2019

Access controls on IP based cameras in IoT ecosystem

Mary Wairimu Muya,
Faculty of Information Technology (FIT)
Strathmore University

Access Controls on IP based Cameras in IoT Ecosystem

Muya, Mary Wairimu

Master of Science in Information Systems Security

2019

Access Controls on IP based Cameras in IoT Ecosystem

Muya, Mary Wairimu

Submitted in partial fulfilment of the requirements for the Degree of Master of Science in Information Systems Security at Strathmore University

Faculty of Information Technology
Strathmore University
Nairobi, Kenya

June, 2019

Declaration

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

Muya, Mary Wairimu

12th June 2019

Approval

This dissertation of Muya, Mary Wairimu was reviewed and approved by the following:

Dr. Humprey Njogu
Senior Lecturer, Faculty of Information Technology
Strathmore University

Dr. Joseph Orero
Dean, Faculty of Information Technology
Strathmore University

Dr. Ruth Kiraka
Dean, School of Graduate Studies
Strathmore University

Abstract

Internet of things (IoT) is a concept of connected things that allows embedded devices, sensors and actuators to interconnect and share data thus bridging the gap between physical devices and virtual objects. The concept of IoT started gaining popularity in 2010, with its popularity impressively outgrowing other concepts up to date. The growth of IoT has seen more than 30% companies globally initiating the process of deploying IoT. IoT security has been a challenge due to its nascent market where manufacturers focus much on getting the product to the market rather than building security from start. Internet Protocol (IP) based cameras are among the most popular IoT devices. Governments, corporations to small business and homeowners are using cameras for surveillance among other activities, with their popularity growing due to their ability to collect and transmit data remotely. As cameras are expected to perform sophisticated tasks, it is important to protect the cameras and data they handle.

The focus of this dissertation is to come up with an access control solution for IP based cameras, in efforts to reduce vulnerabilities associated with identity and access management. This dissertation adopted Rapid Application Development (RAD) methodology to develop the proposed solution. The methodology provided flexibility in changing requirements and testing the prototype at an early stage to continuously improve the system. Must, Should, Could, Would Not (MoSCoW) method was used to identify and rank requirements in evaluating the gaps that existed in the market, as this dissertation could not address all the vulnerabilities the method helped in picking the vulnerabilities to be handled first.

The tested and validated prototype provides a mechanism to restrict factory set authentication credentials, system access lockouts and sending of alerts in cases of suspicious login attempts. The prototype demonstrate how Integrity of camera feeds can be maintained by using a combination of interplanetary file system (IPFS) and Blockchain. The solution also records and stores system logs in immutable format to support forensic investigations

Keywords: Internet of Things, Internet Protocol (IP) based cameras, Attack surface/vulnerability, and security risk

Table of Contents

List of Figures

List of Tables

List of Abbreviations/Acronyms

AES	-	Advanced Encryption Standard
API	-	Application Programming Interfaces
ARM	-	Area Radiation Monitor
CCTV	-	Closed Circuit Televisions
CIA	-	Confidentiality, Integrity and Availability
CID	-	Content Identifier
CIO	-	Chief Information Officer
DAG	-	Directed Acyclic Graph
DB	-	Database
DDoS	-	Distributed Denial of Service
DFD	-	Data Flow Diagram
DHT	-	Dynamic Hash Tables
DNS	-	Domain Name System
E2EE	-	End-to-End Encryption
ERDs	-	Entity Relationship Diagrams
FTP	-	File Transfer Protocol
GPS	-	Global Positioning System
HTTP	-	Hypertext Transfer Protocol
IBM	-	International Business Machines
IDP	-	Intelligent Device Platform
IEEE	-	Institute of Electrical and Electronics Engineers
IoT	-	Internet of Things
IP	-	Internet Protocol
IPFS	-	Interplanetary File System
IPLD	-	Interplanetary Linked Data
IPNS	-	Interplanetary Naming System
IT	-	Information Technology
LAN	-	Local Areas Network
LPWAD	-	Lower power Wide Area Network
MoSCoW	-	Must, Should, Could, Would Not

NVR	-	Network Video Recorder
OPSEC	-	Operational Security
OS	-	Operating System
OWASP	-	Open Web Application Security Project
PAAS	-	Platform as a Service
PKI	-	Public Key Infrastructure
PTZ	-	Pan Tilt Zoom
RAD	-	Rapid Application Development
RAD	-	Rapid Application Development
RFID	-	Radio-Frequency Identification
SDLC	-	Software Development Lifecycle
SoC	-	System on Chip
US	-	United States
USB	-	Universal Serial Cable
VDR	-	Virtual Data Room
WAN	-	Wide Area Network
Wi-Fi	-	Wireless Fidelity

Acknowledgement

This work could not have been a success without leaning on the shoulders of quite a number of people. My special thanks goes to almighty God for the good health and knowledge, my supervisor Dr. Humphrey Njogu who has been there to give me ideas, correct and suggest areas of improvements while working on this dissertation, Dr. Rysavy who assisted me during the early stages of proposal writing and crafting of the idea, to my parents for their care and support all through, aunt Margaret for her undying sincere moral support, Sharon who helped me while joining the course, my siblings, and friends including Steve, Nahashon, James and Elizabeth for their encouragements. Personal thanks to Cyrus who has helped me with brilliant ideas, God bless.

Dedication

I would wish to dedicate this dissertation to my parents who have been the reason for whom I am today, may God bless you.

Chapter 1 : Introduction

1.1 Background of Study

Internet of Things (IoT) overlaps other fields of study including mobile computing, pervasive computing, and cyber physical systems. IoT is more changing field like no other before with many scholars coming up with different descriptions. Banafa (2017) defines IoT as a universe connecting things by providing key physical data and further processing of data on the cloud to achieve business insights.

IoT is referred to as an ecosystem as no one company can be able to do it all. In IoT everything is interconnected with each other to create a mutual value. The three major enablers of an IoT ecosystem is the platform, the market expectation and lastly the network effects. The platform defines how services are built, the market expectation is how the user perceive the platform, is it user friendly, is it secure or not among many other factors. The network effects allow the users and the service provided to interact, that is if there are more services more users will be attracted thus more partners get into the platform (Valdez-de-Leon, 2017).

IoT provides three major products, which include devices, applications and services. Devices include cameras, phones, applications run on the devices while the devices and applications provide services such as surveillance, big data analytics, and cloud storage among many more services. The ability of the IoT devices to handle tasks without human interactions have contributed to their increased user acceptance. One of the many connected devices that have gained the acceptance are the IP based cameras as they able to perform more sophisticated tasks such as performing inference (Higginbotham, 2018).

IP cameras send and receive data via a Local Area Network (LAN), they do not need video cable or Virtual Data Room (VDR) to monitor, but rather they use data connections such as Universal Serial Cable (USB), Ethernet, and Wi-Fi among other technologies. This means that the IP camera feeds can be viewed anywhere in the world. The IP cameras surpass the traditional Closed circuit Televisions (CCTV) as they use computer network to transfer information. Their capability to pass information via Internet provide efficiency and effectiveness as one need not to be in one place to monitor. IP cameras are smaller in size with better visual resolution compared to CCTV, with their

ease to use advantage improving their popularity in a complex world where people are more alert about their security both for business setups and at home (Puiu, 2014).

Although IP cameras are being becoming popular, their security vulnerabilities are worrying. As he notes manufacturers of IP based cameras are slow to bringing security into speed. They develop inexpensive cameras, consequently basic software is installed that are difficult to update (Foxhoven, 2016). There is an increased risk of security threats and breaches due to the increased entry points into the IoT ecosystem. Adding new components in an IoT ecosystem increases the number of threats and breaches. Cameras are among many connected devices in the Internet. Attackers are able to use a single weak link to attack all other devices on the Internet. As mentioned before, IoT is an ecosystem where we have different companies providing different services; this makes it difficult to develop a cohesive security strategy as different technologies are involved. Finally, IoT being an ecosystem with many players it is not clear who should take which role when it comes to security leaving high risk of security breach.

Security risks for IP cameras come along with a number of vulnerabilities owing to their reliance on the Internet. Up to date vulnerabilities are being discovered. This shows that IP based cameras are yet to be completely secure. Examples of recent attacks include a botnet by the name Mirai that launched a mass Distributed Denial of Service (DDoS) attack, the Wanna Cry, and the IoTroop, these among other attacks have caused unimaginable loss just to patch vulnerabilities when the damage is already done.

This dissertation identifies four major vulnerabilities in IP based cameras that are yet to be fully addressed. The vulnerabilities include factory default passwords, insecure web design, lack of integrity verification, mutable system logs to support forensic investigations. The main areas of focus by the dissertation is access controls, authentication, confidentiality/privacy, integrity and trust. This is achieved by restricting use of factory default passwords, designing lockouts and alerts for suspicious authentication attempts and finally using Interplanetary file systems and block chain technologies to maintain integrity of both camera feeds and system logs.

1.2 Problem Statement

IP based cameras are among the many IoT devices that collect and transfer sensitive and confidential information for different users. Having sensitive data passed over the Internet puts the security and privacy of the data into question. The ability to control who accesses the camera and

the data it collects and transmit is critical. IP cameras have a number of vulnerabilities that are targets for attackers. This ranges from major vulnerabilities as design flaws and default passwords to other vulnerabilities like lack proper of encryption, authentication and authorisation mechanisms. When these vulnerabilities are exploited, they cause massive destruction.

For instance, Shah (2018) notes that 47 percent of CIOs and IT managers have allowed IoT devices into their corporate network without change of default passwords. This negligence has led to attackers exploiting the vulnerabilities leading to attacks like Mirai Botnet that infected over 185, 000 IoT devices online leading to a massive DDoS attack that left much of the Internet inaccessible in the U.S (Fruhlinger, 2018). To counter-attack these vulnerabilities it is crucial to ensure proper security mitigations are in place. Restraining access and default passwords, coming up with secure authentication mechanisms, failed access lockouts and alerts, limiting chances of data and activity logs alteration is among the many protective measures towards securing IP based cameras and IoT ecosystem at large.

1.3 Objectives

The major objective of the dissertation is to come up with a solution to provide access controls by minimising common vulnerabilities found in IP based cameras to improve their security.

The specific objectives include:

i. To identify common threats and vulnerabilities on IP based cameras that lead to unauthorised access and their security implications.
ii. To review the existing access control solutions for IP based cameras.
iii. To design, implement and test the proposed solution for IP based cameras.
iv. To validate the effectiveness of the proposed solution.

1.4 Research Questions

The dissertation is based on the following research questions.

i. What are the common threats and vulnerabilities that lead to unauthorised access as well as their security risk to IP based cameras?
ii. What are the strengths and gaps that exist in current access control solutions used in securing IP based cameras?

iii. How to design, implement and test the proposed solution to secure IP based cameras?

iv. How effective is the proposed solution in securing IP based cameras?

1.5 Scope and Limitations

The scope of this dissertation is to use already existing vulnerabilities rather than generating them afresh. Additionally, the dissertation aims to demo physical environments with only one camera that is raspberry pi camera. This dissertation does not intend to develop a fully functional system but rather a prototype to demonstrate its effectiveness controlling access by in order to minimise IP based cameras' attacks.

1.6 Significance of the Study

Securing IP based cameras is a critical endeavor in a fast growing world of IoT. Designing a solution that is well tested and validated will reduce security risks that come along with insecure devices among many digitally connected things. The finding from this research work will support in future research work on how to secure IP based cameras whose popularity is in the rise.

1.7 Chapter Summary

This chapter gives a background study on IP based cameras and the various vulnerabilities associated with the cameras by identifying the commonly exploited vulnerabilities that come along with IP based cameras, the problem statement, research objectives of the study, the related research questions, scope and limitations of the study and finally the significance of the study.

Chapter 2 : Literature Review

2.1 Introduction

This chapter aims at giving a detailed history of IoT. The chapter also dig deeper onto how IP based cameras work, their security in terms of threats, vulnerabilities and their risk implications. The chapter later analyses the current solutions that have been put into place to protect IP based cameras as well as identifying challenges and gaps that are yet to be addressed. Finally, the chapter presents a conceptual framework of the proposed solution.

2.2 Internet of Things

IoT describes a world where ubiquitous devices communicate with the Internet. Although the concept of Internet is not that new the aspect of IoT connecting billions of devices has seen its popular grow day-by-day (Johnston, Cox, & Scott, 2016). DataFlair Team (2018) presents the status and future prospect of IoT as shown in figure 2.1. From the figure shown, throughout the years the number of connected devices is bigger than the global population.

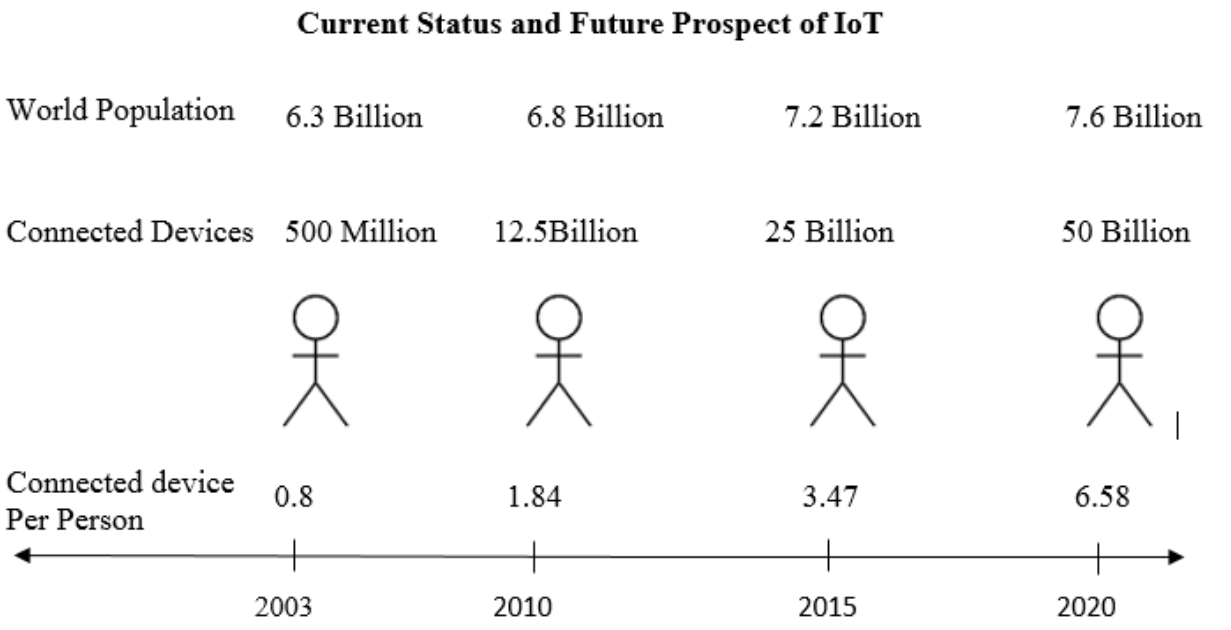


Figure 2.1 : Status and Future Prospect of IoT

Source: DataFlair Team (2018)

By IoT connecting multiple devices it facilitates man to machine and machine-to-machine interactions. Although there are IoT-ready network, compute, application and data management architectures, there is no standard way of understanding and describing these models for IoT (Cisco, 2014). Figure 2.2 shows the three-layer IoT reference model. The three layers are Application layers that comprises of people and processes, the network layer that handles all the communication for IoT and the perception layer that is comprised of all the physical devices that collect and transmit information.

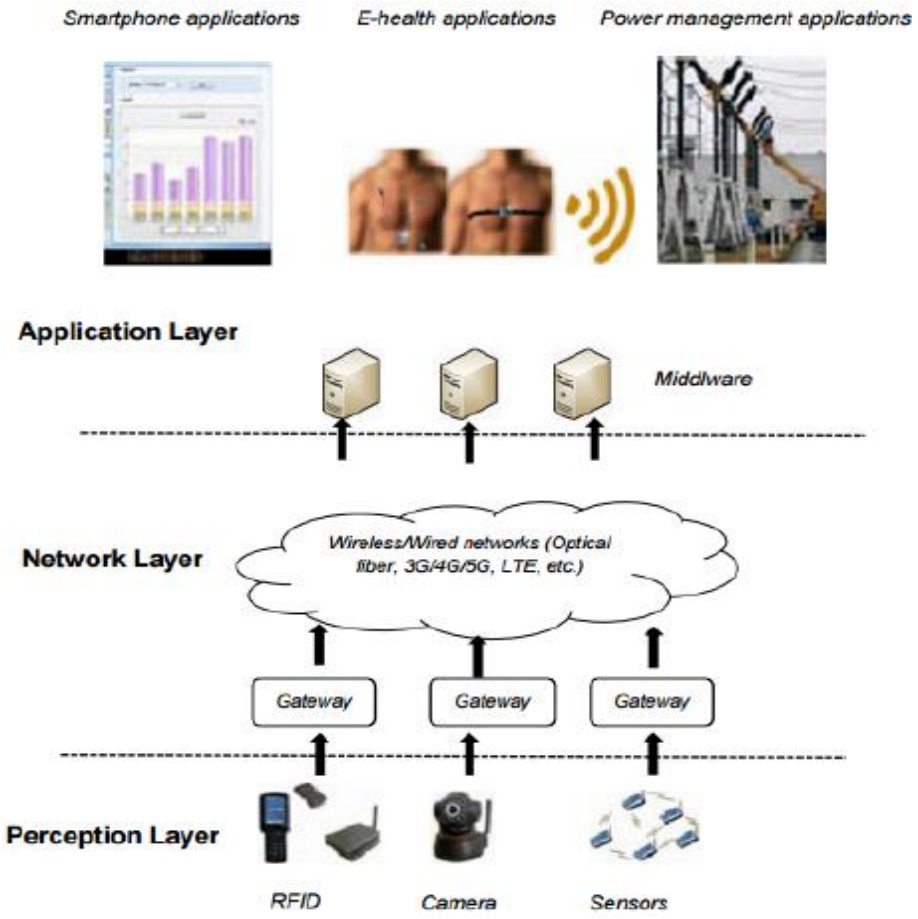


Figure 2.2 : 3 Layer IoT Reference Model

Source: Romdhani, Abdmeziem, and Tandjauol (2015)

Cisco(2014) explain more details on how the proposed IoT reference model work, figure 2.3 shows the model based on information flow. The model aims to simplify, clarify, identify, standardise and organise the IoT architecture. The model has seven levels. Each level is explained with terminology that standardise the architecture so that it can be globally acceptable. The model does not restrict the scope or locality of its components. From the model, it is clear data from in

both directions while in the control pattern, the controls and policies flow from the top level to the bottom.

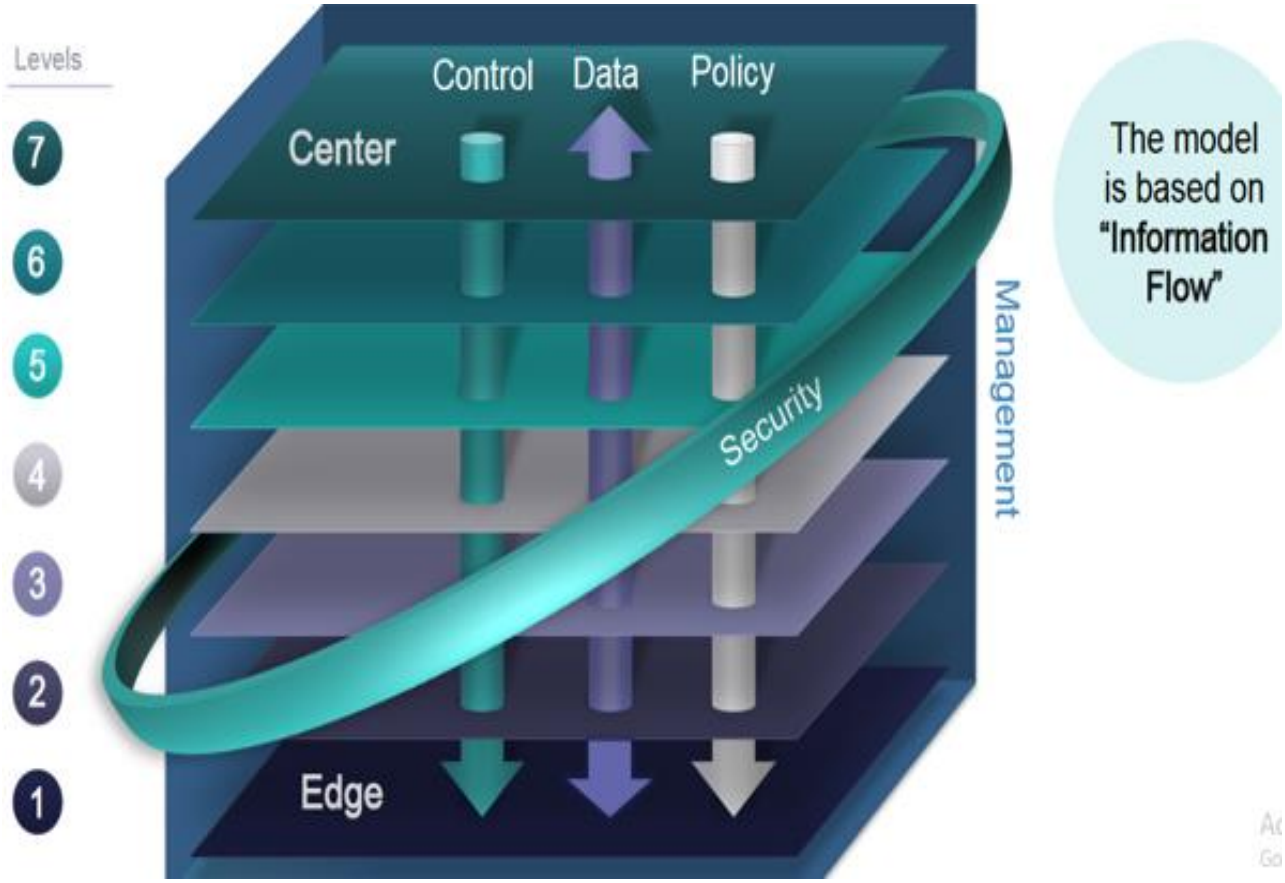


Figure 2.3 : Internet of Things Reference Model

Source: Cisco Connect (2014)

Table 2.1 shows the seven levels in the IoT reference model; they include collaboration and processes, application, data abstraction, data accumulation, edge computing, connectivity, and devices and controllers. The model describe how each level should maintain simplicity, allow scalability and ensure supportability (Cisco, 2014).

Table 2.1 : Levels of Internet of Things Reference Model

Internet of Things Reference Model		
Level	**Description**	**Details**
Level 7	Collaboration and processes (Involving people and businesses)	This level include people and processes that are empowered by application and data in IoT.
Level 6	Application (Reporting, Analytics and Control)	This level helps in interpreting information. This level interacts with level 5 and data at storage. Applications depends on vertical markets, the nature of device data and market needs.
Level 5	Data Abstraction (Aggregation and Access)	This level aims at rendering data and its storage in a form at allowing development of simple and performance-enhanced applications.
Level 4	Data Acquisition (Storage)	This level captures and store data at rest; this allows data to be used in unreal time basis.
Level 3	Edge(Fog) Computing (Data Element Analysis and Transformation)	This help in converting network data flow to information that can be suitable for storage and high level processing at level 4.
Level 2	Connectivity (Communication and Processing Unit)	The level aims at reliable and timely information transmission. Transmission include; between devices and the network, across the network and between the network and low processing at level 3.
Level 1	Devices and Controller (The ‘Things’ in IoT)	These endpoints send and receive information in IoT. As devices are to the system over time, they will become unlimited ranging from small devices like silicon chip to big devices like vehicles. The model describes the level of

		processing for these devices irrespective of their origin or form. The devices should be able to: ♦ Convert analog data to digital form ♦ Generate data ♦ Allow control and querying

Source: Cisco Connect (2014)

2.2.1 IoT Running Environment

For us to understand how IoT technology works and features it entails it is good to know where it runs on. Both the hardware and a software support IoT.

2.2.1.1 IoT Hardware

IoT hardware include a wide range of devices such as sensors, routing and bridges. A good decision for hardware determines the cost, user experience and supported applications of IoT products. IoT hardware is made up of a number of building blocks, which include Thing, Data Acquisition, Data Processing Module and Communication module. Figure 2.4 shows the different building blocks in an IoT architecture (Elicalde, 2017).

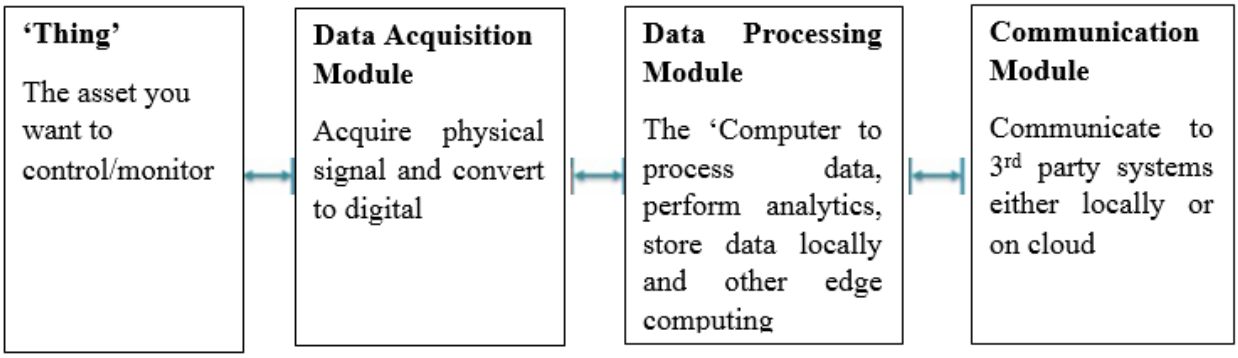


Figure 2.4 : Building Blocks of IoT Hardware

Source: Elicalde (2017)

IoT components can vary from lower power boards. For proper working of the board, the input and output of the board have to be specified by designing a circuit illustrating how the interactions should happen. Figure 2.5 and Figure 2.6 show Arduino Uno and Raspberry pi boards respectively unlike Arduino Uno which is a small single board incorporated into a main board with the aim of improving its functionality by adding new features like GPS or Interactive displays, Raspberry pi

on the other hand is a small computer that incorporate a whole web server with enough power to run windows 10 and IoT core in it (Elicalde, 2017).

Source: Elicalde (2017)

Figure 2.5 : IoT Hardware-Arduino Uno

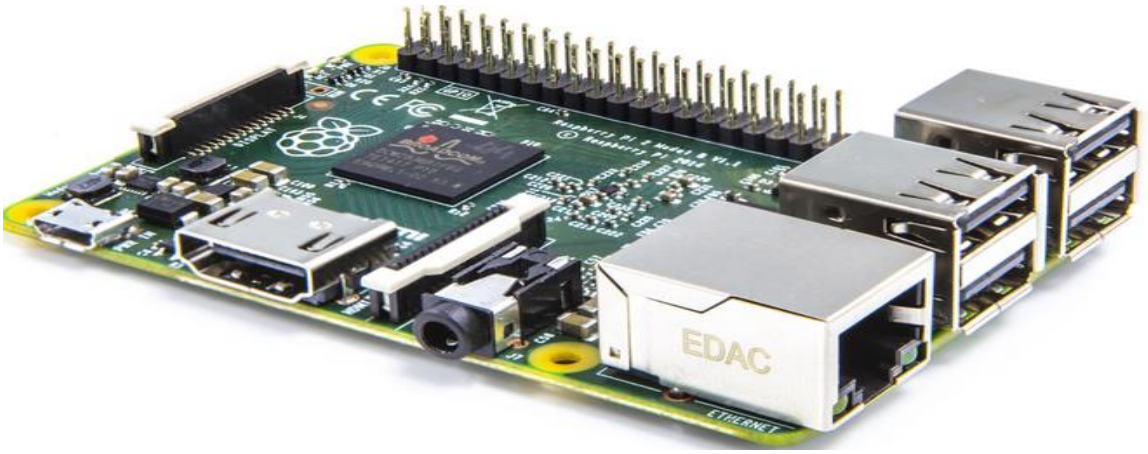


Figure 2.6 : IoT Hardware-Raspberry pi 2

Source: Elicalde (2017)

2.2.1.2 IoT Software

IoT hardware has a software that run on it. As IoT are small and have minimal processing power they run on Linux and Android Operating Systems (OS). A number of programming languages

used include c&c++, java, python among others. Figure 2.7 shows the different hardware and software used by different IoT components (DataFlair Team, 2018).

Source: DataFlair Team (2018)

Figure 2.7 : IoT Software/Hardware by Different IoT Components

2.2.2 Security in IoT

There are quite a number of proposal on how IoT could be made secure. IoT under scrutiny due to its challenges. The proposed IoT reference model proposes a number of measures that include securing each device or system, providing security for processes at all levels and secure movement and communication at each level. Figure 2.8 shows the IoT reference model with securing pervading in the whole model (Cisco, 2014). Identity management is creating a trust among people, devices and processing interacting in the system. Authentication\authorisation ensures identity of entities in IoT is verified and access controls are maintained for each identified entity. Secure storage of data in rest. Tamper resistance deals with ensuring data in motion or software cannot be tampered with. Secure communication protects any communication across the network, secure network access protects the hardware and the protocols used in IoT. Finally, secure content protect the physical devices, sensors and controllers (Cisco, 2014).

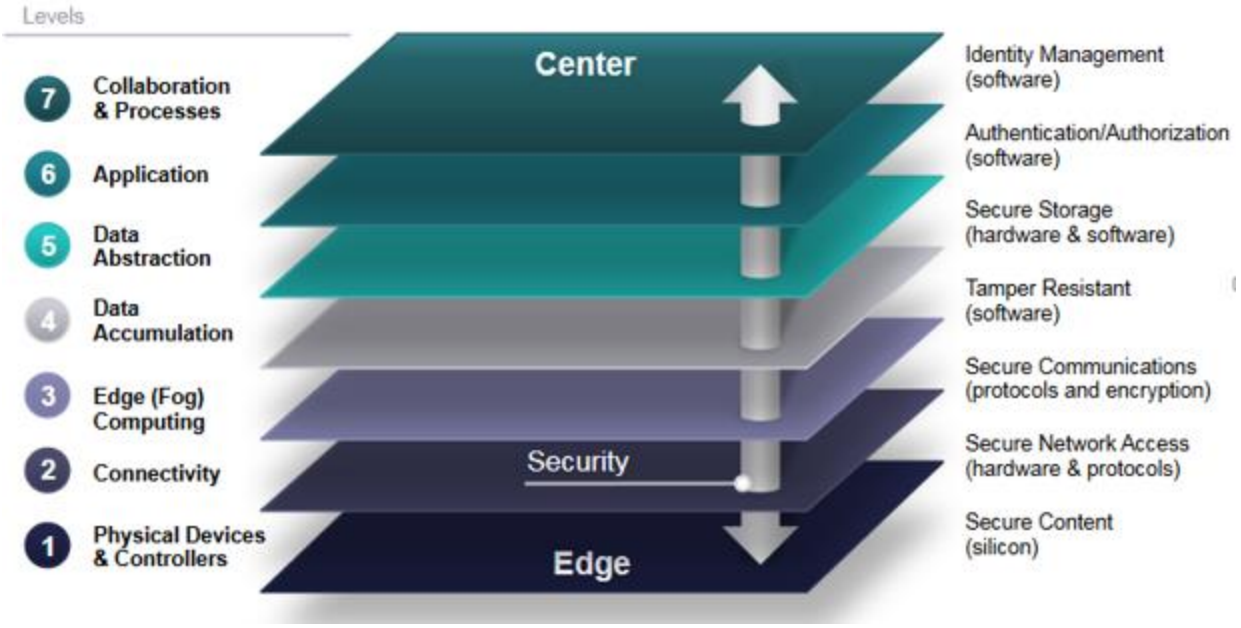


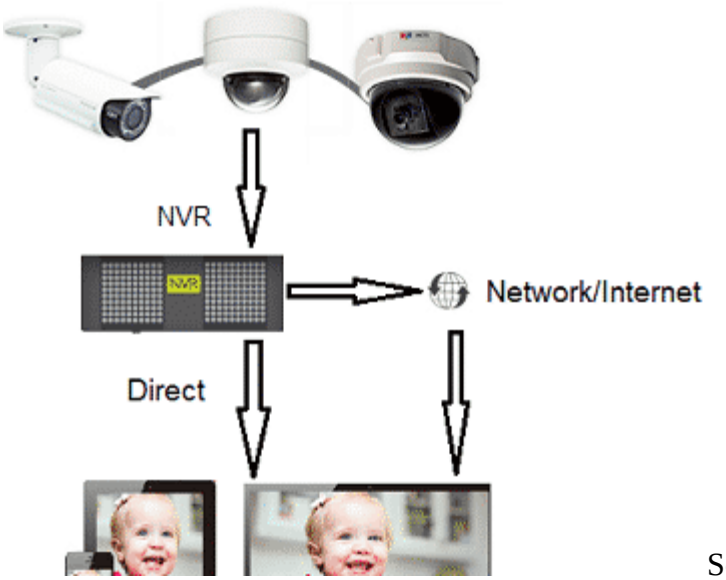
Figure 2.8 : Security in Internet of Things Reference Model

Source: Cisco (2014)

2.3 Internet Protocol (IP) Based Cameras and how they Work

IP camera is a camera that can send information in a LAN or Internet. The camera can monitor and relay important information as well as monitor for physical security. The cameras enables individual to access and monitor what is happening be it in their homes or business in real time while they are away (Brickhouse Security, 2019).

IP based cameras operate like any other cameras but they have a capability to compress the data they capture and send it over a network. The idea of giving a camera an IP address make it an independent unit that can be accessed from anywhere in the network as it have its own software and memory (Subhash, 2017). Figure 2.9 shows how a camera capture and transmit information in a network. The NVR is a software that allows viewing of feeds from different cameras. In this case, the software allows live viewing and remote access also. For remote viewing, the camera need to be assigned a static IP.



Source: Subhash (2017)

Figure 2.9 : How IP based Cameras Work

2.4 Vulnerabilities in IP based Cameras

According to Abomhara & Koien (2015) vulnerability is a weakness in a computer system or its design that allows threat actor to execute a command, access data or conduct a DOS attack. One of the key factors that predispose IP cameras to attacks is the fact that it can be accessed on the Internet. In IoT, most attacks aim other devices other than the target. In addition, IP camera is one of the IoT devices in the ecosystem, this means that there is no one entry and exit point but every other point is an entry to the ecosystem and by than it makes the cameras even more vulnerable. The extent at which these vulnerabilities are exploited is through search engine by the name Shodan that is able to mine a list of all the devices in the internet with all their vulnerabilities how the vulnerabilities can be exploited. Shodan does this by gathering a unit called banner that describe which service a device offer (Matherly, 2017).

IP based cameras have a number of vulnerabilities. Table 2.2 shows IoT vulnerabilities adapted from OWASP Top 10 Vulnerability (OWASP, 2016). The vulnerabilities applies to IP cameras as well.

Table 2.2 : IoT Vulnerabilities

Vulnerability	Attack Vectors
Insecure Interfaces	Weak credentials, capture of plain-text credentials, insecure password recovery systems, or enumerated accounts, and lack of transport encryption may be used to access data or controls.
Insufficient Authentication and Authorization	Weak passwords, insecure password recovery mechanisms, poorly protected credentials, and lack of granular access control may enable an attacker to access a particular interface.
Insecure Network Services	Vulnerable networks services may be used to attack a device or bounce an attack off of a device.
Lack of Transport Encryption/Integrity Verification	The lack of transport encryption allows an attacker to view data being passed over the network.
Privacy Concerns	Insecure interfaces, insufficient authentication, lack of transport encryption, and insecure network services all allow an attacker to access data which is improperly protected and may have been collected unnecessarily.
Insufficient Security Configurability	A lack of granular permissions, lack of encryption or password options may allow an attacker to access device data and controls. An attack (malicious or inadvertent but benign) could come from any device in an IoT system.
Insecure Software/Firmware	Update files captured through unencrypted connections may be corrupted, or an attacker may distribute a malicious update by hijacking a DNS server.
Poor Physical Security	USB ports, SD cards, and other storage means allow attackers access to device data and operating systems.

Source: OWASP (2016)

2.4.1 Factory Default password

Although manufacturers have been designing camera systems with default passwords, 51% of businesses do not have processes to change default passwords in IoT devices (IoT Security Report, 2017), at the same time 63% of security breaches involving IoT has been as a result of default or stolen passwords (Herizon Data Breach Investigation Report, 2016).

When not changed the default passwords make the devices and the whole IoT Ecosystem vulnerable to cyber-attacks as attackers can easily get the passwords and access the vulnerable device on the network giving attackers a chance to take over the device (Cybonet, n.d.). Two popular attacks involving IP cameras where default passwords were exploited are the Mirai botnet and the WannaCry attacks that left a significant portion of the internet at a standstill through launch of massive DDoS attacks (Cybersecurity on Internet of Things, 2017). To manage these vulnerabilities, it is important to have provisions to change default passwords or have rules to ensure manufacturers never design IoT devices with default passwords. Table 2.3 shows a list of IP based cameras with default usernames and passwords.

Table 2.3 : List of camera brands using default username and passwords

Camera Manufacturer	Username	Passwords	Default IP
3xLogic	admin	12345	192.0.0.64
ACTi	Admin or admin	123456	192.168.0.100
America Dynamic	admin	No set password	No default/DHCP
Avigilon Brickcom Dehus Digital watchdog	admin	admin	
Bosch	service	service	192.168.0.1
DRS DVTel	admin	1234	192.168.0.250
costar	root	root	Unknown

Source: NetVu (2017)

2.4.2 Insecure Web Design

Some cameras are designed to broadcast its feeds publicly without restricting access or checking the source of the requestor. This means anyone who has access to the IP address of the camera can view and stream anything recorded by the camera (Jay360, 2017). It is important to ensure confidentiality of data recorded the cameras is maintained by ensuring only authorised access it allowed. This can be achieved by providing strong authentication mechanisms for instance strong passwords, including system lockouts and alerts in case of suspicious authentication requests (Allgeyer, 2018).

2.4.3 Lack of Integrity Verification

IP cameras capture live data and broadcast it on the Internet. Integrity is related to either the camera configurations, the programming code or the data captured by the camera. This data is the main asset for the cameras, which attackers target to exploit. Integrity of data is considered in three states; in motion, rest and in process. Compromise of this data exposes the camera to exploitation including launching of attacks (Baker, 2016). Although IP cameras are resource constrain it is

important come up with ways of ensuring the integrity of data is maintained at any point, why because, data is the IP camera's main asset.

2.4.4 Mutable System Logs

Like any other IoT devices, cameras are meant prone to attacks therefore there is need to record what is done on the IP camera system. Benjamin (2019) notes that system logs are important as the feeds captured by IP cameras, but at the same time attackers can take advange of system logs if they are not properly monitored. Logs are able capture suspicious activities and security incidents among other things. Though these logs can be able to track what has been attackers can modify or delete the logs to to hide their actions (Benjamin, 2019). To overcome this shortcoming it is important to come up with a mechanism to ensure logs cannot be modified and in any case, individuals with rights to access the logs can only be allowed to view. This will also help in provided evidence in case there is need to perform forensic investigations.

2.5 Threats for IP based cameras

According to Vlajic (2013) any action either intentional or unintentional that could cause disclosure, alteration, loss, damage or unavailability of an asset. There are three components of a threat, the target the agents and the event. The target is the organisation that might be attacked; the agent is where the attack is originating be it intentional or unintentional, which include hackers and government agencies among other, while event is the action that pose the threat.

Despite the benefits that come along with IP cameras a number of threats are observed. The attackers are attracted to the IP cameras because of many reasons among them being:

- IP cameras operate unattended by human beings.
- Most cameras are connected through wireless network, thus it becomes easy for attackers to gain access to confidential information through eavesdropping.
- Most IP cameras have low power and resource utilization capability thus unable to support complex security mechanisms.

Table 2.4 shows the common threats in an increasingly connected world.

Table 2.4 : Common Threats in IoT

Threat	Details	Example
Hijacked devices converted into Botnets	IoT devices can be forced to join malicious botnets acting as zombies to conduct DDoS attacks.	Mirai Botnet
Shodan IOT search Engine	Shodan exposes vulnerabilities in IoT devices	Shodan exposes ensure devices on their blog
Privacy leak	Some devices may be identifiable on the internet, some sending unencrypted data.	Attackers identifying a device leaking it IP address
Insecure Devices	IoT devices use default passwords	Users fail to change default passwords
Remote Access	Some IoT can be accessed remotely	Someone monitoring you at home using your camera
Lack of updates	As manufactures churn, quite a number of devices in the market some take long to be update while others are never updated.	A camera having stayed for more than one year without being updated

Source: Guest Writer (2019)

2.6 Risk and Implications of Insecure IP Cameras

While IoT is highly beneficial and influential, it poses a number of security risks (Shea, 2019). Risk is sum of the probability that a threat will be realised and damage cost will be incurred. Generally, the risk may be grouped into; endangers of confidentiality, integrity and availability of any IP Camera component/asset. There are a number of risks on IoT devices which include; discovering devices, most IoT are not easily discovered thus hard to protect them. Patches updates, one of the biggest risk in IoT is to use unsecure or outdated software of firmware, IoT devices face quite a number of challenges when it comes to updates, and these include inaccessibility and inability to have some devices in offline mode for a long period. Lack encryption though

encryption is a secure way to communicate IoT devices are not able to take advantage of it as they have few power, storage and processing resources. Authentication, authorization, considering the huge number of IoT devices it is hard to identify these ids and determine their access rights. IoT passwords and Disruption, DDoS attacks and IoT botnets, Unhygienic routines like using default passwords can be detrimental when exploited. Securing the network, increased number of connected devices increasing the attack vector for IoT devices (Shea, 2019).

Corbin (2016) notes that as systems rely more on IoT, attacks could cause a number of damages to businesses and individuals, for instance a DOS attack could result to temporary outage of some sites causing financial losses, loss of data and breach of privacy. Though these attacks could be nuisance to users and an embarrassment to owners of the affected system their effect could be catastrophic in the physical world, this is causing a real risk to life and property (Corbin, 2016).

2.7 Securing IP Based Cameras

IP camera in an IoT Ecosystem has assets, which include its hardware, the software, the service and the data it captures (Abomhara & Koien, 2015). These assets are the ones that need to be protected. Securing the camera means protecting its hardware and the services it offers from unauthorised access from within the device or externally. This includes the resources, the data and information both in storage and in transit. IP cameras have issues with confidentiality, privacy and trust. In an IoT setup confidentiality means access of data by authorized user/object only, this means two aspect of security need to be addressed, one access control and authorization mechanism. This means IP camera need to verify if one has the right to access a service and after identification if they are permitted to receive a service, finally access control means granting and denying resources. Privacy is a sensitive issue in the era of IoT that is under research by many scholars with an aim of identifying how exchange and transfer of data over the Internet affect privacy. Trust plays an important role in ensuring secure communication between several unsecure devices in the Internet (Abomhara & Koien, 2015).

Safeguarding IP cameras fall into three categories, that is, physical security, Information security and Operational security. According to Goverment Europa (2018) physical security entails all measures put in place to protect the camera hardware against anticipated threats. Physical security for IoT entails smart lighting, connected door locks, smart meter and automotive applications. Physical attacks happen as a result of direct contact with the device System on chip or close

proximity with an aim of exploiting vulnerabilities at silicon level implementation rather than the software or the design flaws (Ashford, 2018).

Operational security is the process of creating policies and procedures as well as administrative controls to protect information about organisation capabilities and vulnerabilities. It is considered as risk management where information managers are encouraged to view data in adversary perspective in order to protect data falling into wrong hands.

Information security handles the confidentiality, which ensures data is kept private and secure both in storage and in transit, this is achieved through encryption, proper authentication and authorisation mechanism. Integrity ensure data is not modified, deleted or added and availability of services ensures the system is available to those who need its services all the time. Integrity and availability can be achieved through access control and trust management (Goverment Europa, 2018). Figure 2.10 below is a CIA architecture showing its three key principles.

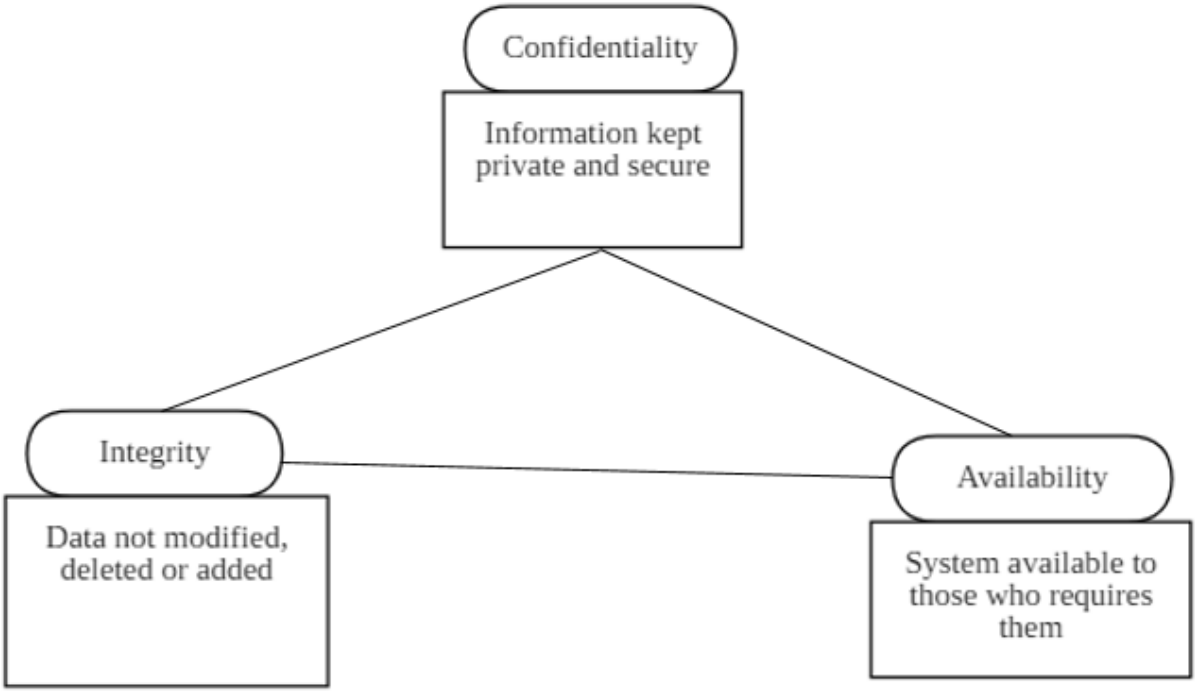


Figure 2.10 : CIA Architecture

Source: Alhassan and Adjei-Quaye (2017)

2.7.1 Blockchain and IoT Security

Security in IoT is an ongoing problem. The challenge is attributed to the many objects connected to the internet that enlarge the attack surface that require end-to-end security mitigation (Minoli & Occhiogrosso, 2018). A number of security mechanism to secure IoT have been proposed.

Blockchain plays a role in security in the context of security in depth. It helps to solve the problem of how digital entities can be passed securely from one entity to another. Blockchain is a database with chunks of blocks. Blocks store information about transactions, people participating in those transactions and information that distinguish one block from another. The blocks are always stored linearly and chronologically. Block works by ensuring once a transaction occur, the transaction is verified, after verification, the transaction is stored in a block and finally the block must be given a hash, which is a unique identifying code (Fortney, 2019). The stored transactions are then shared with all the participating users/nodes, which are tamper proof. Every users/nodes contains a similar copy of the transaction as other nodes/users in the network.

Blockchain acts as a public ledger with transactions where everyone can do the inspection but no one has control over it. The distributed database (Blockchain) records growing transaction cryptographically securing them from alteration. Figure 2.11 shows the three technologies utilized by Blockchain. Instead of having a third party validating transactions Blockchain uses a peer-to-peer network to validate transactions through a formalised predefined set of rules.

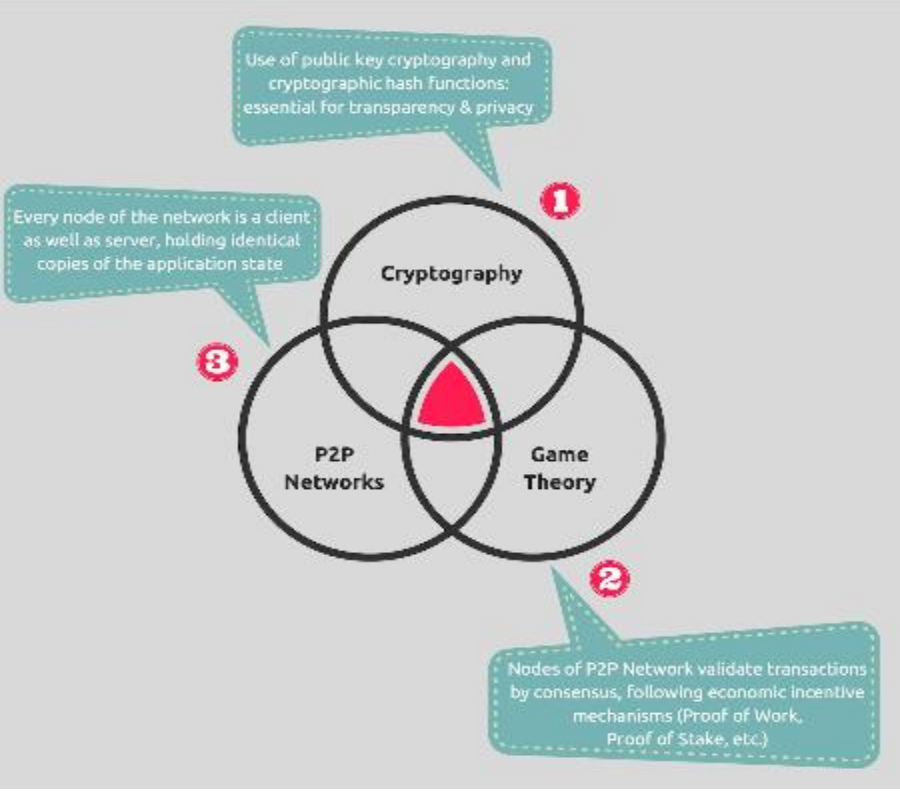


Figure 2.11 : Combination of three technologies in Blockchain

Source: Voshmgir and Kalinov (2017)

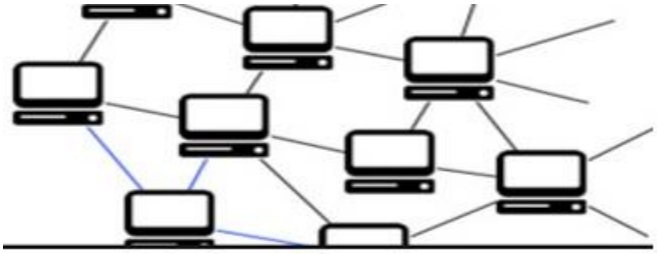
IP based cameras and other IoT devices are all about getting and transmitting large amount data through systems and linkages. This data need to be protected in storage and in communication as well as maintaining privacy for the data. On the other hand, Blockchain is an immutable and append-only ledger that stores the network state (Benet, 2014). Incorporating Blockchain will help solve a number of security challenges in IoT devices, with its capabilities including trustworthiness, decentralisation and scalability. This helps in managing the devices and data at various levels. Since Blockchain technology is based on cryptography, it provide privacy and security. Additionally Blockchain record transactions in an orderly manner providing a history of the connected devices (Mcgrath, 2019).

To provide security to the IoT devices and in particular IP based cameras, the devices are integrated with the Blockchain network while recording and transmitting data. Although Blockchain is a tested technology to provide security to IoT, its inability to large amount of data is a shortcoming. Limitation on the block size requires data to be split and reassembled off-Blockchain. Although smart contracts can be leveraged to store data and provide reassembling information it is a bit expensive. This concludes Blockchain is not the right platform to store and share huge data (Steichen, Norvill, Borja, & Shbair, 2018). This asks for integration of Blockchain with other file sharing platforms to store huge data generated by IoT devices in a convenient and efficient way (Benet, 2014).

2.7.2 Interplanetary File System (IPFS)

IPFS is a peer-to-peer distributed file system that seeks to connect all computing devices with one common file system; IPFS provides a high throughput content-addressed block storage model, with content addressed hyperlinks (Benet, 2014). IPFS acts as a hypermedia protocol that makes it possible to distribute high volumes of data with high efficiency. IPFS ensures data is not duplicated thus saving on space. It also ensures the original vision of open and flat web is realised by decentralising the storage and point of control (Rusnak, 2017). IPFS is an open project that accepts worldwide research and development contributions to enhance the system. Figure 2.12 shows how P2P network of nodes connected to the IPFS. IPFS use cryptographic hashes to identify the stored file content. IPFS identifies, verifies and transfers files relying on the cryptographic hashes of their contents (Steichen, Norvill, Borja, & Shbair, 2018).

IPFS helps in overcoming challenges experienced with hypertext transfer protocol(http), this include inefficiency as data is downloaded from one computer at a time, with IPFS the P2P network saves up to 60% bandwidth. With delete of data history, http keeps a file for one less than 100 days, unlike with http, IPFS provide resilient networks by mirroring data and finally, on network failure, IPFS ensure resilient network by enabling a persistent availability (Tabatabaei, 2018). IPFS uses the following technologies; Distributed Hash Tables (DHT) that helps determine which node has what, BitTorrent to exchange blocks, and Git to control the versions of files. When one wants to add a file into the IPFS, each file and all the blocks under it are given a unique fingerprint known as cryptographic hash. IPFS removes duplication and the history of every file. Each network note stores the content it is interested in and some indexing information to know which note stores, which file (Tabatabaei, 2018).



Source: Benet (2014)

Figure 2.12 : Network nodes in IPFS

2.7.3 Proposed solution on use of IPFS and Blockchain database to achieve IoT Security

Blockchain is immutable and append-only ledger, which makes it popular but at the same time, it is not able to store huge data. For multiple use cases, it may be more efficient to store other huge data in a secure fashion close to the secure level of Blockchain. IPFS becomes a suitable storage medium for this huge data as it allows storage of data that is immune to altering and forgery (Capital, 2018). A combination of IPFS, which has Blockchain properties and Blockchain database, would serve as the best solution to resolve security issues in IoT. This can be done by storing IPFS cryptographic hashes in Blockchain database (Shoikova, Petkova, Donchev, & Jekove, 2017).

2.7.3.1 *Raspberry pi camera*

This is a camera that runs on the raspberry pi (a single computer board). The pi is provided by raspberry pi foundation. The aim of the foundation is to bring the power of computing into the hands of people by educating them. The pi is a cheap computer that runs on linux OS. It provide the general purpose pins that allow one to control the electronic components for physical computing and explore IoT (Red Hart, 2019). For development purposes and to demonstrate how IP cameras work, the raspberry pi camera was chosen as one of the IP based cameras in IoT to capture videos. The raspberry pi housed the camera and was also used to run the developed system to demonstrate the security functionalities. Figure 2.13 shows the raspberry pi camera connected to the raspberry pi computer board.

Figure 2.13 : Raspberry pi camera connected to the raspberry pi computer board

Source: Author's work

2.7.3.2 *BigchainDB 2.0*

BigchainDB is a software that has Blockchain properties (immutability, owner controlled assets and decentralisation) and database properties (high transaction rate, low latency, indexing and querying of structured data). The database was designed first with inclusion of Blockchain properties later. Table 2.5 shows the design goals of the BigchainDB 2.0 in comparison to Blockchain and distributed databases. The database was first released in February 2016

(BigchainDB 2.0 The Blockchain Database, 2018). The design goals for this database would help in eliminating the IoT security challenges. Immutability means once data is saved it is difficult to alter it. Considering IoT generate huge data, its low latency and high transaction rate would be an advantage when incorporated in IoT as it takes few seconds to process large number of transactions and include new transactions in a committed block (BigchainDB 2.0 The Blockchain Database, 2018).

Table 2.5 : Design Goals of BigchainDB

	Typical Blockchain	Typical Distributed Database	BigchainDB
Decentralization	✓		✓
Byzantine Fault Tolerance	✓		✓
Immutability	✓		✓
Owner-Controlled Assets	✓		✓
High Transaction Rate		✓	✓
Low Latency		✓	✓
Indexing & Querying of Structured Data		✓	✓

Source: (BigchainDB 2.0 The Blockchain Database (2018)

The following steps shows the steps involved when using IPFS and Blockchain store feeds captured by IP based cameras.

Step 1: Processing of Camera feeds

This is the initial process where the camera (raspberry pi in this case) capture videos, which are considered to be the camera feeds and then process the feeds by identifying their metadata. These include the name, date, location etc.

Step2: Uploading feeds to IFPS and Asymmetric encryption

Once videos are captured and the metadata is identified, a request is sent to upload the feed to the IPFS, IPFS generates and return a unique fingerprint called cryptographic hash. IPFS uses cryptographic hash function to create the fingerprint. IPFS takes the raw content supplied and the data is run over a hash function to generate a digest that is unique to the content of the feed alone. The hash act as the identifier to the content stored in IPFS. IPFS uses multihashing in that it specifies the hashing function used to hash each file/feed. Every hash starts with letter Qm, which signifies the SHA256 algorithm and the Base58 encoding used by IPFS. IPFS not only hash files

but it breaks down large content into chunks of 256Kbs each. The file system forms a merkle directed Acyclic Graph(DAG), also known as interplanetary linked data(IPLD) of all the chunks of a given file. A hash for each chunk is generated, the chunks are then combined into a hierarchical data structure and a hash for the combined chunks is generated.

IPFS has objects that have data and links. The data is the blob of unstructured data that is equal or less than 256Kbs. The links are arrays of linked structures, that point to other objects in IPFS. Links have a name, hash and size. Name represents the name of the link; hash represents the hash of the IPFS object while size shows the cumulative size of the chunks including the links (Farmer, 2018). Figure 2.14 shows how a file/feed is converted from raw data to a base Content identifier(CID).

Once the hashes are generated they are broadcasted and can be accessed publicly. This means anyone who has the hash can access the file/feed. To avoid this asymmetric encryption is introduced. Before the feed is uploaded to the IPFS system, they are encrypted using public key encryption (public key of the owner of the file/feed). Encryption is done to ensure only parties within the network can only access feeds.

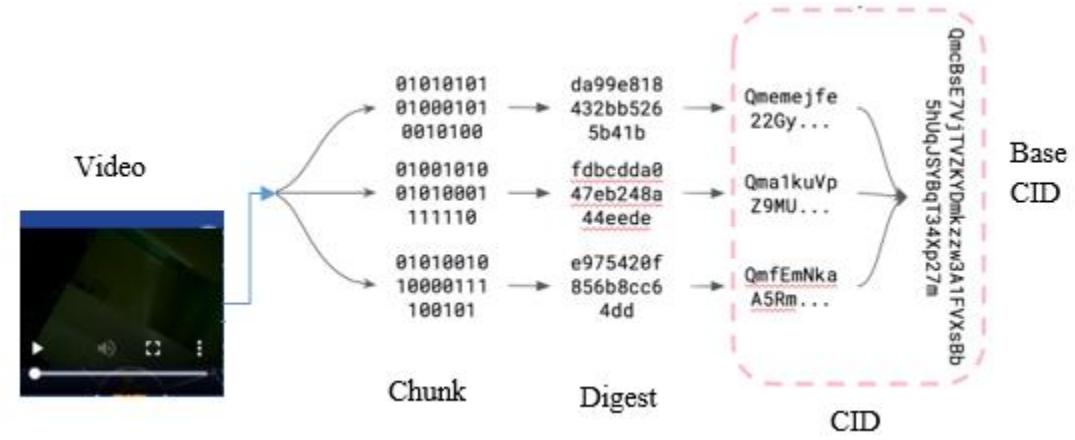


Figure 2.14 : How Large files are chunked, hashed, and organized into an IPLD (Merkle DAG object)

Source: Farmer (2018)

Step3: Storing IPFS hashes using Blockchain

Once IPFS returns the hash, a block in the BigchainDB is created that stores the hash. This ensures same content is not added twice neither can any alteration occur. Use of BigchainDB aims at preventing DDoS attacks in that on one can generate their own hashes and broadcast the same to the IPFS peers. Figure 2.15 shows how hash value are stored in the BigchainDB.

Figure 2.15 : Chains of blocks storing IPFS Hash values

Source: blockchain with hash for IPFS (n.d)

Figure 2.16 shows an overview of how feeds are uploaded to the IPFS and hashes stored in BigchainDB.

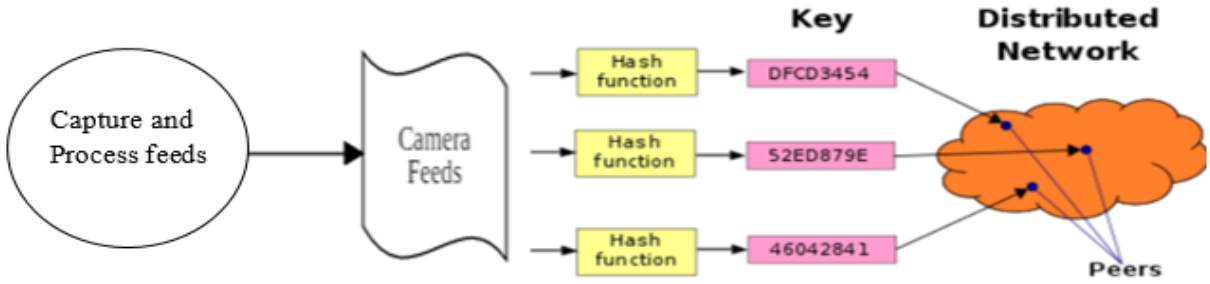


Figure 2.16 : A Peer-to-peer Network using IPFS and Blockchain

Source: Modified from (Kwatra, 2018)

Step 4: Viewing feeds

To view the feeds, the peers on the network have to decrypt the feeds using cryptographic public and private keys. This allows building of trusted chains thus achieving encryption and authentication at the same time. Figure 2.17 shows how the decryption of feeds happens.

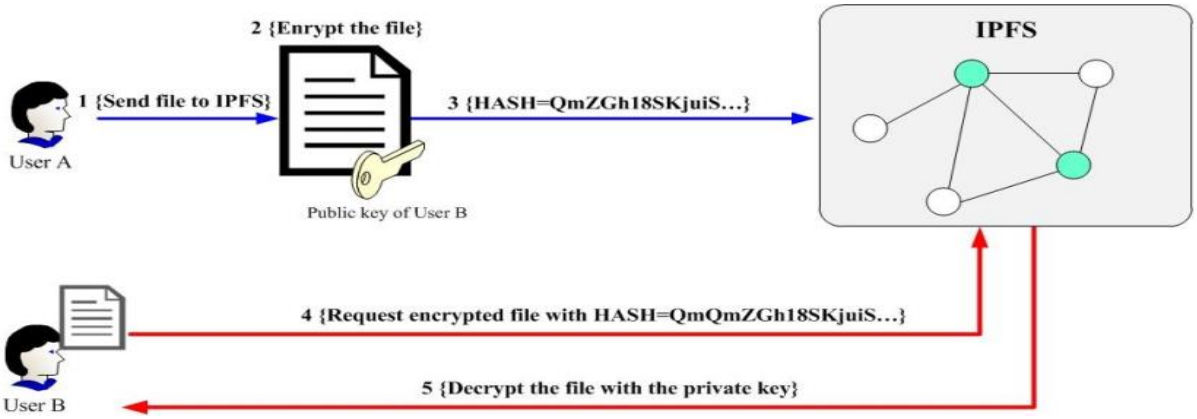


Figure 2.17 : Decrypting feeds stored in IPFS

Source: Jovovic, Sinisa, Forenbacker and Macek (2018)

2.8 Review of Existing Systems

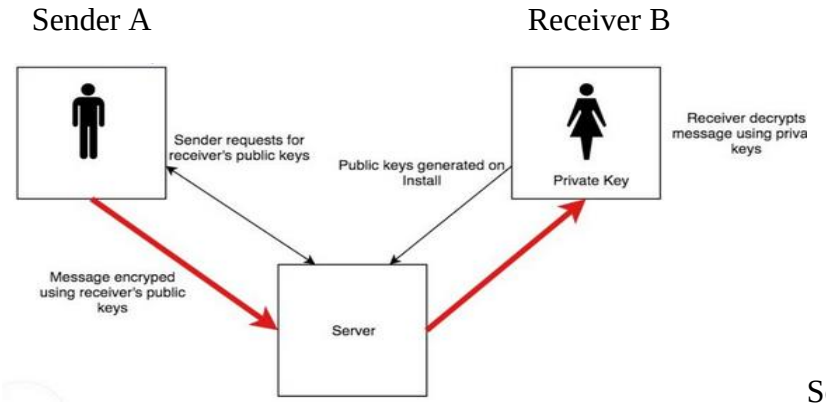
This section reviews related work to this dissertation. The work discussed here is important in providing a background to the research done by the dissertation.

2.8.1 E2EE Camera: A Camera with End-To-End Encryption

Cruz (2017) describes an IP camera that uses E2EE to preserve data privacy. This method is considered the most effective especially when dealing with video data. In principle E2EE allows sending of data in the Internet in such a way that only the sender and receiver can access it. In this case the sender is the camera while the recipient is the viewer. Unlike Advanced Encryption standard (AES) encrypted cameras E2EE ensures that even if one access the password to your system they cannot view the camera feeds. According to Lead Engineer at Cisco Systems, Michael Behringer, E2EE has the following components; identity, protocols, algorithms, secure implementation and secure operation (Behringer, 2009). Identity is used to verify the parties involved, are who they claim to be while the protocol is majorly used to setup everything needed by the E2EE including the key exchange and the algorithm. The purpose of the algorithm is to encrypt the data in such a way that it cannot be unencrypted unless the predetermined key exist. Secure implementation and operation ensure the process of E2EE is protected from cyber-attacks from malwares and viruses.

One algorithm used by E2EE is the Deffie-Hellman Key Exchange Algorithm. According to a paper published in 1976 by Whitfield Diffie and Martin Hellman the algorithm allows transmission of private key on a public channel without compromising the encryption process (Diffie & Hellman, 1976). Figure 2.18 shows how sender A and receiver B communicate through E2EE. The two parties communicate through a server. A and B calculate the private and public key from the common factors as explained in the Deffie-Hellman Algorithm. Sender A encrypts the message using the shared Public key, the message is sent through the server and receiver B decrypts the message using the private key.

While E2EE deals with encryption only, encrypted file can be accessed through other means including include man in the middle attack which can either be impersonation attack where one break integrity of message sent by legit sender or forgery attack where an attacker bypass client-server encryption (Isobe & Minematsu, 2018).



Source: Cruz (2017)

Figure 2.18 : Communication through E2EE

2.8.2 Efficiently Validating Aggregated IoT Data

Kaaniche, Jung and Gehani (2018) present a cryptographic mechanism that enables efficient aggregation of signed data. When data from different sources is sent to a single receiver, the receiver efficiently verify the authenticity and integrity of data. The three proposes a model that take advantage of aggregated data to save local computing and stoarage capabilities in order to remove redeudant data in the network flows.

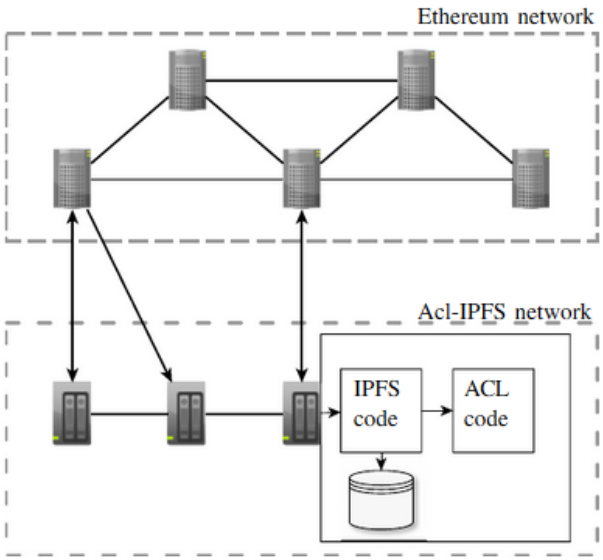
The module allows nodes to create corresponding signaturesof any subsequent aggregated data for authentication without know the private keys of all the original senders of the aggregated data. This eliminate forgery by detecting forgery before it reaches the final reciver (Kaaniche, Jung, & Gehani, 2018).

2.8.3 Blockchain-based, Decentralized Access Control for IPFS

Steichen, Norvill, Borja and Shbair (2018) present a solution called alc-IPFS, where IPFS replaces Blockchain to share large files that contain sensitive information. Taking into consideration that IPFS does not allow sharing of files with specific parties, Steichen, et al(2018) provide a modified version of IPFS that leverage smart contracts to provide access controlled file sharing. . Acl-IPFS leverages Ethereum smart contracts to handle the access control list. Through the smart contract, users can register files, and grant or revoke access to them. This modified IPFS software provides access to the smart contract and enforces the permissions stored in the access control list. Figure 2.19 shows the architecture of the acl-IPFS network. It consists of ethereum nodes that execute ethereum contracts that handle the access controls list and IPFS nodes that enforce permissions.

Each note has the modified IPFS software, local storage and permission package together known as acl code that interact with the Blockchain.

IPFS software interacts with permission package that as a result communicate with smart contracts through Blockchain node. Once a file is added IPFS creates chunks, at the same time content identifiers to the file are passed to smart contracts. The acl-IPFS allow granting of permissions to users. If a required permission have been provided a node is able to request chunks of uploaded files (Steichen, Norvill, Borja, & Shbair, 2018).



Source: Steichen, Norvill, Borja and Shbair (2018)

Figure 2.19 : Architecture of the acl-IPFS network

2.8.4 Forensic Tool for Wireless Surveillance Camera

Alshalawi and Alghamdi (2017) investigate illegal access onto a surveillance camera. The model they developed aims at securing privacy and investigating how to save user privacy. They investigate the possible illegal access (attack) on wireless surveillance camera. The development is done in two stages. One, a new monitoring scheme is built to keep the privacy of data. Two, the investigation process that plays a big role for saving users' privacy and highly secure places that use surveillance camera is facilitated. The model consists of a default Gateway namely (G), an IP Surveillance Camera, both having static IP address, and a trusted PC. G works as a monitoring unit that watches all network traffics using Wireshark. It is connected to the backup server (BS). Every period of time (t), the BS checks G that sends the capture files to the BS. G and BS are run on the same network. Thus, for security issues, the copies of all capture files are transferred to

BS that has a log file which records all copied information (date, size among other data), if BS does not receive any copy it indicates for attack. The gateway of the surveillance camera is set to G. The network communicates with the IP camera and G. The IP camera traffic passes only through G and it can read, capture, and analyze. The destination of this traffic is the trusted PC. The trusted PCs are the only devices allowed to have a legal access to the camera. Trusted PCs also are allowed to connect to each other. Figure 2.20 shows a design overview of their solution. Although this model helps in identifying attacks from unauthorised sources as well as identifying missing packets any attacks done from inside cannot be identified.

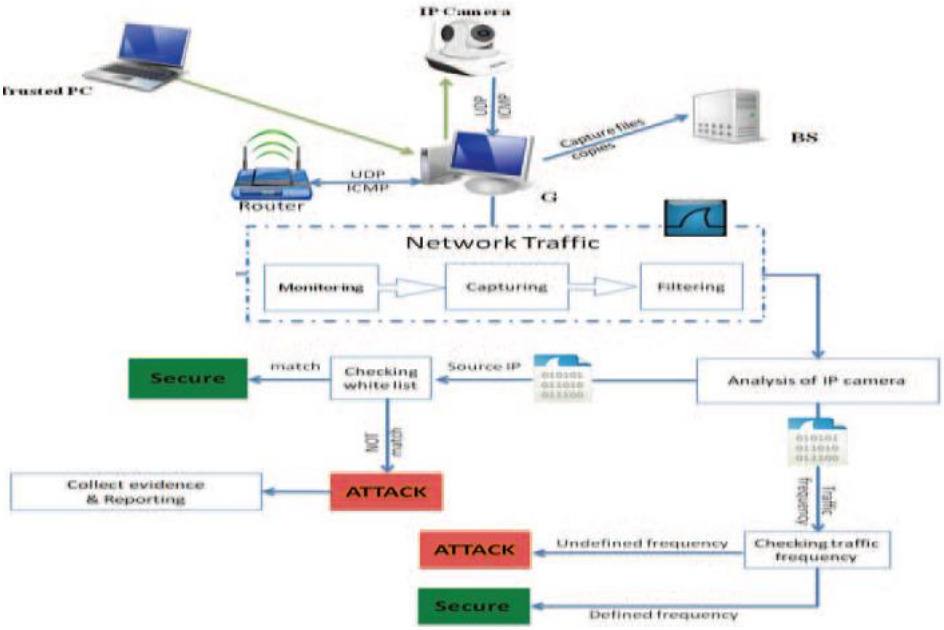


Figure 2.20 : Design Overview of Forensic tool for wireless surveillance camera

Source: Alshalawi and Alghamdi (2017)

2.9 Conceptual Framework

According to Regoniel (2015) conceptual framework is the the research understanding on how variables in the area of study interact with each other. The framework sets on stage the presentation of research questions that drive the investigation in answering the problem statement. The problem statement on the other hand gives the context and the reasons of conducting the study (Regoniel, 2015).

Figure 2.21 shows a diagram of the conceptual framework for the proposed system that was designed. The diagram is divided into inputs, processes and the outputs of the proposed system.The

inputs are the variables the system will be the user login credentials which are the variables that determines if one will given access to the camera and its feeds and which roles each user should perform, the second input is the camera feeds which is one of the resources of the camera that need to be protected as attackers are looking either to invade security or use it to cause network jam. The third input of interest is the user activities, these are stored as logs, the purpose of the logs is to identify any malicious activities done on either the cameras itself or its feeds.

The processes are divided into three levels, the first level include checking factory set passwords during account creation, and when found, and option to change credentials to more secure ones is provided, checking for malicious logins, where alerts are sent incase of positive results and giving access based on user roles. The second level deals with transfering and storing camera in a secure way in that they cannot be altered and the third level ensures logs are stored in non-editable format.

As per the formulated framework there are three categories of outputs, one enforced login credentials and alerts incase of suspicious login attempts, the second output is unaltered camera feeds and finally untampered system logs to facilitate forensic investigations.

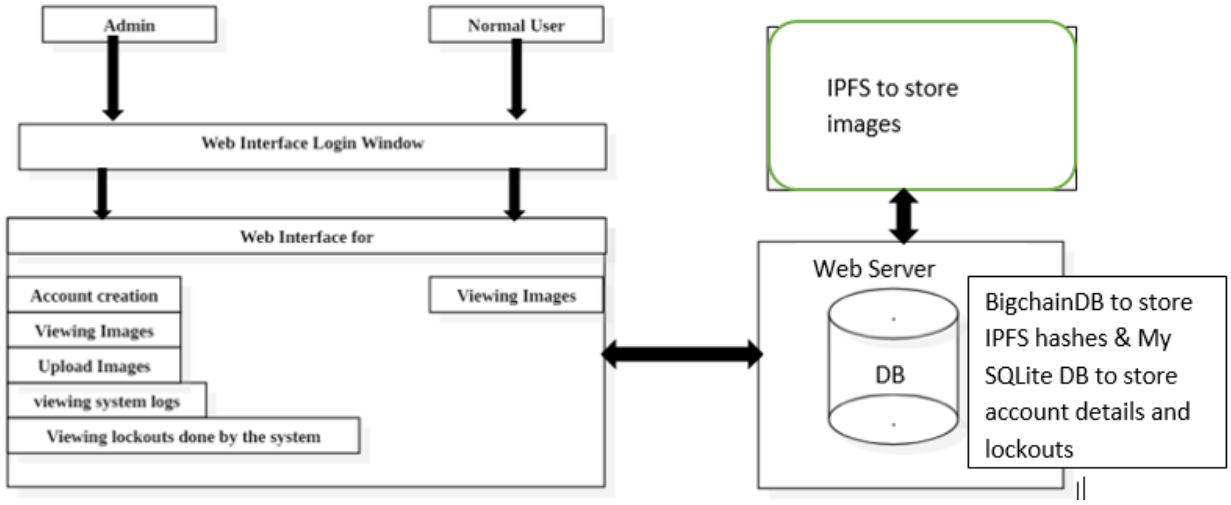


Figure 2.21 : Conceptual Framework of the proposed System

Source: Author's work

2.10 Conclusions

With the risk insecure IP cameras and IoT devices come along with, it has been an interest by all the involved parties including organisation, consumers, manufacturers and the government to see

that security in these devices is taken care of earlier enough to avoid cases where solutions are provided after an attack that leave behind unimaginable losses. From the analysed studies a number of gaps still exist, in terms of proper authentication and access control, forensics and integrity verification. Although there are some laws passed, solutions proposed, some of them have not been incorporated into the devices, the proposed system add value so as to seal gaps or enhance solutions that already exist, towards achieving secure environments for IP based cameras.

Chapter 3 : Methodology

3.1 Introduction

This chapter is aimed at describing the research approach taken for this dissertation. To answer the formulated research questions a literature review was done to understand the area of study. MoSCoW method was used to priotise the areas the dissertation would focus on as opposed to working on all the gabs identified during the literature review phase. Later a prototype of the proposed system was created, executed, tested and verified to understand if it solves the gaps that were found during the literature review phase.

3.2 System Development Methodology

Before starting off the dissertation, there was need to select a system development methodology, putting into consideration the platform and the time available for the development process.

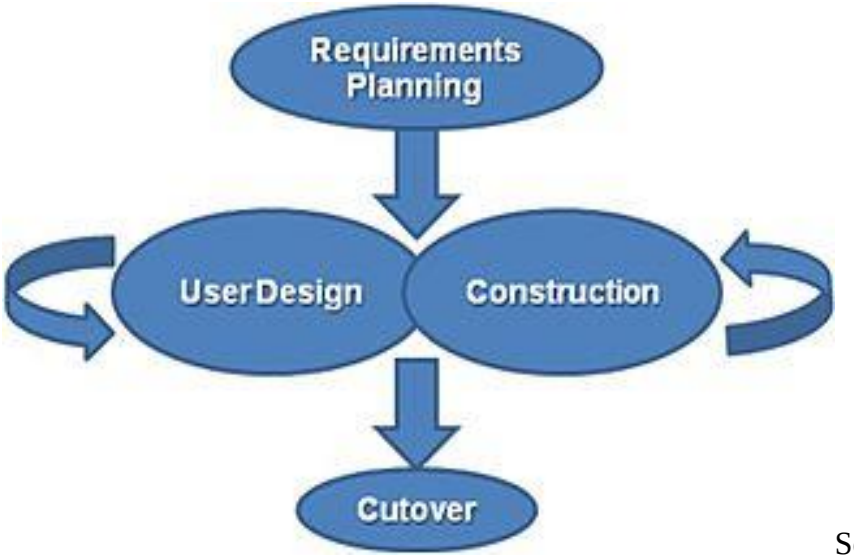
There exist a number of software development methodologies each having its own weaknesses and strengths. Rather than looking for end users of IP based cameras, to gather requirements literature review was used to identify the gaps that existed across the various IP based cameras in the market.

Rapid Application Development(RAD) system development methodology proved to be the best, as is a software development lifecycle (SDLC) model that focus on prototyping and iterative development without much dwelling on the planning process (Hirschberg, n.d). The main objectives of RAD is ensuring the following is achieved; high speed, high quality and low cost. Choosing RAD methodology meant that the intention was not to present a product at the end of the development process but rather evolve the product based on feedback on the desired end product (Gottesdiener, 1995).

RAD methodology was picked as the best model for this dissertation as it has a number of advantages; requirements were gathered by doing a literature review rather than directly engaging IP camera users, the methodology provided flexibility in changing requirements, compared to other models, the methodology allowed development and testing of prototype at an early stage, this provided a room for continuous improvement of the program under development. Additionally the reviews with my supervisor were conducted with less formality (Rouse, 2016).

3.3 Rapid Application Development Methodology

This dissertation aimed at proofing a concept by coming up with a program to show how IP based cameras can be made more secure. RAD methodology was used to guide the process of developing the program. The methodology guided the flow of developing the program from requirements gathering to testing and verifying the program significance in securing IP based cameras. The methodology has four phases, the requirement planning, user design, construction phase and finally the cutover stage as illustrated in figure 3.1.



Source: RAD Model (n.d)

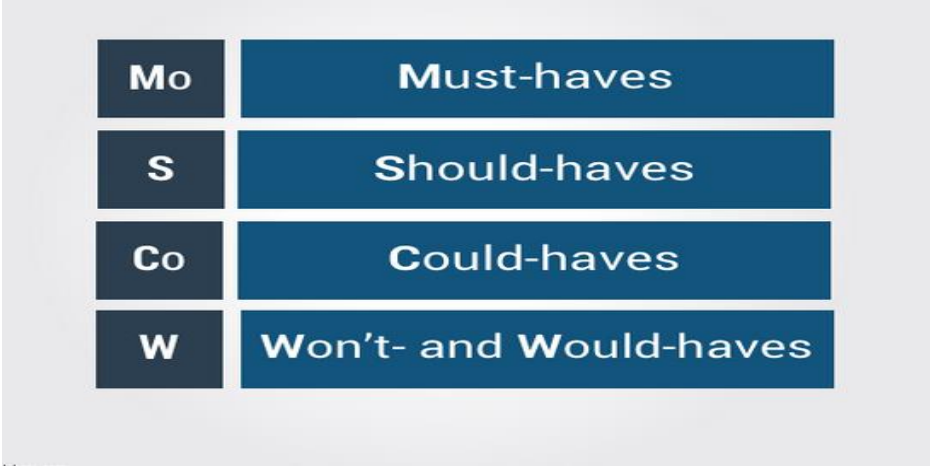
Figure 3.1 : Rapid Application Development Stages

3.3.1 Requirement Planning

This was the initial phase of the methodology, the purpose of this phase was to identify the objectives and research questions of the dissertation. After identifying the objectives, the next step was to understand the study area. To understand the study area a literature review was done, focusing on IoT Ecosystem and then narrowing down to IP based cameras. The goal of the literature review was to have a background knowledge on how IoT devices work, and IP cameras to be specific. Security being the area of focus the literature review aimed at further identifying the major threats, vulnerabilities and security risks that exist in IP based cameras and IoT devices at large, this was to answer the first research question, to answer the second research question a review of existing solutions was done, to identify if there were any gaps that existed. After the

gaps were found the next step was to identify the role the proposed system was to play by determining the system area model and the scope.

After identifying the gaps the next stage was to gather the requirements. Must, Should, Could, Would (MoSCoW) method was used as a prioritisation method to decide what to complete first, the ones that could come later and the requirements that can be exclude completely (Haughey, 2014). According to Patty Mulder a Dutch expert in management skills, the MoSCoW Method is about setting requirements by order of priority. The most important requirements need to be met first for a greater chance of success (Mulder, 2018). Figure 3.2 shows the MoSCoW method. The Must have requirements are those that without them the system is not usable, these requirements are essential in any system development. The should have requirements are additional requirements that are most desired with high priority. The requirements add value to the end product but without them the system will still work. The could have requirements have lower priority than should have requirements, they are implemented if there is time left, their absence have no negative results to the system. Finally, the would have requirements are the future wishes, most of the time the requirements are hard to realise due to time and cost.



Source: Mulder (2018)

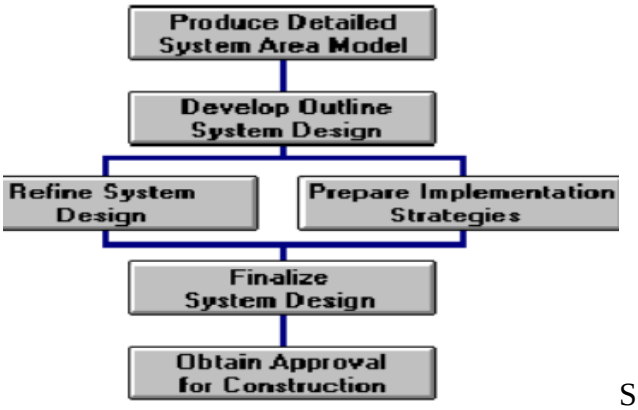
Figure 3.2 : MoSCoW Method

Once the requirements were identified the final stage of this phase was to determine the cost and timelines of the dissertation.

3.3.2 User Design

This phase more or less helped in coming up with the design of the system so as to prepare for the construction phase. Figure 3.3 illustrate the main activities that were involved in this phase adapted

from (Rapid Application Development, 2015). The activities included detailing a system area model, developing an outline system design, refining the system design, preparing the implementation strategies, finalizing the system design and finally obtaining an approval to go to the next phase of developing the system.



Source: Rapid Application Development (2015)

Figure 3.3 : Activities of RAD Design Phase

Produce Detailed System Area Model

This phase involved identifying the main activities of the system to be developed in this dissertation plus the data that was needed to come up with system area model. This assisted in identifying critical functions of the system.

Develop Outline *System Design*

Using the system area model, an outline system design was formulated by identifying the actors on the system and how they could interact with the system (Mikec, 2017).

Refine System Design

This was a confirmation and verification of developed system area model and outline system design. The requirements were confirmed, with corrections done on need bases.

Implementation Strategy

Once the design was ready an implementation plan was developed, parallel development and time box development were picked to drive the implementation process.

Finalise System Design

The design and developed prototypes were reviewed to check requirements are being met; this included more research that enabled necessary changes to the design and the prototypes.

Obtain Approval for Construction

Once the design was finalised, the phase of design was closed and rapid construction began so as to complete the system within the stipulated timeframe.

3.3.3 Construction

This phase involved creating a proof of concept. The following activities were involved:

Preparation for construction

Before the actual development it was wise to identify the best language that would be used to develop the proposed system. Python was the programming language of choice (Download the latest python version, 2017). IPFS and bigchain database (BigchainDB) was used to create a trusted distributed peer to per network. Rather than working with a physical camera this dissertation used Django to upload feeds to IPFS. Since IP camera systems uses limited resources SQLITE was the best database to implement this system, a database was created as per the data structures formulated in the design phase. At this stage black box testing was chosen to be the best system testing method of choice.

Rapid Construction

The actual development of the proposed system was done at this stage. Each function was developed depending on priority attached to it during requirement planning phase. Image validation using Blockchain was the first functionality followed by user authentication, followed by checking suspicious logins and finally storing logs in un-editable format. Black box testing was conducted on each developed function

Generation of Test Data

At this stage test data that was meant to test operational capacity of the developed system formulated.

3.3.4 Cutover

System Testing

Although each function was tested during development, all components of the developed system was tested at this stage. Black-box testing was used to test the system end to end to determine if it fulfills its functional requirements. Figure 3.4 below shows how black-box testing was done courtesy of (RAD Model, n.d). The testing focused on the input and output only, without paying much attention on how the implementation was done.

Source: RAD Model (n.d)

Figure 3.4 : Black Box Testing

Testing done at this level majorly focused on the following:

- Identifying if all functions were working as expected.
- Verifying if all named functions at the requirement planning stage were all implemented.
- Ensuring all data structure and external database accesses were functioning correctly.
- Checking if there were any behavior and performance errors.
- checking if there were any initiation and termination errors in the system

System Validation

To validate the system, test data was used to check if the system could allow image alteration. Second we checked if one logged in with factory set passwords would be allowed in. Finally the checking if the system was alerting in case one logged into the system more than 4 times with incorrect credentials and at the same time if the system could allow one to alter log activities.

3.4 Chapter Summary

This chapter gave details on the methodology used to come up with the system to proof the concept of this dissertation. Methods used to collect data helped used to determine the work done by other researchers so as to identify the gaps in place. The chapter gave more details on what was done at each stage of the rapid development methodology in relation to the focus of the dissertation.

Chapter 4 : System Design and Architecture

4.1 Introduction

System design is a crucial process in every system development as it helps in defining the architecture, modules, interfaces and data of a system towards meeting the system requirements, but before then the user requirements of the system were identified. This chapter presents the system requirements, the architecture and how the proposed system was design.

4.2 User Requirements

Requirements are the services that a software system must provide and the constraints it can work under (Software Requirements, n.d). The prototype developed was based on the attack surfaces that currently exist in IP camera and the security gaps that currently exist in these cameras. According to the requirement analysis done in Chapter 3 the requirements were divided into two, that is:

i. Functional Requirements
ii. Non-Functional Requirements

Chung, L. (n.d). defines functional requirements as the technical services the system should provide that is, how the system should respond to certain inputs and how it should behave in particular situations.

4.2.1 Functional Requirements

Results of chapture two shown IP based cameras are facing serious security challenges. This disseration focused on information security for IP based cameras to reduce the identified challenges, In efforts to reduce the gaps that currently exist, this disseration employed MoSCoW priotisation method to come up with a number of user functional requirements that when implemented would play a signifacant role in reducing vulnerabilities in IP based cameras.

From the discussion in chapter two cameras feeds are the main assets when it comes to IP based cameras, these feeds are the main target by attackers with the mission to breach confidentiality and integrity of data. with that in mind this dissertation aims at protecting the feeds from any mishandling thus creating a trusted network within which the camera feeds can be accessed is the first in the implementation list having the highest priority.

The following are the functional requirements of the proposed system; Creating of accounts and restricting use of factory default passwords, authenticating login requests based on access rights, checking and locking out login attempts exceeding four times at ago, sending alerts incase of a lockout, uploading feeds, providing an interface to view uploaded feeds, recording system logs, restricting editing of system logs and providing an interface to view the logs.

4.2.2 Non Functional Requirements

Nonfunctional requirements are the constraints on the services systems should provide (Software Requirements, n.d). The proposed system had three main non-functional requirements; performance, security and ease of use. First, the system aims at performing it functions within the least time possible, secondly, the systems aims to be simple and clear so that user is able to request for a service and navigate through the interfaces provided. Finally, the system aims at providing security to the data and services handled by the system.

4.3 System Architecture

The proposed system is comprised of three components. **Front End Web interface** which provide access to the system**, IPFS** which store and render uploaded feeds, provide IPFS file viewer and store system logs and **web server** comprising of BigchainDB which stores hashes generated by IPFS and SQLite DB user account details and recorded lockouts. The system has two main actors the system admin and normal user. The system admin has unlimited access to the system. He is able to create user accounts, view recorded lockouts receive alerts, upload feeds, view feeds and view system logs. The normal user has limited access to view camera feeds only. Figure 4.1 shows an overview of how the whole system works. The system architecture of the proposed system can be explained in terms of its inputs, processes and outputs. The three are explained in respect to each vulnerability the proposed system is meant to address

4.3.1 Inputs

The system has four main inputs, the first input is the account details captured at the point of account creation, the second input is the authentication details, these are provided at the front end web interface requesting access to the system, and the third input is the feeds that are uploaded to the IPFS. The fourth input is the system logs; the system logs all the activities done on the system.

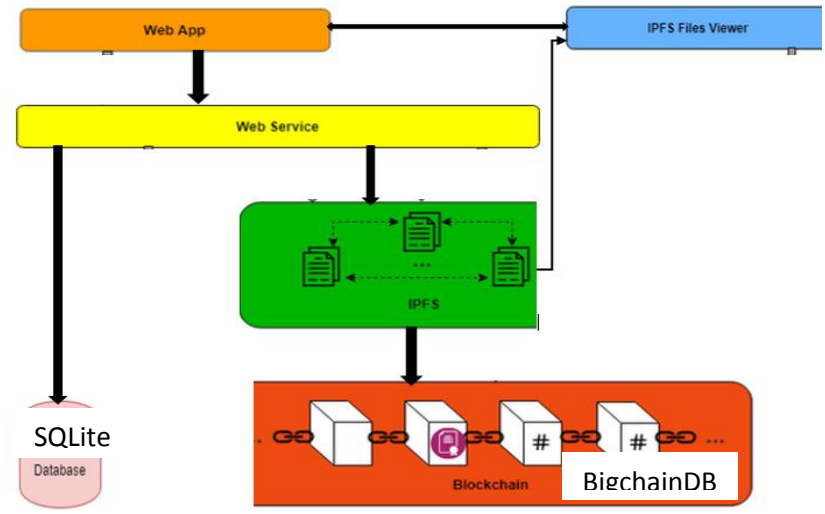


Figure 4.1 : System Architecture of the Proposed System

Source: Modified from (Processing, 2018)

4.3.2 Processes

The processes of the proposed system are classified depending of the input. The first class of process in on account creation. The system checks for factory default passwords during account creation once identified the system prompt user to enter strong password before allowing an account to be created, once the password pass the set rules of being non factory set an account is created by and storing the details in SQLite DB.

The second class of processes is on the authentication details, once a login attempts the system checks if the authentication details exist, if details exist the system checks access rights of the account requesting access after which access is allowed based on access rights, if not user is prompted to enter the correct authentication details. At the same time, the system records the number of login attempts made at a time using the same IP address and username. If more than four requests are made at a go, the system locks the account, record the lockout in SQLite DB and send an alert to the system admin informing them on the lockout done by the system.

The third class of process happens on the uploaded feeds, once the image is uploaded the system captures the metadata of the image, after that the image is encrypted and uploaded to the IPFS. The IPFS store the image and return a hash value that is stored in a BigchainDB. Once the users request to view the feeds stored in IPFS, the BigchainDB returns the hash pointing to the requested feeds in IPFS, after which the image decrypted and displayed to the user. The final process in on

system logs, the system stores the recorded system logs in the IPFS and links to those records stored in the BigchainDB.

4.3.3 Outputs

The outputs generated by the system depends on the input and processes of the system. Upon account registration and restriction on factory default passwords, the output of the system is to have strong passwords for created accounts. On authentication there are three outputs, access based on access rights, lockouts and alerts on lockouts. On the third input the feeds are stored inefficient distributed file system while maintain the integrity of the feeds, this means unaltered feeds act as the output. Finally, recorded feeds in IPFS generates immutable logs

Figure 4.2 shows the main modules and functionalities of the system as explained by processes, inputs and outputs.

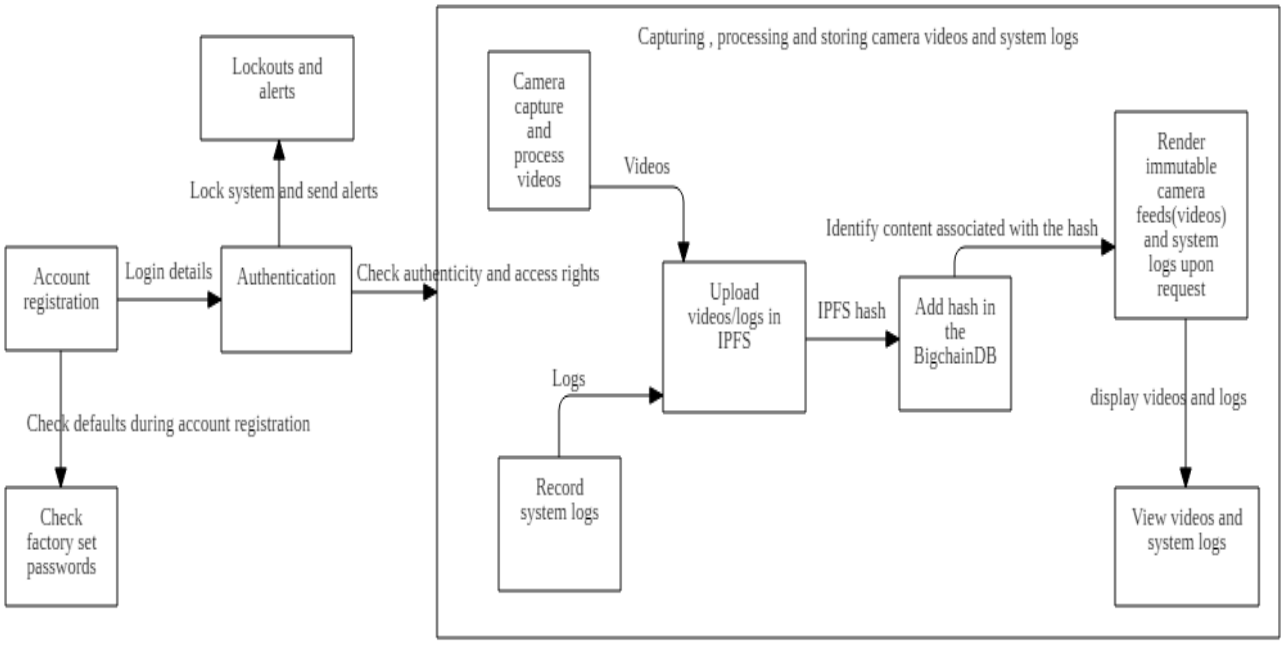


Figure 4.2: Main Modules and Functionalities of the proposed system

Source: Author's Work

4.4 System Design Tools

System design was used to depict the processes of the system, context diagram and data flow diagrams were used to depict the system area model.

4.4.1 Data Flow Diagrams (DFDs)

This dissertation used a context diagram to show the boundaries of the proposed system in terms of the entities that were meant to interact with the system. The entities were the camera and the user as shown in figure 4.3.

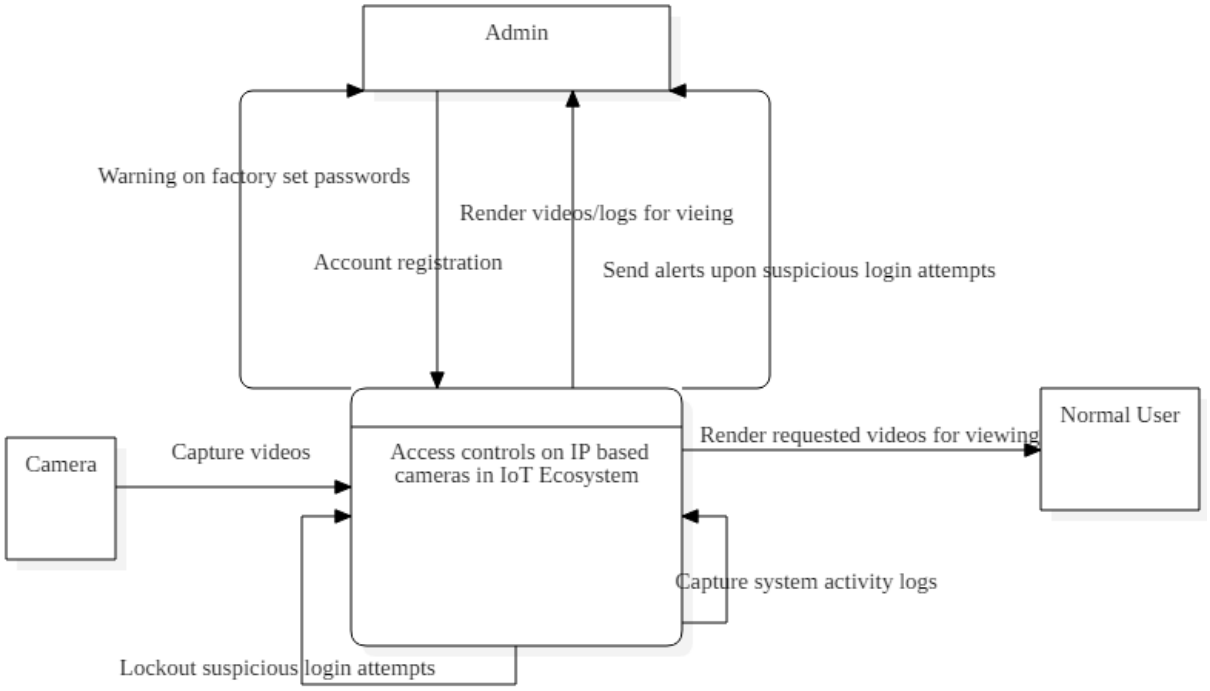


Figure 4.3 : Context Diagram of The proposed System

Source: Author's work

To show more details on how the system works, this dissertation also used a level 1 DFD to show how data moved from one entity to another. Figure 4.4 shows level one DFD for the proposed system.

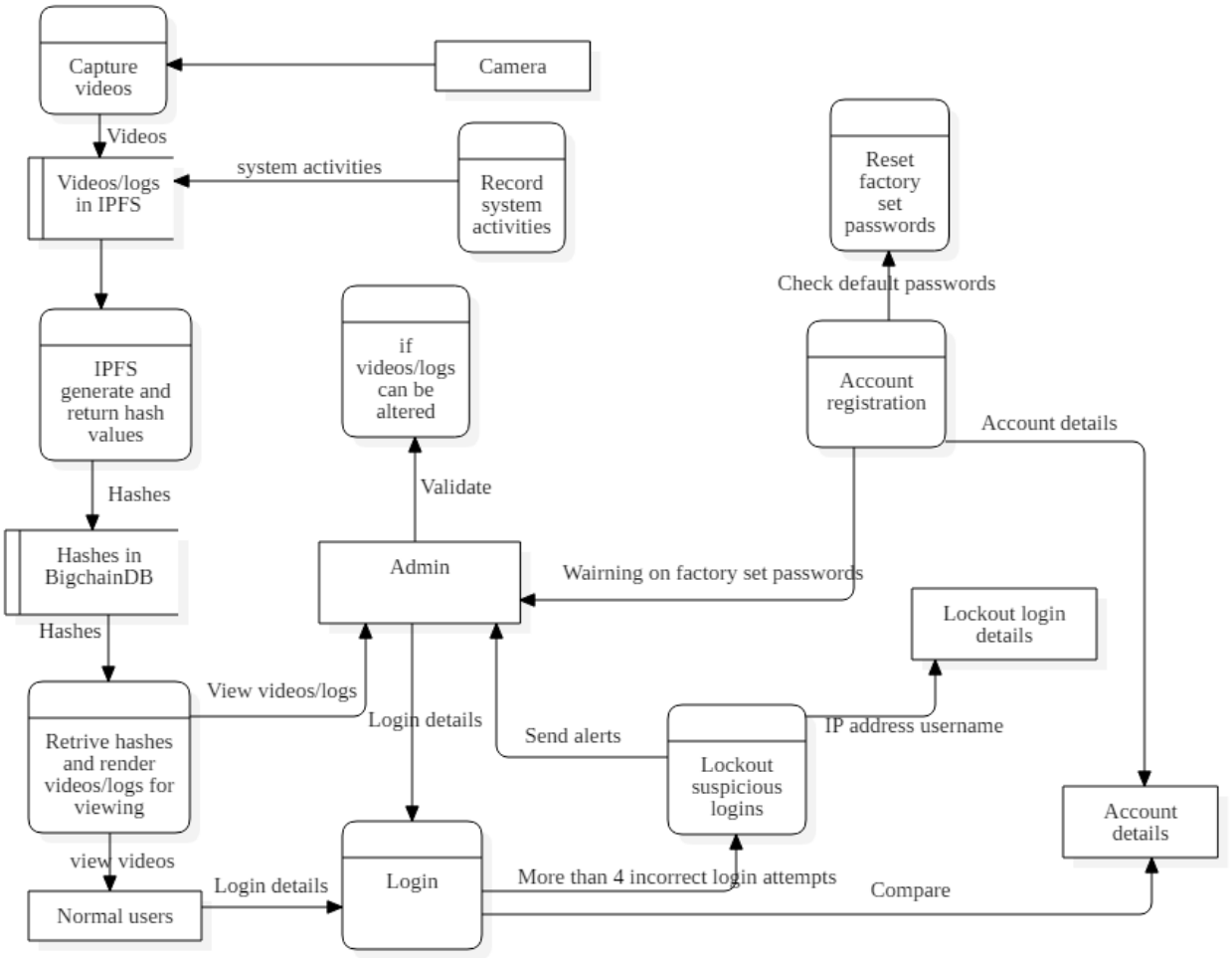


Figure 4.4 : Level 1 Data Flow Diagram of the proposed System

Source: Author's work

4.5 Sequence Diagram

A sequence diagram describes how different parts of a system interact with each other to achieve a certain function and the order in which the interactions occur when a certain use case is executed (Athuraliya, 2018). Figure 4.5 shows a sequence diagram for the proposed system.

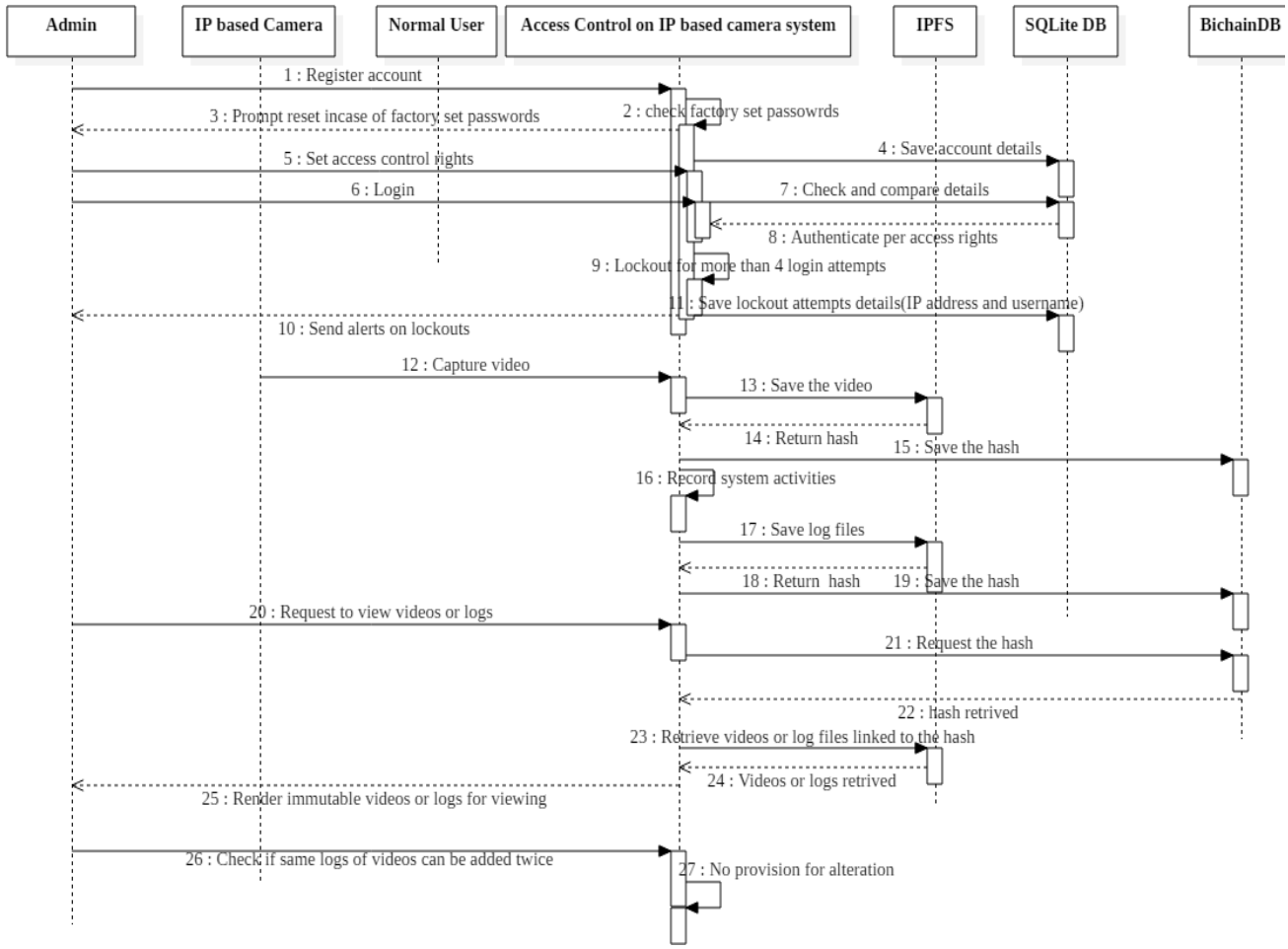


Figure 4.5 : Sequence Diagram of the Proposed System

Source: Author's work

4.6 Use cases

A use case captures what the system should do i.e. the system functional requirements. They are modeling diagrams that define interactions between external actors and the system to accomplish a certain goal (Massimo, 2011). Figure 4.6 below shows use case for different processes of the proposed system.

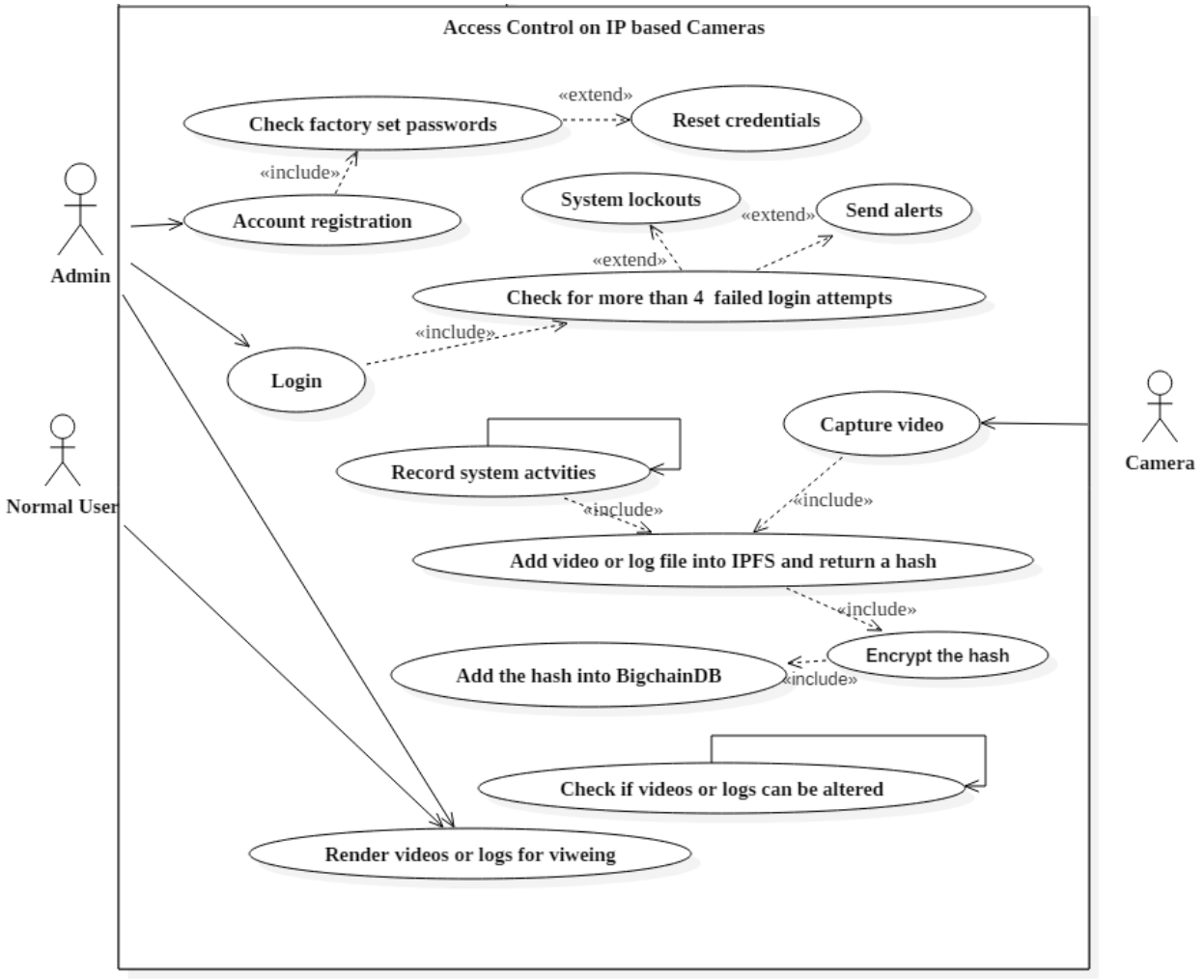


Figure 4.6 : Use Case Diagram for the Proposed System

Source: Author's work

4.6.1 Use case 1: Capture camera feeds

Figure 4.7 : Capture and process feeds use case

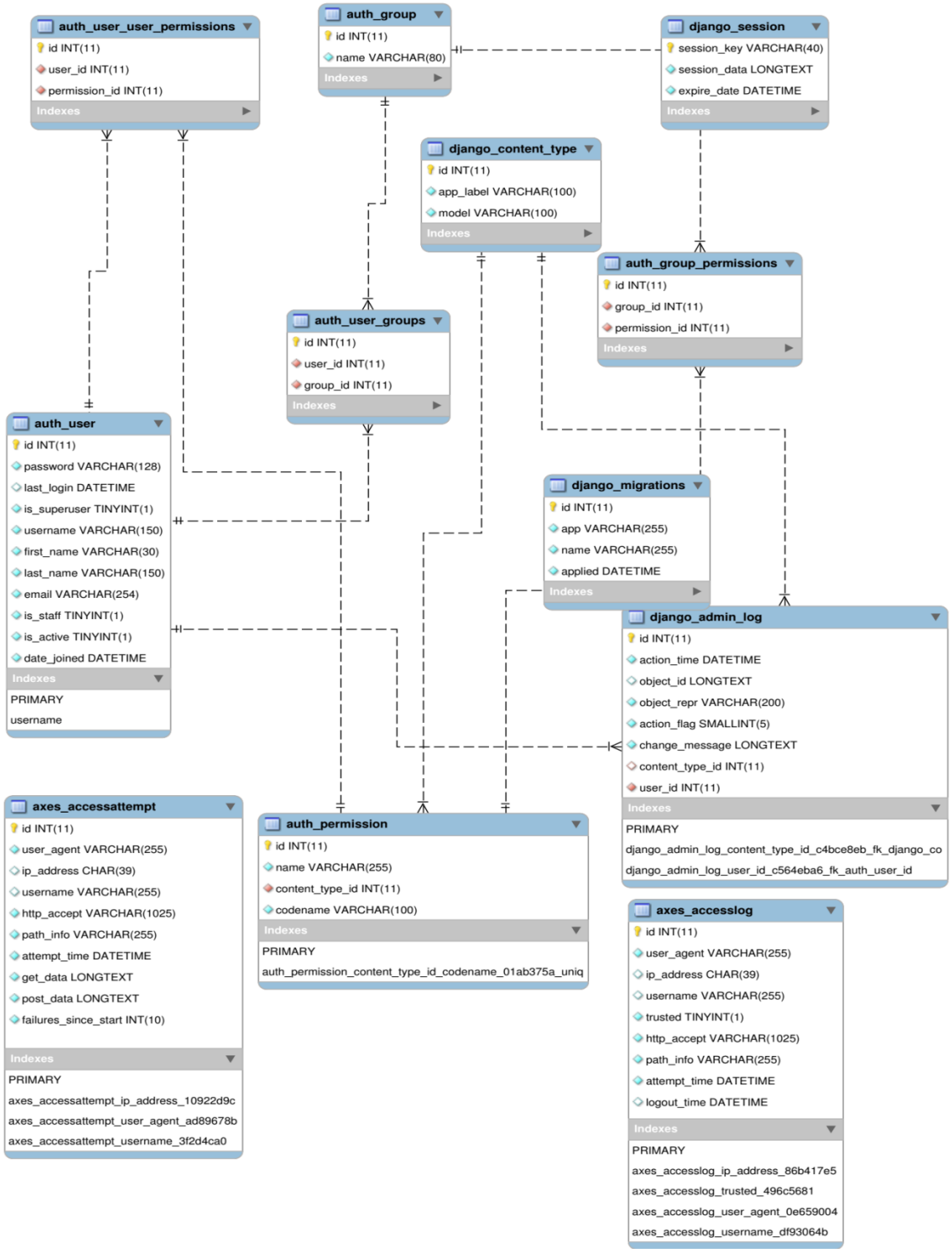
Use Case Title	Capture feeds
Primary Actors	Camera
Secondary Actor	System
Brief Description	This command is meant to capture video from physical environment before uploading the same on the IPFS network.
Include use cases	Add the captured video to IPFS network after which a hash is returned Add a hash to the BigchainDB
Pre-Conditions	Camera working and available content to be captured

Post Conditions	The videos/feeds captured are securely uploaded to the IPFS network and a hash associated with the uploaded feeds stored into the BigchainDB.
Path/flow of events	The camera capture and process the feeds, the feeds are added to the IPFS after which a hash value is returned. The hash value of the feed is encrypted using public key encryption. The encrypted hash is recorded as a block in the BigchainDB; the block contained the access path to the feeds stored in the IPFS. Any peer that want to view the feeds it decrypts the hash using the public key encryption, if they have permission the hash is decrypted and access path to the feeds is provided, thus the imaged is rendered for viewing.
Alternatives	Failed to capture video, try again

Source: Author's Work

4.7 Entity Relationship Diagram (ERD) for User Accounts used in the proposed system

The ERD is meant to show the actual the relationship between different entities of the proposed system. Figure 4.8 shows the representation of the database with the relationships between different tables.



Source: Author's work

Figure 4.8 : ERD for the user accounts database

4.8 Network Design

The network technology used by the proposed system is a P2P network. The camera is connected to the internet either through cabled network or through wireless network, the IP based camera is connected to the IPFS and BigchainDB database. Once the feed is captured, it is stored to the IPFS and the access path recorded in the BigchainDB after which broadcasting is done to all connected peers.

The system has another DB server that stores user information and system access request. Figure 5.11 shows the adopted network design for the proposed system.

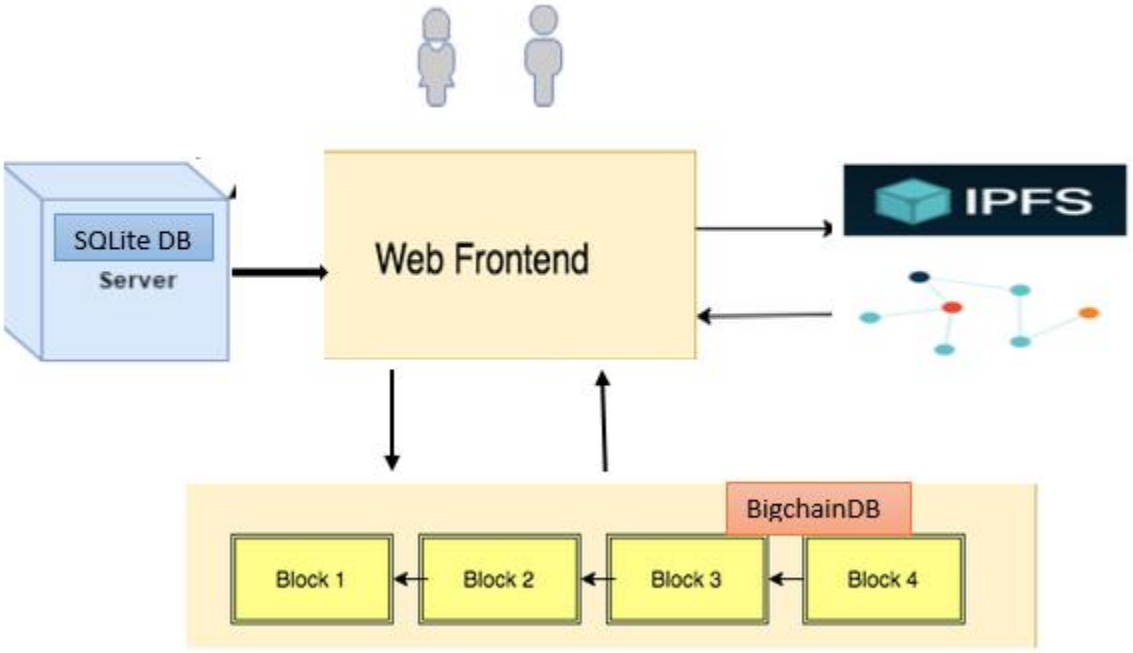


Figure 4.9 : Network design for the proposed system

Source: IPFS Structure (2018)

4.9 Security Design

The focus of this dissertation was to implement the small but important factors that are left behind when developing a system. This being the case before system was developed different parties were consulted on the common vulnerabilities and malpractices that are experienced with IP based cameras in use to try to minimise the risk of attacks in those cameras.

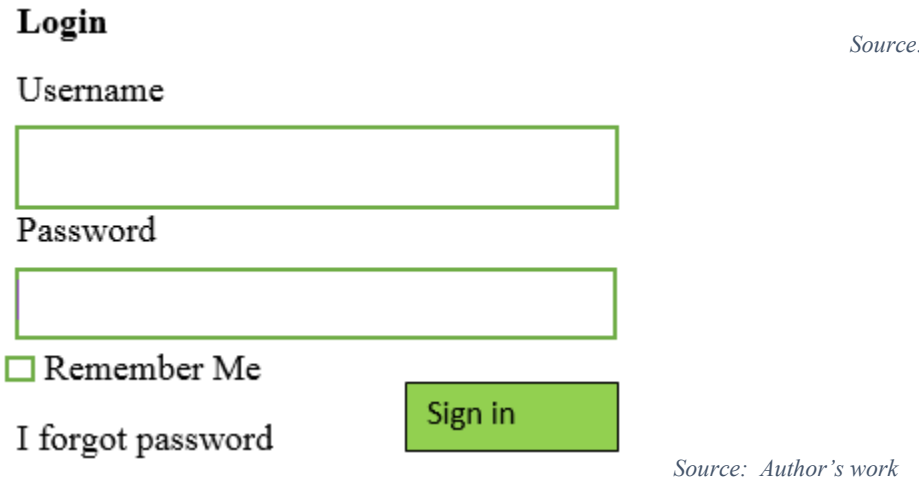
This dissertation adopted encryption of data, EAS was the encryption algorithm adopted. This ensured integrity of data in both storage and transit. IPFS and Blockchain through use of BigchainDB helped in performing integrity verification to all feeds and system logs handled by the system.

4.10 Wireframes

Wireframes shown in figure below are meant to show the skeleton backbone for the proposed system. Wireframes sketched before the implementation of the proposed system are shown below.

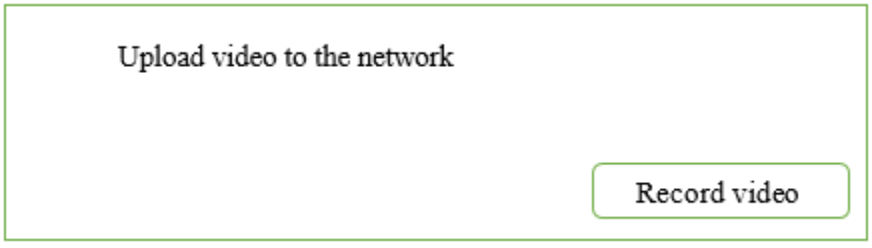
Figure 4.10 : Account Registration Wireframe

Source: Author's work



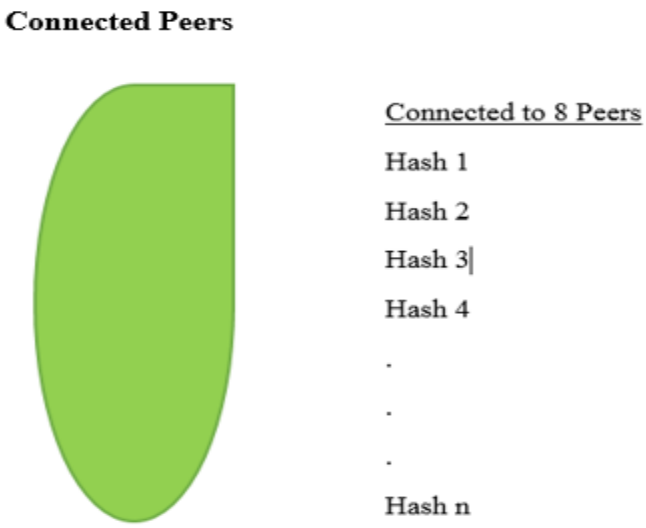
Source: Author's work

Figure 4.11 : Login Wireframe



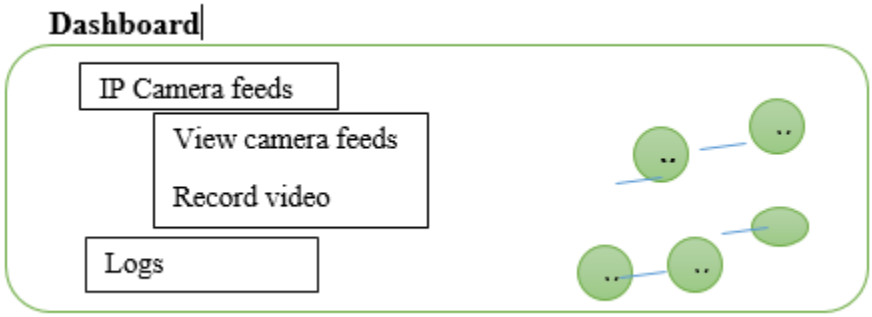
Source: Author's work

Figure 4.12 : Feed capture/simulation wireframe



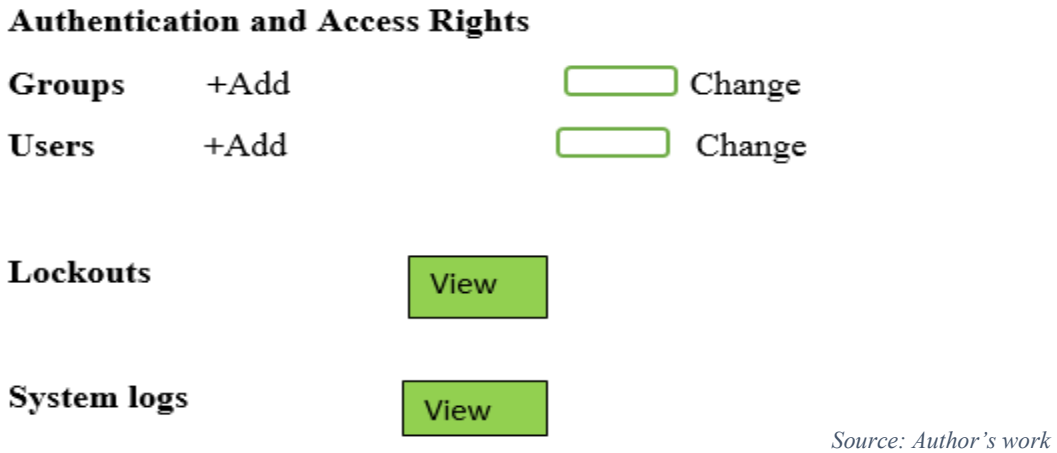
Source: Author's work

Figure 4.13 : Viewing feeds on IPFS wireframe



Source: Author's work

Figure 4.14 : Viewing feeds on IPFS wireframe



Source: Author's work

Figure 4.15 : Authentication and Access rights window Wireframe

Chapter 5 : System Implementation and Testing

5.1 Introduction

This chapter discusses the implementation of the proposed system, this include the development environments, the processes of the proposed system, the data sources, the screenshots of the processes, inputs and outputs and finally the system testing and validation.

5.2 Hardware Environment

The system need the following as the minimum hardware requirements to run the proposed system;

i. Random Access memory 2GB and above
ii. Central processing unit 2.53 GHZ
iii. Raspberry pi 3 model B V1.2
iv. Raspberry pi camera V2.1
v. Straight Ethernet cable
vi. USB cable
vii. HDMI display
viii. Mouse, keyboard and HDMI cable
ix. Micro SD card

5.3 Implementation Environment

The proposed solution is a web-based platform to demonstrate how capturing, access and broadcasting of IP based camera feeds can be made more secure. The following tools were used to develop the proposed system:

i. Ubuntu 16.04 LTS (Target Operating System)
ii. Python 3.6.7 (Backend Development tool)
iii. Interplanetary File system (Storing and accessing camera feeds)
iv. BigchainDB (Blockchain database for storing hash values returned by IPFS)
v. Bigchaindb-driver V0.6.2 (Python library to connect to the BigchainDB)
vi. Django web Framework (Developing the web interface)
vii. Python IPFS API Library (To create peer network and send http requests from IPFS client to IPFS server)

viii. SQLite (Database server for storing user account details)

ix. Raspbian stretch OS (OS for raspberry pi)

x. Picamera Library (Library for accessing the camera module in python)

xi. Ngrok software (for secure tunneling of local IP to the Internet)

xii. nodejs, npm (for installing Ngrok software globally in the local machine)

xiii. gpac (Convert videos captured by raspberry pi camera from .h264 to .MP4 format)

5.4 Software implementation Environment Setup

The best operating system for this system to run was Ubuntu 16.04 LTS, this was selected as it provided a good environment to interface with raspberry pi. Python 3.6 was chosen to develop the backend part of the system. The advantage of using python is availability of compatible libraries to the IPFS, which was the file system selected to store the camera feeds and for running raspberry pi computer board which housed the IP camera.

Before the set was done the microSD card to be used with the raspberry pi was formatted using balenaEtcher tool, it was the inserted into the raspberry pi and raspbian OS was installed in the raspberry pi. After that, the camera module connected to the raspberry pi. The Pi was powered using USB cable and then connected to a laptop using Ethernet cable. Figure 5.1 shows a hardware setup of the raspberry pi and laptop that acted as an interface instead of using HDMI display.

Figure 5.1: Hardware setup of Raspberry pi and the laptop used as the display

The next step was to set up an environment for running and executing code in raspberry pi, which was achieved by establishing SSH connection to the raspberry pi, figure 5.2 shows a command that establishes connection to the raspberrypi. On successful connection relevant python libraries and dependencies were installed, this included IPFS, BigchainDB client and setting up local repository nodes.

```
                    :~/camera_venv/lpcamera/web$ ssh pi@10.42.28
pi@10.42.28's password:
Connection to 10.42.28 closed by remote host.
Connection to 10.42.28 closed.
                    :~/camera_venv/lpcamera/web$ ssh pi@10.42.28
pi@10.42.28's password:
Linux raspberrypi 4.19.42-v7+ #1219 SMP Tue May 14 21:20:58 BST 2019 armv7l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed Jun  5 15:53:34 2019 from 10.42.0.1

SSH is enabled and the default password for the 'pi' user has not been changed.
This is a security risk - please login as the 'pi' user and type 'passwd' to set a new password.

pi@raspberrypi:~ $
```

Figure 5.2 : Bash command showing connection to the raspberry pi computer board

Setting up IPFS repository

Once the IPFS software was installed in the raspberry pi OS, a local repository was initialized and converted into a IPFS peer by running the IPFS daemon command. Figure 5.3 and 5.4 shows the commands used to initialize and start the daemon.

```
pi@raspberrypi:~/Desktop/camera_env $ cd ipcamera/
pi@raspberrypi:~/Desktop/camera_env/ipcamera $ ipfs init
initializing IPFS node at /home/pi/.ipfs
generating 2048-bit RSA keypair...done
peer identity: QmbJTkH8WXumppkjfzbk5GdR18YEWVymTVcMD6yqnnwivp
to get started, enter:

        ipfs cat /ipfs/QmS4ustL54uo8FzR9455qaxZwuMiUhyvMcX9Ba8nUH4uVv/readme

pi@raspberrypi:~/Desktop/camera_env/ipcamera $
```

Figure 5.3: IPFS repository initialization command

```
pi@raspberrypi:~ $ ipfs daemon
Initializing daemon...
go-ipfs version: 0.4.21-
Repo version: 7
System version: arm/linux
Golang version: go1.12.5
Swarm listening on /ip4/10.42.0.28/tcp/4001
Swarm listening on /ip4/127.0.0.1/tcp/4001
Swarm listening on /ip6/::1/tcp/4001
Swarm listening on /p2p-circuit
Swarm announcing /ip4/10.42.0.28/tcp/4001
Swarm announcing /ip4/127.0.0.1/tcp/4001
Swarm announcing /ip6/::1/tcp/4001
API server listening on /ip4/127.0.0.1/tcp/5001
WebUI: http://127.0.0.1:5001/webui
Gateway (readonly) server listening on /ip4/127.0.0.1/tcp/8080
Daemon is ready
```

Figure 5.4 : start IPFS daemon command to create IPFS peer in the local machine

Transferring the source code and running it on raspberry pi

The development was done independently on the Linux OS. This needed to be transferred to the raspberry pi. Since the prototype a was web based the web server was setup that controls the raspberry pi cameras and the web dashboard. Figure5.5, Figure5.6, Figure5.7, and Figure5.8, shows the different commands that were used to run the web application.

```
pi@raspberrypi: ~/Desktop/camera_env                pi@raspberrypi: ~/Desktop/camera_env/ipcamera
(camera_env) pi@raspberrypi:~/Desktop/camera_env/ipcamera $ python manage.py runserver
Performing system checks...

System check identified no issues (0 silenced).
June 05, 2019 - 18:04:15
Django version 2.1.7, using settings 'ipcamera.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

Figure 5.5 : Command for Running python Django web server

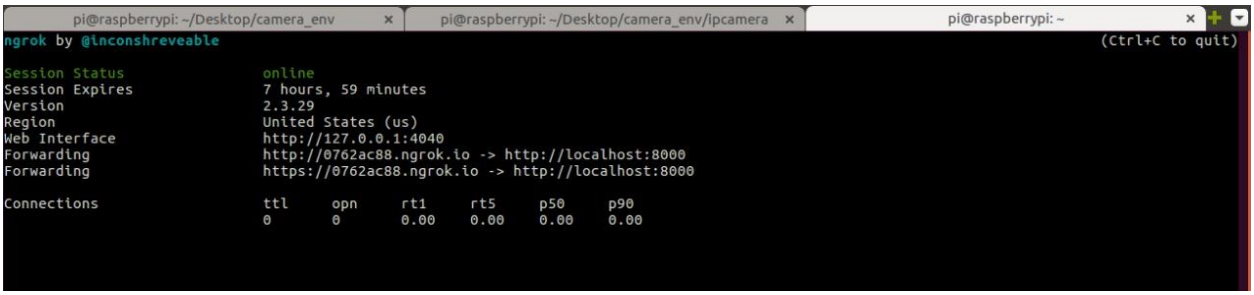


Figure 5.6 :Command to tunnel Django server local IP to the Internet

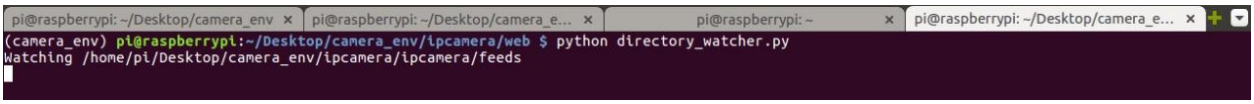


Figure 5.7 : Command to run directory watcher script that is used to check and upload captured camera feeds

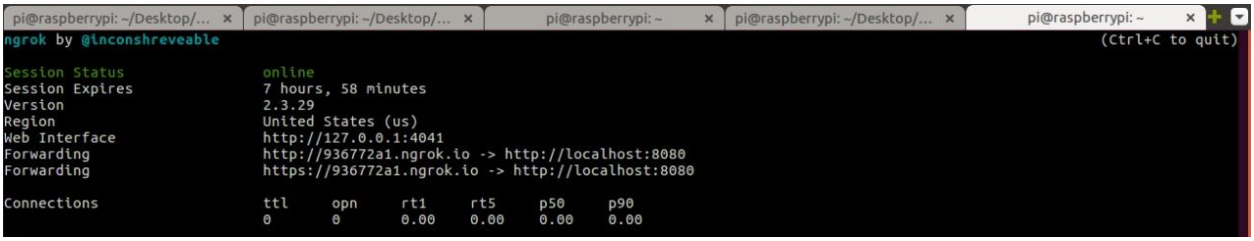


Figure 5.8 : Command to tunnel IPFS web interface

5.5 System Modules

After setting up the development the system modules were developed based on the requirements identified in chapter 4 of this dissertation.

5.5.1 Web Interface

The user interaction with the system is through a web interface. The interface has three main purposes, creating accounts, logging in to the system and viewing feeds and system logs. To access the system, you need to type the IP address. Figure 5.9 and 5.10 shows the login page and the home page respectively.

Figure 5.9 : Login Page

Figure 5.10 : Homepage of the proposed system

5.5.2 Using IPFS and Blockchain to store feeds

This is the main functionality of the system, with an aim of demonstrating how cameras feeds can be captured, stored and broadcasted with no alteration or unauthorised access, this feature has to parts, the process of capturing videos and uploading the videos in the IPFS and viewing the uploaded videos/feeds using IPFS and Blockchain technology, a major part of this functionality happens on the backend of the system with front providing an interface to capture a video, view IPFS hashes and view uploaded videos/feeds.

Capturing videos and Uploading them to IPFS

This dissertation used raspberry pi camera to capture videos that were uploaded to the IPFS system. Under IPcamera feeds, once you click on capture video, the window in Figure 5.11 is displayed. Once you click record video, the system activates the connected raspberry pi camera and capture a video of length 10ms as specified by the system. Once the specified time elapsed the video is uploaded into the IPFS and a hash of the uploaded video is displayed on the screen as showed in Figure5.12.

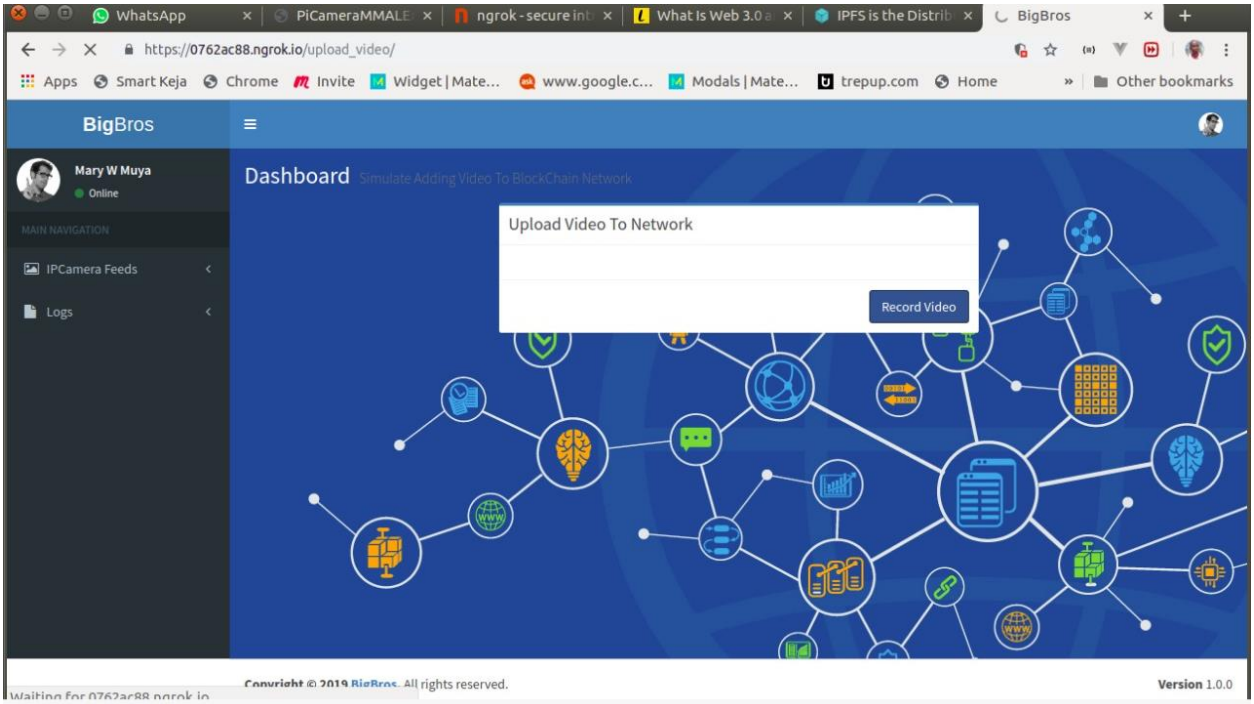


Figure 5.11 : System window to capture and upload video to IPFS

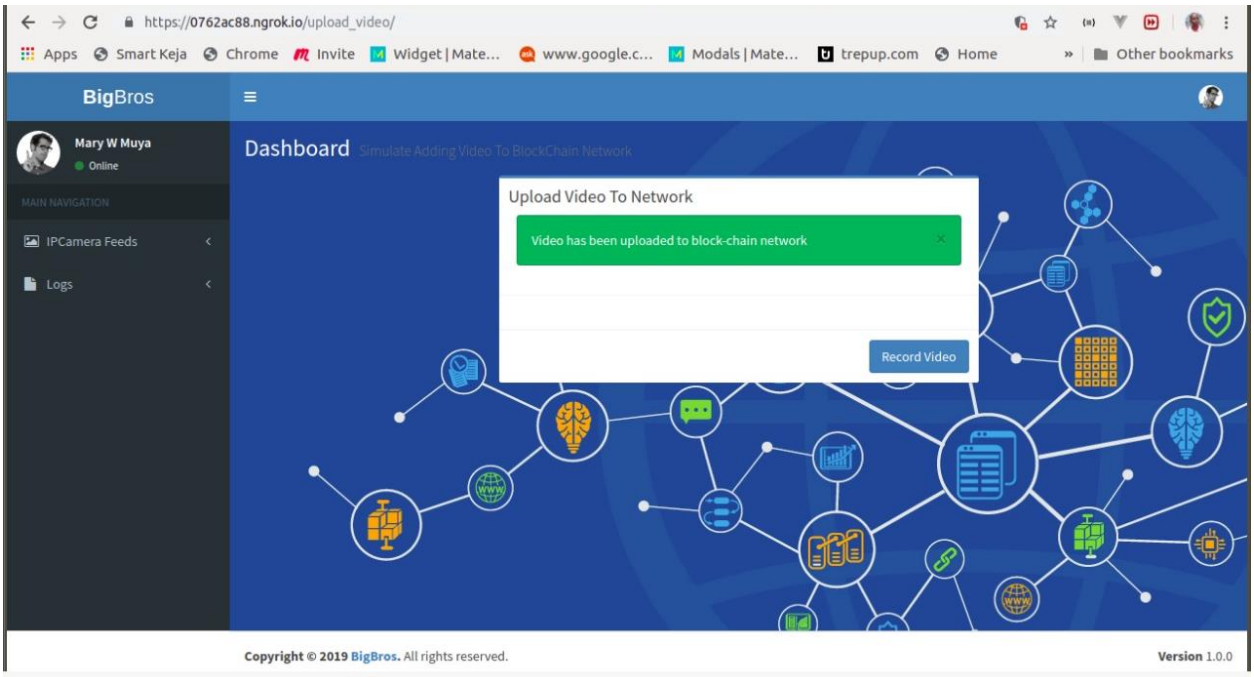


Figure 5.12 : Hash returned by IPFS after uploading a video captured by raspberry pi camera

Once the IPFS hashes are returned a block is created on the BigchainDB, the block has the index, timestamp, hash value from IPFS and hash. The block has also a hash value for the previous video that was uploaded in IPFS.

Viewing camera videos/feeds from IPFS

Once a peer in the network want to access the feeds, it requests the access path in the BigchainDB. The videos/feeds are guaranteed no alteration as they are stored in an immutable format. Only peers within that network are allowed to view the videos/feeds. The web interface provide access to the stored feeds. Access to the web interface is allowed after a successful authentication process. Figure 5.13 shows a simulated list of peers on the IPFS network while Figure 5.14 shows a list of videos/feeds displayed on request.

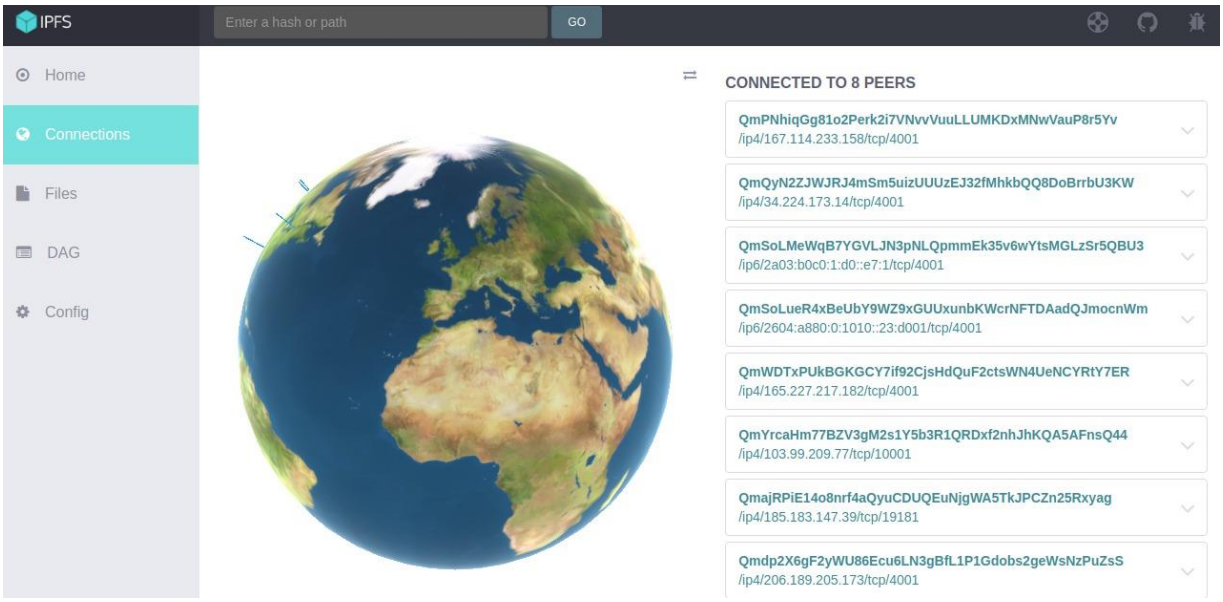


Figure 5.13 : Connected peers to the IPFS network

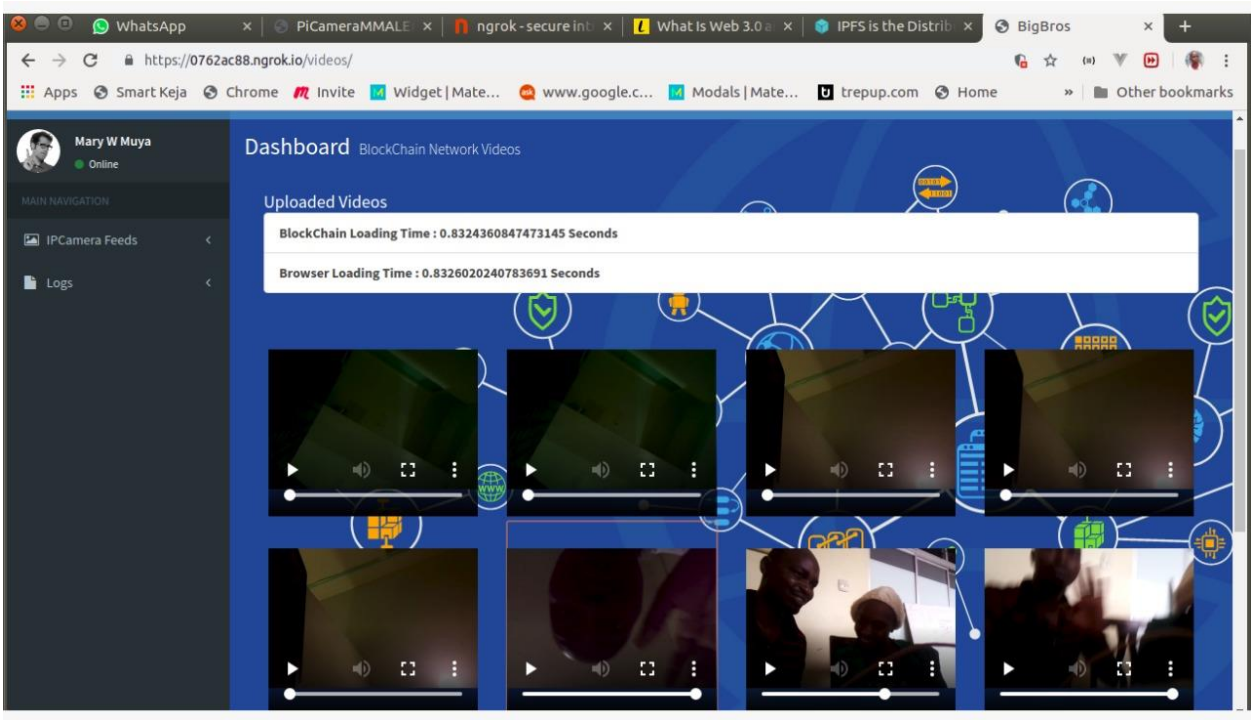


Figure 5.14 : A list of uploaded videos to the IPFS network.

5.5.3 Account creation, User roles and Restricting factory set passwords

The proposed system has a requirement to restrict factory set credentials are well as separating different user roles in the IP based camera system. The system is able to group users depending on the roles they should perform on the system. Figure 5.15 shows an interface where Admin can change access right for different group of users to access the system.

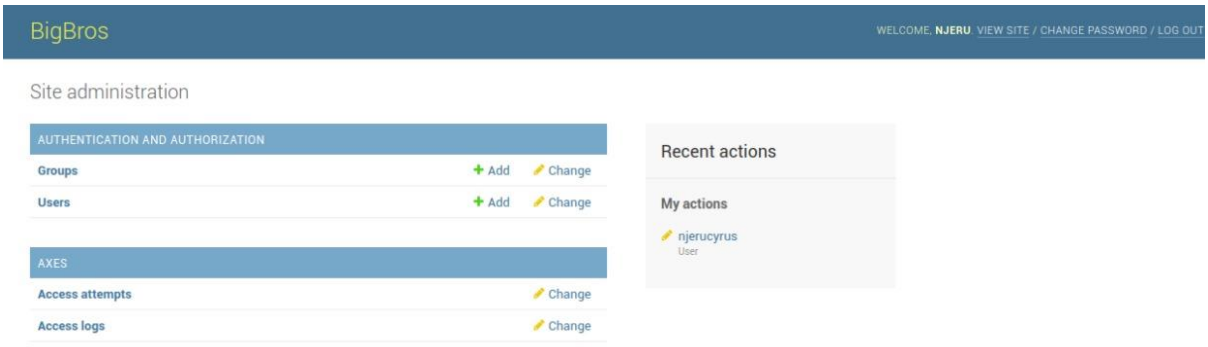


Figure 5.15 : Page to change Access rights

To achieve the security configuration aspect where factory set passwords are avoided the system checks common passwords during account creation or password change stage. The system is trained to check default passwords that have been used by manufacturers before or any common

passwords for instance 123456, root and admin, these default passwords were identified in chapter 2. In this case when one enters and confirm the login password, the system compares this to commonly known credentials, if this is detected, the password is reset and a prompt to enter a strong password is provided to the user. Figure 5.16 shows a warning issued by the system in case of detection of common password during account creation

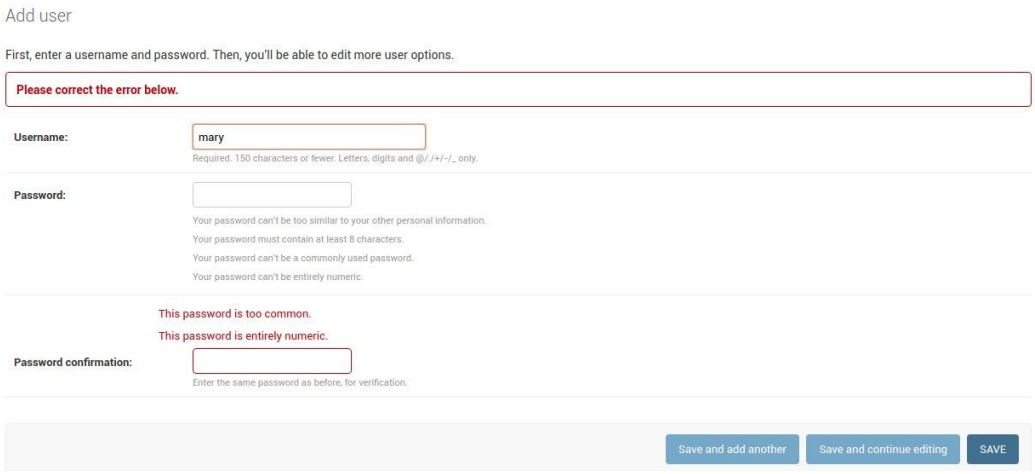


Figure 5.16 : System warning upon detection of common password during account creation

5.5.4 Locking out suspicious login attempts and sending alerts

To minimise brute force attacks where attacker takes advantage of guessing password combination until a possible guess is found the system implement a functionality to track login attempts at a go. The system limits such requests to 4. Incase request are more than the limit number a combination of IP address and username is lockout out from access the system. In addition, an email notification is sent to the system admin to notify possibility of unauthorized system access requests. Figure 6.9 and 5.18 shows lockout message and email notification respectively.

Figure 5.17 : Account lockout message upon more than 4 login attempts

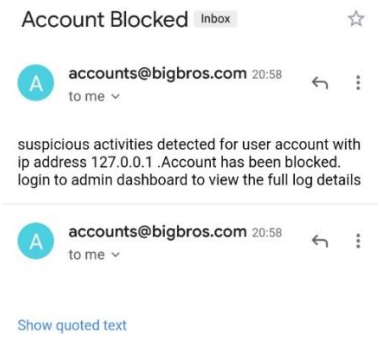


Figure 5.18 : Email Notification to System Admin upon lockout

5.5.5 Storing system logs in immutable format

To support forensic investigation, the dissertation was to come up with a functionality that ensures system logs cannot be interfered with, with was achieved by storing logs in an immutable format through use of IPFS and Blockchain. Figure 5.19 shows a web interface to view feeds on anything done on the SQLite database that details with creation of user accounts and access controls. Figure 5.20 shows a list of hashes to the log files, while Figure 5.21 shows the logs in each files stored non-editable format. The system works in such a way if any activity s done in the system a log file is created. If another activity is done it is appended to a list of existing system activities and a new log file is created.

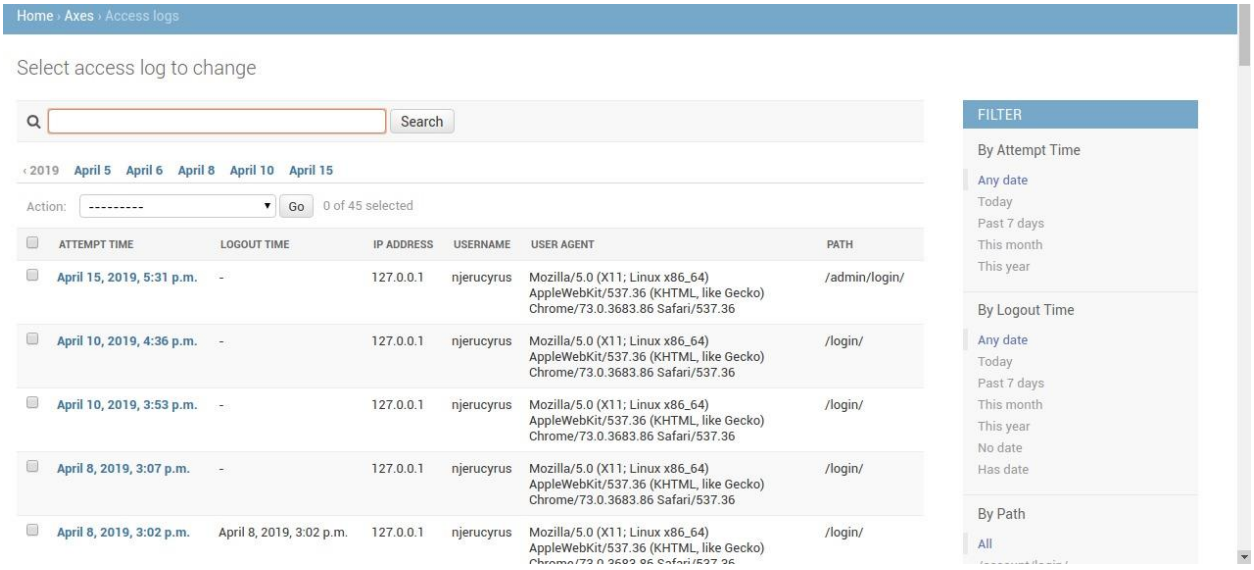


Figure 5.19 : System logs in an immutable format

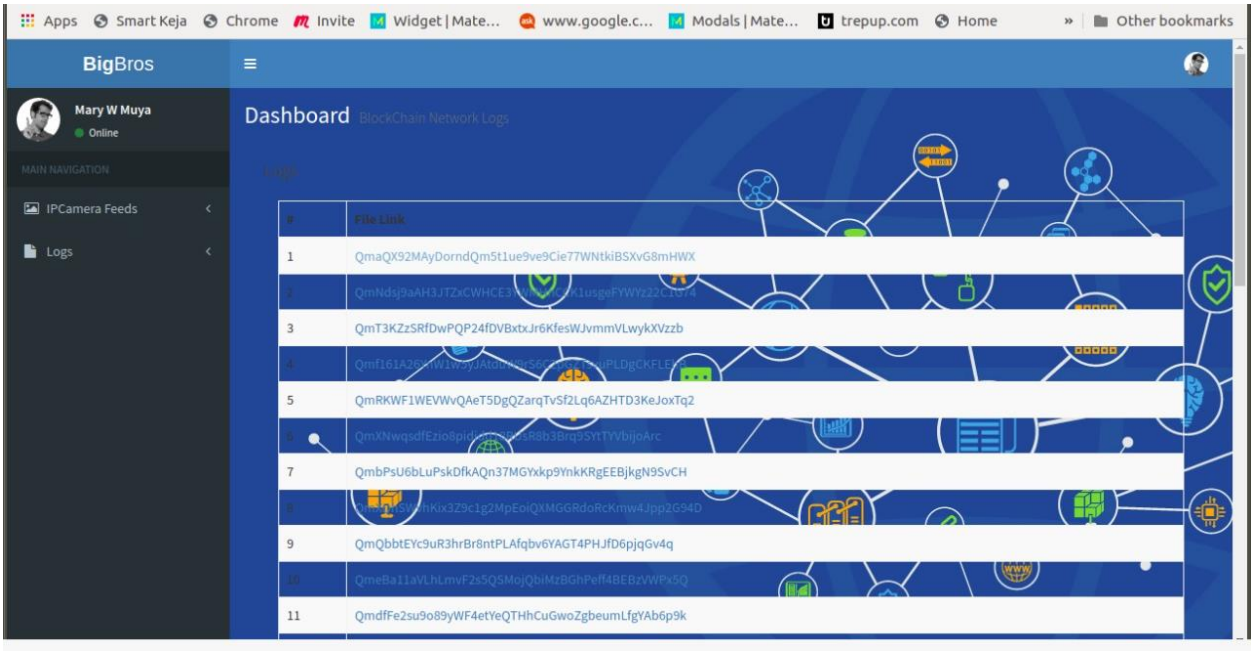


Figure 5.20 : Hashes of the log files

```
2019-06-03 15:02:36,291:INFO:requested /videos/ with {} in method
2019-06-05 12:38:05,371:INFO:requested /videos/ with {} in method
2019-06-05 12:42:16,989:INFO:requested /videos/ with {} in method
2019-06-05 18:37:04,747:INFO:requested /videos/ with {} in method
2019-06-05 18:39:14,664:INFO:added video feed :  20190605183844
2019-06-05 18:40:30,690:INFO:requested /logs/ with {} in method
2019-06-05 18:51:17,546:INFO:requested /logs/ with {} in method
```

Figure 5.21 : Log activities in the log file

5.6 System Testing

The purpose of testing was to demonstrate that the system met its requirements. Testing involved functional testing where each functionality was working as expected. The other testing done was is system testing where the whole system was tested back to back. Nonfunctional testing was done to check if the system met the requirements listed in chapter 4. Black box testing was applied in this case to identify if the system worked as expected.

5.6.1 Functional testing

The system had four main functions which was tested individually;

Integrity verification on Feeds and system logs

When one request to view the videos/feeds/log files, the video/log file is displayed in a format that cannot be edited. IPFS provides file-level encryption providing integrity of the feeds/log files. Additionally, use of IPFS and BigchainDB guarantees storage of feeds/files in immutable format, thus integrity of the feeds/files is verified as they are immutable. The system provided an interface where the raspberry pi camera is activated once a capture request is made and an upload to the IPFS system done immediately after capturing process is over. This ensures no video can be added twice. The hashes generated by IPFS are stored in BigchainDB, this ensures no fake hashes can be generated and injected to the IPFS network. Storing the logs in the IPFS network and storing the hashes to the log files in the BigchainDB means access to the logs is secure as it cannot be viewed by authorized parties without allowing any alterations.

Testing for factory set passwords

Four factory set credentials were used to do a black box testing, the credentials were used to create account to check if the system could pass. The following figures shows a combination of different common credentials used;

i. Username: root Password: No password

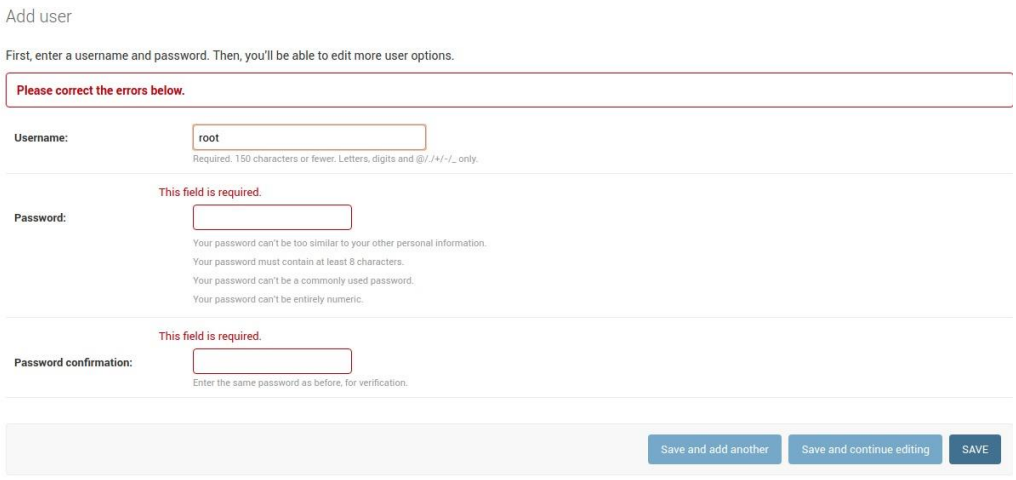


Figure 5.22 : Factory set credentials username: root password:

ii. Username: Admin Password: 1234/4321/0000

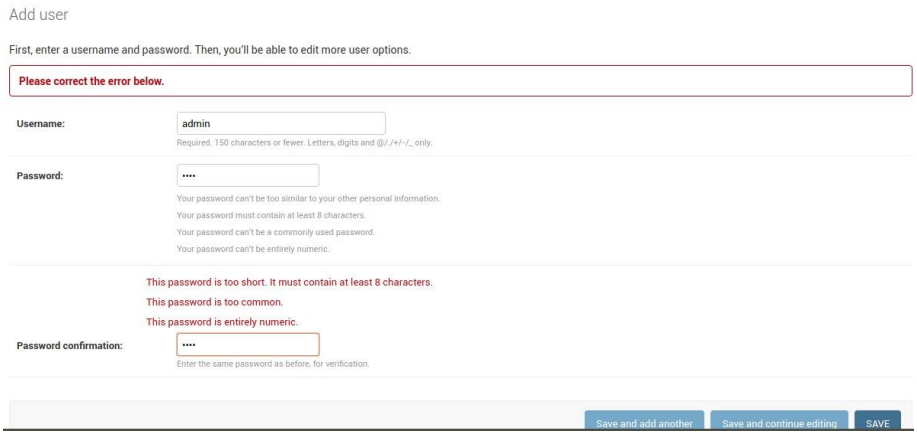


Figure 5.23 : Factory set credentials username: Admin password: 1234/4321/0000

iii. Username: Admin Password: 12345/123456

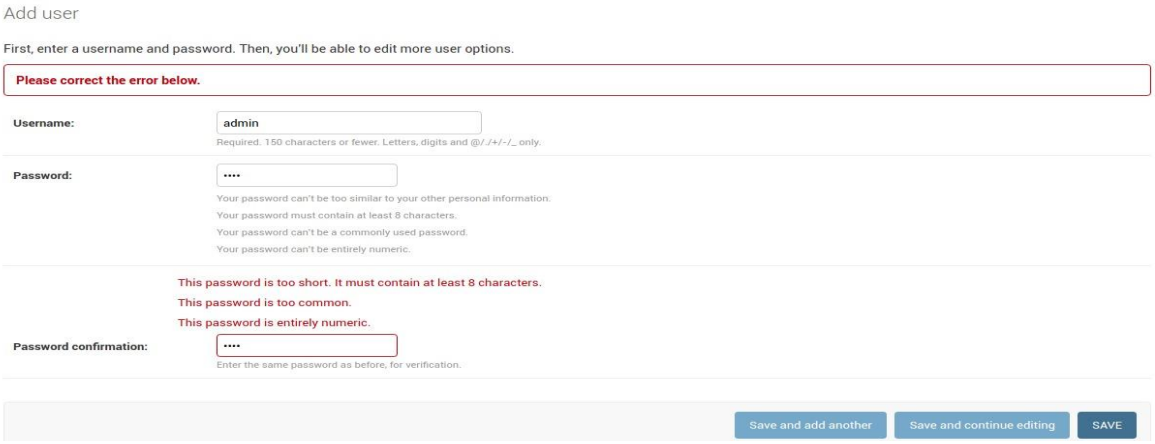


Figure 5.24 : Factory set credentials username: Admin password: 12345/123456

iv. Username: root Password: root

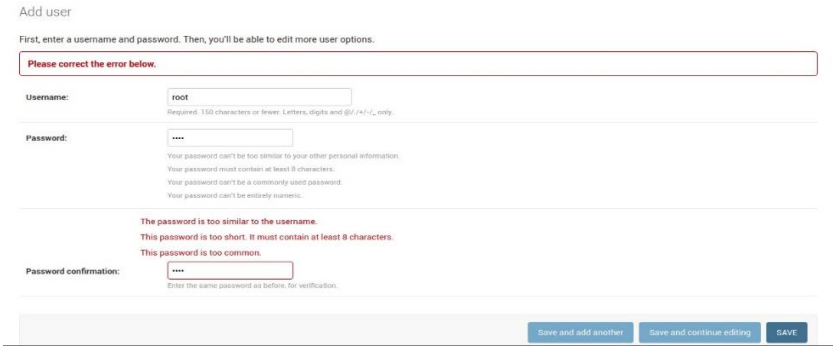


Figure 5.25 : Factory set credentials username: root password: root

v. Username: service Password: service

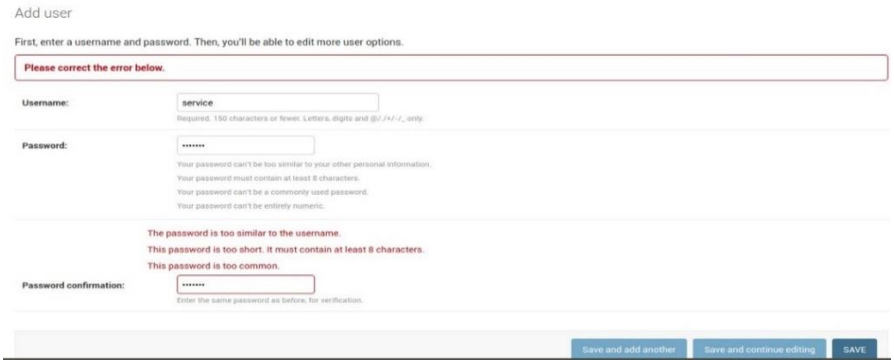


Figure 5.26 : Factory set credentials service password: service

Testing for login lockouts and alerts

Logins with incorrect credentials were used. Four attempts were done using the incorrect credentials and it was checked if the system could allow the user to login. Figures shows 4 login attempts at a go. The system blocks user after 4 attempts. The message showed in Figure5.27 is displayed any time a combination of the locked IP address and user name tries to access the system after that.

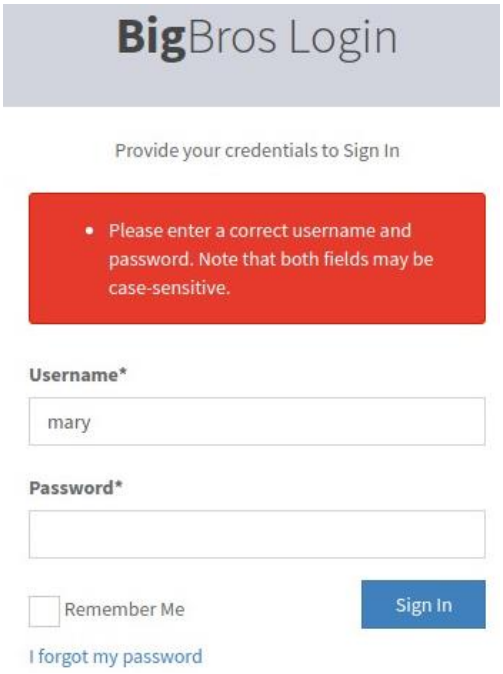


Figure 5.27 : Failed login attempt

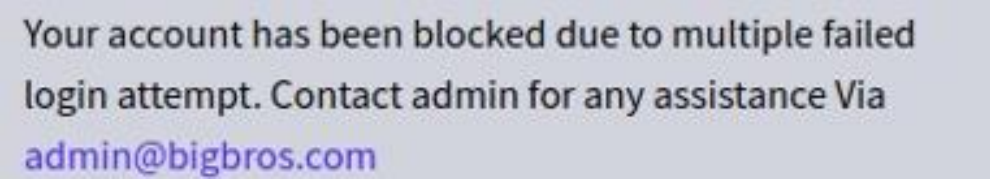


Figure 5.28 : Lockout message

5.6.2 Non Functional testing

i. After setting up the virtual IPFS client and server the system was always available upon request.
ii. Uploading and viewing of feeds was fast enough.
iii. Use of Blockchain secured feeds and system logs from alterations.
iv. The system blocked access account with common/factory set credentials.
v. They system proved to be verifiable as all its activities were logged and could be audited
vi. The system proved to be easy to use due to its well design and simple front end web interface.

Chapter 6 : Discussion of Results

6.1 Overview

The purpose of this dissertation was to come up with a system prototype that would demonstrate how IP based cameras could be made secure by minimising the attack surfaces that exist currently. The prototype was designed implemented and tested. The aim of this chapter is to ascertain if the objectives of the dissertations were met. The advantage most attackers have been taking to use factory set credentials to gain access to the camera was fully eliminated my ensuring no account can be created with factory set credentials.

6.2 Common threats and vulnerabilities on IP based cameras and their security implications

The first objective of this dissertation was to identify common threats, vulnerabilities/attack surfaces that exist in IP cameras and their security implications. Section 2.4, 2.5 and 2.6 detailed the threats, vulnerabilities and risk implication. Section 2.4 outlined four major vulnerabilities the dissertation aimed to address.

6.3 To review the existing security solutions for IP based cameras

The second objective was to review existing solutions for IP based cameras. A number of previous work was reviewed as explained in section 2.8. From the findings of section 2.9 there was no single system that had addressed all the identified vulnerabilities and therefore it was in deed right to go ahead design the proposed prototype.

6.4 To design, implement and test the proposed solution for IP based cameras

A solution was designed, implemented and tested per objective three. The solution was meant to address the four main vulnerabilities; use of factory set passwords, allowing unauthorized access in case of brute force attacks, image validation and allowance to support forensic investigations

The advantage most attackers have been taking to use factory set credentials, insecure web interfaces, and lack of proper authentication to gain access to the camera was fully eliminated my ensuring no account can be created with factory set credentials, a combination of IPFS and BigchainDB to ensure feeds and system logs cannot be edited at all. This was combined with encryption to ensure attackers cannot take advantage of eavesdropping to access what is not meant to be seen by intruders. The development meant to achieve Integrity verification, at the same time

the immutable logs were to provide authenticate evidence to support forensic investigations. Finally, the system was trained to identify brute force attacks and lockout suspicious access attempts doubled with alerts, this enhanced the authentication process thus reducing the insecurity that come along with insecure web interfaces and poor authentication mechanisms. These functionalities were tested in section 5.6.

6.5 System Validation

Through the testing done to the functions of the system in chapter 6 it was proved that the system protected the camera, its feeds and activity logs from attacks by minimising the vulnerabilities that come with factory default passwords, nonvalidated feeds, insecure web designs and editable system logs. This guarantee integrity of data by providing integrity verifications to feeds and system logs through use of IPFS and BigchainDB, improving system configurations by restricting default passwords. Additionally, the system ensured confidentiality of data through authentication, lockouts, access controls and sending of alerts. Reducing the vulnerabilities identified in chapter two, guaranteed availability of the data and services provide by the system.

The proposed system is an easy to use system with a web interface that is easy to use. Its performance and response rate is high; this is achieved by use of BigchainDB and taking advantage of its design goals. The functionalities of the system were also tested on windows environment and provided same results as what was achieved using Linux environment.

Chapter 7 : Conclusion, Recommendations and Future Work

7.1 Conclusion

IP based cameras and IoT devices at large hold a game changing potential to make our lives easier, better and more convenient. Unfortunately, if the security risks that come along with these devices are not addressed the value of the IoT will be outdone by the security problems associated with the IoT devices.

The objectives of this dissertation was come up with a solution that could demonstrate how IP based cameras can be made more secure by reducing the vulnerabilities attackers take advantage of. The developed solution proved it could improve the security of the IP based cameras by coming up with functionalities to reduce unauthorized access, limited access management, image validation and how the camera systems can be auditable and verifiable.

7.2 Recommendations

Security is a major concern in the internet world. It is therefore important for developers to be more alert on security related issues when they are designing IoT and more so IP based cameras whose demand has been on the rise. Use of Blockchain is a necessity when it comes to matters on integrity, IoT developers should embrace the power of combing IPFS and Blockchain technology to achieve more secure systems.

7.3 Future Work

This dissertation only used a prototype to prove the concept by using development environment for IPFS and test BigchainDB. Future work should ensure the developed functionalities are tested on the production environment for both IPFS and BigchainDB. The dissertation also used Ubuntu 14.06 LTS to run the solution and windows and Linux for testing. The concept can be extended to other operating systems including android and MacOS

Additionally, this dissertation didn’t address all the vulnerabilities that currently exist but rather identified t some in order of priority in terms of the damage the exposure of the vulnerabilities has to the IP camera and to the world of IoT. Future study can check other vulnerabilities.

References

Abomhara, M., & Koien, G. M. (2015). Cyber Security and the Internet of Things:. *Journal of Cyber Security, Vol. 4*, 65-88.

Alhassan, M. M., & Adjei-Quaye, A. (2017). Information Security in an Organization. *International Journal of Computers*, 100-115.

Allgeyer, T. (2018, March 22). *IoT Security-Threats and Vulnerabilities*. Retrieved from FRENUS: https://www.frenus.com/4603

Alshalawi, R., & Alghamdi, T. A. (2017). Forensic tool for wireless surveillance camera. *2017 19th International Conference on Advanced Communication Technology (ICACT)* (pp. 536-540). 10.23919/ICACT.2017.7890148.

Ashford, W. (2018, May 2). *ARM aims to boost physical security of IoT*. Retrieved from www.computerweekly.com: https://www.computerweekly.com/news/252440406/ARM-aims-to-boost-physical-security-of-IoT

Athuraliya, A. (2018). *Sequence Diagram Tutorial: Complete Guide with Examples*. Retrieved 2018, from creately blog: https://creately.com/blog/diagrams/sequence-diagram-tutorial/

Avital, N. (2019, January 9). *The State of Web Vulnerabilities in 2018*. Retrieved from Imperva: https://www.imperva.com/blog/the-state-of-web-application-vulnerabilities-in-2018/

Baker, A. (2016, August 23). *Maintaining Integrity in the IoT applications.* Retrieved from ICC Media: files.iccmedia.com/pdf/windriver160823.pdf

Banafa, A. (2016, July 12). *IoT Standardization and Implementation Challenges*. Retrieved from IoT.ieee.org: https://iot.ieee.org/newsletter/july-2016/iot-standardization-and-implementation-challenges.html

Banafa, A. (2017, March 4). *Major Challenges Facing IoT.*

Banafa, A. (2018, February 17). *Components and its Challenges of an IoT Implementation*. Retrieved from DATAFLOQ: https://datafloq.com/read/5-components-iot-implementation-challenges/1860

Behringer, M. H. (2009). End-to-End Security. *The Internet Protocol Journal Vol12 No3*, 2.

Benet, J. (2014). *IPFS - Content Addressed, Versioned, P2P File System.* Retrieved from https://ipfs.io: https://ipfs.io/ipfs/QmV9tSDx9UiPeWExXEeH6aoDvmihvx6jD5eLb4jbTaKGps

Benjamin. (2019, feb 17). *Web Server Security Friendly logging and Monitoring.* Retrieved from https://infosec-handbook.eu/blog/wss6-logging-monitoring

BigchainDB 2.0 The Blockchain Database. (2018, May). Berlin, Germany.

blockchain with hash for IPFS. (n.d). Retrieved from https://www.google.com/search?q=blockchain+with+hash+for+IPFS&client=firefox-b-d&source=lnms&tbm=isch&sa=X&ved=0ahUKEwiz-7S9jMrhAhVJzIUKHe8BBngQ_AUIECgD&biw=1093&bih=502#imgrc=V2CVIKCPmrL5fM:

Boone, A. (2018, August 30). *Complexity is the enemy of security' … especially in IoT*. Retrieved from https://www.timesys.com: https://www.timesys.com/security/complexity-enemy-security-iot/

Brandnenburg, L. (n.d). *How to create an Entity Relationship Diagram*. Retrieved Jan 31, 2018, from bridging-the-gap.com: http://www.bridging-the-gap.com/erd-entity-relationship-diagram/

Brickhouse Security. (2019). *Internet Cameras Explained*. Retrieved from BrickHouse Security: https://www.brickhousesecurity.com/security-cameras/ip-camera-guide/

Capital, Z. (2018, october 3). *IPFS: A Complete Analysis of the Distributed Web*. Retrieved from https://hackernoon.com/ipfs-a-complete-analysis-of-the-distributedweb-6465ff029b9b

Check point Researcher. (2018). *2018 Security Report.* Israel: Point Software Technologies Ltd.

Cisco. (2014). *The Internet of Things Reference Model.* San, Jose, CA.

Cisco Connect. (2014). *Building the Internet of Things.* Cisco.

Corbin, K. (2016). *The Real World Damage of IoT Attacks*. Retrieved from CyberSecurity: https://totalsecuritydailyadvisor.blr.com/cybersecurity/real-world-damage-iot-attacks

Cruz, b. (2017, 11 21). *E2EE Camera: A Camera With End-To-End Encryption*. Retrieved from https://homealarmreport.com/e2ee-camera-camera-end-end-encryption/

Cusack, B., & Tian, Z. (2017). Evaluating IP Surveillance . *15th Australian Security management Conference* (pp. 23-32). perth, West Australia: Edith Cowan University Research Online.

Cybersecurity on Internet of Things, 115-40 (Subcommittee On Information Technology Of The committee On Oversight And Government Reform House Of Representatives One Hundred Fifteenth Congress October 3, 2017).

Cybonet. (n.d.). *The Danger of Default Passwords*. Retrieved from cybonet: www.cybonet.com/en/news/blog/191-the-danger-of-default-passwords

DataFlair Team. (2018, September 15). *How IoT Works – 4 Main Components of IoT System*. Retrieved from Data Flair: https://data-flair.training/blogs/how-iot-works/

DataFlair Team. (2018, December 21). *IoT Tutorial for Beginners | What is Internet of Things?* Retrieved from DataFlair: https://data-flair.training/blogs/iot-tutorial/

Diffie, W., & Hellman, M. E. (1976). New Directions in Cryptography. *IEEE Transacctions on Information Theory*, 11.

Download the latest python version. (2017, December 19). *Download the latest python version*. Retrieved January 2018, from python.org: https://www.python.org/downloads/

Elicalde, D. (2017, February 24). *IoT Hardware – Introduction and Explanation*. Retrieved from IoTforAll: https://www.iotforall.com/iot-hardware-introduction-explanation/

Fabric, S. (2015). *The Internet of Things- A Conceptual Model*. Retrieved from The Sand Reckoner: https://www.thesandreckoner.co.uk/model-viewing-internet-things/

Farmer, C. (2018, August 28). *What's really happening when you add a file to IPFS?* Retrieved from textile: https://medium.com/textileio/whats-really-happening-when-you-add-a-file-to-ipfs-ae3b8b5e4b0f

Fortney, L. (2019, May 21). *Blockchain, Explained*. Retrieved from https://www.investopedia.com/: https://www.investopedia.com/terms/b/blockchain.asp

Foxhoven, P. (2016, November 8). *Security risks from Internet of Things*. Retrieved from TechTarget: https://internetofthingsagenda.techtarget.com/blog/IoT-Agenda/Security-risks-from-the-internet-of-things

Fruhlinger, J. (2018, May 9). *The Mirai botnet explained: How teen scammers and CCTV cameras almost brought down the interne*. Retrieved from CSO: https://www.csoonline.com/article/3258748/security/the-mirai-botnet-explained-how-teen-scammers-and-cctv-cameras-almost-brought-down-the-internet.html

Gottesdiener, e. (1995, August). *RAD Realities*. Retrieved January 25, 2018, from www.ebgconsulting.com: https://www.ebgconsulting.com/Pubs/Articles/RAD_Realities_Beyond_the_Hype_Gottesdiener.pdf

Goverment Europa. (2018, March 28). *Information security and privacy protection aspects of CCTV systems*. Retrieved from www.governmenteuropa.eu: https://www.governmenteuropa.eu/information-security-cctv-systems/85930/

Guest Writer. (2019, March 9). *The 7 Most Common IoT Security Threats in 2019*. Retrieved from https://www.iotforall.com/

Hashim, A. (2018, June 21). *Multiple Vulnerabilities In Axis Cameras Allow Hackers To Take Control* . Retrieved from Latest Hacking News: https://latesthackingnews.com/2018/06/21/multiple-vulnerabilities-in-axis-cameras-allow-hackers-to-take-control/

Haughey, D. (2014). *MOSCOW METHOD*. Retrieved January 29, 2018, from Project Smart: https://www.projectsmart.co.uk/moscow-method.php

(2016). *Herizon Data Breach Investigation Report.*

Higginbotham, S. (2018, November 20). *IoT Cameras Need to be Quick, Clever, and Able to Assign Meaning*. Retrieved from Spectrum.ieee.org: https://spectrum.ieee.org/telecom/internet/iot-cameras-need-to-be-quick-clever-and-able-to-assign-meaning

Hirschberg, M. A. (n.d). *Rapid Application Development(RAD) : A brief Overview.* Retrieved January 13, 2018, from Software Tech News: https://www.csiac.org/wp-content/uploads/2016/02/1998_03_01_RapidApplicationDevelopment.pdf

IBM. (2017, November 17). *Top 10 IoT security challenges*. Retrieved from https://developer.ibm.com: https://developer.ibm.com/articles/iot-top-10-iot-security-challenges/

IEEE Standard Association. (2015, January). *Internet of Things (IoT) Ecosystem Study.* Retrieved from IEEE: https://iot.ieee.org/images/files/pdf/iot_ecosystem_exec_summary.pdf

(2017). *IoT Security Report.* RSA Conference.

IPFS Structure. (2018). Retrieved from reddit.com: https://www.reddit.com/r/ethereum/comments/6ynhjz/decentralized_ebay_on_ethereum_ipfs/

Isobe, T., & Minematsu, K. (2018). Breaking Message Integrity ofan End-to-End Encryption Scheme of LINE? Hyogo, Japan. Retrieved from https://eprint.iacr.or: https://eprint.iacr.org/2018/668.pdf

Jay360. (2017, January 20). *The Ultimate Guide to Surveillance Camera Vulnerabilities*. Retrieved from Jay360: https://jay360.ca/ultimate-guide-to-surveillance-camera-vulnerabilities/

Jeffrey, C. (2018, April 2018). *Securing IP Surveillance Cameras in the IoT Ecosystem*. Retrieved from Trend Micro Websire: https://www.trendmicro.com/vinfo/au/security/news/internet-of-things/securing-ip-surveillance-cameras-in-the-iot-ecosystem

Johnston, S. J., Cox, S. J., & Scott, M. (2016). Recommendations for securing Internet of Things devices using commodity hardware. *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)* (p. 23). Reston, VA, USA: IEEE.

Jovovic, I., Sinisa, H., Forenbacker, I., & Macek, S. (2018). 5G, Blockchain and IPFS: A General Survey with Possible Innovative Applications in Industry 4.0. *MMS 2018, November 06-08,* (p. 10). Dubrovnik, Croatia: University of Zagreb, Faculty of Transport and Traffic Sciences,.

Kaaniche, N., Jung, E., & Gehani, A. (2018). *Efficiently Validating Aggregated IoT Data Integrity.* Retrieved from http://www.csl.sri.com: http://www.csl.sri.com/users/gehani/papers/BDS-2018.Homomorphic.pdf

Kwatra, K. (2018, March 15). *What is IPFS*. Retrieved from Wolverine Blockchain: https://medium.com/wolverineblockchain/what-is-ipfs-b83277597da5

Laeeq, K., & Shamsi, J. (2015). A Study of Security Issues, Vulnerabilities and Challenges. In K. Laeeq, & J. Shamsi, *Securing Cyber-Physical Systems* (p. 9). Pakistan.

Lemos, R. (2019, January 9). *Web Vulnerabilities Up, IoT Flaws Down*. Retrieved from Dark Reading: https://www.darkreading.com/vulnerabilities---threats/web-vulnerabilities-up-iot-flaws-down/d/d-id/1333625

Malafronte, K. (2018, October 11). *California Bans Default Passwords for All IoT Devices*. Retrieved from Security Technology: https://www.campussafetymagazine.com/technology/california-bans-default-passwords-iot-devices/

Malafronte, K. (2018, October 11). *California Bans Default Passwords for All IoT Devices* . Retrieved from Technology Security: https://www.campussafetymagazine.com/technology/california-bans-default-passwords-iot-devices/

Market&Research. (2018, Dec 11). *Global IP Camera Market Outlook 2017-2026 - Growing Security Concerns, Better Image Resolutions and Ease of Installation are Driving Market Growth* . Retrieved from PRNewswire: https://www.prnewswire.com/news-releases/global-ip-camera-market-outlook-2017-2026---growing-security-concerns-better-image-resolutions-and-ease-of-installation-are-driving-market-growth-300763551.html

MarketsandMarkets. (n.d.). *Global Video Surveillance Market Applications and Management Services Forecasts (2010-2015)* . Retrieved from MarketsandMarkets: https://www.marketsandmarkets.com/Market-Reports/surveillance-277.html

Massimo, F. (2011). *Use Cases.* Retrieved 2018, from http://www.inf.ed.ac.uk: http://www.inf.ed.ac.uk/teaching/courses/seoc/2011_2012/notes/SEOC03_notes.pdf

Mathapai, S. (2017). IoT Ecosystem and Business Opprtnities. *Tizen Conference for Developers* (p. 43). San Francisco: Samsung.

Matherly, J. (2017, 8 23). *Complete Guide to Shodan.* Retrieved from https://the-eye.eu: https://the-eye.eu/public/Books/qt.vidyagam.es/library/zz_unsorted/shodan.pdf

Mcgrath, S. (2019, Jan 21). *Resolving IoT Security Issues with blockchain Technology*. Retrieved from Hackernoon: https://hackernoon.com/resolving-IoT-securiy-issues-with-blockchain-technology-3ffb36357094

Miessler, D. (2015). SSecuring the Internet of Things:Mapping Attack Surface Areas Using the OWASP IoT Top 10. *RSAConference2015* (p. 12). San Franscisco: ASD.

Mikec. (2017, Jan 22). *Interaction Diagrams.* Retrieved January 31, 2018, from www.cs.ucsb.ed: https://www.cs.ucsb.edu/~mikec/cs48/project/InteractionLarman.pdf

Millien, R., & George, C. (2016, November 28). *The Enabling Technologies of the Internet of Things*. Retrieved from ipwatcgdog: https://www.ipwatchdog.com/2016/11/28/enabling-technologies-internet-things/id=75039/

Minoli, D., & Occhiogrosso, B. (2018). Blockchain mechanisms for IoT security. *Internet of Things*, 1-13.

Mulder, P. (2018, January 23). *MoSCoW*. Retrieved Jan 25, 2018, from www.toolshero.com: https://www.toolshero.com/project-management/moscow-method/

NetVu. (2017, July 27). *Internet of Things and the risks of vulnerabilities*. Retrieved from www.netvu.org.uk: https://netvu.org.uk/wp-content/uploads/2017/08/Cyber_Security_presentation_NetVu_July17.pdf

NewgenApps. (2018, May 28). *IoT Ecosystem Components: The Complete Connectivity Layer*. Retrieved from New Gen Apps: https://www.newgenapps.com/blog/iot-ecosystem-components-the-complete-connectivity-layer

Operational Security. (n.d). Retrieved from www.sc.edu: http://www.marines.mil/unit/13thmeu/Pages/OperationalSecurity.aspx

OWASP. (2016, May 18). *Top IoT Vulnerabilities*. Retrieved from www.owasp.org: https://www.owasp.org/index.php/Top_IoT_Vulnerabilities

Pasqua, E. (2018, September 28). *LPWAN emerging as fastest growing IoT communication technology – 1.1 billion IoT connections expected by 2023, LoRa and NB-IoT the current market leaders*. Retrieved from IoT Analytics: https://iot-analytics.com/lpwan-market-report-2018-2023-new-report/

Processing, F. (2018, 11 22). *Storing files in a distributed file system using blockchain technology*. Retrieved from Future Processing: https://www.future-processing.pl/blog/storing-files-in-a-distributed-file-system-using-blockchain-technology/

Puiu, T. (2014, Aril 23). *How IP Camers Work:Basics of Modern Surveillance*. Retrieved from ZME Science: https://www.zmescience.com/tech/how-ip-cameras-work-the-basics-of-modern-surveillance/

RAD Model. (n.d). *RAD Model.* Retrieved 2018, from https://studyregular.in: https://studyregular.in/category/software-engineering/?print=pdf-search

Rajiv. (2018, January 10). *What are the major components of Internet of Things*. Retrieved from RF Page: https://www.rfpage.com/what-are-the-major-components-of-internet-of-things/

Rapid Application Development. (2015, september 1). *Rapid Application Development*. Retrieved January 25, 2018, from http://www.ftms.edu.my: http://www.ftms.edu.my/images/Document/IMM006%20-%20RAPID%20APPLICATION%20DEVELOPMENT/Chapter%202nnote.pdf

Red Hart. (2019). *What is Raspberry Pi*. Retrieved from opensource.com: https://opensource.com/resources/raspberry-pi

Regoniel, P. A. (2015, Jan 5). *Conceptual Framework: A Step by Step Guide on How to Make One*. Retrieved from SimplyEducate.Me: https://simplyeducate.me/2015/01/05/conceptual-framework-guide/

Romdhani, I., Abdmeziem, R., & Tandjauol, D. (2015). Architecting the Internet of Things: State of the Art.

Rouse, M. (2016, July 29). *rapid-application-development*. Retrieved Jan 29, 2018, from SearchSoftwareQuality: http://searchsoftwarequality.techtarget.com/definition/rapid-application-development

Rusnak, P. (2017). *Interplanetary File System.* Retrieved from ipfs.io: https://thunder.org/sites/default/files/2017-11/IPFS%20-%20The%20distributed%20Web.pdf

SDLC-Rapid Application Development Model. (n.d). *Software Development Lifecycle-Rapid Application Development Model*. Retrieved January 25, 2018, from tutorialspoint.com: https://www.tutorialspoint.com/sdlc/sdlc_rad_model.htm

Shah, S. (2018, April 5). *IoT security: Half of IT departments don't change default passwords*. Retrieved from https://internetofbusiness.com: https://internetofbusiness.com/password-iot/

Shea, S. (2019, January). *The top 7 enterprise IoT risks to consider*. Retrieved from https://internetofthingsagenda.techtarget.com: https://internetofthingsagenda.techtarget.com/tip/Internet-of-Things-IOT-Seven-enterprise-risks-to-consider

Sheridan, K. (2017, 9 5). *New IoT Botnet Discovered, 120K IP Cameras At Risk of Attack*. Retrieved from DarkReading: https://www.darkreading.com/attacks-breaches/new-iot-botnet-discovered-120k-ip-cameras-at-risk-of-attack/d/d-id/1328839

Shoikova, E., Petkova, P., Donchev, D., & Jekove, B. (2017). Study on the IoT Ecosystem Business Models and the Segment of Startups. *ICERI2017 10th annual International Conference of Education, Research and Innovation* (p. 11). Seville (Spain): Research Gate.

Software Requirements. (n.d). Retrieved 2018, from http://www.inf.ed.ac.uk: http://www.inf.ed.ac.uk/teaching/courses/cs2/LectureNotes/CS2Ah/SoftEng/se02.pdf

Spring, T. (2017, August 3). *Two Popular IP Cameras Riddled With Vulnerabilities*. Retrieved from ThreatPost: https://threatpost.com/two-popular-ip-cameras-riddled-with-vulnerabilities/127172/

Steichen, M., Norvill, R., Borja, B., & Shbair, W. (2018, July). Blockchain-based, Decentralised Access Control for IPFS. *2018 IEEE Confs on Internet of Things, Green Computing and Communications, Cyber, Physical and Social Computing,.* Luxembourg: IEEE.

Subhash, D. (2017, April 29). *What is an IP Camera? How does IP Surveillance Camera Work? [Complete Guide]*. Retrieved from IT4NextGen: http://www.it4nextgen.com/what-is-ip-camera/

Swedish Civil Contingiencies Agency. (2018). *IoT Related Risks.* Karlstad: MSB.

Tabatabaei, M. H. (2018, April 4). *Advanced BlockchainStorage.* Retrieved from UIO: https://www.uio.no/studier/emner/matnat/ifi/IN5420/v18/timeplan/resources/summary-twelfth-topic/advanced20blockchain20storage.pdf

Tracy, P. (2017, May 12). *Major cyberattack holds computers hostage in at least 99 countries*. Retrieved from The DailyDot: https://www.dailydot.com/debug/wanna-cry-ransomware-malware-attack-nhs/

Valdez-de-Leon, O. (2017, May 17). *Key Elements and Enablers for Developing an IoT Ecosystem*. Retrieved from IoT.ieee.org: https://iot.ieee.org/newsletter/may-2017/key-elements-and-enablers-for-developing-an-iot-ecosystem.html

Vlajic, N. (2013). *Introduction.* Retrieved from www.eecs.yorku.ca: https://www.eecs.yorku.ca/course_archive/2013-14/F/4482/CSE4482_01_Introduction_2013_posted.pdf

Voshmgir, S., & Kalinov, V. (2017, September 30). *Blockchain-A beginer's guide.* Retrieved from BLOCKCHAINHUB: https://s3.eu-west-2.amazonaws.com/blockchainhub.media/Blockchain+Technology+Intro.pdf

Wikipedia contributors. (2019, January 15). *Vulnerability (computing)*. Retrieved from wikipedia: https://en.wikipedia.org/w/index.php?title=Vulnerability_(computing)&oldid=878548043

Willson, S. (2018, May 22). *What the Internet of Things is Missing*. Retrieved from CA: https://www.ca.com/en/blog-automation/what-the-internet-of-things-is-missing.html

Zelijka, Z. (2018, October 10). *9 million Xiongmai cameras, DVRs wide open to attack*. Retrieved from HelpNetSecurity: https://www.helpnetsecurity.com/2018/10/10/vulnerable-xiongmai-cameras/

Appendices

Appendix A Code Snippet on how Feeds are encrypted using AES

```
def encrypt_file(self, key, in_filename, out_filename=None, chunksize=64 * 1024):
    """ Encrypts a file using AES (CBC mode) with the    given key.
        key:
            The encryption key - a string that must be either 16, 24 or 32 bytes long. Longer keys
            are more secure.
        in_filename:
            Name of the input file
        out_filename:
            If None, '<in_filename>.enc' will be used.

    chunksize:
        Sets the size of the chunk which the function
        uses to read and encrypt the file. Larger chunk
        sizes can be faster for some files and machines.
        chunksize must be divisible by 16.
"""
if not out_filename:
    out_filename = in_filename + '.enc'


iv = ''.join(chr(random.randint(0, 0xFF)) for i in range(16))
encryptor = AES.new(key, AES.MODE_CBC, iv)
filesize = os.path.getsize(in_filename)
```

Appendix B Use Cases

Use Case Title	**Create Account**
Primary Actors	Admin
Secondary Actor	System
Brief Description	This command is to add users with roles and rights to access the system
Include use cases	Check factory set credentials
Extends use case	Reset Password
Pre-Conditions	None
Post Conditions	No account created with factory set login credentials
Path/flow of events	The admin creates accounts to access the system, the system check the login set details by the admin, the system prompt the user to enter strong passwords if factory set credentials are detected.
Alternatives	Once the details are confirmed they meet requirements an account is created

Use Case Title	**Login**
Primary Actors	User(Admin/Normal User)
Secondary Actor	System
Brief Description	The command checks and authorise or deny access to the system
Include use cases	Check for more than four failed login attempts
Extend use cases	Send alerts
Pre-Conditions	To login one must have a registered account
Post Conditions	Login allowed to authorised users with accounts and the login details match.
Path/flow of events	The system checks if the user requesting access has authority to access, if unauthorized access is detected the login attempts is declined. When user tries for than four times the system locks the account and the IP address sending the access request. Upon lockout an email alert is send to the admin notifying the blocked access request, the lockout request is then stored for reference

Alternatives	Once login details are confirmed user is allowed to access the system

Use Case Title	**Capture system activities**
Primary Actors	System
Admin	System
Brief Description	A command to log all system activities
Include use cases	Add file to IPFS Add a block of the file on the BigchainDB
Pre-Conditions	Activities must have been done on the system
Post Conditions	System activities are stored in an immutable format
Path/flow of events	The system record all activities done on the system, the activities are stored as files in the IPFS and a block created for each session per user
Alternatives	None.

Use Case Title	**View System Logs**
Primary Actors	Admin
Secondary Actor	System
Brief Description	A command to view system logs
Include use cases	None
Pre-Conditions	none
Post Conditions	Logs must be displayed in immutable format.
Path/flow of events	Admin request access to the feeds, an access path to the request logs is provided, logs are retrieved from IPFS in immutable format
Alternatives	Failed request incase user is not an admin.