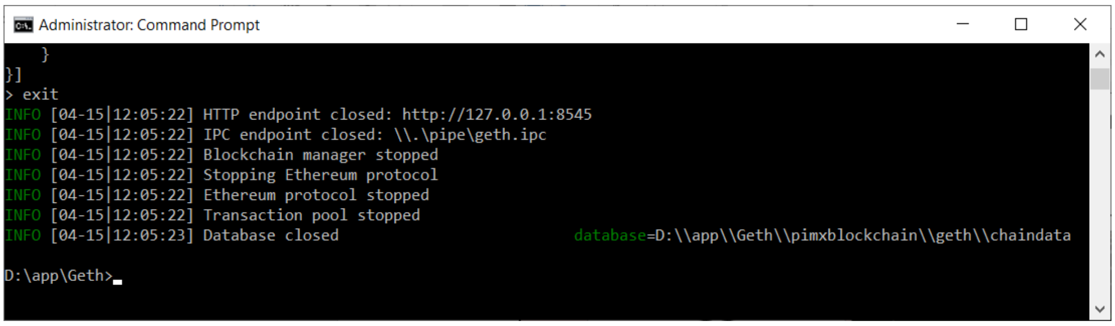
0xCe0ECd9646574847ABf2888801caC76a0812823A

For access to the Internet, please tap or click Continue.

Copyright © The Nodogsplash Contributors 2004-2019.
This software is released under the GNU GPL license.

Figure 6.17 Second Raspberry Pi pimx Captive Portal (successful login after verification of blockchain)

Now, we will demonstrate a failed process whereby the blockchain is not running in the machine. The blockchain is not detected, thus, the user will not be able to proceed to the next step which is to connect to the internet. He will be stuck in the wireless AP captive portal without internet access.


Administrator: Command Prompt

```
    }
}]
> exit
INFO [04-15|12:05:22] HTTP endpoint closed: http://127.0.0.1:8545
INFO [04-15|12:05:22] IPC endpoint closed: \\.\pipe\geth.ipc
INFO [04-15|12:05:22] Blockchain manager stopped
INFO [04-15|12:05:22] Stopping Ethereum protocol
INFO [04-15|12:05:22] Ethereum protocol stopped
INFO [04-15|12:05:22] Transaction pool stopped
INFO [04-15|12:05:23] Database closed                              database=D:\\app\\Geth\\pimxblockchain\\geth\\chaindata

D:\app\Geth>_
```

Figure 6.18 End device Geth console (stop blockchain from running)

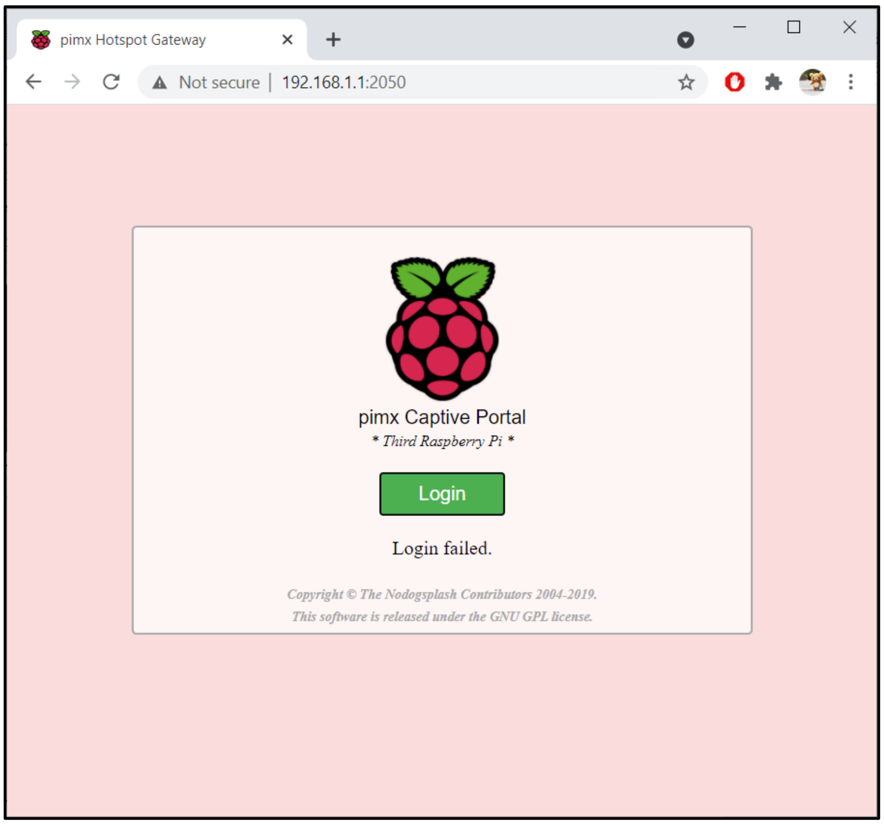

pimx Hotspot Gateway
Not secure | 192.168.1.1:2050
pimx Captive Portal
* Third Raspberry Pi *
Login
Login failed.
Copyright © The Nodogsplash Contributors 2004-2019.
This software is released under the GNU GPL license.

Figure 6.19 Third Raspberry Pi pimx Captive Portal homepage (not running, fail to authenticate)

This also applies to other devices not inside the blockchain, which may also result to failure in authenticating in the captive portal.

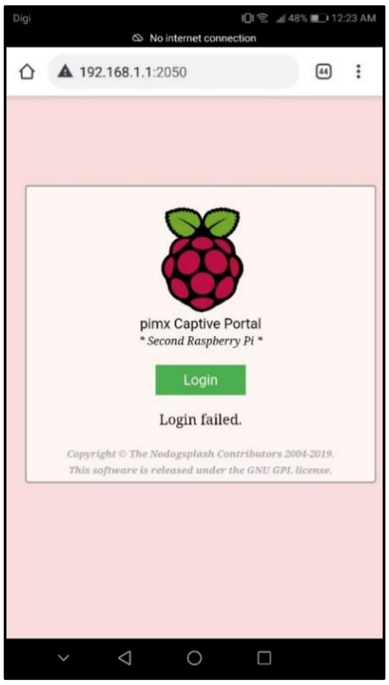

Digi
No internet connection
192.168.1.1:2050
pimx Captive Portal
* Second Raspberry Pi *
Login
Login failed.
Copyright © The Nodogsplash Contributors 2004-2019.
This software is released under the GNU GPL license.

Figure 6.20 Second Raspberry Pi pimx Captive Portal homepage (end device not peer, fail to authenticate)

When the first Raspberry Pi is nearer to the end device than the third Raspberry Pi, the end device gets connected to it, but is unable to proceed with the authentication. This is because end device is not part of first Raspberry Pi's trusted peer.

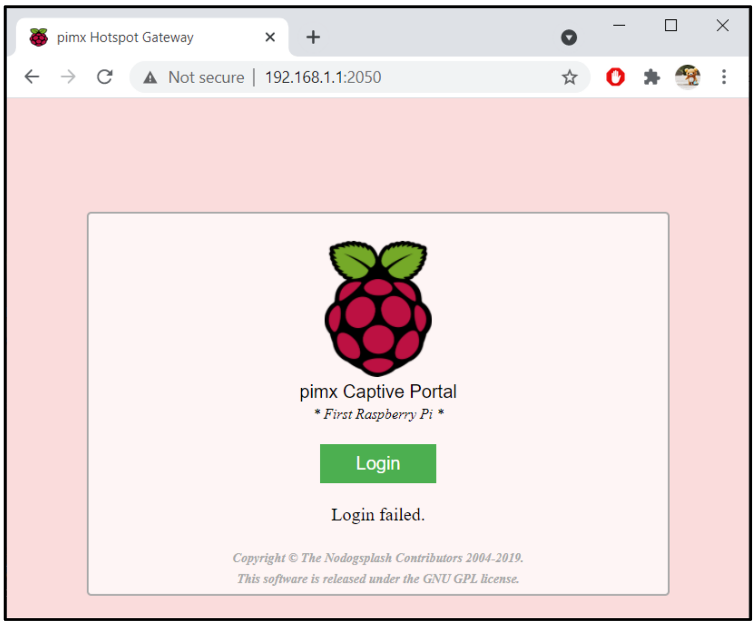

pimx Hotspot Gateway
Not secure | 192.168.1.1:2050
pimx Captive Portal
* First Raspberry Pi *
Login
Login failed.
Copyright © The Nodogsplash Contributors 2004-2019.
This software is released under the GNU GPL license.

Figure 6.21 First Raspberry Pi pimx Captive Portal homepage (wireless AP not peer, fail to authenticate)

6.4 Experimental Results and Testing

In this section, we evaluate the performance of the proposed context-aware authentication method using blockchain for pervasive computing against existing Wi-Fi authentication method.

6.4.1 Experimental Setup

To configure the test bed, we will compare the proposed Ethereum blockchain-based authentication method with the existing WPA2 authentication method. As shown in the network architecture section in Chapter 4, the first Raspberry Pi is a blockchain network with no trusted devices. Now, we will modify the first Raspberry Pi to use the existing WPA2 authentication method. The network architecture diagram will be as follows:

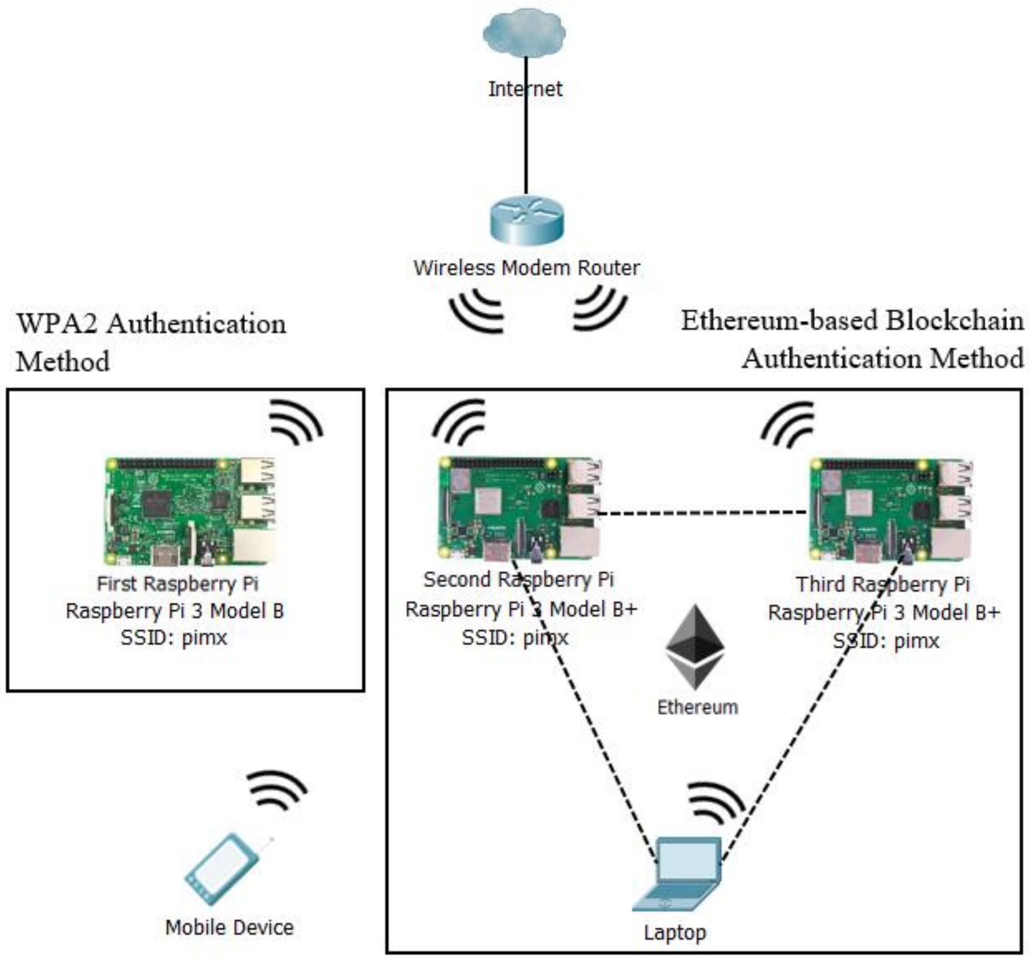


Figure 6.22 Network Architecture Diagram (WPA2 compared to Blockchain Authentication Method)

6.4.2 Evaluation Metrics

To evaluate the proposed system compared to the WPA2 authentication system, we will measure the time used for different device to obtain Internet connectivity. The formula to measure the time is as follows:

Blockchain Authentication Method

Time = Discover SSID + Connect and Redirect to Captive Portal + Obtain Internet

WPA2 Authentication Method

Time = Discover SSID + Type Password + Obtain Internet

Device Operating System	**Blockchain Authentication Method (ms)**	**WPA2 Authentication Method (ms)**
Windows 10	08.920	10.629

Table 6.2 Time for device to connect to the Internet using different authentication methods

Based on the table above, it is known that the Blockchain authentication method is significantly faster than WPA2 authentication method, as it does not require user to key in password upon authentication. Once user is added into the blockchain same as the wireless AP, it is simple to obtain Internet access, unlike WPA2 Authentication Method whereby user will need to know the password to be able to authenticate.

6.4.3 A/B Testing

A/B Testing is also known as Split Testing. We will collect and study the data of the reviews on the proposed blockchain method compared with the WPA2 method. This is to find out which authentication method performs better in a controlled environment.

Step 1: Form Hypothesis

Null hypothesis (H0): WPA2 authentication method is more secured, efficient and preferable than Ethereum blockchain-based authentication method.

Alternative hypothesis (H1): Ethereum blockchain-based authentication method is more secured, efficient and preferable than WPA2 authentication method.

Step 2: Create Control Group and Test Group

We will randomly select 100 volunteers, 50 each for Control group and Test group. Control group will be using the WPA2 authentication method while Test group will be using the Ethereum blockchain-based authentication method. Random sampling method will be used to eliminate bias of the A/B test.

Step 3: Collect the Data and Conduct the A/B Test

The table below shows 10 sample data which are selected randomly:

Volunteer	**Ethereum Blockchain-based Authentication Method (ratings out of 5)**	**WPA2 Authentication Method (ratings out of 5)**
Volunteer 1	5	3
Volunteer 2	5	2
Volunteer 3	4	4
Volunteer 4	5	3
Volunteer 5	3	5
Volunteer 6	5	3
Volunteer 7	4	4
Volunteer 8	5	5
Volunteer 9	4	5
Volunteer 10	5	2

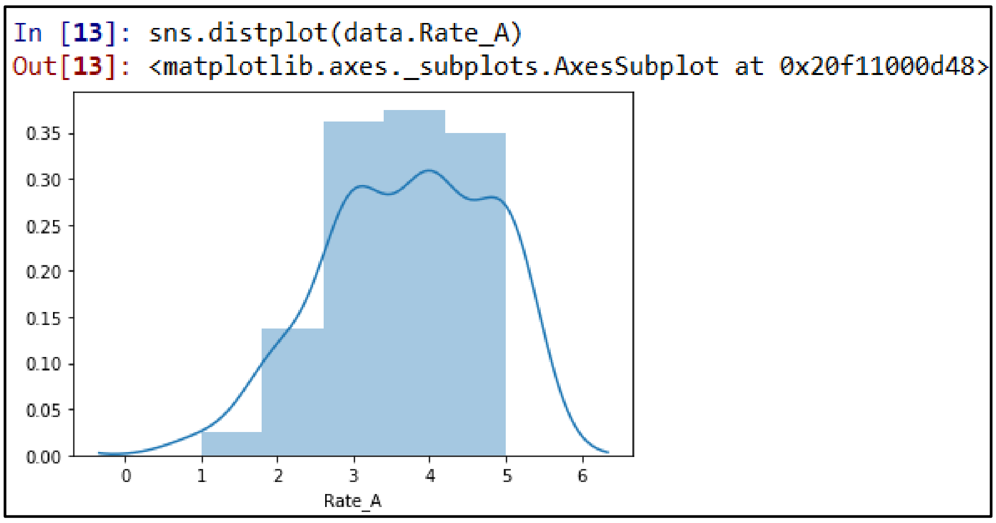
Table 6.3 10 Sample Volunteer Data of Ratings on Authentication Methods

Mean ratings for Control group: 3.97 over 5

Mean ratings for Test group: 4.54 over 5

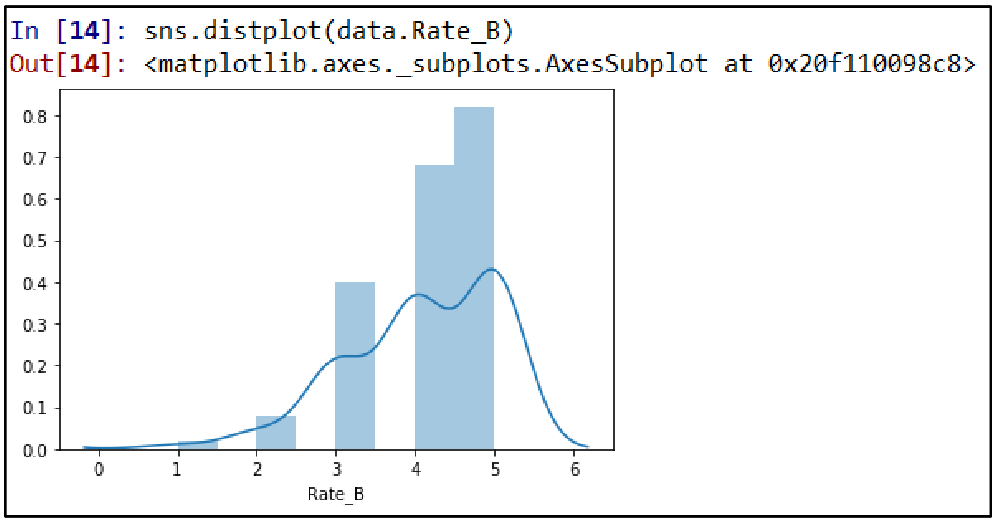
Step 4: Statistical Significance of the A/B Test

Rate_A represents the rating for WPA2 authentication method which is the control group, whereby Rate_B represents the rating for Ethereum Blockchain-based Authentication Method which is the test group.



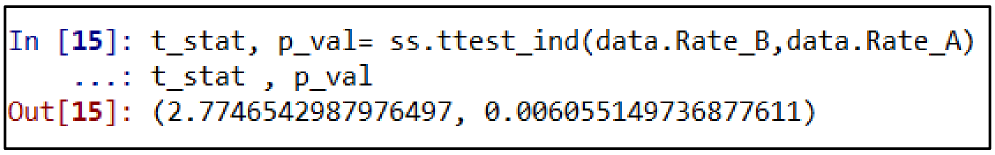
```
In [13]: sns.distplot(data.Rate_A)
Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0x20f11000d48>
```

Figure 6.23 Distribution of control group (Rate_A)



```
In [14]: sns.distplot(data.Rate_B)
Out[14]: <matplotlib.axes._subplots.AxesSubplot at 0x20f110098c8>
```

Figure 6.24 Distribution of test group (Rate_B)



```
In [15]: t_stat, p_val= ss.ttest_ind(data.Rate_B,data.Rate_A)
    ...: t_stat , p_val
Out[15]: (2.7746542987976497, 0.006055149736877611)
```

Figure 6.25 t-test

The mean ratings for the test group is observed as 4.54 over 5. The hypothesized value which is the mean of the control group is 3.97 over 5. The t-score we achieved is 2.7747, and the p-value is 0.0061. The p-value is less than the significance level (0.05). Therefore, we reject the null hypothesis. To conclude, H1 is accepted. Ethereum blockchain-based authentication method is more secured, efficient and preferable than WPA2 authentication method.

CHAPTER 7: CONCLUSION

Security issues rise, cases of identity theft are increasing, leading to people feeling insecure of their identity on the online world. People needing to memorise their passwords on every platform leads to them using the same passwords in all the platforms including important ones. This increases the risk of account exploitation due to data breach in different platforms.

It is a motivation to implement a secured authentication system as security issues arises everywhere. Blockchain is used to improve the security of the network as it is an emerging technology to eliminate the need for centralised systems. This can highly decrease the risk of the center server being the single point of failure, leading to the vulnerability of the entire network.

Hence, this project proposes a Wi-Fi authentication system by implementing blockchain technology in the wireless APs. Once the user is connected to any of the wireless APs, they are able to indirectly connect to the other wireless APs in the network because they are chained together using the blockchain technology.

It is a problem whereby there is a limitation of wireless distribution system on the Raspberry Pi. IP address allocation for the end devices to be able to connect to the blockchain is also a factor to be improved. Besides, there is a limit for number of nodes connecting to the small network, therefore, it is a challenge whereby it exceeds the limit and what could be done to solve the problem. Further features for private Ethereum blockchain will also be explored in details to allow more flexibility and accessibility of the nodes to the blockchain.

In future work, end devices can easily be added into the blockchain to allow easy connection when it is inside the list of trusted networks inside the wireless AP. It is hoped that in the future, Android and iOS devices can be running the Ethereum blockchain easily. The integration for the authentication system with the blockchain will also be strengthen to improve user experience. A trusted Wi-Fi authentication system will enable users to connect to the network effortlessly, yet securely.

BIBLIOGRAPHY

Agrawal, H., 2019. *What Are Private Blockchains & How Are They Different From Public Blockchains?* Available from: https://coinsutra.com/private-blockchain-public-blockchain/. [Accessed 6 April 2020].

Albaugh, D., 2020. Biggest data breaches in history. Available from: https://www.comparitech.com/blog/information-security/biggest-data-breaches-in-history/ [Accessed 6 April 2020].

Block42 Team, 2018. *ICO Review: Marconi Protocol*. Medium, [blog] 29 June. Available from: https://medium.com/block42-blockchain-company/ico-review-marconi-protocol-ffc572cf0aaa. [Accessed 11 April 2020].

Buntinx, 2015. *Is Blockchain Technology the Future of Two-Factor Authentication?* Available from: https://news.bitcoin.com/blockchain-technology-future-two-factor-authentication/. [Accessed 7 April 2020].

Douglas, R., 2020. *2020 Identity theft statistics*. Available from: https://www.consumeraffairs.com/finance/identity-theft-statistics.html. [Accessed 6 April 2020].

Ethereum Foundation, n.d. Ethereum is a global, open-source platform for decentralized applications. Available from: https://ethereum.org/en/. [Accessed 16 August 2020].

Geier, E., 2014. *8 ways to improve wired network security*. Available from: https://www.networkworld.com/article/2175048/8-ways-to-improve-wired-network-security.html. [Accessed 10 April 2020].

Gisolfi, D., 2018. *Decentralized Identity: An alternative to password-based authentication*. Available from: https://www.ibm.com/blogs/blockchain/2018/10/decentralized-identity-an-alternative-to-password-based-authentication/. [Accessed 14 April 2020].

HBUS, 2018. *The How and Why of Blockchain Transparency*. Medium, [blog] 19 December, Available from: https://medium.com/hbus-official/the-how-and-why-of-blockchain-transparency-b3f3465f6989. [Accessed 5 April 2020].

Hunt, T., n.d. *Have I Been Pwned*. Available from: https://haveibeenpwned.com/. [Accessed 5 April 2020].

Kline, K., 2019. *Why New Protocol Marconi Is Restructuring Core Internet Technology*. Available from: https://www.inc.com/kenny-kline/why-new-protocol-marconi-is-restructuring-core-internet-technology.html. [Accessed 10 April 2020].

Koetse, M., 2019. *Cybersecurity Experts Warn: Flicking the V-Sign in Photos Could Give Away Your Fingerprint Data*. Available from: https://www.whatsonweibo.com/cybersecurity-experts-warn-flicking-the-v-sign-in-photos-could-give-away-your-fingerprint-information/. [Accessed 5 April 2020].

Larcheveque, E., 2016. *Bitcoin address authentication protocol (BitID)*. Available from: https://github.com/bitid/bitid/blob/master/BIP_draft.md. [Accessed 10 April 2020].

Marconi Foundation, 2018. *Marconi Protocol*. [Online]. Available at: https://marconi.org/. [Accessed 15 April 2020].

Muzzy, E., 2020. What Is Proof of Stake? Available from: https://consensys.net/blog/blockchain-explained/what-is-proof-of-stake/#:~:text=The%20core%20of%20the%20Ethereum,Work%20(PoW)%20consensus%20mechanism. [Accessed 17 August 2020].

Nagpal, R., 2018. *Blockchain-based authentication of devices and people*. Available from: https://medium.com/blockchain-blog/blockchain-based-authentication-of-devices-and-people-c7efcfcf0b32. [Accessed 13 April 2020].

Nelson, P., 2019. *How blockchain will manage networks*. Available from: https://www.networkworld.com/article/3356496/how-blockchain-will-manage-networks.html. [Accessed 8 April 2020].

Nicolls, D., 2019. *What is Biometric Authentication?* Available from: https://www.jumio.com/what-is-biometric-authentication/. [Accessed 5 April 2020].

R Joosten, 2018. *A conceptual analysis on sovrin*. [Online]. Available at: https://www.researchgate.net/publication/323144927. [Accessed 15 April 2020].

Raspberry Pi Foundation, n.d. Raspberry Pi 3 Model B. Available from: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/. [Accessed 15 August 2020].

Raspberry Pi Foundation, n.d. Raspberry Pi 3 Model B+. Available from: https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/. [Accessed 15 August 2020].

Reiff, N., 2020. *Blockchain Explained*. Available from: https://www.investopedia.com/terms/b/blockchain.asp. [Accessed 5 April 2020].

Rouse, M., 2019. *pervasive computing (ubiquitous computing)*. Available from: https://internetofthingsagenda.techtarget.com/definition/pervasive-computing-ubiquitous-computing. [Accessed 6 April 2020].

Rouse, M., 2014. *Possession factor*. Available from: https://searchsecurity.techtarget.com/definition/possession-factor#:~:text=The%20possession%20factor%2C%20in%20a,conjunction%20with%20a%20software%20token. [Accessed 6 April 2020].

Ruff, T., 2018. *7 Myths of Self-Sovereign Identity*. Medium, [blog] 31 October. Available from: https://medium.com/evernym/7-myths-of-self-sovereign-identity-b16648c3090d. [Accessed 13 April 2020].

Sacolick, I., 2020. *What is agile methodology? Modern software development explained*. Available from: https://www.infoworld.com/article/3237508/what-is-agile-methodology-modern-software-development-explained.html. [Accessed 15 April 2020].

Sharma, L., 2016. Waterfall Model. Available from: https://www.toolsqa.com/software-testing/waterfall-model/#:~:text=The%20waterfall%20model%20is%20a,Production%2FImplementation%2C%20and%20Maintenance. [Accessed 19 August 2020].

Sobers, R., 2020. *107 Must-Know Data Breach Statistics for 2020*. Available from: https://www.varonis.com/blog/data-breach-statistics/. [Accessed 6 April 2020].

Southurst, 2016. *BitID Will Verify Your Identity with the Bitcoin Blockchain.* Available from: https://news.bitcoin.com/bitid-verify-id-bitcoin-blockchain/. [Accessed 7 April 2020].

Sovrin Foundation, 2017. *The Inevitable Rise of Self-Sovereign Identity*. [Online]. Available at: https://sovrin.org/. [Accessed 15 April 2020].

Tar, A., 2018. *Proof-of-Work, Explained*. Available from: https://cointelegraph.com/explained/proof-of-work-explained. [Accessed 6 April 2020].

TeamGantt., 2020. Online gantt chart project planning software. Available from : https://www.teamgantt.com/. [Accessed 22 August 2020].

Windley, Phillip J., 2019. *An overview of Self-Sovereign Identity: the use case at the core of Hyperledger Indy.* Available from: https://www.hyperledger.org/blog/2019/05/01/an-overview-of-self-sovereign-identity-the-use-case-at-the-core-of-hyperledger-indy. [Accessed 15 April 2020].

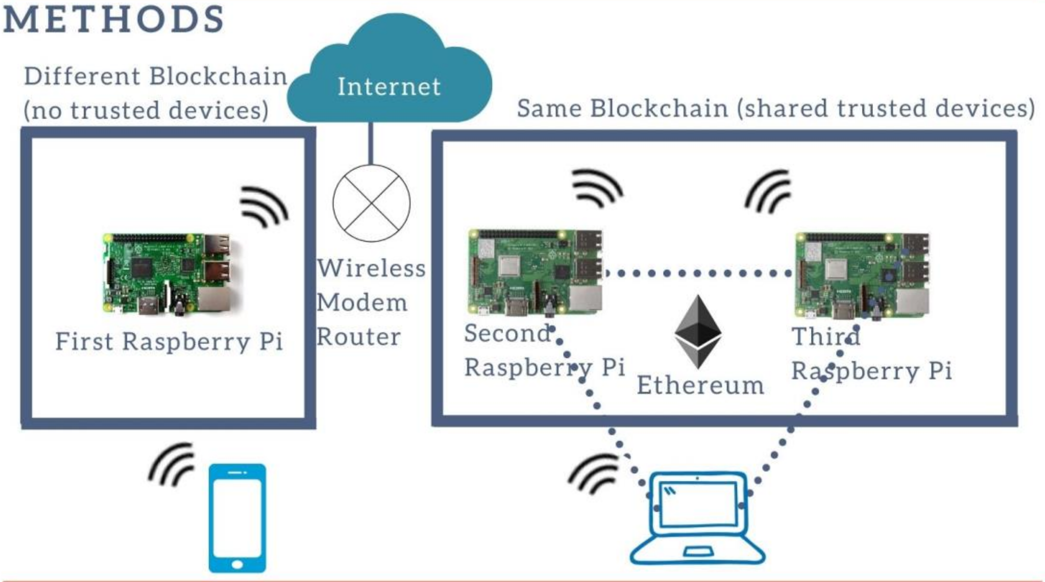
APPENDIX A : POSTER

A CONTEXT-AWARE AUTHENTICATION METHOD USING BLOCKCHAIN FOR PERVASIVE COMPUTING

INTRODUCTION

A trusted Wi-Fi authentication system using blockchain will be implemented. Blockchain is an emerging technology which improves security, it can ease the Wi-Fi authentication system whereby user does not need to remember password.

METHODS



RESULTS

End device in the same blockchain as the Raspberry Pi wireless AP will be able to obtain Internet Access by authentication process through Captive Portal.

DISCUSSION

Performance of Blockchain Authentication System is slightly faster than WPA2 Authentication System. Password will be needed for WPA2, but blockchain ID can be automatically detected in Blockchain Authentication System after wireless AP added the end device into list of trusted peers.

CONCLUSION

Future implementation will be implementing Ethereum blockchain in Android and iOS devices, and improve the performance and security of the Blockchain Authentication System.

Prepared by: Ng Miao Xuan   Supervised by: Dr. Aun Yichiet

APPENDIX B : FINAL YEAR PROJECT WEEKLY REPORT

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S1	**Study week no.: 1**
Student Name & ID: Ng Miao Xuan 17ACB01924	
Supervisor: Dr. Aun Yichiet	
Project Title: A CONTEXT-AWARE AUTHENTICATION METHOD USING BLOCKCHAIN FOR PERVASIVE COMPUTING	

1. WORK DONE [Please write the details of the work done in the last fortnight.] Researched on further features of Ethereum blockchain.
2. WORK TO BE DONE To find out which feature of Ethereum blockchain is able to perform authentication.
3. PROBLEMS ENCOUNTERED Uncertain which method in Ethereum blockchain is able to perform authentication.
4. SELF EVALUATION OF THE PROGRESS Self-assigned tasks are yet to be completed within the expected timeframe.

[illegible] [illegible]

Supervisor's signature Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S1	**Study week no.: 3**
Student Name & ID: Ng Miao Xuan 17ACB01924	
Supervisor: Dr. Aun Yichiet	
Project Title: A CONTEXT-AWARE AUTHENTICATION METHOD USING BLOCKCHAIN FOR PERVASIVE COMPUTING	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

Discussed on making wireless AP into open authentication to simplify the process.

2. WORK TO BE DONE

Research on how to make ledger transaction into containing MAC address.

3. PROBLEMS ENCOUNTERED

Lack of link layer knowledge.

4. SELF EVALUATION OF THE PROGRESS

Self-assigned tasks are yet to be completed within the expected timeframe.

[illegible] [illegible]

Supervisor's signature Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S1	**Study week no.: 5**
Student Name & ID: Ng Miao Xuan 17ACB01924	
Supervisor: Dr. Aun Yichiet	
Project Title: A CONTEXT-AWARE AUTHENTICATION METHOD USING BLOCKCHAIN FOR PERVASIVE COMPUTING	

1. WORK DONE [Please write the details of the work done in the last fortnight.] Researched on blockchain smart contracts.
2. WORK TO BE DONE Research on adding client to blockchain using smart contract, and scan nearby devices to allow auto authentication.
3. PROBLEMS ENCOUNTERED Lack of knowledge on blockchain smart contracts.
4. SELF EVALUATION OF THE PROGRESS Able to complete the tasks given on time.

[illegible]

Supervisor's signature

[illegible]

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S1	**Study week no.: 7**
Student Name & ID: Ng Miao Xuan 17ACB01924	
Supervisor: Dr. Aun Yichiet	
Project Title: A CONTEXT-AWARE AUTHENTICATION METHOD USING BLOCKCHAIN FOR PERVASIVE COMPUTING	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

Received wireless USB adapter to be able to use the wireless interface card as monitor mode, to see nearby device connection.

2. WORK TO BE DONE

Research on adding end devices into the same blockchain as wireless AP.

3. PROBLEMS ENCOUNTERED

None.

4. SELF EVALUATION OF THE PROGRESS

Assigned tasks are yet to be completed within the expected timeframe.

[illegible]

Supervisor's signature

[illegible]

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S1	**Study week no.: 9**
Student Name & ID: Ng Miao Xuan 17ACB01924	
Supervisor: Dr. Aun Yichiet	
Project Title: A CONTEXT-AWARE AUTHENTICATION METHOD USING BLOCKCHAIN FOR PERVASIVE COMPUTING	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

Added end devices into the same blockchain as wireless AP.
Configured captive portal on the wireless AP to perform authentication before end device obtain Internet connectivity.

2. WORK TO BE DONE

Perform analysis on the authentication process and connectivity of end devices with wireless AP.

3. PROBLEMS ENCOUNTERED

None.

4. SELF EVALUATION OF THE PROGRESS

It is a challenging task to work with blockchain technology.

[illegible]

Supervisor's signature

[illegible]

Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

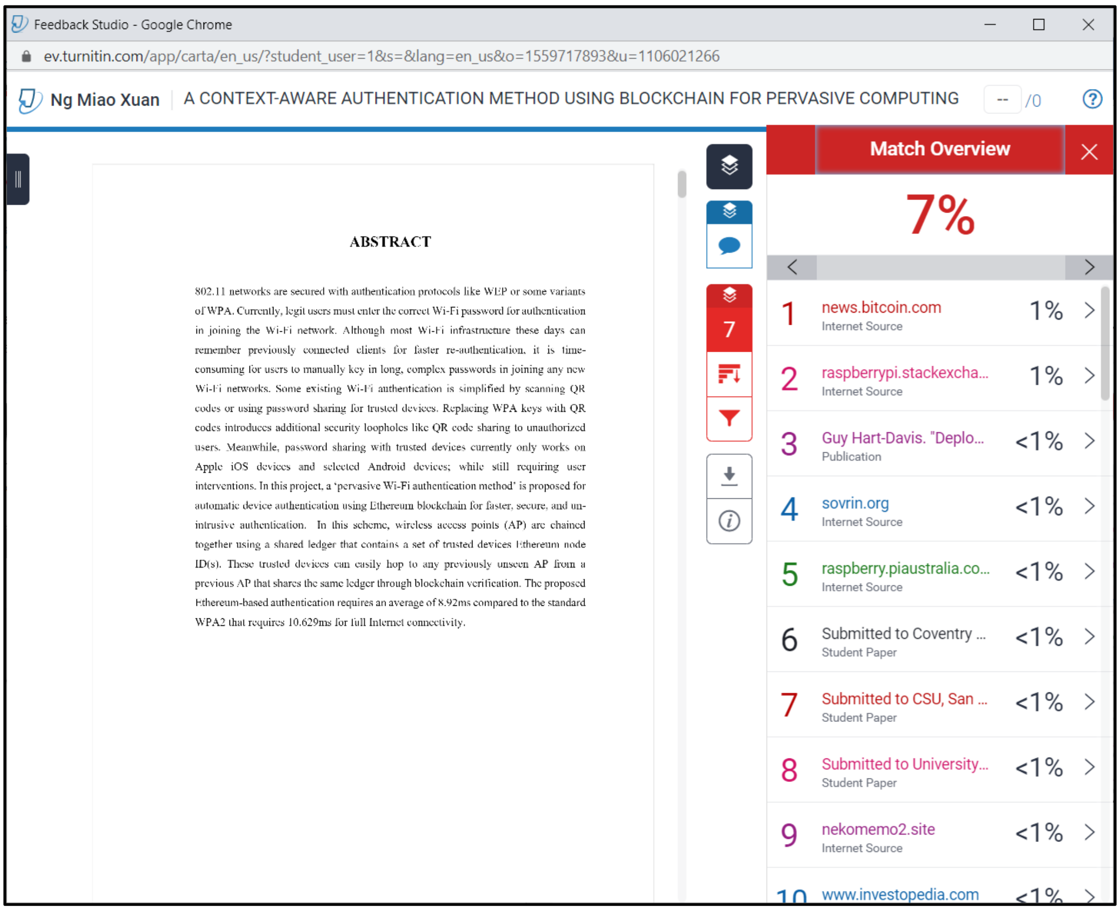
Trimester, Year: Y3S1	**Study week no.: 11**
Student Name & ID: Ng Miao Xuan 17ACB01924	
Supervisor: Dr. Aun Yichiet	
Project Title: A CONTEXT-AWARE AUTHENTICATION METHOD USING BLOCKCHAIN FOR PERVASIVE COMPUTING	

1. WORK DONE [Please write the details of the work done in the last fortnight.] Performed analysis on the network connectivity and completed report for draft check.
2. WORK TO BE DONE Evaluate the performance study and finalise the report.
3. PROBLEMS ENCOUNTERED None.
4. SELF EVALUATION OF THE PROGRESS Doing everything within the expected timeframe and able to complete the project.

[illegible] [illegible]

Supervisor's signature Student's signature

APPENDIX C : PLAGIARISM CHECK RESULT



Feedback Studio - Google Chrome

ev.turnitin.com/app/carta/en_us/?student_user=1&s=&lang=en_us&o=1559717893&u=1106021266

Ng Miao Xuan | A CONTEXT-AWARE AUTHENTICATION METHOD USING BLOCKCHAIN FOR PERVASIVE COMPUTING -- /0

ABSTRACT

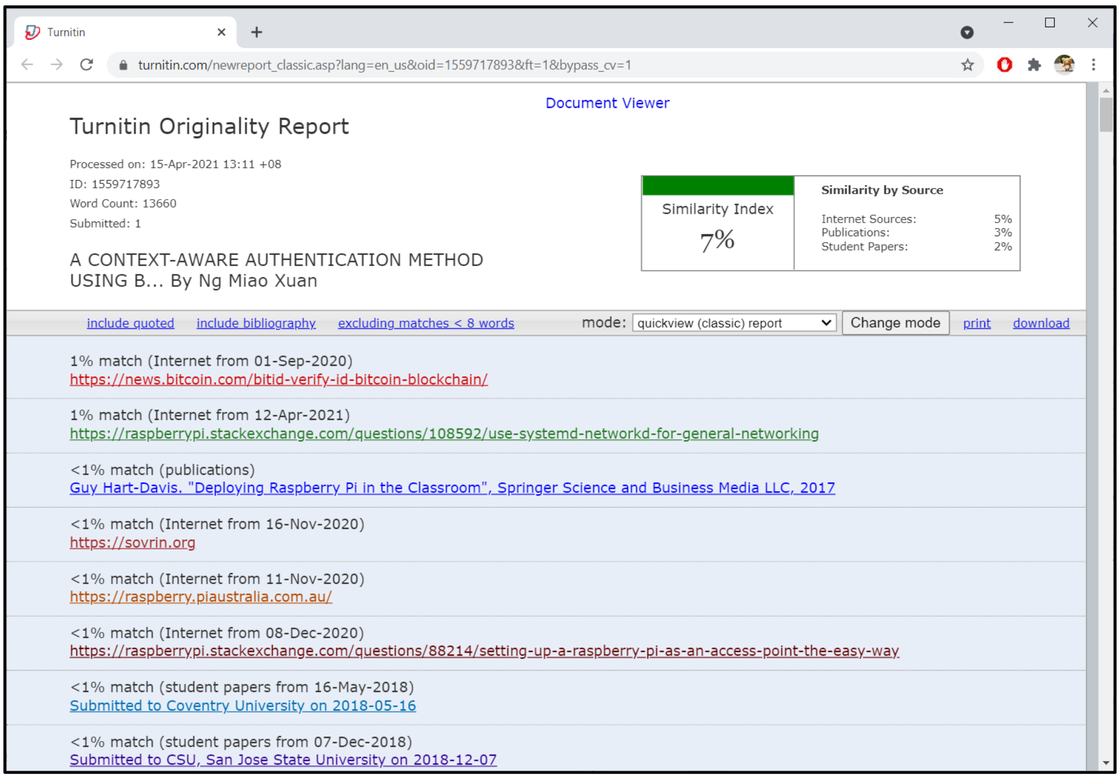
802.11 networks are secured with authentication protocols like WEP or some variants of WPA. Currently, legit users must enter the correct Wi-Fi password for authentication in joining the Wi-Fi network. Although most Wi-Fi infrastructure these days can remember previously connected clients for faster re-authentication, it is time-consuming for users to manually key in long, complex passwords in joining any new Wi-Fi networks. Some existing Wi-Fi authentication is simplified by scanning QR codes or using password sharing for trusted devices. Replacing WPA keys with QR codes introduces additional security loopholes like QR code sharing to unauthorized users. Meanwhile, password sharing with trusted devices currently only works on Apple iOS devices and selected Android devices; while still requiring user interventions. In this project, a 'pervasive Wi-Fi authentication method' is proposed for automatic device authentication using Ethereum blockchain for faster, secure, and un-intrusive authentication. In this scheme, wireless access points (AP) are chained together using a shared ledger that contains a set of trusted devices Ethereum node ID(s). These trusted devices can easily hop to any previously unseen AP from a previous AP that shares the same ledger through blockchain verification. The proposed Ethereum-based authentication requires an average of 8.92ms compared to the standard WPA2 that requires 10.629ms for full Internet connectivity.

Match Overview

7%

1	news.bitcoin.com Internet Source	1%
2	raspberrypi.stackexcha... Internet Source	1%
3	Guy Hart-Davis. "Deplo... Publication	<1%
4	sovrin.org Internet Source	<1%
5	raspberry.piaustralia.co... Internet Source	<1%
6	Submitted to Coventry ... Student Paper	<1%
7	Submitted to CSU, San ... Student Paper	<1%
8	Submitted to University... Student Paper	<1%
9	nekomemo2.site Internet Source	<1%
10	www.investopedia.com	<1%

APPENDIX C: PLAGIARISM CHECK RESULT



Turnitin

turnitin.com/newreport_classic.asp?lang=en_us&oid=1559717893&ft=1&bypass_cv=1

Document Viewer

Turnitin Originality Report

Processed on: 15-Apr-2021 13:11 +08
ID: 1559717893
Word Count: 13660
Submitted: 1

A CONTEXT-AWARE AUTHENTICATION METHOD USING B... By Ng Miao Xuan

Similarity Index	Similarity by Source	
7%	Internet Sources:	5%
	Publications:	3%
	Student Papers:	2%

include quoted   include bibliography   excluding matches < 8 words   mode: quickview (classic) report   Change mode   print   download

1% match (Internet from 01-Sep-2020)
https://news.bitcoin.com/bitid-verify-id-bitcoin-blockchain/

1% match (Internet from 12-Apr-2021)
https://raspberrypi.stackexchange.com/questions/108592/use-systemd-networkd-for-general-networking

<1% match (publications)
Guy Hart-Davis. "Deploying Raspberry Pi in the Classroom", Springer Science and Business Media LLC, 2017

<1% match (Internet from 16-Nov-2020)
https://sovrin.org

<1% match (Internet from 11-Nov-2020)
https://raspberry.piaustralia.com.au/

<1% match (Internet from 08-Dec-2020)
https://raspberrypi.stackexchange.com/questions/88214/setting-up-a-raspberry-pi-as-an-access-point-the-easy-way

<1% match (student papers from 16-May-2018)
Submitted to Coventry University on 2018-05-16

<1% match (student papers from 07-Dec-2018)
Submitted to CSU, San Jose State University on 2018-12-07

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	Ng Miao Xuan
ID Number(s)	17ACB01924
Programme / Course	BACHELOR OF INFORMATION TECHNOLOGY (HONOURS) COMMUNICATIONS AND NETWORKING
Title of Final Year Project	A CONTEXT-AWARE AUTHENTICATION METHOD USING BLOCKCHAIN FOR PERVASIVE COMPUTING

Similarity	**Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)**
Overall similarity index: 7 **%** **Similarity by source** Internet Sources: 5 % Publications: 3 % Student Papers: 2 %	
Number of individual sources listed of more than 3% similarity: 0	
Parameters of originality required and limits approved by UTAR are as Follows: **(i) Overall similarity index is 20% and below, and** **(ii) Matching of individual sources listed must be less than 3% each, and** **(iii) Matching texts in continuous block must not exceed 8 words** ***Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.***	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

[illegible]

Signature of Supervisor	Signature of Co-Supervisor
Name: DR. AUN YICHIET	Name:
Date: 15 APRIL 2021	Date:

UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	17ACB01924
Student Name	NG MIAO XUAN
Supervisor Name	DR. AUN YICHIET

TICK (√)	DOCUMENT ITEMS Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
√	Front Cover
√	Signed Report Status Declaration Form
√	Title Page
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
√	Appendices (if applicable)
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report. [illegible] (Signature of Student) Date: 15 April 2021	Supervisor verification. Report with incorrect format can get 5 mark (1 grade) reduction. [illegible] (Signature of Supervisor) Date: 15 April 2021