Securing mHealth - Investigating the Development of a Novel Information Security Framework

Nattaruedee Vithanwattana

M00530498

Middlesex University

School of Science and Technology

A thesis submitted to Middlesex University in partial fulfilment of the requirements for the degree of Doctorate of Philosophy

Director of Studies: Dr. Carlisle George

Supervisor: Dr. Glenford Mapp

February 2022

Abstract

The deployment of Mobile Health (mHealth) platforms as well as the use of mobile and wireless technologies have significant potential to transform healthcare services. The use of mHealth technologies allow a real-time remote monitoring as well as direct access to healthcare data so that users (e.g., patients and healthcare professionals) can utilise mHealth services anywhere and anytime. Generally, mHealth offers smart solutions to tackle challenges in healthcare. However, there are still various issues regarding the development of the mHealth system. One of the most common difficulties in developing the mHealth system is the security of healthcare data. mHealth systems are still vulnerable to numerous security issues with regard to their weaknesses in design and data management. Several information security frameworks for mHealth devices as well as information security frameworks for Cloud storage have been proposed, however, the major challenge is developing an effective information security framework that will encompass every component of an mHealth system to secure sensitive healthcare data. This research investigates how healthcare data is managed in mHealth systems and proposes a new information security framework that secures mHealth systems. Moreover, a prototype is developed for the purpose of testing the proposed information security framework. Firstly, risk identification is carried out to determine what could happen to cause potential damage and to gain insight into how, where, and why the damage might happen. The process of risk identification includes the identification of assets those need to be protected, threats that we try to protect against, and vulnerabilities that are weaknesses in mHealth systems. Afterward, a detailed analysis of the entire mHealth domain is undertaken to determine domain-specific features and a taxonomy for mHealth, from which a set of the most essential security requirements is identified to develop a new information security framework. It then examines existing information security frameworks for mHealth devices and the Cloud, noting similarities and differences. Key mechanisms to implement the new framework are discussed and the new framework is then presented. Furthermore, a prototype is developed for the purpose of testing. It consists of four layers including an mHealth secure storage system, Capability system, Secure transactional layer, and Service management layer. Capability system, Secure transactional layer, and Service management layer are developed as main contributions of the research.

Acknowledgements

During a few years of my research, it has not always been smooth. In this space, I would like to express my gratitude and appreciation for the following people for supporting me along this journey. Without you all, I would not be where I am now.

Firstly, I would like to express my sincere gratitude to my supervisors, Dr. Carlisle George and Dr. Glenford Mapp, for their patience, endless support, and valuable advice during my Ph.D. research. On many occasions, they worked during their holidays, early morning, or even very late at night to support me in order to get my work done. Thank you very much.

I would like to thank my parents for everything. They have always been very supportive in everything I do and every decision I make. I am so lucky to be their daughter. Without them, I would not have a chance to come this far.

To my cat, Lunar, she is my greatest companion. She always sleeps right next to me (or sometimes on me) no matter what I do. Even when I am lacking sleep, undoubtedly, she is still sleeping.

From the bottom of my heart, I would like to say a big thank you to the special person, Ercument. He has supported me in both academics and mentality. Even though he has to finish his Ph.D. research himself, he helped me look into the code when I struggled, dropped my groceries when I was in self-isolation, and many more.

I would like to extend my special thanks to all my dear friends who were always there for me, gave me continuous moral support, and understanding when I could not join them on many occasions. Hopefully, I will be able to compensate them after my thesis submission.

Last but not least, I would like to say thank you to every other person who I met along this journey. You all have taught me something. I may not remember your name or even your face, but I will always cherish the lessons.

Contents

List of Abbreviations

AAAC	Authentication, Authorisation, Accounting, and Control
AES	Advanced Encryption Standard
AI	Artificial Intelligence
AHP	Allied Health Profession
API	Application Programming Interface
APT	Advanced Persistent Threat
ARM	Advanced RISC Machines
ARP	Address Resolution Protocol
AWS	Amazon Web Services
BAN	Body Area Network
BSN	Body Sensor Network
CA	Certificate Authority
CL	Capability List
CIA	Confidentiality, Integrity, Availability
CSF	Cloud Security Framework
CSS	Cascading Style Sheet
CU	Capability Unit
DAR	Data at Rest

DDoS	Distributed Denial of Service
DES	Data Encryption Standard
DIT	Data in Transit
DNS	Domain Name System
DoS	Denial of Service
E2EE	End-to-End Encryption
EBS	Elastic Block Store
EC2	Elastic Compute Cloud
ECG	Electrocardiogram
EEG	Electroencephalogram
eHealth . . .	Electronic Health
EHR	Electronic Health Record
EMG	Electromyography
FUSE	Filesystem in Usersapce
GDPR	General Data Protection Regulation
GOe	Global Observatory for eHea;th
GPRS	General Packet Radio Service
GPS	Global Positioning System
GS	Gateway Server
HELEDD . .	Human Embedded Light Emitting Diode Display
HI	Health Informatics
HIPAA	Health Insurance Portability and Accountability Act 1996
HTML	Hypertext Markup Language
IaaS	Infrastructure as a Service

IAS	Information Assurance and Security
ICT	Information Communication Technology
ID	Identification
IDL	Interface Definition Language
IoMT	Internet of Medical Things
iOS	iPhone Operating System
IoT	Internet of Thing
IP	Internet Protocol
IPv4	Internet Protocol Version 4
IPv6	Internet Protocol Version 6
IT	Information Technology
ITSM	Information Technology Service Management
J2ME	Java 2 Platform, Micro Edition
Kbps	Kilobits per second
LAN	Local Area Network
LDAP	Lightweight Directory Access Protocol
LTE	Long-term Evolution
MCS	Multiprogrammed Computer System
MDCS	Mobile Data Collection System
mHealth	Mobile Health
ML	Machine Learning
MRL	Master Resource List
NIST	National Institute of Standards and Technology
NMS	Network Memory Server

OS	Operating System
OWASP	Open Web Application Security Project
PaaS	Platform as a Service
PDA	Personal Digital Assistant
PHR	Personal Health Record
PIN	Personal Identification Number
PRL	Process Resource List
QoS	Quality of Service
RASP	Resource Allocation Security Protocol
RBAC	Role-based Access Control
RPC	Remote Procedure Call
RSA	River Shamir Adleman
SaaS	Software as a Service
SAML	Security Assertion Markup Language
SLTP	Simple Lightweight Transport Protocol
SManL	Service Management Layer
SMF	Service Management Framework
SMS	Short Message Service
SOD	Separation of Duties
SQL	Structured Query Language
SP	Simple Protocol
SRPC	Secure Remote Procedure Call
SSAT	Single Sign-on Access Token
SSL	Secure Socket Layer

TCP	Transport Control Protocol
TLS	Transport Layer Security
TTP	Trusted Third Party
UDP	User Datagram Protocol
UI	User Interface
URL	Uniform Resource Locators
VMM	Virtual Machine Manager
VPN	Virtual Private Network
WAN	Wide Area Network
WS	Web Service
XACML	eXtensible Access Control Markup Language

List of Figures

List of Tables

Chapter 1

Introduction

Mobile Health or mHealth is an emerging phenomenon in healthcare. This is because we now have real-time access to information through the Internet via our mobile devices. These mobile and wireless technologies facilitate data collection, enhanced patient engagement, and support for healthcare professionals.

The use of mobile and wireless technologies to support achievements in healthcare systems has an enormous potential to transform the face of healthcare across the globe [1]. In recent years, there has been a huge increase in the number of these technologies to facilitate mHealth. mHealth covers "medical and public health practice supported by mobile devices, such as mobile phones, patient monitoring devices, personal digital assistants (PDAs), and other wireless devices" [2]. mHealth is a subset of eHealth, using the benefits from information and communication technologies to support healthcare services. mHealth solutions include the use of mobile devices, such as mobile phones, body sensors, wireless infrastructures. These devices are used to clinical health data and to deliver healthcare information to patients, medical professionals, and researchers. They are also used for real-time monitoring of patients' vital signs, such as heart rate, blood glucose level, blood pressure, body temperature, and brain activities [3]. mHealth enables users to monitor their health status and directly facilitates healthcare data sharing with healthcare professionals anytime and anywhere.

Since late 2019, the world has faced the global situation due to the Coronavirus pandemic. The COVID-19 virus has resulted in major changes in our lives. In an attempt to ensure that hospital facilities do not become overwhelmed, many countries announced lockdown policies to prevent and slow down the transmission. This strict measure advised people to stay at home which also included limited movement (e.g., on trips to see a doctor). Furthermore, several vaccines were developed which have been deployed globally to prevent people from getting severely ill due to COVID-19, hence, the number of people needing to go to the hospital should be significantly reduced.

Due to the COVID-19 situation, there have also been significant changes in the way healthcare is being delivered. The majority of services have moved to video, telephone, or online sessions. Face-to-face meetings are conducted only when necessary or are unavoidable such as to do medical tests or surgeries. One of the other dramatic changes is the use of medical devices to monitor and record patients' health parameters. Thus, the use of mHealth devices appears to be more accepted going forward and remote monitoring is now seen as an essential requirement for future healthcare systems. Moreover, many governments, private sectors, and citizen movements have developed mHealth initiatives to keep the population informed and help manage the crisis [4].

mHealth provides a significant potential to tackle the financial challenges of healthcare systems. It delivers more patient-focused healthcare and improves the efficiency of healthcare systems. mHealth provides sustainable healthcare through better planning of patients' treatment which reduces the number of unnecessary consultations and is well organised in delivering guidance in treatment and medication from healthcare professionals. Moreover, mHealth solutions can help patients to take more responsibility for their health through devices which can detect and report their vital signs, as well as mobile applications that will help them to be more focused on their diet and medication [5].

In mHealth systems, sensors that are generally embedded into mobile devices will collect healthcare data from users using a body area network communication. Collected healthcare data will be stored in different databases including the databases of mobile devices, hospital servers, and Cloud storage. Healthcare data is classed as "sensitive data" since it may reveal the state of someone's health which he/she may not want to share with everyone. It is also classed as special category data under Article 9 of the General Data Protection Regulation (GDPR), incorporated in the UK Data Protection Act 2018, resulting in stricter conditions for processing such data. The security of healthcare data is an essential requirement mandated by data protection legislation. Article 5(1)(f) of the GDPR states that personal data should be "processed in a manner that ensures appropriate security of the personal data, including protection against unauthorised or unlawful processing and accidental loss, destruction or damage, using appropriate technical or organisational measures" [6].

However, this ongoing transformation in healthcare has resulted in an intense spotlight being placed on the security of mHealth environments because there are now many more ways in which these systems are open to attack, leading to major disruptions in services which will result in longer recovery times and even increased deaths. Several vulnerabilities need to be addressed including unauthorised access to patient records, ransomware attacks on hospital data, Distributed Denial of Service (DDoS) attacks on hospital IT infrastructure, the tampering or misuse of hospital equipment, stealing sensitive data from remote devices, and impersonation of people by others during remote monitoring. Therefore, it is necessary to tackle these challenges by developing a subtle approach to secure mHealth systems.

1.1 Background

mHealth has become a rising phenomenon within the healthcare industry. Initially, the mHealth industry was a marketplace that was mainly dominated by fitness and well-being applications and wearable devices. However, the use of mHealth in remote consultation and monitoring is growing especially since the beginning of the COVID-19 pandemic. Nowadays, mHealth solutions can support the management of temporary illnesses and injuries as well as chronic diseases including diabetes, cancer, and asthma.

In the UK, the total percentage of smartphone users has been steadily increasing in recent years and around 87 percent of the adult population owns at least one mobile device in 2020 [7]. In 2021, there were over 350,000 mHealth applications available on the marketplace with over 200 new applications added each day for both general users and healthcare professionals. The number of available mHealth applications has approximately doubled since 2015 driven by increased smartphone adoption and ongoing heavy investment in the digital healthcare market [8]. The most common mHealth applications are concerned on fitness, lifestyle, diet and nutrition, and healthcare monitoring and consulting. They can be divided into 3 main categories: (1) General health and wellbeing, (2) Telemedicine, and (3) Health management. The majority of mHealth applications are free to download, allowing any users with a mobile device the possibility to use.

While dealing with healthcare data, information security is an important concern. Security refers to the safeguards, techniques, and tools used to protect against inappropriate access or disclosure of information [9].

In mHealth systems, healthcare data is collected from users through mobile devices. Collected healthcare data will then be transmitted over a network and stored in different databases including the databases of mobile devices, on-site hospital servers, and Cloud storage. Hence, it is necessary to develop an effective information security framework that will be able to provide a safeguard for mHealth systems.

Health Informatics

Health Informatics (HI) is defined as "the interdisciplinary study of the design, development, adoption, and application of IT-based innovation in healthcare services delivery, management, and planning" [10]. Health informatics is concerned with information and communication technologies that are used in the healthcare sector. The purpose of health informatics is to develop and improve the organisation and management of information and thereby improve the overall quality of care for patients [11]. Health informatics can be identified into different categories including clinical informatics, bioinformatics, computational health informatics, and clinical research informatics [12].

During the last couple of decades, the focus of activity in health informatics has moved from departmental or ward-based systems to institutional-based systems that operate at a regional or global level [11]. Healthcare records have also shifted from paper-based systems to electronic-based systems. Recently, the term eHealth (Electronic Health) has been widely used. It refers to health services and information delivered or enhanced through the Internet and related technologies [13]. In this research, we are particularly interested in the following eHealth sub-categories: Mobile Health (mHealth), Personal Health Records (PHR), and Electronic Health Record (EHR).

Mobile Health (mHealth)

The Global Observatory for eHealth (GOe) defined mHealth as medical and public health practice supported by mobile devices, such as mobile phones, patient monitoring devices, PDAs, and other wireless devices [1]. According to major mHealth activities, it can be identified into four following categories [14]:

• **Physiological monitoring**: measuring, recording, and reporting physiological parameters such as heart rate and blood pressure.

• **Activity and behavior monitoring**: measuring, recording, and reporting movement, physical and social activity as well as health-related behaviors such as eating and addictive behaviors.

• **Information access**: accessing health-related data. For example, medical records, activity, or behavior data and decision-support tools.

• **Telemedicine**: communication between patients and caregivers and/or providers. For example, a virtual doctor visit or a patient receiving personal encouragement from carers

Personal Health Record (PHR)

The definition of a PHR is "An Internet-based set of tools that allows people to access and coordinate their lifelong health information and make appropriate parts of it available to those who need it" [15]. A PHR allows the user to be able to control, add information, have access to his/her medical record, and share when it is needed. A PHR contains a range of healthcare data including health records, physician appointments, prescriptions, test results. A PHR enables the user to conduct various activities. For example:

• Communicate with healthcare professionals through secure communication

• Share daily healthcare information about user's condition such as blood pressure, glucose level, body temperature, disease activity scores.

• Capture and share images or videos of a symptom

• Complete pre-consultation questionnaires in order to help the healthcare professional to get a better understanding of recent medical conditions in order to make best use of face-to-face meetings.

Electronic Health Record (EHR)

According to HIPAA (Health Insurance Portability and Accountability Act 1996), the term EHR means an electronic record of health-related information on an individual that is created, gathered, managed, and consulted by authorised healthcare clinicians and staff [16]. The difference between a PHR and an EHR is that a PHR is mainly managed by a patient, whereas an EHR is managed by healthcare professionals. The EHR contains healthcare-related information including a patient's medical history, diagnoses, medications, treatment plans, immunisation dates, allergies, radiology images, and test results.

1.2 Problem Statement

The growing number of mHealth sectors alerts mHealth application users, healthcare professionals, application developers, and public sectors to be concerned about the security of data and how healthcare data is processed by devices or applications. It cannot be denied that mHealth provides a lot of benefits to healthcare in terms of improving the quality of healthcare which affects the quality of an individual's life. However, it also generates concerns regarding information security [17].

Security difficulties are some of the key challenges in mHealth. They include dealing with those who have the right to access healthcare records, either patients or healthcare professionals, and ensuring that only applications or devices that have been approved will be able to access healthcare data [18]. It is, therefore, necessary that there must be legal requirements that control data mining of healthcare data in terms of personal data protection as well as ethical guidelines. For example, a third party (outside the doctor-patient relationship) must receive permission from a patient to use his/her personal data before conducting future research in healthcare. Indeed, it is a criminal offence for a person to obtain knowingly, recklessly, or unlawfully personal data under Section 170 of the UK Data Protection Act 2018 [6].

In the past, many software vendors of healthcare information and some healthcare providers adopted the philosophy of making it work first, then think about the security later. However, rapid technological changes and new developments have made information security a priority to protect the confidentiality of healthcare information [19]. Various methods should be applied to mHealth systems to secure healthcare data from the exploitation of threats. To ensure the security of healthcare data, different essential security requirements (e.g., Confidentiality, Integrity, Availability) should be applied to mHealth systems.

From the previous studies, several information security frameworks for mHealth devices as well as information security frameworks for Cloud storage have been proposed. However, the major challenge is developing an effective information security framework that provides a complete set of security requirements for mHealth systems.

1.3 Research Aim and Objectives

1.3.1 Research Aim

This research aims to develop a novel information security framework that secures mHealth systems.

1.3.2 Research Objectives

- To understand an mHealth architecture and how healthcare data is collected, stored, processed, and shared in mHealth systems
- To discover a detailed analysis of security issues in mHealth systems
- To identify essential security requirements for mHealth systems

- To propose possible key mechanisms that provide a complete set of security requirements for mHealth systems
- To develop an information security framework for mHealth systems which combines key security mechanisms to achieve security requirements
- To develop a practical prototype of the proposed information security framework for the purpose of implementation, testing, and evaluation
- To develop a Service Management Layer, a Secure Transactional Layer, a Capability system, and an mHealth Secure Storage System that are components of the implemented prototype and the novel contribution of the research
- To evaluate how the implemented information security framework for mHealth systems achieves all the practical security requirements

1.4 Thesis Structure

The purpose of the research is to develop a novel information security framework for mHealth systems. The research covers six major areas including: (1) Investigating security issues to be concerned in mHealth systems, (2) Examining existing studies of information security frameworks for healthcare environments and data storage, (3) The analysis of security requirements, (4) Exploring possible security mechanisms those provide security for healthcare data, (5) Proposing the information security framework for mHealth systems, and (6) Developing a practical prototype which provides an experimental system for the purpose of implementation, testing and evaluation.

Chapter 2 presents the research methodology. The engineering design method is selected to be a predominant research strategy. A series of steps in conducting the research includes: (1) Define the problem, (2) Literature review, (3) Specify requirements, (4) Evaluate a solution, (5) Prototype developing, (6) Implementation, and (7) Communicate results.

Chapter 3 reviews an overview of mHealth systems focusing on an mHealth system architecture. Various information security models were observed to identify a complete set of information security requirement that provides a full protection to mHealth systems.

In Chapter 4, general security approaches for mobile devices are investigated. The finding shows that various approaches (e.g., user and device authentication, encrypted data communication, general policy) are applied to mobile devices to provide security. Existing security frameworks are examined in this chapter. The evaluation presented that all existing security frameworks provide CIA (Confidentiality, Integrity, Availability). However, only a few of them provide Non-repudiation, Authentication, and Authorisation which are also important security requirements for any system. Moreover, possible security mechanisms are studied. These include Encryption, Remote Procedure Call (RPC), Information Technology Service Management (ITSM), Capabilities, Role-based Access Control (RBAC), and Blockchain.

The information security framework for mHealth systems is proposed in Chapter 5. The chapter begins by illustrating an mHealth system scenario to clearly understand the entire workings of an mHealth system and its components. Moreover, identification of assets, threats, and vulnerabilities in mHealth systems is also presented. The security issues regarding mHealth are examined and divided into five subsystems: (1) The provision of AAAC (Authentication, Authorisation, Accounting, and Control), (2) The protection of devices, (3) The protection of healthcare data, (4) The protection of hospital infrastructure, and (5) The access control of physical sites and locations in hospital environments. A taxonomy of an mHealth system is developed. It clarifies the structure of mHealth system into four main components including (1) System architecture, (2) Healthcare data, (3) Stakeholder, and (4) Security requirements. Finally, a new security framework for mHealth systems is proposed. The framework consists of several security mechanisms namely Encryption as a Service, Capability system, Storage management system, Digital filter, Secure transport layer, Blockchain, Secure transactional layer, and Service management layer.

In chapter 6, new mechanisms including: Capability, Secure Remote Procedure Call (SRPC), and Service Management Layer are introduced as key components of the implemented prototype. The implemented prototype is developed for the purpose of testing. Each security mechanism and its benefits are explained in detail.

Chapter 7 describes the implementation of the developed prototype using the relevant technologies in detail. The testing is also conducted in this chapter. Furthermore, the evaluation shows that the developed prototype provides a complete set of security requirements as well as practical security for users, devices, digital data, IT infrastructure, and access to physical sites in hospital environments.

Finally, the research is concluded in Chapter 8. This chapter outlines a summary and results of the research. It also provides details in contributions to the research and the field. Moreover, limitations of the study and future work are discussed.

1.5 List of Publications

The main scientific publications have been published as a part of this Ph.D. research are as follows:

1. Vithanwattana, N., Mapp, G. and George, C. (2016) mHealth – Investigating an Information Security Framework for mHealth Data: Challenges and Possible Solutions. In 12th International Conference on Intelligent Environments. London IEEE, pp.258-261, doi: 10.1109/IE.2016.59.
2. Vithanwattana, N., Mapp, G. and George, C. (2017) Developing a comprehensive information security framework for mHealth: a detailed analysis. Journal of Reliable Intelligent Environments 3, pp.21–39, doi: 10.1007/s40860-017-0038-x.
3. Vithanwattana, N., Karthick, G., Mapp, G., and George, C. "Exploring a

New Security Framework for Future Healthcare Systems," 2021 IEEE Globecom Workshops (GC Wkshps), 2021, pp. 1-6, doi: 10.1109/GCWkshps52748.2021.9681967.

4. Vithanwattana, N., Karthick, G., Mapp, G. et al. Securing future healthcare environments in a post-COVID-19 world: moving from frameworks to prototypes. J Reliable Intell Environ 8, 299–315 (2022). doi: 10.1007/s40860-022-00180-7

1.6 Chapter Summary

Generally, mHealth offers smart solutions to tackle challenges in healthcare. However, there are still various issues regarding the development of mHealth systems. One of the most common difficulties in developing mHealth systems is the protection of healthcare data. mHealth systems are still vulnerable to numerous security issues relating to weaknesses in their design and data management. Therefore, there is a need to develop a comprehensive information security framework for mHealth systems.

Chapter 2

Research Methodology

2.1 Introduction

This research has the aim to develop a novel information security framework for mHealth systems and so it should be viewed as applied science. Applied science is a discipline of science that applies existing scientific knowledge to develop more practical applications, such as technology or inventions. Technology is the study and knowledge of the practical, especially industrial, use of scientific discoveries [20]. Therefore, the technology is supported by scientific knowledge. However, technology starts with a human need or desire and ends with a solution.

2.2 Research Design Method

Realistically, the distinction between science and engineering is sometimes blurry. There is a grey area between science and engineering that the research might fall into. However, if the research has an objective to solve the problem by inventing a new product, system, computer program, experience, or environment, then it is adequate to follow the engineering design process [21].

As a result, the Engineering Design Method was selected as a predominant research strategy. The Engineering Design Process broke down into a series of steps as seen in the diagram below.

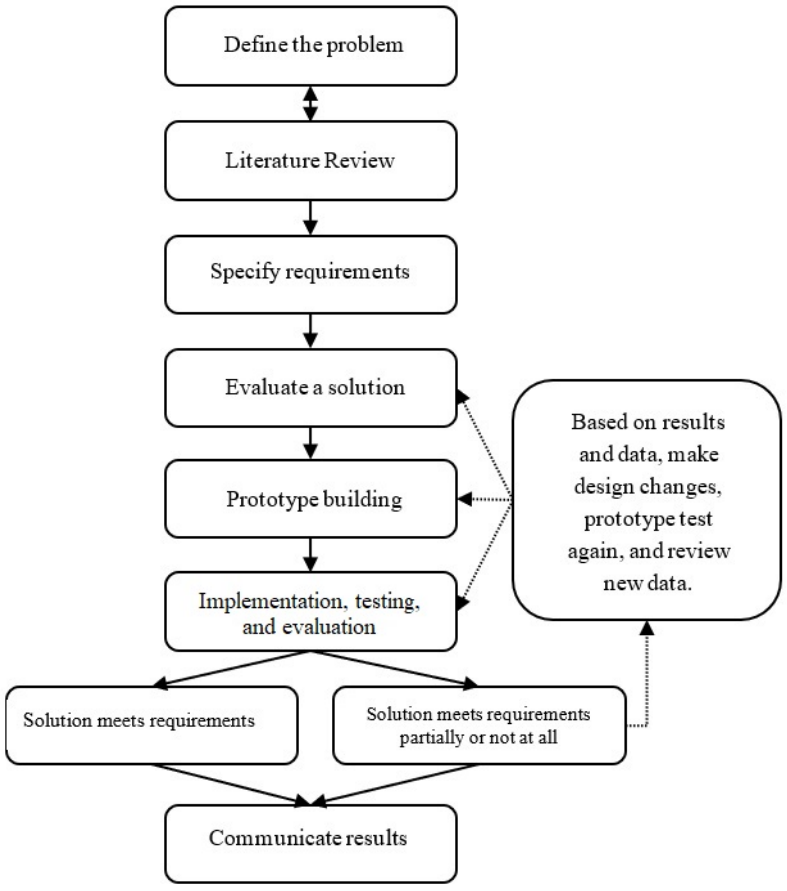

Define the problem
Literature Review
Specify requirements
Evaluate a solution
Prototype building
Implementation, testing, and evaluation
Solution meets requirements
Solution meets requirements partially or not at all
Communicate results
Based on results and data, make design changes, prototype test again, and review new data.

Figure 2.1: The engineering design method

2.3 Methodology

2.3.1 Define the Problem

Firstly, the engineering design method began with defining the problem of the research. The questions were needed to be observed may include: What is the problem or need? Who has the problem or need? And why is it important to solve this problem? Therefore, rather than scientific curiosity, the engineering design research is driven by the need of society. As we are aware, the security of healthcare data is the main difficulty that is faced by mHealth. From previous studies, several information security frameworks for mobile devices as well as information security frameworks for Cloud storage were proposed. So, the question that this research had to observe was *"What information security framework that will provide a complete set of security requirements for mHealth systems could be used as a solution to identify this problem?"*

2.3.2 Literature Review

The second step was the literature review. The literature review refers to the process involved in creating the review that appears in the research [22]. It can also be defined as the selection of available documents (both published and unpublished) on the topic, which contains information, ideas, data, and evidence written from a particular standpoint to fulfill certain aims or express certain views on the nature of the topic and how it is to be investigated, and the effective evaluation of these documents in relation to the research being proposed [23]. The purpose of the literature review is to locate the research project, to form its context or background, and to provide insights into previous work [24]. The literature review should be a coherent argument that leads to the description of a proposed research [25]. This step helps the researcher to discover existing solutions to similar problems and be able to propose a new solution that is more efficient to solve the research problem.

The particular area of mHealth that deals specifically with data collection is called Mobile Data Collection System (MDCS). The literature review of this research started with the observation of MDCS. MDCS allows the collection and transmission of data from remote geographical locations to centrally located data storage repositories through a wireless or cellular network. The study of MDCS helped the researcher to understand the basic components of MDCS, data flow in MDCS, and its security aspects. The research then continued looking into the architecture of mHealth systems.

Major components of mHealth systems, including mobile devices, Cloud storage infrastructure, and connectivity infrastructure, and their interconnections were clarified. The development of the information security model was discussed in this research. Some of the most well-known information security models were identified, including CIA Triad, The McCumber's Cube, The Parkerian Hexad, and IAS Octave.

As a result, an effective information security framework was developed to resolve this problem. General security approaches that can be regularly applied to mobile devices were then discussed. Furthermore, the existing security frameworks were examined to develop an efficient security framework that encompasses both mHealth devices and Cloud storage. Finally, possible security mechanisms were observed to identify which one of them could be developed and combined as major components of the new information security framework for mHealth systems.

2.3.3 Specify Requirements

Once understanding the problem and existing solutions, the third step was to specify requirements. It is crucial to specify the requirements clearly and accurately. Design requirements state the important characteristics that the solution must meet to accomplish the aim of the research. Requirement specifications can be identified from the information that was gathered in the literature review (e.g., existing solutions, possible security mechanisms). This step corresponded to the formulation of a hypothesis or proposing an explanation. After gathering some information in the literature review, a taxonomy of mHealth systems was identified and various security requirements were analysed to discover general issues to concern in mHealth systems, security requirements for mHealth systems, and key mechanisms that provide security requirements for mHealth systems.

The purpose of this study was to discover security requirements that need to be achieved to accomplish the security attributes of mHealth systems. Moreover, assets, threats, and vulnerabilities in mHealth systems were observed. The discussed topic included the identification of assets, threats, and vulnerabilities in mHealth systems and the mHealth system threat analysis. The identification of assets, threats, and vulnerabilities in mHealth systems helped the researcher gain more knowledge about what could happen to cause a loss and gain to mHealth systems as well as how, where, and why the loss might happen.

2.3.4 Evaluate a Solution

The data analysis took place this stage. The findings related to the research objectives guided the study. Once the researcher identified specific requirements, multiple solutions which may correspond to the defined objectives were gathered. This process started with analysing existing solutions in terms of focusing on their weaknesses and how to improve them. Furthermore, several new ideas and methods were generated and compared to produce the best solution for the define objective.

After conceiving alternative solutions that may correspond to the defined objectives, various possible solutions were evaluated for solving the research problems. The researcher needed to observe the best solution that achieves the research objective as well as meets all (or most) requirements including providing a complete set of essential security requirements and protecting healthcare environment components.

In this stage, the researcher analysed security requirements provided by existing security frameworks proposed by Firesmith [26], Takabi, Joshi, and Ahn [27], Brock and Goscinski [28], Zissis and Lekkas [29], and Mapp et al [30], Pirbhulal et al. [31], Rathee et al. [32], and Yahya [33]. Security requirements mentioned in existing proposed security frameworks included confidentiality, integrity, availability, non-repudiation, authenticity, and reliability. However, some essential security requirements, such as non-repudiation, accountability, and auditability, were still missing from most previous proposed security frameworks. Therefore, it was necessary to develop a security framework that provides a full set of essential security requirements which includes confidentiality, integrity, availability, non-repudiation, authentication, authorisation, accountability, auditability, and reliability. Moreover, the developed security framework must also protect major healthcare environment components including users, devices, digital data, IT infrastructure, and locations.

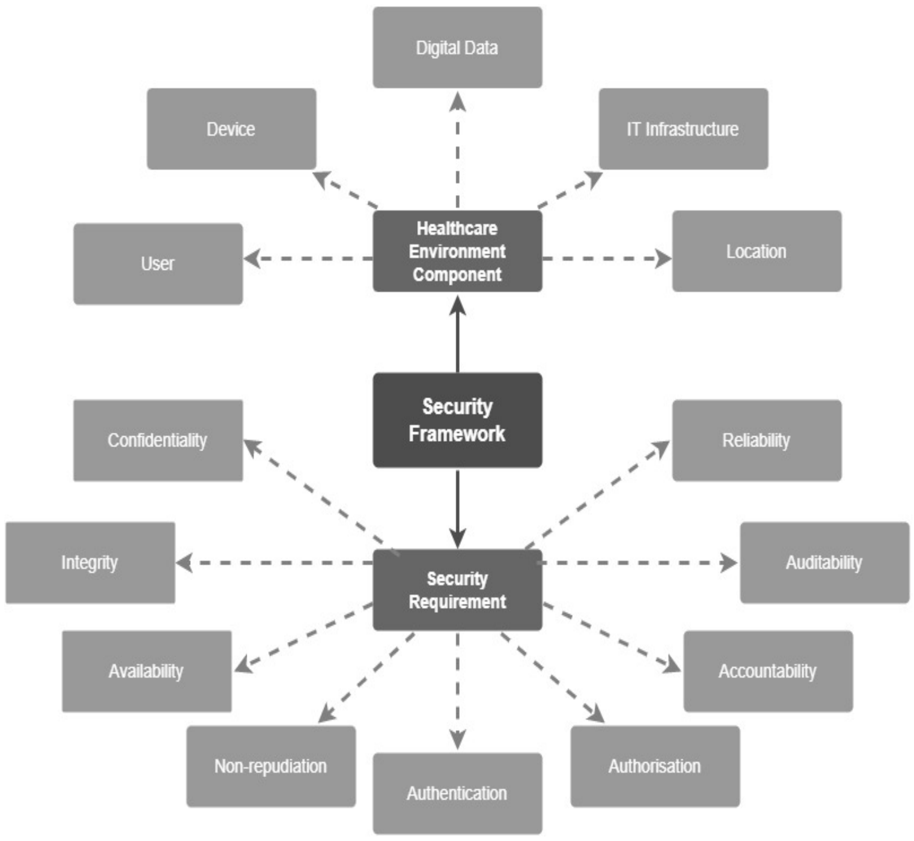


Figure 2.2: The information security framework matrix

After analysing existing proposed security frameworks, key mechanisms for providing possible solutions to manage mHealth systems were proposed. These mechanisms enabled the delivery of essential security requirements required by mHealth systems. Proposed key mechanisms included Encryption as a Service, Capabilities, Storage Management, Digital Filter, Secure Transport Layer, Blockchain, Secure Transactional Layer, and Service Management Layer. Finally, a new information security framework for mHealth systems, that combines key mechanisms mentioned above together, was proposed.

2.3.5 Prototype Building

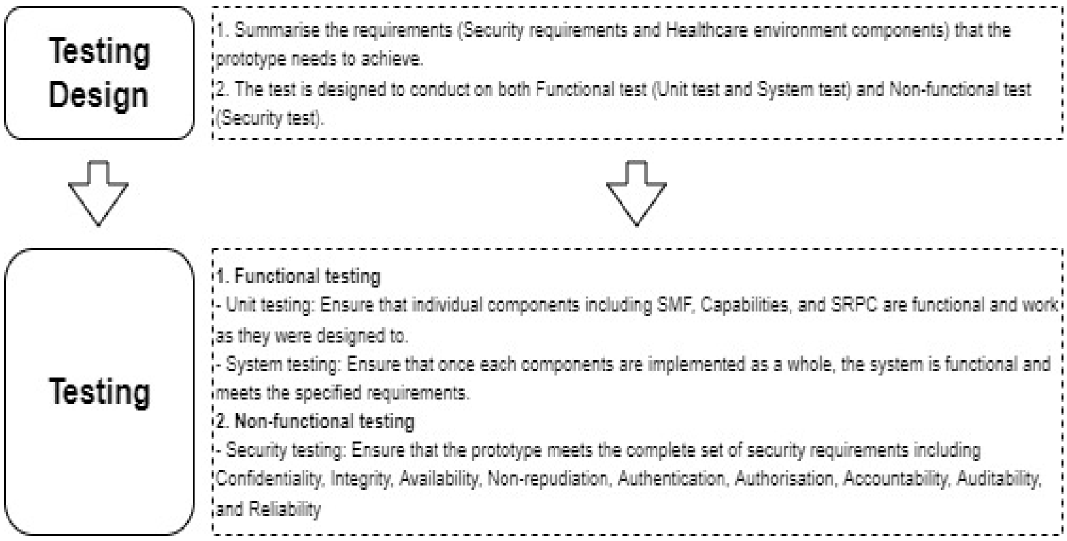
A prototype is an operating version of a solution and a preliminary model of the system. It is built, tested, and then reworked as necessary until an acceptable prototype is finally achieved from which the complete system can now be developed. Building a prototype is a key step in the development of a final solution. This step allowed the researcher to test if the proposed framework provides all essential security requirements.

Since the proposed framework consists of many different mechanisms and each of them is rather complex to implement. Therefore, developing a viable prototype that clearly embodies all features will require a significant effort as well as a time constraint.

As a result, a new security prototype was developed for the purpose of testing. A prototype was developed as a part of novel contributions. The main goal of this prototype is to provide an experimental system that contains the necessary functionalities to meet the security requirements that were identified. The prototype consists of 4 layers: an mHealth Secure Storage Application, Service Management Layer, Secure Transactional Layer, and Capability System.

2.3.6 Implementation, Testing, and Evaluation

This stage involves multiple iterations and redesigns the final solution. If the solution does not meet or partially meet the research requirements, the researcher may have to go back to previous steps to make changes and test new solutions before settling on the final prototype design. A developed prototype from the previous stage was implemented.



1. Summarise the requirements (Security requirements and Healthcare environment components) that the prototype needs to achieve.
2. The test is designed to conduct on both Functional test (Unit test and System test) and Non-functional test (Security test).

1. Functional testing
- Unit testing: Ensure that individual components including SMF, Capabilities, and SRPC are functional and work as they were designed to.
- System testing: Ensure that once each components are implemented as a whole, the system is functional and meets the specified requirements.

2. Non-functional testing
- Security testing: Ensure that the prototype meets the complete set of security requirements including Confidentiality, Integrity, Availability, Non-repudiation, Authentication, Authorisation, Accountability, Auditability, and Reliability

Figure 2.3: The rational for testing

All healthcare services, servers, users, and devices are registered with the Service Management Layer (SManL) in the Service Management Framework (SMF). mHealth applications can find out about services by sending a request to the SManL. The SMF was tested to ensure that the service (i.e., securely storing and retrieving healthcare records) is assigned to a relevant secure server and is given the correct parameters in order to use the designated server.

In the capability system, every object in the system has their capability with assigned access rights to other objects (e.g., Hospital policy documents, EHRs, email). The file directory was created, and each directory was assigned for different types of accesses including Public (Available to access by everyone), Role-based (Available to access only by a designated group), and Private (Available to access only by a designated person). This was tested by checking if each directory can be accessed based on their access right or not. For example, everyone in the system should be able to access the hospital policy documents since they are located in the public directory, whereas the doctor role-based directory should only be able to access by doctors. If nurses try to access the doctor role-based directory, they should be blocked from accessing it.

In a secure transactional layer, a Secure Remote Procedure Call (SRPC) was tested to ensure that all the parameters including the data type with its value passed between the client and the server are encoded. Capabilities were included in this layer to provide authentication and authorisation mechanisms. The result showed that SRPC is more secure than a traditional RPC since SRPC supports the type array which is an improvement from a traditional RPC as type information is not passed in RPC calls.

Moreover, the Filesystem in Userspace (FUSE) was used as a filesystem for an mHealth secure storage application to control how healthcare records are stored and retrieved in the system. In the system, files are managed using inode structures to control access to the file. FUSE was then connected to the Network Memory Server (NMS) which stores blocks of healthcare data in secure random memory (RAM) over the network.

2.3.7 Communicate Results

In this stage, the researcher reviewed the results of the study and identified the main conclusion. Significantly, it is important to go back through the research objectives to see whether the study is able to achieve them. Moreover, the needed further research areas were identified. An important part of this stage was the recognition of the limitations of the study. Finally, the researcher communicated results to others in a form of a thesis and publications.

2.4 Chapter Summary

The Engineering Design Method was applied as a strategy to guide the studies in this research. It contained several stages including (1) Defining the problem of the research that was what the author aimed to achieve, (2) Conducting the literature review about an mHealth system architecture, essential security attributes required by mHealth systems, existing security frameworks, and possible security mechanims, (3) Specifying requirements to develop a taxonomy of mHealth systems, (4) Evaluating a solution by proposing an information security framework for mHealth systems with a combination of various security mechanisms, (5) The implemented prototype was then built to verify the proposed information security framework, (6) The implemented prototype was tested in this stage and the results were evaluated, and (7) The results of this research was communicated in the forms of thesis and publications.

Chapter 3

An Overview of mHealth Systems

3.1 Introduction

This chapter reviews an overview of mHealth systems concentrating on architecture of mHealth systems, and observing various information security models.

The chapter starts with an identification of the system architecture of mHealth systems to discover major components in mHealth systems and to understand how healthcare data is collected, transmitted, and stored in the system. Secondly, general mHealth architecture security issues are described to gain more understanding in weaknesses of mHealth systems. Finally, essential security requirements are identified to achieve the security attributes of mHealth systems. These essential security requirements are therefore the security baselines to identify and take appropriate mitigation to manage risks that may pose harms to mHealth systems.

3.2 An Architecture of mHealth Systems

3.2.1 Mobile Data Collection System (MDCS)

Mobile Data Collection System (MDCS) allows the collection and transmission of data from remote geographical locations to centrally located data storage repositories through wireless or cellular networks. It is a combination of a client application running on mobile devices, wireless infrastructure, and accessibility to remote server databases [34]. Mobile devices such as smartphones, tablets, or PDAs apply MDCS in the system to gather structured data. MDCS requires two-way communication including immediate and delayed synchronisation of data [35]. MDCS consists of three basic components as below:

1. **Data Collection**: Mobile devices are used in the process of data collection. This process could be either manually coded by using Short Message Service (SMS) forms (RapidSMS, FrontlineSMS, and SouktelAidLink) or electronically by using a client application running on a mobile device. However, SMS coded forms-based data collection does not handle complex forms and lacks skip logic and validation techniques. Therefore, more complex forms have been proposed to provide higher security.

• **Native apps**: Specific to a given mobile platform (Android, iOS, BlackBerry or Windows Mobile) using the development tools and languages that the respective platform supports (e.g., Objective-C with iOS, Java and C with Android, J2ME with Windows Mobile and BlackBerry). Native apps look and perform the best [36].

• **Web/HTML5**: Use standard web technologies – typically HTML5, JavaScript and CSS. This write-once-run-anywhere approach to mobile development creates cross-platform mobile applications that work on multiple devices [36].

• **Hybrid apps**: Make it possible to embed HTML5 applications inside a thin native container, combining the best (and worst) elements of native and HTML5 applications .

2. **Form Designer (Administrator Interface)**: It is used to create an electronic form from scratch or used to convert a traditional paper-based form into an electronic form. In the process, the mobile application will also be designed. It also sometimes allows for data entry and viewing of the collected data. The administrator interface acts as the analysis platform and generally provides basic descriptive statistic functions as well as line graphs and bar charts [35].

3. **Data Management**: MDCS uses a centrally located server to manage, distribute form definitions, and aggregating collected data [36]. Once data has been collected, the server will present the data via the administrator client interfaces, and to sometimes mobile devices.

MDCS data flow

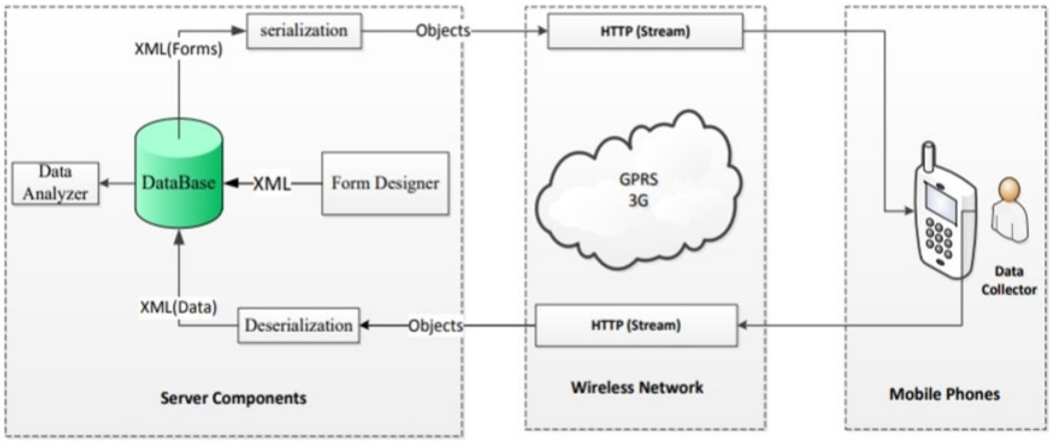


Figure 3.1: MDCS data flow[36]

Firstly, the form definition designs a form that consists of a set of questions for collecting the relevant data and uploads in an accessible server database. Therefore, the collectors will be able to download these forms onto their mobile devices and use them to collect actual data on the field. Each form that has been filled is stored on the mobile device until it is possible to upload them to the central server [36]. The MDCS data flow is shown in Figure 3.1.

MDCS Security Aspects

The research of Gejibo [37], identified the different security aspects of MDCS based on the OWASP Top Ten Mobile Risks as shown below:

1. Insecure data storage: The nature of mobile devices is that they are more likely to be lost or stolen. Most application data on mobile devices is stored openly on their memory card without any appropriate protections. Therefore, it is easy for data to be taken from a mobile device. As a result, data on mobile devices must be encrypted before storing.

2. Insufficient transport layer protection: In wireless communication, if there is no encryption during the communication, it may be easy to eavesdrop on the data traffic and sensitive data might be stolen. Even though, the first router or wireless hotspot may be encrypted. However, there is no guarantee that an entire communication will be secure from a malicious attacker. Therefore, an end-to-end encryption should be applied to the server.

3. Poor authorisation and authentication: Poor authorisation and authentication are major problems on both the client and server. There should be mechanisms that guarantee that access to data is only allowed to its collector, and this can be done only if some form of authentication and access control is in place on the client application.

4. Data recovery: It is important to have some recovery solution in case the collectors or even the project administrators lose their encryption keys or passwords.

5. The process to process communication and separation: The data captured in the form of GPS coordinates, video clips, and pictures are not under the direct control of the application, which only gets a copy or a link to it, and it is, therefore, very difficult to secure it or make sure it is deleted from the phone memory.

3.2.2 Components of an mHealth System

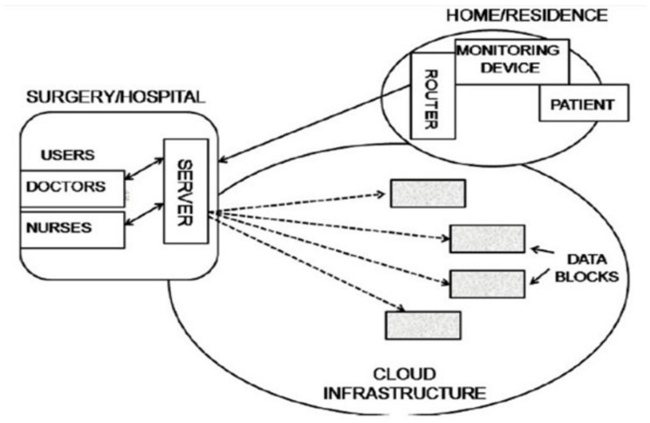


Figure 3.2: mHealth system scenario

As seen in Figure 3.2 above, in mHealth systems, healthcare data is collected from mHealth devices such as wearable or implanted devices. Collected healthcare data is generally transferred to mobile phones/tablets/PDAs, on which mHealth applications are installed. The healthcare data is stored on mobile devices until the data can be transferred over the network to healthcare professionals' servers and then stored in the Cloud. Therefore, healthcare professionals will be able to access their patients' healthcare data through Cloud storage without the need for a physical meeting between the healthcare professionals and patients. The following describes components of mHealth system architecture and their interconnections.

mHealth Devices

Recent advances in the development of Information and Communications Technologies (ICT) have opened new possibilities for using mHealth devices (also known as wearable devices) in the digital health ecosystem to achieve a range of health outcomes. Several large technology companies have introduced various mHealth devices to expand the market of population health.

The definition of mHealth devices is defined as "devices which can be worn or mated with human skin to continuously and closely monitor an individual's activities, without interrupting or limiting the user's motions" [38]. Nowadays, the range of mHealth devices includes a smart watch, a smart ring, a smart wristband, a smart pants, a smart shirt, a smart belt, a smart glasses, a Bluetooth key tracker, and a SGPS/GPRS baby control. Most of mHealth devices can be classified by their uses into five following categories [39]:

- **Health and wellness monitoring**: The devices that monitor physiological data of people and individuals with chronic conditions. This type of device is also widely used by individuals who focus on the condition of their health, fitness, and wellbeing.

- **Safety monitoring**: The devices that have been developed to detect a health condition in people who have a chronic disease (e.g., heart attacks and stroke, cancer) and then send alarm signals to carers or healthcare professionals.

- **Home rehabilitation**: This mHealth technology is sometimes combined with the use of interactive gaming or virtual reality environments to facilitate home-based rehabilitation for patients, aging individuals, and those requiring physiotherapists.

- **Treatment efficacy assessment**: mHealth devices assist in tracking physiological changes from chronic conditions, as well as the progress of treatments that may provide better clinical outcomes.

- **Early detection of disorders**: mHealth devices can be used for early detection of symptoms and adverse changes in a patient's health condition which could help healthcare professionals to make timely medical interventions.

The use of mHealth devices provides multiple benefits including decreasing cost, improving outcomes of treatment, improving disease management, allowing monitoring of chronic diseases, enhancing patient experience, and improving drug management. However, there are still some challenges in different areas such as data privacy and security concerns, lack of common standards, regulatory issues, and patient safety issues. The term of mobile device can be categorised into two sub-units.

A. Body Area Sensor Unit

Body area sensor units are front-end components of mHealth systems. The body area sensors can be identified into two categories which are On-Body contact sensors and Peripheral non-contact sensors. The body area sensors are primarily responsible for collecting healthcare data either directly through on-body contact sensors or from peripheral non-contact sensors providing indirect information of the body and its behaviors [40]. The mHealth sensors directly communicate with the mobile unit (e.g., mobile phones, tablets, PDAs) via Bluetooth or Zigbee.

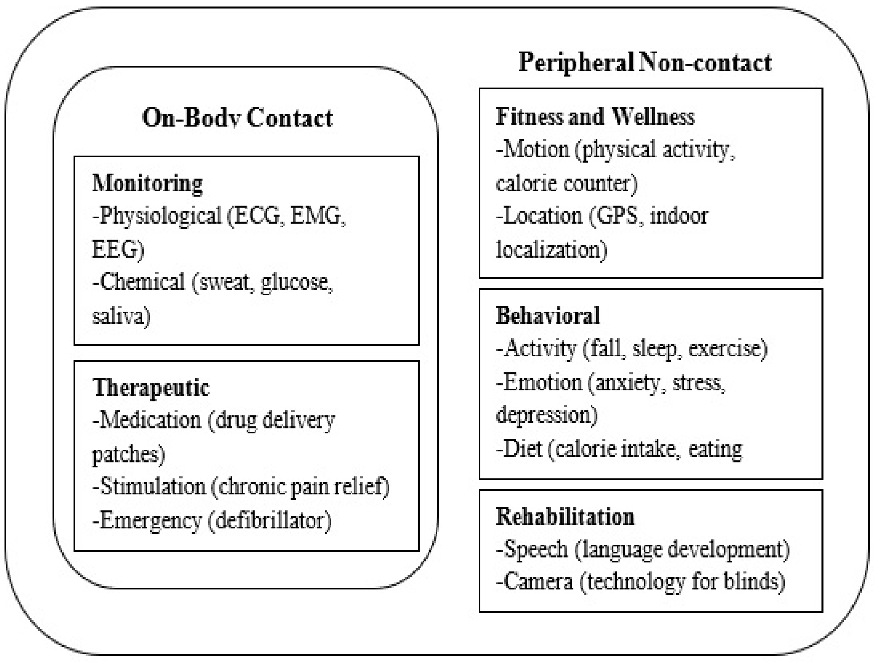

Peripheral Non-contact
On-Body Contact
Monitoring
-Physiological (ECG, EMG, EEG)
-Chemical (sweat, glucose, saliva)
Therapeutic
-Medication (drug delivery patches)
-Stimulation (chronic pain relief)
-Emergency (defibrillator)
Fitness and Wellness
-Motion (physical activity, calorie counter)
-Location (GPS, indoor localization)
Behavioral
-Activity (fall, sleep, exercise)
-Emotion (anxiety, stress, depression)
-Diet (calorie intake, eating
Rehabilitation
-Speech (language development)
-Camera (technology for blinds)

Figure 3.3: Body area sensor categories

B. Mobile Unit

Smartphones, tablets, PDAs are the most common devices used in mHealth systems. An authorised mHealth application is installed on these devices to communicate with the body area sensor unit. A mobile unit receives collected healthcare data from a body area sensor unit. Received healthcare data may be collected in these devices' databases and/or will be transferred to be stored in the Cloud storage.

Cloud Computing Infrastructure

Cloud computing refers to a distributed architecture that centralised server resources on a scalable platform [41]. Cloud computing provides an on-demand, self-service Internet infrastructure that enables the user to access computing resources anytime from anywhere.

Generally, Cloud computing provides several types of services, including virtualised storage and development platforms. One of the well-accepted Cloud computing service models is NIST's SPI (SaaS, PaaS, IaaS) model. The main objective of this model is to encompass various service models of Cloud computing. The SPI model comprises three following categories [42]:

- **Software as a Service (SaaS)**: The Cloud service that provides access to application software on the server. This type of software is referred to as on-demand software. SaaS provides an opportunity for users to reduce the cost of software ownership by removing the need for technical staff to install, manage and upgrade software, as well as reduce the cost of a software licence.

- **Platform as a Service (PaaS)**: PaaS provides users with computing platforms which include databases, programming language execution environments, operating systems. PaaS function is at a lower level than SaaS.

- **Infrastructure as a Service (IaaS)**: IaaS delivers Cloud computing infrastructure including storage, server, operating system, and network as an on-demand service. IaaS is the most flexible Cloud computing service model in which users will have fully control over their infrastructures.

Cloud computing is also classified by the development model. There are four primarily Cloud deployment models. They represent the different categories of the Cloud environment and are mainly distinguished by proprietorship, size, and access.

The four types of Cloud deployment model are:

- **Private Cloud**: The Cloud infrastructure is provisioned for exclusive use by a single organisation comprising multiple consumers.

- **Community Cloud**: The Cloud infrastructure is provisioned for exclusive use by a specific community of consumers from organisations that have shared concerns.

- **Public Cloud**: The Cloud infrastructure is provisioned for open use by the general public.

- **Hybrid Cloud**: The Cloud infrastructure is a composition of two or more distinct Cloud infrastructures (private, community, or public) that remain unique entities, but are bound together by the underlying.

This layer considers the implementation of Cloud storage to store and manage healthcare data. A Cloud computing infrastructure can facilitate the management of healthcare data and can support advanced functionality of data mining, machine learning, and medical big data analytic [40]. Only authorised users (e.g., patient, doctor, nurse) will be able to access healthcare data that is stored in the Cloud.

Cloud computing has attracted more attention from users by providing features such as 24/7 availability and elasticity. However, it still inherits some challenges regarding its weaknesses.

Without doubt, information security issues have become one of the main challenges of Cloud computing. Some security issues, including unauthorised access, data loss, phishing, botnet, pose serious threats to data store on the Cloud.

Connectivity Infrastructure

The connectivity infrastructure refers to the communication technology implemented between each layer of an mHealth system. Healthcare data is generally transferred between a body area sensor unit and a mobile unit via Bluetooth and Zigbee.

However, body area sensor units are rarely standalone systems due to the limit of computing power and communication bandwidth [40]. Hence, there is a need to use a cellular network or a local area network service to transmit collected healthcare data from a mobile unit to the Cloud storage.

An mHealth system architecture is shown in Figure 3.4 below.

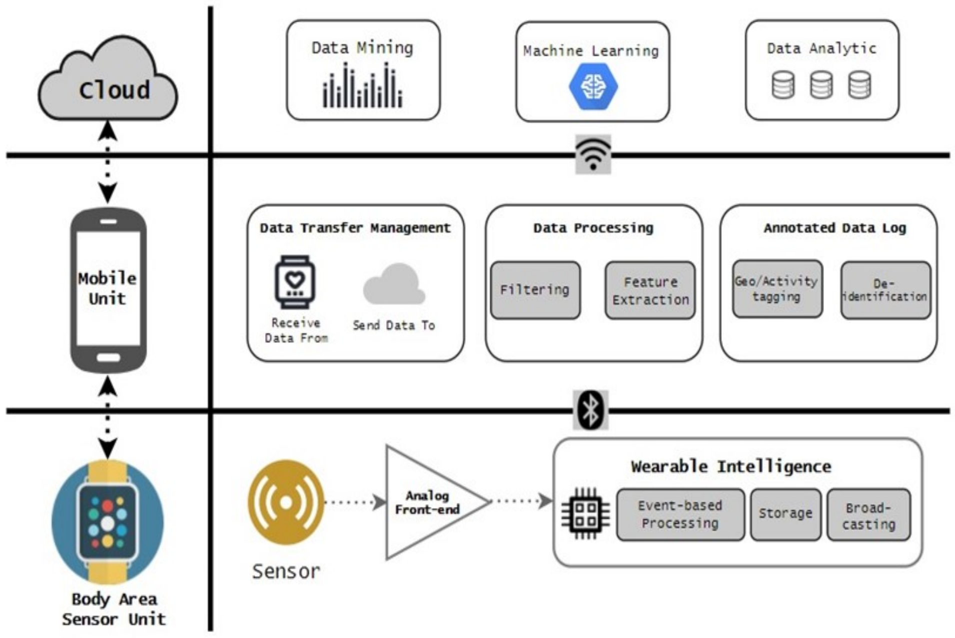


Figure 3.4: mHealth system architecture [37]

3.2.3 General mHealth Architecture Security Issues

The issues regarding information security have been cited as primary obstacles to the development of mHealth. mHealth systems are suffering from several security issues which affect all entities of a system such as users, devices, data storage in varying degrees.

Every application, including mHealth, at the highest level of abstraction tends to have the same basic kinds of potentially vulnerable assets. In mHealth systems, assets include healthcare data, mHealth devices, IT infrastructure, and data storage.

Sensitive healthcare data stored on mHealth devices and Cloud data storage are vulnerable to various security threats which are anything that can exploit the vulnerability, either accidentally or intentionally. In mHealth systems, threats, including unauthorised access to patient records, Distributed Denial of Service (DDoS), and ransomware, can severely pose the greatest risk to the security of healthcare data residing in an mHealth system.

Due to the weakness of design and data management, mHealth systems still have various vulnerabilities which are the weaknesses of the system that can be exploited by threats to damage its assets. An unauthorised user access is one of the most serious security vulnerabilities since it leads to an unauthorised privilege escalation (exploitation of vulnerabilities to gain access to protected data) and to data theft or loss [43]. Therefore, there is a need for effective steps to be taken over the existing security issues in mHealth systems.

3.3 Observing Information Security Models

3.3.1 CIA Triad

The CIA Triad is a simple but widely well-known model for information security policy development. It is used to identify problem areas and necessary solutions for information security. CIA stands for Confidentiality, Integrity, and Availability and these are three main objectives of information security.

• **Confidentiality**: The assurance that data cannot be viewed by an unauthorised user [44].

• **Integrity**: The assurance that data has not been altered in an unauthorised (which includes accidental) manner [44].

• **Availability**: The assurance that authorised parties are able to access the information when needed.

3.3.2 The McCumber's Cube

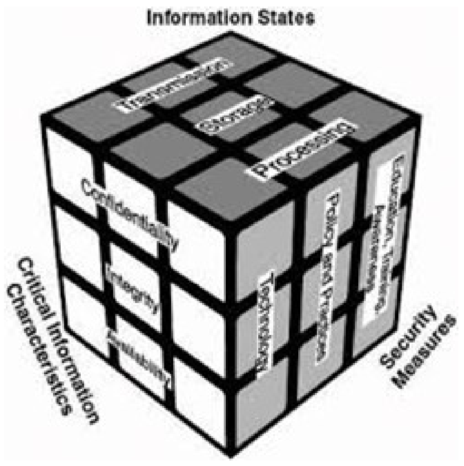


Figure 3.5: McCumber's cube [45]

The McCumber's Cube is the first information security comprehensive model which was proposed by John McCumber in 1991 [46]. The concept of this model is to define the relationship between the communications and computer security disciplines [45]. To develop information assurance systems, the interconnectedness of all the different factors that impact organisations must be considered.

Dimensions and Attributes

1. Safeguards: The X-axis represents the three primary categories of security measures including:

- *Education, Training, and Awareness*: Ensuring that the users can require standards and are aware of their roles and responsibilities regarding the security of information systems.

- *Policy and Practices*: Administrative controls provide a foundation for how information assurance is to be implemented within an organisation.

- *Technology*: Hardware and software solutions designed to protect information systems.

2. Information States: The Y-axis of the model represents the states of information including:

- *Transmission*: Also known as Data in Transit (DIT). This process transfers data between information systems.

- *Storage*: Also known as Data at Rest (DAR). This includes data in an information system that is stored in memory or any kind of data storage.

- *Processing*: The operations perform on data to accomplish an objective.

3. Desired Goals: The vertical axis comprises the three security perspectives based on CIA triad: Confidentiality, Integrity, and Availability.

The McCumber's cube provides the advantage of integrating three separate viewpoints of this classification scheme. However, the dimensions of the cube do not provide a good partition [47].

3.3.3 The Parkerian Hexad

The Parkerian Hexad is a set of six fundamental elements of information security proposed by Donn B. Parker in 1998 [48]. After the CIA Triad was introduced, there was a wide discussion as to its completeness. Some major information security issues, such as Authenticity, are not included within the three core concepts and there is a need to extend this classic trio. The Parkerian Hexad is an expanded version of the CIA triad to form a more comprehensive and complete security model.

The Parkerian Hexad consists of six elements as below:

1. Confidentiality: The quality or state of being private or secret; known only to a limited few [49].

2. Possession or Control: The assurance that the information is in control or possession of the authorised parties.

3. Integrity: Unimpaired or unmarred condition; soundness; entire correspondence with an original condition; the quality or state of being complete or undivided; material wholeness [49].

4. Availability: The assurance that authorised parties can access the information when needed.

5. Authenticity: The assurance that a message, transaction, or other exchange of information is from the source it claims to be from [48].

6. Utility: The assurance that the information is usable for its intended purpose [50].

The Parkerian Hexad considers "Possession" as one of the important elements which may impact confidentiality. It also considers "Authenticity" as a different property that refers to the validity or genuineness of the information rather than just the unimpaired condition of the information.

3.3.4 IAS Octave

In 2013, Cherdantseva and Hilton [51] developed and proposed the Information Assurance and Security (IAS) Octave as an extension of the CIA triad because the CIA triad does not cover new threats that emerge in the collaborative de-parameterised environment. It is essential to integrate these requirements into healthcare systems. The IAS Octave security goal includes [50]:

1. Accountability: An ability of a system to hold users responsible for their actions.

2. Auditability: An ability of a system to conduct persistent, non-bypassable monitoring of all actions performed by humans or machines within the system.

3. Authenticity/Trustworthiness: An ability of a system to verify/identity and establish trust in a third party and in the information it provides.

4. Availability: A system should ensure that all system components are available and operational when they are required by authorised users.

5. Confidentiality: A system should ensure that only authorised users access information.

6. Integrity: A system should ensure completeness, accuracy, and the absence of unauthorised modifications in all its components.

7. Non-repudiation: The ability of a system to prove occurrence/non-occurrence of an event or participation/non-participation of a party in an event.

8. Privacy: A system should comply with privacy legislation and it should enable individuals to control, where feasible, their personal information.

3.4 Chapter Summary

This chapter examined the identification of assets, threats, and vulnerabilities of mHealth systems. It results showed that assets of mHealth systems, including mobile devices, Cloud storage, and network connectivity, are prone to be attacked by one or more threats that may result in harm to the system. Moreover, each of them has several vulnerabilities that could potentially be exploited by threats. Therefore, it is necessary to develop an information security framework to protect the security of assets in mHealth systems.

Chapter 4

Existing Studies: Security Approaches, Frameworks, and Mechanisms

4.1 Introduction

This chapter investigates various approaches to provide security for mobile devices. Existing security frameworks are examined to discover which security requirements they provide and what requirements are missing. Moreover, possible security mechanisms to secure mHealth systems are explored. These include Encryption, Remote Procedure Call (RPC), Information Technology Service Management (ITSM), Capabilities, Role-Based Access Control (RBAC), and Blockchain.

4.2 Existing Studies in Security Approaches and Frameworks

4.2.1 General Security Approaches for Mobile Devices

There are a number of security approaches that can be regularly applied to mobile devices to manage their security. Some of them are:

1. General policy: An organisation should define the security policy regarding which types of the organisation's resources may be accessed via mobile devices and which types of mobile devices can be accessed via the organisation's information technology system. If organisations allow their employees to use personally owned devices (Bring Your Own Device), they will need to have an appropriate policy in place. The centralised technology policy should enforce security on mobile devices. Some policies include managing the wireless network interface, how the organisation's system is administered, restricting user and application access to hardware, or automatically monitoring, detecting, and reporting when policy violations occur [52].

2. Data Communication and Storage: Encryption is a technique used for protecting the confidentiality of data. Strongly encrypted data communications are recommended to be applied between mobile devices and organisations. To prevent potential eavesdroppers, end-to-end encryption (E2EE) should be used in the process of data communications. The encryption should apply to both built-in mobile storage and removable mobile storage. Moreover, mobile devices should be configured to wipe themselves after a certain number of incorrect authentication attempts [52].

3. User and Device Authentication: Password-based authentication is a simple and popular technique for controlling access to data in mHealth systems. However, during a critical medical situation, the owners of a healthcare record may be unconscious and unable to provide the password to access their healthcare records [53]. For this reason, biometrics is another possible solution for regulating access to healthcare records as they are physical attributes that always stay with the owner. User Authentication is another solution to maintaining the confidentiality of healthcare data that is stored in mobile devices. In general, automated authentication mechanisms can be broken down into three categories:

• *Something the user knows* (e.g., password, Personal Identification Number (PIN), passphrase): This is the most common authentication method used for system users. Sometimes, the system may require a minimum length or special characters for the password to minimise the risk of guessing the correct password. As a result, something that the user knows can become something the user forgets.

• *Something the user possesses* (e.g., token, smart card): This authentication mechanism can replace the problem of forgetting a PIN or password. However, the user must carry this object with him/her at all times to gain access to mobile devices. Moreover, such an object might be lost or stolen or can fall into the possession of an attacker [54].

• *Something the user is* (e.g., fingerprint, retina scan, voiceprint): This authentication mechanism is based on something intrinsic, also known as biometrics. Biometrics uses a physical feature unique to a person which can therefore be used to identify that person [55]. The main advantage of using biometrics in authentication mechanisms is that the user will always have the biometric component. In mHealth systems, voice printing and facial recognition have the most potential because of current audio and visual recording standards on many of the latest mobile devices [56]. However, biometric sensors are quite expensive and are sometimes inaccurate. See the comparison of different types of biometrics in Table 4.1:

Type of biometric	Security level	Cost	Size of device
Fingerprint recognition	Medium	Low	Small
Finger vein pattern	High	Medium	Small to Medium
Palm vein pattern	High	Medium	Medium
Facial recognition	Low	Medium	Medium to Large
Iris recognition	High	Medium to High	Large

Table 4.1: The comparison of biometrics [56]

The authentication mechanisms that have been mentioned above may provide some level of security for mHealth devices. Therefore, the Multi-Factor Authentication, which is strong encryption that meets at least two requirements of something the user knows, something the user possesses, and something the user is [56], should be introduced as another solution to mHealth systems to increase the security of the authentication mechanism.

4. Applications: This includes restrictions regarding which applications may be installed on the device, restricting the permissions (e.g., location service, camera) assigned to each application, ensuring that applications have been installed and updated properly, and verifying digital signatures on applications to ensure that only applications from trusted entities are installed on the device and that code has not been modified.

4.2.2 Existing Security Frameworks

Previously, several security frameworks were developed. They identified security requirements that are needed to be achieved to accomplish essential security attributes.

1. A reusable security requirements template was developed by Firesmith [26]. It attempts to provide a comprehensive security framework which can be defined as follows:

- *Access Control*: The degree to which the system limits access to its resources only to its authorised externals.

- *Intrusion Detection*: The degree to which attempted or successful attacks are detected, recorded, and notified.

- *Integrity*: The degree to which components are protected from intentional and unauthorised corruption.

- *Non-repudiation*: The degree to which a party to an interaction or event is prevented from successfully repudiating any aspect of the interaction.

- *Privacy*: The degree to which unauthorised parties are prevented from obtaining sensitive information.

- *Security Auditing*: The degree to which security personnel is enabled to audit the status and use of security mechanisms by analysing security-related events.

- *Physical Protection*: The degree to which the system protects itself and its components from physical attack.

Every application, including those in mHealth systems, at the highest level of abstraction will tend to have the same basic kinds of valuable and potentially vulnerable assets. In mHealth systems, assets include healthcare information, mHealth devices, and Cloud storage. Likewise, these assets are vulnerable to the same basic kinds of security threats from attacks by the same basic kinds of attackers who can be profiled with motivations and their typical levels of expertise and tools.

The idea behind the reusable security template is to develop security requirements that potentially can be reused by or be extended for any system. Unlike typical functional requirements, security requirements can potentially be highly reusable, especially if specified as instances of reusable templates [26]. Although this security framework has not been directly implemented in either the context of mHealth or Cloud computing, it provides a detailed overview of the security requirements for any secure information system. However, this framework can be extended to develop a new information security framework for mHealth systems.

2. A comprehensive Secure Cloud framework was proposed by Ahn, Joshi, and Takabi [27]. These authors proposed a comprehensive security framework for Cloud computing environments that deals with issues such as identity management, access control, policy integration among multiple Clouds, trust management between different Clouds and between a Cloud and its users, secure service composition and integration, and semantic heterogeneity among policies from different Clouds. This framework ensures that only authorised users will be granted access to the stored data. The main focus of this security framework is to understand data protection and resources from a security breach in a Cloud that provides shared platforms and services.

The key components of the Cloud computing environment include Service Integrator and the Security Management Component. Service Integrator has components that are responsible for the establishment and maintenance of trust between the local provider domains and between the providers and the users. The security management component provides the security and privacy specification and enforcement functionality. The comprehensive Secure Cloud framework consists of the six following modules:

• *Access Control Module*: This module is responsible for supporting providers' access control needs. Role-Based Access Control (RBAC) has been introduced as a method used in the Access Control Module.

• *Policy Integration Module*: SAML, XACML, and WS standards are viable solutions that satisfy the need for a Specification framework to ensure that cross-domain accesses are properly specified, verified, and enforced.

• *Service Management Module*: Responsible for secure service discovery, composition, and provision but using an approach that also considers security and privacy issues.

• *Trust Management Module*: Responsible for negotiation, establishment, and evolution in the overall system.

- *Heterogeneity Management Module*: Responsible for providing a global ontology and supporting semantic heterogeneity concerns related to policies.

- *Authentication and Identity Management Module*: Responsible for authenticating users and services based on credentials and characteristics.

3. Cloud Security Framework (CSF) was suggested by Brock and Goscinski [28]. To develop the Cloud Security Framework (CSF), Brock and Goscinski characterised the security problems of Clouds, evaluated the security of current Cloud environments, and presented current security countermeasures. The main focus of this security framework is on Cloud infrastructure protection, communication and storage security, authentication, and authorisation.

The Cloud Security Framework (CSF) is influenced by the Information Flow Control Model and Kerberos. There are two main components in CSF: a Gateway Server (GS) and a Single Sign-on Access Token (SSAT). GSs are located in the Cloud and manage the security of those Clouds in which they are located. A SSAT is a one-time token that is a time-limited, non-forgeable, and non-transferable entity, which is granted to Cloud users. This token identifies the user, services that the user wishes to use, and provides verification tokens to prove the SSAT itself is valid. Only authorised users are allowed to use the token. Moreover, the token cannot be reused once it expires.

In terms of functionality, the CSF is similar to the Information Flow Control model which uses trusted capabilities and some elements from Kerberos. The CSF framework aims to use time-based access to grant authorised users access to the Cloud, protects against forgery of authorisation and grants access to services in remote Clouds on behalf of users.

4. Zissis and Lekkas examined **user-specific security requirements for end clients** [29]. They proposed a security solution to a number of challenges in a Cloud environment, including Confidentiality and Privacy, Integrity, and Availability, by using

a trusted Third Party. The authors proposed a Trusted Third Party (TTP) service as a solution to providing end-to-end security services in the Cloud environment. Trusted Third Party services will lead to the establishment of the necessary Trust level and provide ideal solutions to preserve the confidentiality, integrity, and authenticity of data and communications. Moreover, TTP is an ideal security facilitator in Cloud environment where entities belonging to separate administrative domains, with no prior knowledge of each other, require the establishment of secure interactions.

In a Cloud environment, the users are required to use their digital certificates (sometimes called TLS/SSL certificates), as a reliable passport, to authenticate themselves to a Cloud service provider to validate their access rights to the Cloud resources. This certificate is used in combination with the service provider's certificate to create a secure SSL connection between them, thus encrypting exchanged data and assuring their security through the Cloud infrastructure. The application providers are required to use their digital certificates to authenticate themselves when communicating with the Cloud as well as encrypting and decrypting application data. The proposed solution makes use of a combination of Public Key Cryptography, Single-Sign-On technology, and Lightweight Directory Access Protocol (LDAP) to securely identify and authenticate implicated entities in a Cloud environment system.

5. **A security framework based on Capabilities** was proposed by Mapp et al [30]. A capability is a communicable, unforgeable token of authority. It refers to a value that references an object along with an associated set of access rights granted to the user of the capability. A user program on a capability-based operating system must use capabilities to access objects. A capability is defined to be a protected object which, by virtue of its possession by a user process, grants that process the right to interact with an object in certain ways. The capability logically consists of a reference that uniquely identifies a particular object and a set of one or more of these rights. In this framework, everything in the system including users, devices, and patient data must be represented by a capability. Capabilities, therefore, need to be

carefully managed and need to be protected against being created or changed in an inappropriate manner [30]. The security framework defines five layers including user, application, hypervisor, transport and storage, and the method that happens in each layer. As seen in Figure 4.1 below:

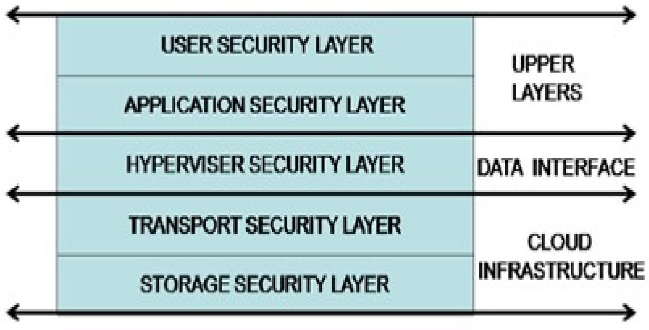


Figure 4.1: Security framework based on Capabilities [30]

• *User Security Layer*: In this layer, the Cloud users will authenticate themselves to local devices and applications. This authentication will enable authorisation to grant access to authorised users to access application resources.

• *Application Security*: This layer is used to authenticate the application to the hypervisor and is responsible for Presentation Security which encodes and decodes data between the application and the Cloud Storage System.

• *Hypervisor Security Layer*: This layer is used to authenticate the application and user security layers to the Cloud Infrastructure and is also used to generate capabilities that allow applications to access the required resources in the Cloud Infrastructure.

• *Transport Security Layer*: This layer provides the security of moving data between the application and the Cloud Infrastructure by using the Simple Protocol (SP) [30] which is a mechanism that provides quick authentication using a key exchange.

• *Storage Security Layer*: This block uses encryption techniques including Data Encryption Standard (DES) and Advanced Encryption Standard (AES) algorithms to secure blocks of data in the Cloud Infrastructure.

However, this security framework does not specify security requirements for Cloud storage in general. This is because this framework has been derived from the Firesmith framework [26] and looks at the functions of different parts of the Cloud system to provide secure Cloud storage.

6. A joint resource-aware and security framework for the Internet of Medical Things (IoMT) based remote healthcare systems was proposed by Pirbhulal et al. [31]. This research considered the following requirements: Data Confidentiality, Data Integrity, Data Availability, Data Freshness, Scalability, and Secure Key Distribution, as critical security and privacy requirements for IoMT based healthcare. The framework applied a bio-keys generation mechanism for medical data encryption. The authors argued that this framework can assure the security of medical data transmission between patients and physicians as well as decrease the economical healthcare solution.

7. A hybrid framework for IoT – Healthcare using blockchain technology was proposed by Rathee et al. [32]. They claimed that currently, blockchain technology is the best technique to provide secrecy and protection for control systems in real-time conditions. Results of their research showed that this framework offers an 86 percent success rate and can prevent wormhole and falsification attacks. However, the drawback of their framework is that hashing of all blocks (nodes) becomes very complicated to predict since the entire network is maintained by the blockchain.

8. Research by Yahya in [33] aimed to develop **a security framework for Cloud storage**, which is the main component of mHealth systems, by exploring existing proposed security frameworks. The result from this research will identify which security requirements can be used as baselines to protect data in Cloud storage. Security requirements mentioned in this research include (1) Security policies implementation; (2) Data access protection; (3) Modifications of data stored; (4) Data accessibility; (5) Non-repudiation; (6) Authenticity; (7) Reliability; (8) Accountability; and (9)

Auditability.

According to interview results [57], all security experts commented that all proposed security requirements are important. However, Confidentiality, Integrity, and Availability are the most basic and common security requirements to form any security framework.

While the previously discussed frameworks achieve some protection of healthcare data, there is still a need to develop a more comprehensive security framework that protects the end-to-end security of an mHealth system. This means security from when healthcare data is collected, then transferred over the network, and stored in both on-site storage and Cloud storage. The framework must support the necessary security requirements (Confidentiality, Integrity, Availability, Non-repudiation, Authentication, Authorisation, Accountability, Auditability, and Reliability) and protects every healthcare environment component (devices, hospital infrastructure, digital data, healthcare data storage) in healthcare systems.

Tables 4.2 and Table 4.3 show the comparison of different security frameworks mentioned above in terms of providing security requirements and protecting healthcare environment components. The two tables demonstrate that no existing framework caters to all security requirements and all healthcare environments at the same time.

Security requirement	**[26]**	**[27]**	**[28]**	**[29]**	**[30]**	**[31]**	**[32]**	**[33]**
Confidentiality	*	*	*	*	*	*	*	*
Integrity	*	*	*	*	*	*	*	*
Availability	*	*	*	*	*	*	*	*
Non-repudiation	*						*	*
Authentication	*	*	*	*	*			*
Authorisation	*	*	*		*			
Accountability		*					*	*
Auditability	*						*	*
Reliability			*	*	*			*

Table 4.2: Synthesis of security requirements

Healthcare environment components	**[26]**	**[27]**	**[28]**	**[29]**	**[30]**	**[31]**	**[32]**	**[33]**
Device	*				*			
Hospital infrastructure	*				*			
Digital data	*				*	*	*	
Cloud storage	*	*	*	*	*			*

Table 4.3: Synthesis of healthcare environment component

4.3 Possible Mechanisms to Secure mHealth Systems

4.3.1 Encryption

Encryption is a process of concealing data or information (Plaintext) and converting it into a code (or known as "Ciphertext") so that only authorised parties are able to access it. Only authorised users can access encrypted data using an encryption key, then decrypt it using a decryption key. The main purpose of encryption is to provide the confidentiality of data. The main components of an encryption system include plaintext, key (public key/private key), encryption algorithm, and ciphertext.

There are several types of encryption methods. Each of them has been developed with different techniques to achieve different needs. The two main types of encryptions are symmetric encryption and asymmetric encryption. Both are described as below.

• In *symmetric encryption*, the encryption key and the decryption key are essentially the same [44]. The symmetric key is used to encrypt the plaintext and decrypt the ciphertext. Some common symmetric encryption methods include the Data Encryption Standard (DES), the Advanced Encryption Standard (AES), Triple DES, and Twofish.

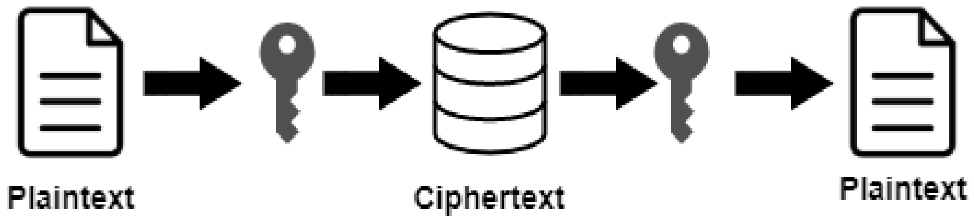


Figure 4.2: Symmetric encryption

• In *asymmetric encryption* (sometimes referred to as "public-key encryption"), the encryption key and the decryption key are different [44]. These two keys are known as a public key and a private key. A private key remains secret while a public key is available to anyone who needs it. There are various asymmetric encryption methods such as Rivest Shamir Adleman (RSA), the Diffie-Hellman exchange method, TLS/SSL protocol, etc.

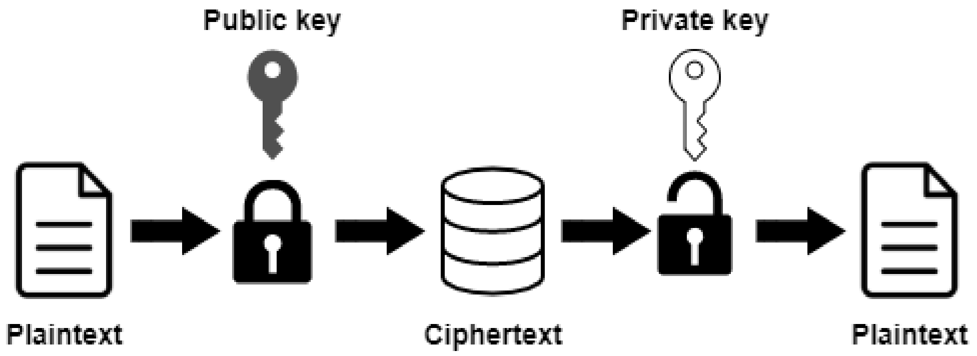


Figure 4.3: Asymmetric encryption

Encryption may protect unauthorised users from knowing the content of plaintext that is encrypted, however, encryption does not prevent communication interceptions as well as does not guarantee end-to-end confidentiality.

4.3.2 Remote Procedure Call (RPC)

RPC (Remote Procedure Call) is a technique of inter-process communication between client and server to exchange data or execute some instructions. It provides the high-level communications paradigm used in the operating system [58]. The two processes, the client process and the server process, could be on the same system or on different systems with a network connecting between them. RPC is a crucial part of client-server-based applications. It supports the communication between client and server to be more secure as well as efficient.

RPC Major Components

1. Client Application

The client application is a program that makes remote procedure calls to request operations. It defines the user interfaces, the calls to the remote procedures of the server, and the client-side processing functions. The client is responsible for sending instructions to the server. The client application reads data from the file and will encode every single instruction before processing them. After that, it will make a procedure call and will pass the argument to the client stub.

2. Server Application

The server application is a program that performs the operation. It implements the calls offered by the server. The server application receives a request for an operation from a client and sends a response containing the result of the operation. The server application decodes the encoded arguments received from a client and verifies each argument before performing a final operation. In case of any mismatches are found, it will send an error back to the client and terminate the current process. If argument validation is successful, the server will perform the requested operation, and will then encode the response before sending it back to the client.

3. Client/Server Stub

A stub is a communication interface that implements the RPC protocol and specifies how messages are constructed and exchanged. Stubs are placeholder functions that make the calls to the runtime library functions which manage the remote procedure call. A stub is an application-specific code, but it is not directly generated by the application programmer. The programmer specifies interfaces using an Interface Definition Language (IDL), and the IDL compiler generates stub automatically from the specification. The client and server process communication using two stubs: a client stub and a server stub.

When an application calls a remote procedure, the stub performs the process called "Marshalling" which is preparing the input arguments for transmission. A stub will also unmarshall received arguments. Unmarshalling is a process of disassembling incoming network data and translating it into a standardized format that a local system can understand.

Both marshalling and unmarshalling occur twice in each remote procedure call. The client stub marshalls input arguments and unmarshalls output arguments. On the other hand, the server stub unmarshalls input arguments and marshalls output arguments [59].

4. RPC Run-time Library

The core of RPC model is the RPC run-time system, which is a library of routines and a set of services that handle the network communications that underlie the RPC mechanism. The RPC run-time library contains the routines, tables, and data that support the communication between client and server. Run-time operations perform tasks including controlling communication between client and server, establishing communications over an appropriate protocol, and handling communication errors.

In addition to handling all communications between client and server applications, the RPC run-time library provides the following utilities:

- An interface that lets applications access various name servers (which can be used to locate various network resources).

- Management services such as monitoring servers, monitoring run-time operations, and stopping servers [60].

A client and a server stub exchange arguments through their local RPC run-time. The client run-time sends the request to the server, whereas the server run-time accepts the request and calls the server stub procedure. Moreover, the server run-time also returns the call result back to the client run-time over the connection.

When an RPC is established, the client initiates the request for connecting to the server and waits for a response to be returned from the server. Once successfully establishing the connection to the server, the client sends instructions to the server which processes the request and returns it back to the client. While waiting for the response from the server, the client will not be able to execute any further instructions. Figure 4.4 shows the flow of activity that takes place during an RPC mechanism between two systems.

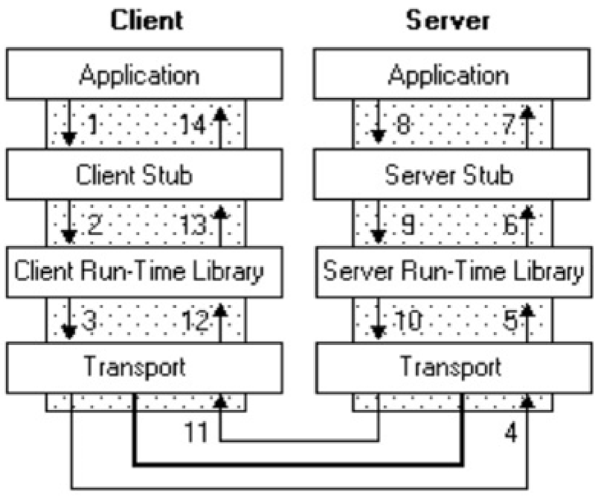


Figure 4.4: Remote Procedure Call Flow [61]

1. Client application invokes the procedure – Makes a procedure call and passes the argument to the client stub. Since the client stub is linked to the client, no network connection needs to be made.

2. Client stub marshalls the parameters – Client stub will convert the parameters to a standardised message format and copy them into a message to reprocess called “Marshalling”.

3. Calling function in the RPC client run-time library – The RPC client run-time library sends the request and its parameters to the server.

4. The message is sent to the server – The message is sent to the server using binding information that is given. Binding is a process in which the client to connect to the server. This can be done statically or dynamically. The static binding uses hard code binding information. Dynamic binding is more complex and adds another layer to the tier architecture.

5. The server RPC run-time library functions accept the request – After the server RPC run-time library functions accept the request, they call the server stub procedure.

6. The server stub receives and unmarshalls the messages – When a message arrives at the server, it is received by the server stub. Since the argument has been marshalled, it must be translated back into the format that is usable for the server-side procedure call. This is done in a process called "Unmarshalling".

7. The server stub invokes and passes the argument to a server application - Once the argument has been converted back to the proper format, the server stub will invoke and pass the argument to the procedure in a server application.

8. The server application executes the procedure and returns the result to the server stub – Once the procedure finishes the execution, the server component will return the result back to the server stub. The server stub then needs to marshall the result into a standardized format.

9. The results are returned back to the server RPC run-time library function – This process does not need to bind to the client as the connection is already established by the client.

10. The server RPC run-time library functions return the message back to the client – After receiving the result from the server stub, it returns the message back to the client through the connection.

11. The client accepts the message – After the message is transmitted over the network, the client can complete the process by accepting the message and returning it to the calling function.

12. The client RPC run-time library receives the message – It receives the remote-procedure return values and passes them back to the client stub.

13. The client stub unmarshalls the result - The client stub unmarshalls the result to the message and then returns it to the client application.

14. The client application receives and processes the results – The client can now close the connection to the server.

RPC provides an authentication process to verify the client and server to each other. RPC has been widely applied as a security mechanism in a client-server environment since it is powerful yet simple and flexible. It supports both process-oriented and thread-oriented models. RPC offers a higher-level abstraction and provides numerous advantages including maximise security and efficiency.

4.3.3 Information Technology Service Management

Information Technology Service Management (ITSM) refers to an approach to IT operations that is characterised by its emphasis on IT services, customers, service level agreements, and an IT function's handling of its daily activities through processes [62]. The service management platform is responsible for operating and monitoring applications, data, and services residing in the system. The service management platform ensures that resources are working properly and optimally interacting with users and other services. The service management covers some basic processes as follows:

1. Service catalogue management: Service catalogue acts as knowledge management for users. It contains information about a description of the service, processes for requesting the service, and other service-related topics.

2. Service-level management: Ensuring that the system provides an adequate level of service, as well as maintaining and improving the level of service.

3. Availability management: The main goal of availability management is to define, analyse, plan, measure, and improve all aspects of the availability of services. It is responsible for ensuring that all infrastructures, processes, tools are appropriate for the agreed availability targets [63].

4. Capacity management: Managing and monitoring the resource capacity in the system. Lack of capacity information can result in too many workloads and slow application response times.

5. Service continuity management: It covers the processes by which solutions are launched and managed to ensure that services will be able to recover and continue after an unexpected incident occurs.

6. Security management: It assures that an adequate information security framework is applied to the system to prevent unauthorised access from malicious users.

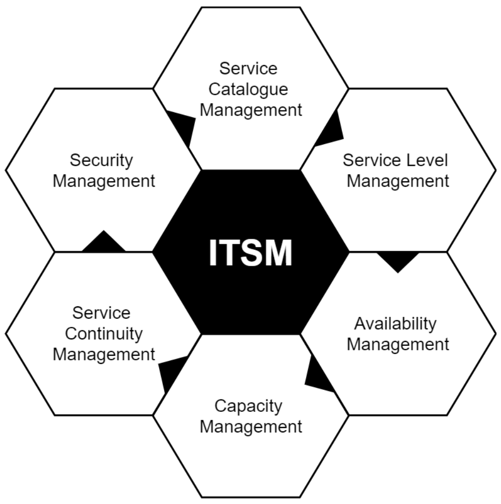


Figure 4.5: Service management process

ITSM ensures that IT services provided by an organisation are delivered in a way that meets requirements stated by end-users. It provides several benefits such as reducing cost, providing better services, increasing efficiency and productivity, and decreasing disruption. Moreover, ITSM will be able to develop to support the Internet of Things (IoT) and social media integration.

4.3.4 Capabilities

The term “Capabilities” originated in 1966 and was coined by Dennis and Van Horn [64]. It refers to a token that permits authorised users to access certain objects in a system.

Over the years, research into capability systems diverged in two different directions. The first was the development of hardware capabilities which were used to control physical access to computer resources as seen by the development of the Cambridge CAP Operating System in the late 1970’s [65]. This system had a Capability Unit (CU) which prevented the execution of programs unless the right capability was loaded into the CU. Modern processor designs made by Advanced RISC Machines (ARM) also use this method to restrict access to certain memory regions, hence, increasing the security of the Operating System [66].

Software capabilities have also been explored as seen in the design of the Amoeba Distributed Operating System [67]. A capability was used to invoke operations on a system server via a communication port. To send data to the port, the correct capability had to be provided.

Dennis and Van Horn’s Supervisor

Dennis and Van Horn [64] defined a hypothetical operating system supervisor for a Multi programmed Computer System (MCS) and its use of capabilities. They presented the concept of capability addressing which can be widely found in many hardware and software systems. The operating system supervisor can be defined into two main components including a set of objects and a set of object operations. These operations, implemented by the supervisor, are called meta-instructions. To fully understand meta-instructions, the following concepts and terminology are described below.

1. Segments: The unit that stores information that is of interest in the present discussion or known as a word.

2. Protection: Only authorised accesses are able to access memory words and other objects of a system. In an environment, an executed process is called a sphere of protection and it is specified by a list of capabilities (C-list). Objects in the system are named and given access rights by each capability in the C-list.

3. Processes: A thread of control that is capable of executing algorithms that are specified by sequences of instructions. Processes must indicate the C-list applicable to the computation to which the process belongs.

4. Computations: A set of processes that are working together on the same job. A set of processes that share a common C-list are members of the same computation.

5. Principals: An individual or group of individuals to whom charges are made for the expenditure of system resources or known as users of the system.

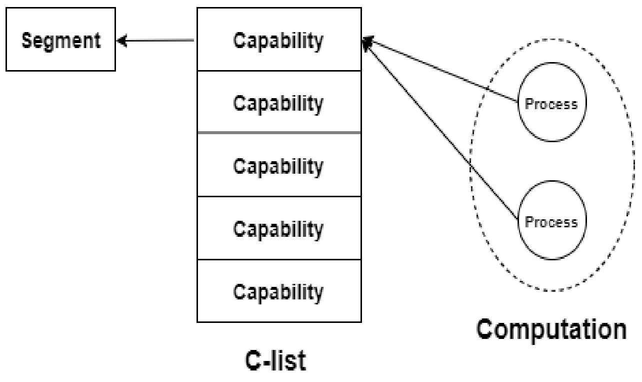


Figure 4.6: A relationship between processes, computation, and C-list

Dennis and Van Horn conceptual design became widely implemented on later systems. However, there are still many ways that the concept of capability can be applied and developed. Others will be examined in the following sections below.

The Cambridge CAP Operating System

The concept of the Cambridge CAP Operating System [65] was developed due to the need for hardware memory protection since operating systems that allow to run users' programs written in various languages came into use.

The memory of a machine with a capability architecture contains two main components, namely, data words (which term includes instructions) and capabilities. Data words and capabilities are stored separately in different segments and the same segment cannot contain both data words and capabilities. It is necessary to assign a capability to access a segment of memory. Hence, capabilities must not be forgeable.

In the CAP system, a process is a basic execution and must be protected. A process can be defined by a set of data words that describes a list of accessible segments and other resources.

As seen in Figure 4.7, the CAP system structure is a process tree structure that eliminates a privileged mode of operation. The Master coordinator allocates on the top of a process tree (Level 1). It controls all system hardware resources, which allocate among user processes (Level 2). Therefore, user processes can create further subprocesses acting as a coordinator for them.

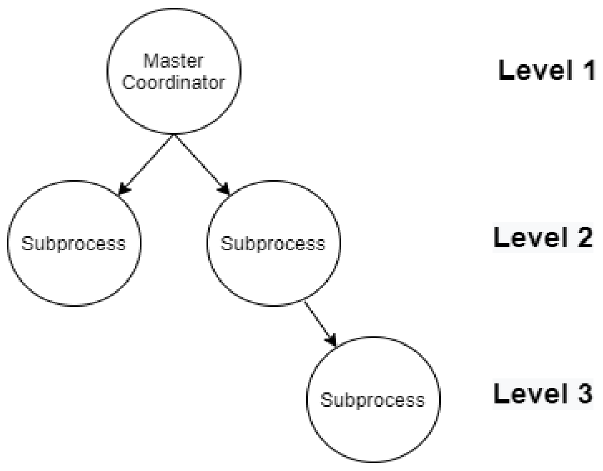


Figure 4.7: CAP process tree

The set of capabilities available to the master coordination are contained in a segment, namely, the Master Resource List (MRL), and the set of capabilities available to a subprocess running under the master coordinator are contained in a segment, namely, the Process Resource List (PRL). Therefore, each subprocess has only one PRL.

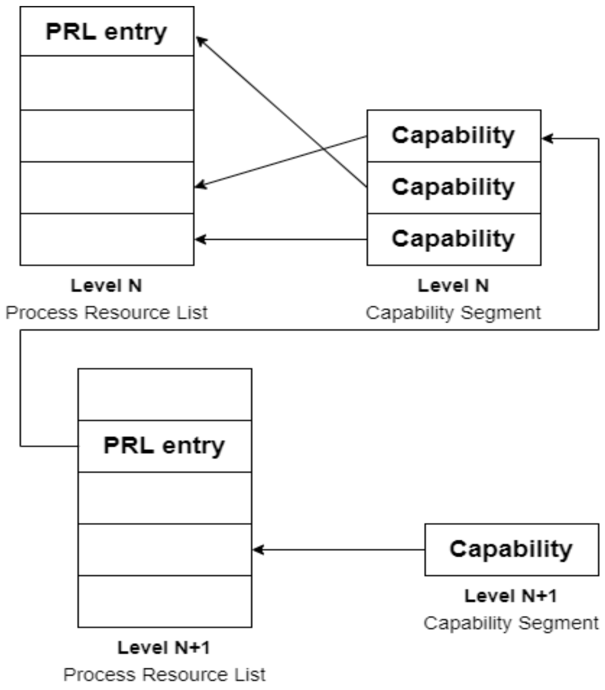


Figure 4.8: The CAP process addressing

In the CAP system, a process must acquire a capability for any object to be accessed. Capabilities, therefore, are stored in capability segments. There are six capability segments including G, A, N, P, I, and R. The G capability segment contains global capabilities and cannot be changed during the process life. The P, I, and R capability segments are changed when an ENTER instruction takes effect and the process enters a new protected procedure. The P segment is commonly used to contain capabilities for code segments. The I segment contains capabilities for a stack and other segments peculiar to the process. And the R segment contains capabilities for data segments permanently associated with the procedure.

The Cambridge CAP operating system is the first successful university-built capability system. Since it became operational, it has proven to be flexible for further research and experimentations with a capability architecture.

The Amoeba Distributed Operating System

The Amoeba Distributed Operating System was developed in 1983 by Tanenbaum et al. [67]. It is a distributed system that allows several machines connected over the network to operate as a single system. The purpose of developing the Amoeba system is to create a transparent distributed system that allows users to log into the system as a whole [68].

In Figure 4.9, the Amoeba hardware consists of four components including workstation, pool processor, specialised server, and gateway.

• *Workstation*: The workstations execute only processes that require intense user interaction. They allow users to gain access to the Amoeba system.

• *Pool processor*: The pool processor consists of many processors which contain private memory and a network interface. The group of processors can be dynamically allocated as required by users and the system. They can be allocated to run many compilations in parallel.

• *Specialised server*: Specialised servers are machines for running dedicated processes with unusual resource demands. They carry out and synchronise the fundamental operations of the kernel.

• *Gateway*: Gateways allow other Amoeba systems to access over Wide Area Network (WAN). It protects local machines from the idiosyncrasies of protocols that must be used over the wide-area links.

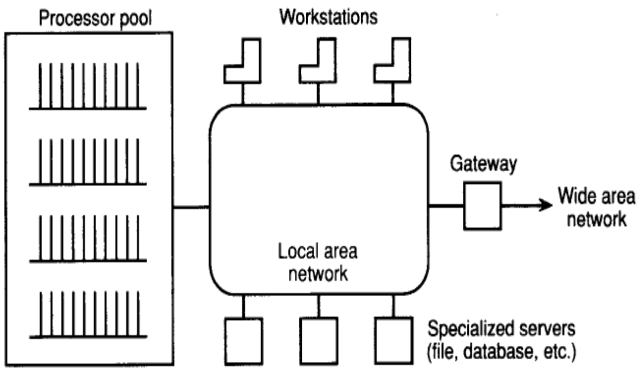


Figure 4.9: Amoeba hardware architecture [67]

The Amoeba software is an object-based system using clients and servers. Client processes use remote procedure calls to send requests to server processes for carrying out operations on objects. An object is a piece of data on which well-defined operations can be performed by authorised users, independent of the user's and object's locations. Objects are managed by server processes and named using capabilities chosen randomly from a sparse namespace [67]. Capabilities provide a set of operations and contain cryptographic protection so that it is not possible to guess an object's capability. In the Amoeba system, the key protection is to keep capabilities confidential by embedding them in a huge address space. The capability structure of the Amoeba system is shown in Figure 4.10 below.

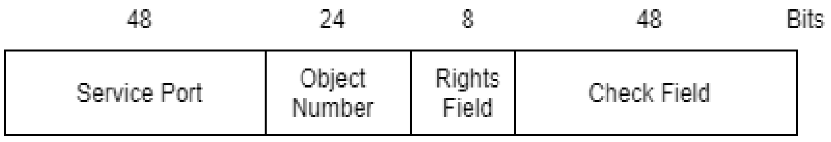


Figure 4.10: Amoeba capability structure [67]

Objects are implemented by the server processes that manage them. The Service Port field identifies the server process that manages the corresponding object. The Object Number field clarifies the object. The Right Field manages an allowance of operations. The Check Field must be protected cryptographically to prevent tampering by users.

In the Amoeba system, many processors are connected, and a single program can use multiple processors. This provides advantages of increasing performance and simplicity. It is suitable for an environment that consists of multiple computers.

4.3.5 Role-based Access Control (RBAC)

Role-based access control (RBAC) was originated in 1992 by the US National Institute of Standards and Technology (NIST) [69]. It refers to the network access restrictions based on the roles of users within an organisation. In RBAC, a role is essentially a collection of permissions. All users in the system received permissions to access resources based on the roles to which they are assigned.

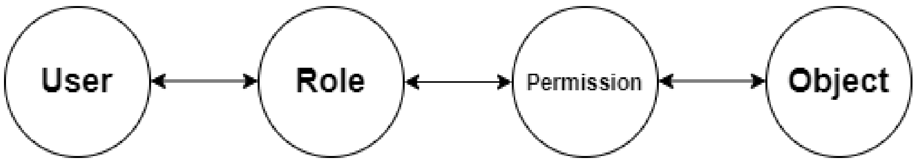


Figure 4.11: RBAC model

The NIST RBAC model is organised in a four-step sequence of increasing functional capabilities as below.

• *Flat RBAC*: This model consists of the essential components of RBAC including user, role, and permission. The basic concept is roles are assigned to users, permissions are assigned to roles, and users gain permission by being members of their roles. Relationships can be many-to-many which means an individual user can be assigned to many roles and a single role can contain many users as well as permissions.

• *Hierarchical RBAC*: This model provides support to role hierarchies. A hierarchy defines the seniority of each role whereas senior roles also obtain the permission of their junior roles.

• *Constrained RBAC*: This model adds a requirement for enforcing separation of duties (SOD). SOD is a technique that is designed to prevent the possibility of fraud, theft, misuse of information, accidental damage, and other security compromises by ensuring that at least two individuals are responsible to complete a task.

• *Symmetric RBAC*: This model provides support for permission-role review with performance similar to user-role review introduced in a Flat RBAC. An idea behind developing this model is that organisations are able to periodically review and modify permission-role accordingly.

Multiple benefits are provided by RBAC including reducing administrative work and IT support, increasing operational efficiency, enhancing compliance, and reducing the risk of breaches and data leakage. Therefore, RBAC is one of the most well-known approaches for advanced access control.

4.3.6 Blockchain

Blockchain, sometimes referred to as Distributed ledger technology, is a shared, immutable ledger that facilitates the process of recording transactions and tracking assets in a network [70]. A blockchain is simply a chain of blocks that holds information.

A blockchain technique was firstly described in 1991 by science researchers, Stuart Haber and W. Scott Stornetta [71]. The original concept of blockchain is intended to cryptographically secure chain of blocks to store the time-stamped documents so it is not possible to tamper with them. However, it was not widely known until Satoshi Nakamoto created the digital cryptocurrency Bitcoin in 2008 [72].

In a blockchain system, each block contains three major elements: data, the hash of the block, and the hash of the previous block. The stored data inside each block depends on the type of blockchain. For example, a patient record may store information such as a patient's name, date of birth, medical history. Each block also has a hash. A hash is always unique and is used to identify a block and its contents. Once a block is created, a block's hash is calculated. The hash is changed if contents inside the block are changed. The hash of the previous block is an effective element that creates a chain of blocks which makes this technique secure and provides a security requirement of non-repudiation.

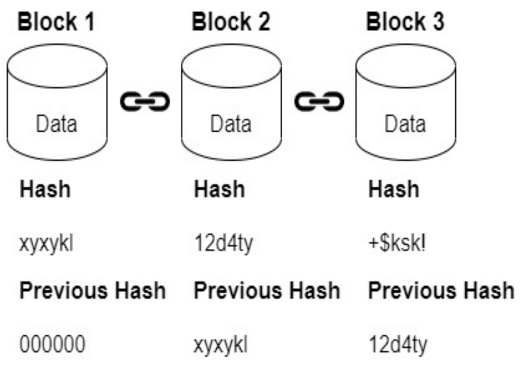


Figure 4.12: Blockchain components

A blockchain system provides the following key characteristics [70]:

• *Consensus*: All participants must agree on the transaction validity to ensure that the transaction is valid.

• *Provenance*: Participants have knowledge of where assets are from and how their ownership has changed over time.

• *Immutability*: A transaction cannot be tampered by any parties once it has been recorded to the ledger.

• *Finality*: The declaration that all blocks will not be revoked once they are committed to the blockchain.

Blockchain technology offers a benefit to a network where members do not necessarily trust each other. It also provides greater security for all network members as well as increases efficiency. Therefore, it has become one of the most common security mechanisms that have been widely used in several types of systems.

4.4 Chapter Summary

While the previously discussed frameworks achieved some protection of healthcare data, there is still a need to develop a more comprehensive security framework with a combination of various security mechanisms to provide end-to-end security for mHealth systems. This means security from when healthcare data is collected, then transferred over the network, and stored in both on-site storage and Cloud storage.

Tables 4.2 and 4.3 in Section 4.2 showed the comparison of different security frameworks mentioned above in terms of providing security requirements and protecting healthcare environment components. The two tables demonstrated that no existing framework catered to all security requirements and all healthcare environments at the same time.

Chapter 5

Proposing a new Information Security Framework for mHealth Systems

5.1 Introduction

In Chapter 5, an mHealth system scenario is demonstrated to gather information about mHealth system components. Moreover, assets, threats, and vulnerabilities in mHealth systems are identified in this chapter. A detailed analysis of security requirements for each component of an mHealth system is then examined. Furthermore, the taxonomy of an mHealth system is developed to clarify the structure of an mHealth system which includes system architecture, healthcare data, stakeholder, and security requirements. A new information security framework for mHealth systems, consists of several security mechanisms is proposed in this chapter. Finally, the result shows that the combination of proposed security mechanisms can deliver the full set of security requirements that are required in mHealth systems.

5.2 mHealth System Scenario

Overall, mHealth systems combine the setting of three environments: (1) Body Area Network (BAN) environment, (2) Hospital environment, and (3) Cloud storage environment. The BAN environment consists of two major units including the Body area sensor unit (or mHealth device) and the Mobile unit (Mobile phone, Tablet). Collected healthcare data in an mHealth system is transferred over the network through the connectivity infrastructure including Bluetooth, Local Area Network (LAN), and cellular network. This healthcare data may be stored on a hospital server in the hospital environment and/or on the Cloud storage.

mHealth systems generally store a large quantity of healthcare information such as patient's name, date of birth, medical information, or symptom descriptions. Cloud storage also plays an important role in mHealth systems as seen in an mHealth system setting in Figure 5.1:

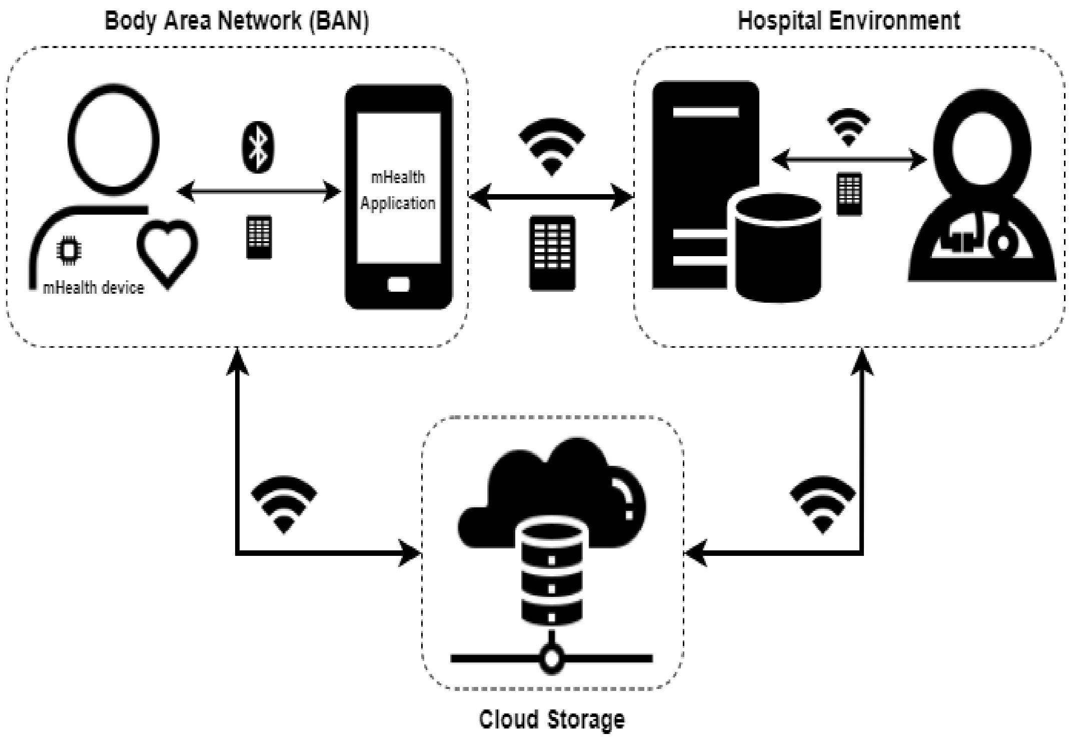


Figure 5.1: An mHealth system setting

In mHealth systems, mHealth devices that may be embedded, or implanted inside the user's body, or mounted to a user's body interface in a fixed position collect healthcare data from the user using Bluetooth communication within a Body Area Network (BAN). Collected healthcare data is then transferred to remote terminals (e.g., hospital server, Cloud storage) via a Wide Area Network (WAN). Healthcare data is stored in different databases including a mobile device's database, a hospital server, and Cloud storage. Authorised users, including healthcare professionals, patients, and mHealth users, are able to access healthcare data in the storage via the Internet.

The main benefit of using Cloud computing is it provides an opportunity for end-users to process large quantities of mHealth data in the Cloud and supports a real-time data collection service. However, the security of mHealth data protection is still challenging. Only authorised users should be able to access mHealth data on any data storage system. Therefore, an appropriate information security framework must be applied to both mHealth devices and the Cloud storage to secure mHealth data from unauthorised parties.

5.3 Identification of Assets, Threats, and Vulnerabilities in mHealth Systems

To inform the safe clinical use of mHealth, it is necessary to understand and be able to verify different kinds of risk that may occur in mHealth systems. To distinguish different kinds of risks, we must understand the key variables that can influence risk in mHealth systems [73]. This process includes the identification of assets, threats, and vulnerabilities. The purpose of this process is to determine what could happen to cause a loss and gain insight into how, where, and why the loss might happen [74]. As a result, an effective information security framework will be developed to address this problem.

5.3.1 Identification of Assets

An asset is anything that has value in each system, and which therefore requires protection [75]. In mHealth systems, assets could consist of more than hardware and software. Assets in mHealth systems can be identified as below:

1. Mobile Device can be defined as an mHealth device used in mHealth systems. Mobile devices in mHealth systems consist of wearable devices, implanted devices, and mobile phones/tablets/PDAs.

2. Cloud Storage: Cloud storage is where healthcare data in an mHealth system is stored. Healthcare data that is stored in Cloud storage will be able to be accessed by authorised stakeholders including patients, healthcare professionals, or insurers.

3. Network Connectivity is defined as wired or wireless connections to transmit, receive, and exchange data in mHealth systems. There are various types of connectivity used in mHealth systems such as Bluetooth, Zigbee, 4G LTE, and other wireless connections.

4. Data: Data refers to data that is stored and transmitted in mHealth systems. Data in mHealth systems can categorise into personal data (e.g., name, address, healthcare record) and authentication data (e.g., PIN, password).

5.3.2 Identification of Threats

A threat is a potential cause of an incident that may result in harm to assets in the system. Threats may be caused by natural means or human efforts and could be either accidental or deliberate; hence both accidental and deliberate threat sources should be identified. Threats may arise from within or from outside the system. Moreover, each threat may affect more than one asset and may cause different impacts depending on which assets are affected.

In mHealth systems, assets that need protection include mobile devices, Cloud storage, network connectivity, and data. This analysis will identify threats based on where data in mHealth systems is stored and transmitted (i.e., mobile devices, Cloud storage, and network connectivity).

Mobile devices security threats

Mobile devices deal with several threats that may pose significant harm to data which they store. The key mobile device security threats are identified as follows:

MT1. Loss and stolen: By the nature of mobile devices, they are prone to be lost or stolen. As a result, users have a high risk of losing data stored on mobile devices.

MT2.Mobile malware: Mobile devices are vulnerable to malware such as viruses, worms, trojans, spyware, ransomware. Mobile malware are usually hidden inside some malicious mobile applications that users installed or were deceived to install. Malware can disrupt mobile operations, gain access to sensitive personal data, or track users' activities.

MT3.Unauthorised access: Users normally store their login credentials for applications on their mobile devices. In this way, malicious attackers can easily access users' sensitive data in email and social network accounts.

MT4.Unlicensed and unmanaged software: The mobile software must be licensed and are required to be updated regularly. Failure of action could lead to unauthorised access to data or a significant loss of data.

MT5.Security of Biometrics: While a biometric system can enhance user convenience and strengthen the security of the system, it is also vulnerable to various types of threats as identified below [76]:

- Attacker can present fake biometric data to the sensor using prosthetic fingers created out of latex.

- Attacker modifies his behavior (e.g., voice) to impersonate a weak biometric template.

- Attacker exploits a residual biometric image left on the sensor to impersonate the last authorised user.
- Attacker modifies (adding/replacing) biometric templates from storage.
- Attacker steals the biometric template database.

Cloud computing security threats

According to the Treacherous 12 Cloud Computing Top Threats [77], the analysis identifies some key threats that may exploit the security of Cloud computing as below:

CT1. Insufficient identity, credential, and access management: The failure to use multifactor authentication, weak password, and poor key or certificate can lead to breaches and other attacks to Cloud storage.

CT2. Data breaches: Sensitive data is released, accessed, stolen, or used by an unauthorised user.

CT3. Insecure interfaces and APIs: The security and availability of Cloud services are dependent on the security of user interfaces (UIs) and application programming interfaces (APIs). UIs and APIs are generally the most exposed parts of the system and will be the target of heavy attacks.

CT4. System vulnerabilities: The attackers can use system vulnerabilities, or exploitable bugs, to penetrate a system for the purpose of stealing data, taking control of the system, or disrupting service operations.

CT5. Account hijacking: If an attacker gains access to an individual's credentials, they can eavesdrop on the individual's activities, manipulate data, return falsified information, and redirect the individual's clients to illegitimate sites.

CT6. Malicious insiders: Insider threats could be from any stakeholders who have authorised access to the system, network, or data intending to exceed or misused that access in a manner that negatively affects the confidentiality, integrity, or availability of data in the system.

CT7. Advanced persistent threats (APTs) are the parasitical forms of attack. APTs will penetrate the system to create a foothold in the computing infrastructure so that they can smuggle data and intellectual property over an extended period of time.

CT8. Data loss: Data stored in the Cloud can be lost not only by deliberate actions but also by accident, such as an accidental deletion by the Cloud service provider or a physical catastrophe including fire or earthquake. This can cause the permanent loss of data unless the provider or Cloud consumer takes adequate measures to back up data.

CT9. Insufficient due diligence: Choosing Cloud service providers without performing due diligence may lead to a myriad of commercial, technical, financial, legal, and compliance risks that jeopardize success.

CT10. Abuse and nefarious use of Cloud services: Poorly secured Cloud service deployments expose Cloud computing models to malicious attacks. Some examples of misuse of Cloud service-based resources include launching DDoS attacks, email spam, phishing campaigns, and hosting malicious or pirated content.

CT11. Denial of Service (DoS): Attacks that are meant to prevent users from being able to access their data or their applications.

CT12.Shared technology issues: Cloud service providers deliver their services scalable by sharing infrastructure, platforms, or applications. As a result, some users might be able to gain access to other users' actual or residual data and network traffic.

Network connectivity security threats

Threats could be harmful to the ability of the network to operate efficiently and effectively with its desired level of confidentiality [78]. Several common threats occur and potentially harm network connectivity which can be defined as follow:

NT1. Spoofing is a malicious process that access can be unauthorised - data cannot be unauthorised enable attackers to access to unauthorised information in a wireless network by masquerading as a different machine. Attackers can use this process to gain access to private data, steal data, spread malware, or bypass access controls [79]. Network spoofing attacks are easy to launch by many tools available in the market and can significantly damage the performance of networks [80]. Network spoofing can take on many forms including Internet Protocol (IP) spoofing, Address Resolution Protocol (ARP) spoofing, and Domain Name System (DNS) spoofing [79].

NT2. Scanning is a process of discovering the wireless networks and the vulnerabilities and threats associated with them. Network scanning can be used for either network security assessment or attacking the network. Network scanning is classified into two main categories: network port scanning and vulnerability scanning.

Network port scanning is a technique conducted by network administrators and attackers to gather information from computers that are connected to a network. Network administrators perform port scanning to identify open ports of a system so they may limit access to those ports or shut them off completely. However, attackers use port scanning in the same way as network administrators do but with malicious intent to exploit a network [81].

Network vulnerability scanning is the process of identifying weaknesses in a system or a network that can provide a backdoor to an attacker to perform an attack [82]. This process helps to detect loopholes in the system which could be used by attackers to conduct malicious attack or compromise it for undesired purposes.

Both network port scanning and network vulnerability scanning are techniques used in gathering information. However, it can be carried out by intruders which can be viewed as a prelude to an attack.

NT3. Denial-of-Service (DoS) and Distributed Denial-of-Service (DDoS) Attacks: A DoS attack is any intended attempt to prevent legitimate users from accessing a specific network resource [83]. DoS attacks are conducted by sending multiple requests to a server in an attempt to slow it down, flooding a server with large packets of invalid data, and sending requests with an invalid or spoofed IP address [84].

DoS attack is a malicious attempt by an attacker to cause a system to deny service to legitimate users. When this attempt derives from a single host or network, it is considered as a DoS attack. However, if the attack takes place simultaneously from multiple points, this type of attack is called a DDoS attack [85].

NT4. Eavesdropping refers to an unauthorised monitoring of private communication by an individual for whom it is not intended. The main objective of eavesdropping is to obtain sensitive data such as any kind of confidential information, passwords, or session tokens. Network eavesdropping can be conducted in the form of sniffing for data. A specific program is used in sniffing and recording packets of data communications from a network and then subsequently listened to or read using cryptographic tools for analysis and decryption [86]. Since eavesdropping will not affect the normal operation of network transmission, it can be difficult for both of sender and receiver to notice that the communication has been intercepted.

NT5. Jamming is defined as the disruption of existing wireless communications by decreasing the signal-to-noise ratio through the transmission of interfering wireless signal [87]. An attacker with a radio transceiver could perform an attack by intercepting the transmission, injecting spurious packets, and blocking the legitimate transmission. Jammers interrupt wireless communication by generating high-power noise across the entire bandwidth near the transmitting and receiving nodes. As a result, the performance of wireless networks can dramatically degrade [88].

Table 5.1 represents an mHealth system threat list in which each threat is associated with the security requirements, Confidentiality (C) - The assurance that data cannot be viewed by an unauthorised user, Integrity (I) - The assurance that data has not been altered (which includes accidental alteration) in an unauthorised manner, and Availability (A) - The assurance that mHealth data will be available and accessible to all authorised users every time it is needed, which are the fundamental parts of any security system.

Security threat	**Confidentiality**	**Integrity**	**Availability**
MT1	*	*	*
MT2	*	*	*
MT3	*	*	*
MT4	*	*	*
MT5	*	*	*
CT1	*	*	*
CT2	*	*	*
CT3	*	*	*
CT4	*	*	*
CT5	*	*	*
CT6	*	*	*
CT7	*	*	*
CT8			*
CT9	*	*	*
CT10	*	*	*
CT11			*
CT12	*	*	
NT1	*	*	*
NT2	*		
NT3			*
NT4	*		
NT5			*

Table 5.1: An mHealth system threat analysis

Most threats that occur in mHealth systems attempt to exploit weaknesses in confidentiality, integrity, and availability which are the most common security requirements for mHealth systems. Therefore, developing security frameworks that can provide all mentioned security services is an essential part of managing data in mHealth systems.

5.3.3 Identification of Vulnerabilities

Vulnerability refers to the security weaknesses of the system that could potentially be exploited by one or more threats. The presence of vulnerability does not cause harm, as there needs to be a threat present to exploit it. Vulnerability testing should be regularly performed by the responsible parties to identify weaknesses of the system and prevent unexpected events that may harm the system.

Mobile Device Vulnerabilities

Mobile devices are a preferred attack target in mHealth systems. Many factors lead to insecure mobile devices being released due to the weaknesses of mobile devices. Some vulnerabilities of mobile devices that could be exploited by threats are identified as below:

MV1. Insufficient transport layer protection: Typically, the data is exchanged in a client-server fashion. Mobile devices may use the Secure Socket Layer and Transport Layer Security (SSL/TLS) protocol during the data transmission. However, it could fail to perform SSL/TLS if an SSL certificate is outdated or improperly configured. As a result, data can be easily leaked or exposed to interception and enabling attackers to interfere with the data. This is more vulnerable when mobile devices are connected over Wi-Fi, this will allow the attacker to be able to perform a simple Man-in-the-Middle attack [89].

MV2. Unintended data leakage: Without user knowledge, some custom keyboard applications could be keylogging which can be used to intercept passwords and other confidential information entered via the keyboard. As a result, an intruder can get some confidential information, such as PIN, online banking account number, email password, from the user.

MV3. Poor authentication and authorisation: Mobile devices often lack user authentication and authorisation, and access control to data stored on them. Many devices may support some authentication mechanisms such as passwords, PINs, or pattern screen locks. However, some users normally have passwords and PINs that can be easily determined or bypassed such as users' date of birth, 0000, or 1234. Moreover, two-factor authentication is not always used when conducting sensitive transactions on mobile devices.

MV4. Malware: Users may unknowingly download some applications that may contain malware because it can be easily disguised as a game, security patch, utility, or other useful applications.

MV5. Lack of security software: Some mobile devices may not provide security software that protects against spyware, malware, and malicious application. Moreover, some users are not willing to install security software on their mobile devices because they understand that security software may slow down mobile operations and affect battery life.

MV6. Inadequate or outdated mobile software: The mobile device software should be licensed and updated regularly. Lack of performing, these actions can lead to an open invitation for attackers to steal user's data.

MV7. Lack of limited Internet connections: Some mobile devices may not provide firewalls to limit Internet connections. As a result, when mobile devices connect to WAN (Wide Area Network), an attacker could gain access to mobile devices through a port that is not secure and obtain sensitive information from mobile devices.

MV8. Unauthorised modifications: Some mobile devices may have unauthorised modifications (known as rooting or jailbreaking). Jailbreaking will allow users to bypass the application vetting process established by the manufacturer and install unauthorised software and applications. This process could change how security has been managed on mobile devices. This process will lead to the increase of security risks. As a result, mobile devices will have less protection against unintentionally installing malware [90].

Cloud Computing Vulnerabilities

Cloud computing has attracted more attention from users by providing features such as 24/7 availability and elasticity. However, it still inherits some challenges regarding its own weaknesses. According to previous research [91][92], Cloud computing vulnerabilities can be described as below:

CV1. Session Riding and Hijacking: Session hijacking (sometimes also known as cookies hijacking) refers to the use of a valid session key to gain unauthorised access to the information residing on a system. In Cloud computing, a session is created when a user logs on to a system. This session tracks the user information, i.e., session ID, to authenticate the user's request for data. This session ID will no longer make a valid request when the user logs out from the system. However, session data will still be stored within a cookie in the URL. Session hijacking takes over an existing active session. It occurs when an attacker takes over a user session by obtaining a valid session ID. An attacker could masquerade as an unauthorised user and make requests to the system. In Cloud computing, session hijacking could lead to a breach of stored data.

CV2. Virtual Machine Escape refers to the process that an attacker gains unauthorised access to the primary hypervisor and its created virtual machines. In virtualised environments, the physical server runs multiple virtual machines on top of hypervisors. An attacker can exploit a hypervisor remotely by using a vulnerability existing in the hypervisor [92]. This allows an attacker to access the host operating system and all other virtual machines running on that particular host [91].

CV3. Reliability and Availability of Service: Cloud computing is still not a perfect technology in terms of availability and reliability even it provides some features such as real-time data collection service. For example, in June 2016, as storms struck Sydney

(Australia), an Amazon Web Services region in the area lost power, and several of EC2 instances and EBS volumes hosting critical workloads for name-brand companies subsequently failed. As a result, websites, and online services from banking services to pizza deliveries went down across the Australian AWS availability zone for roughly 10 hours [93].

Bad weather and power outages are common issues causing the failure of Cloud computing operating. This can create a domino effect causing data loss and taking down large amounts of Internet-based services and applications.

CV4. Insecure Cryptography: Weak cryptographic Cloud storage is a common vulnerability in Cloud computing which leads to occurring of security data risks. Attackers may decode any cryptographic mechanism or algorithm to gain access to data stored in Cloud computing. It is common to find crucial flaws in cryptographic algorithm implementations. As a result, it can turn strong encryption into weak encryption or sometimes no encryption at all [91].

CV5. Data Protection and Portability: Generally, Cloud services are offered based on a contract between a client and a provider. In the case where the client would like to switch the Cloud provider or where the client goes out of the business, what happens to the data stored with the Cloud service provider is of concern. The sensitive data of the client could be deleted or misused by the Cloud service provider. For this reason, data protection and portability remain as two main vulnerabilities of Cloud computing.

CV6. Internet Dependency: Cloud computing provides services relying on the connectivity of the Internet. The Internet may temporarily fail due to unexpected events such as a bad weather or internet service provider maintenance. Therefore, users will not be able to access Cloud services. As a result, it can cause an inconvenience to a very crucial mHealth system which needs to run 24/7.

Network Connectivity Vulnerabilities

The security of network connectivity is also as important as the security of data storage because it is the route for data to transmit from one place to another. Unfortunately, network connectivity is one of the targets that are most prone to be exploited by attackers. Some common network connectivity vulnerabilities are identified as below:

NV1. End-to-end security is not performed: Wireless transmissions are not always encrypted. As a result, sensitive data which is transmitted over the network can be easily intercepted by attackers who may gain unauthorised access to it.

NV2. No user authentication exists in Bluetooth connection: There is no user authentication performed in Bluetooth connectivity, only device authentication is provided by the specification [94].

NV3. Attempts for authentication are repeatable in Bluetooth connection: There is a need for a mechanism that could prevent unlimited authentication requests. The Bluetooth specification requires an exponentially increasing waiting interval between successive authentication attempts. However, it does not require such a waiting interval for authentication challenge requests, so an attacker could collect large numbers of challenge responses (which are encrypted with the secret link key) that could leak information about the secret link key [94].

NV4. Connectable/discoverable devices are prone to attack: Any devices in connectable mode can become the main target of attackers. Any devices that are in a connectable mode to pair or connect should only do so for a minimal amount of time. A device should not always be in connectable mode [94].

NV5. Lack of physical security: mHealth devices mainly use the wireless connection to transmit data between each node. Unlike wired networks, the signals of a wireless network are broadcasted among the communication nodes. An attacker with a compatible wireless device is to be able to intercept the signals when the intercepting device is within the broadcast range of the communication paths [95].

5.4 A Detailed Analysis of Security Requirements for mHealth Systems' Components

The research examined security issues in mHealth environments, and the security issues can be divided into five subsystems.

1. The first requirement is **the provision of AAAC** (Authentication, Authorisation, Accounting, and Control) for all human users including medical staff, patients, technicians, administrative staff, and visitors. The system should allow users to use the hospital environment simply and intuitively. One way of addressing this is to look at using mechanisms that support Role-Based Access Control (RBAC) which is a security framework for controlling user access rights to objects in the system, based on their roles [96][97].

2. The second requirement is to **protect devices** from being misused, tampered with, or stolen. This includes not just medical devices in hospitals and surgeries, but also devices used in the home or by mobile users with eHealth or mHealth functions.

3. The third requirement is the need to **protect digital data** such as the Electronic Health Records (EHRs) of patients. The misuse of EHRs can cause personal as well as economic damage. Hence, it is a legal requirement to protect EHRs as highlighted by the GDPR [98] and HIPAA [99].

4. There is now also a need to **protect hospital infrastructure**. This is due to the fact that new types of attacks, such as ransomware, are being developed. Ransomware typically attacks victim machines in several ways including phishing emails, malicious links, and malvertising [100][101]. Network-based attacks such as Denial-of-Service (DoS), Distributed Denial of service (DDoS), and buffer-overflow attacks are

on the rise [102][103].

5. Finally, the presence of COVID-19 increases the need to **protect access to certain physical sites and locations**. This is becoming more important in the United Kingdom, where several large hospitals have many departments and access to different parts of the hospital needs to be controlled. Some areas, such as car parks and concourse areas, clearly need to be publicly accessible while a large number of areas, such as offices and wards, need to have restricted access.

5.5 Developing a Taxonomy of an mHealth System

Taxonomies have been used in many fields for a long time. For example, in botany, taxonomy has been used to classify plants. Taxonomy is the basis of classification schemes and indexing systems in information management [104]. A taxonomy is usually a hierarchy of concepts that only shows the relationship between each concept such as parent and child, or superclass and subclass. One purpose of a taxonomy is to give a knowledge representation of a classification [105].

Previously, there have been some developments of taxonomy for mobile devices and Cloud computing which are key components of mHealth systems. However, no taxonomy encompasses mobile devices, hospital environment, and Cloud storage environment to address a complete set of mHealth system concepts/components including system architecture, healthcare data, stakeholders, and security requirements has been proposed to secure healthcare data in mHealth systems.

A taxonomy of an mHealth system was developed from the mHealth system scenario that was previously presented in Figure 5.1. Healthcare data is collected by a mobile device that has an mHealth application installed on it. It then transfers over the network connectivity to store in different data storage including on-device storage, a hospital server, and Cloud storage. Due to the characteristic of healthcare data, it has to be shared among various stakeholders such as patients, healthcare professionals, researchers, etc. Therefore, appropriate security measures must be placed into the system to protect the security of healthcare data.

Figure 5.2 shows a taxonomy of an mHealth system developed by the author. To develop this taxonomy, a domain analysis of the mHealth system was carried out. The structure of mHealth systems was categorised into several components, including system architecture, healthcare data, stakeholder, and security requirements.

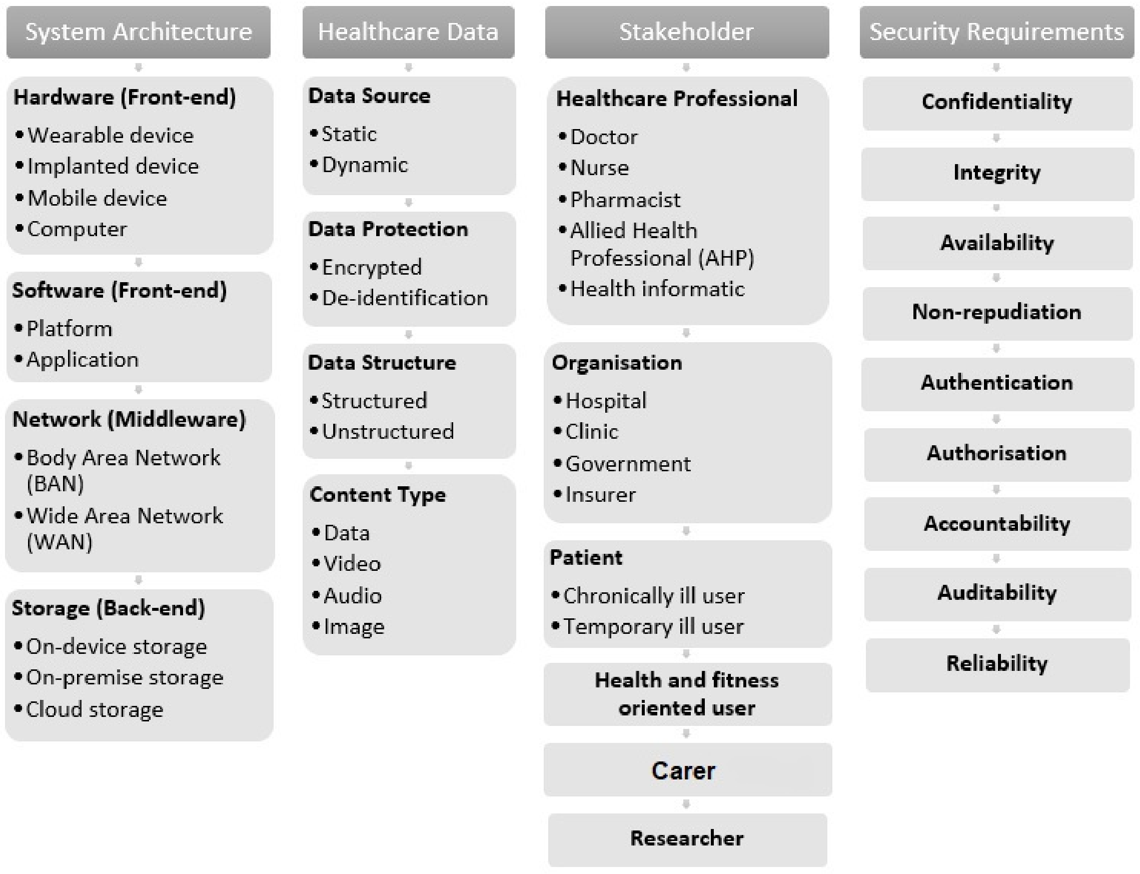


Figure 5.2: mHealth Taxonomy

5.5.1 System Architecture

The first-level taxonomy of mHealth systems is based on the architecture of mHealth systems. This can be defined in terms of Front-end (Hardware and Software), Middleware (Network), and Back-end (Storage).

1. Hardware can be defined as physical components (or mHealth devices) in mHealth systems. Hardware in mHealth systems consist of:

- *Wearable Devices*: Wearables refer to hand-held health electronic devices that are mounted on a user's body interface. They can perform many computing tasks that are not typically seen in mobile and laptop devices including sensory and scanning features. These devices are designed to collect the data of users's health, diet, and exercise. They can pass this information to healthcare professionals in real-time. Some wearable devices such as Fitness-tracking bands (e.g., Fitbit, Jawbone, Runtastic) are good examples of the Internet of Things. Other examples of wearable devices include Smartwatches (e.g., Android Wear, Apple Watch, Pebble Watch), Smart Glasses (e.g., Google Glass, Sony's SmartEyeGlass), Wearable ECG Monitors, Wearable Blood Pressure Monitors, and Biosensors.

- *Implanted Devices*: An implant is a medical device manufactured to replace a missing body structure, supporting a damaged body structure, or enhancing an existing body structure. An implanted device can be placed permanently or can be removed once it is no longer needed [106]. Examples of implanted devices include an artificial heart, neurostimulator, and Circadia (Human Embedded Light Emitting Diode Display (HELEDD)) [107].

- *Mobile devices (Mobile phone/Mobile computer)*: These devices are used to perform a variety of information management functions in mHealth systems. An authorised mHealth application is installed on mobile phones or mobile computers to communicate with wearable devices and implanted devices embedded in a user's body. Wearable devices and implanted devices collect healthcare data from the user and transfer it to a mobile phone or a mobile computer. Healthcare data may be collected on these devices' storage areas or will be transferred to be stored in Cloud storage.

- *Computer (Personal computer, medical computer, laptop)*: These devices can be used as local and remote servers. Authorised users can access, store, and manage healthcare data through these devices.

2. Software can be defined as various kinds of computer programs operated by mHealth devices which are used to manage healthcare data in mHealth systems.

• *Platform*: In mHealth systems, a platform generally refers to a mobile operating system (e.g., Android, iOS, Windows, Ubuntu Touch OS) which is an operating system for mobile devices such as mobile phones and mobile computers. Mobile phones and mobile computers are designed to be run by a specific operating system.

• *Application*: Software applications are designed to run on mobile phones and mobile computers. mHealth applications include the use of mobile devices in collecting healthcare data from users. Different mHealth applications may be designed to operate on different operating system platforms. Nowadays, more mHealth applications are designed to be compatible with many operating system platforms to allow more users to be able to make use of them.

3. Network is defined as wired or wireless connections to transmit, receive, and exchange healthcare data in mHealth systems.

• *Body Area Network (BAN)*: or sometimes referred to as Wireless Body Area Network (WBAN) or Body Sensor Network (BSN) is a communication standard optimised for low power devices and operation on, in, or around the human body (but not limited to humans) to serve a variety of applications including medical, consumer electronics, personal entertainment, and others [108]. A BAN is characterised as an easily configured, low-cost, low-power, and highly reliable sensor system. The radius of operation of a BAN is just over a few feet. Healthcare data is generally transferred within a BAN via Bluetooth and ZigBee. Bluetooth is a developed technology that is widely used in many mobile phones and mobile computers. It allows communication bandwidth speeds of up to 720 kbps which is more than adequate for most body sensors. ZigBee is an emerging wireless standard for low-data rates, ultralow-power usage with potential for use in mHealth systems. The maximum data rate of ZigBee is 250 kbps which is still sufficient for wearable devices and implanted devices [109].

• *Wide Area Network (WAN)*: refers to a geographically distributed communication network that connects multiple Local Area Networks (LAN). In mHealth systems, WAN is a network that connects several entities including mHealth devices, data storage, and stakeholders all together in beneficial to gain access, store, and manage healthcare data. Healthcare data communications that are established between a BAN and the remote terminals will be achieved by using WAN technologies such as 5G Cellular, 4G LTE, SSL VPN, or Fibre Optic.

4. Storage is where healthcare data in mHealth systems is stored. Healthcare data could be stored using either on-device storage, on-premise storage, or Cloud storage.

• *On-device Storage*: Every mobile phone and mobile computer provides on-device storage which makes it as a reliable place to store healthcare data once it has been collected from wearable devices before transmitting to healthcare professional server or Cloud storage.

• *On-premises storage (On-site server)*: Traditionally, healthcare organisations have a preference to store healthcare data on on-premise storage as it is convenient to gain access and they can have fully control over the management of healthcare data. Moreover, on-premise storage does not rely on the use of the wireless connection, there is no risk of downtime. Authorised personnel can access healthcare data (e.g., patient health information, radiology images, laboratory and test results) that are stored on a server (a central computer) via a network that requires login access.

• *Cloud storage*: Nowadays, there is a tremendous increase in the capacity of healthcare data and multiple devices being used in healthcare systems with more complex applications that require remote access to healthcare data. Cloud storage has become a more practical option to store healthcare data. Cloud storage offers a flexible and scalable environment at a lower cost than other on-premise storage. Moreover, authorised users can also access healthcare data that is stored on Cloud storage from anywhere and anytime.

5.5.2 Healthcare Data

In mHealth systems, healthcare data refers to data that hold patient information and is stored and transmitted in the system. Healthcare data are categorised by the source of data, the presence of data protection mechanism, the structure of data, and the content type.

1. Data Source

• *Static*: Once static data have been created, their contents do not change over time. If they are changed, the data, that have been changed, will be considered as stateless, or no longer existing, and will be replaced by a new set of data. Examples of static data in mHealth systems include medical histories, allergies, and lab and test results.

• *Dynamic*: Dynamic data refers to the data that are asynchronously changed over time because new further updates are become available. Vital signs (e.g., heart rate, body temperature, blood pressure) are good examples of dynamic data in mHealth systems.

2. Data Protection

• *Encrypted*: Strong encryption is the most effective way to achieve data security. Encryption is one of the most effective methods for the protection of healthcare data. Encrypted data refers to data that has been transformed into another form in such a way that only authorised parties will be able to gain access to it. Unencrypted data are extremely vulnerable when it comes to healthcare data breaches. Although, healthcare data encryption may not be required, however, it is a very crucial measure that should be considered to protect the confidentiality of healthcare data.

• *De-identification*: The de-identification of healthcare data refers to the process that identifiers are removed from health information. The purpose of de-identification is to obscure the identifiable data items within the healthcare records sufficiently that the risk of potential identification of the subject or records is minimised to acceptable levels. Even though the potential risk may not be fully removed, it can be minimised with the use of multiple pseudonyms [110]. De-identification can be conducted in various ways including removing direct patient identifiers, replacing a real name with an ID, or using age ranges instead of actual age.

3. Data Structure

• *Structured data* refers to healthcare data that are standardised and easily transferable between health information systems. It can be stored and presented in a reliable and organised manner as well as easily examined over time. Some examples of structured healthcare data include patient demographics, blood type, and lab test results.

• *Unstructured data* refers to healthcare data that are not standardised and cannot be easily organised using pre-defined structures. It is much more complicated to examine and interpret than structured healthcare data. Physician notes, audio recordings, personal correspondences are good examples of unstructured healthcare data. Therefore, they must be converted to a more structured format before processing.

4. Content type

• *Data*: Healthcare data that are in the form of files. E.g., Electronic Health Record (EHR), Consent form, Insurance record

• *Video*: Healthcare data that are in the form of live casting and all real time data streaming. (e.g., Video recording, Live vital sign monitoring)

• *Audio*: Healthcare data that are in the form of sound and speech. E.g., Audio recording

- *Image*: x-ray film, clinical photograph, printouts from monitoring equipment

5.5.3 Stakeholder

In mHealth systems, stakeholders refer to authorised parties who have the right to gain access to healthcare data that are stored. There are six main stakeholders in mHealth systems.

1. Healthcare Professionals: Healthcare professionals manage human health through the application of the principles and procedures of evidence-based medicine and caring [111]. They provide healthcare and well-being guidelines to mHealth application users. Healthcare professionals include doctors, nurses, pharmacists, Allied Health Professionals (AHPs), and health informatics.

2. Organisations: Organisations refer to authorised organisations who are involved in the provisional of healthcare. They include hospitals, clinics, governments, and insurers.

3. Patient: A patient is a person who requires medical care. In an mHealth system, patients who use mHealth applications can be both of chronically ill patients and temporary ill patients.

4. Health and Fitness Oriented User: This group of users uses mHealth applications to support and focus on their healthy lifestyles which include weight control, diet, and exercise. Some examples of these users are general users, athletes, and personal trainers.

5. Carer: A carer plays a significant role in terms of supporting a patient through a recovery process. On most occasions, carers may have the patient's consent to access patient's healthcare data as well as contact healthcare professionals.

6. Researcher: A researcher is responsible for proposing and performing experiments as well as analysing results to increase healthcare knowledge. Occasionally, they are required to access prior healthcare data to gain more understanding. Their tasks may include developing medicines and medical products.

5.5.4 Security Requirements

Security requirements describe the functional and non-functional requirements that need to be achieved to accomplish the security attributes of an mHealth system. From the threat analysis of mHealth systems and previous research by Yahya, Walters, and Wills [57], the security requirements that are vital to protect the security of mHealth systems can be identified as follows:

1. Confidentiality: The assurance that data cannot be viewed by an unauthorised user [44]. Confidentiality is a common security component that is required in any security system.

2. Integrity: A key aspect of information security is integrity. Integrity is an assurance that data cannot be altered (which includes accidental alteration) in an unauthorised manner [44].

3. Availability: The assurance that healthcare data is available and accessible to all authorised users every time it is needed [57].

4. Non-repudiation: The assurance that an entity cannot deny a previous commitment or action [44].

5. Authentication: Authentication is the process or action of verifying the identity of a user or process. [112]. Some processes could be applied to verify legitimate nodes between mHealth devices in mHealth include using passwords and biometrics (e.g., fingerprint, retina scan, voice recognition) as previously discussed in Section 4.2.1.

6. Authorisation: Authorisation is a process by which a system determines the security level for access or using resources within the system by each user. Whereas authentication is the process of identifying legitimate nodes or users within an mHealth system, authorisation is required to allow users such as patients or healthcare professionals to access stored healthcare data to populate information required [112].

7. Accountability: Accountability is the process of keeping track of users' activity while accessing resources in the system. Accounting simply tracks which users accessed the mHealth system, what they were granted access to, the amount of data transferred during the session, the amount of time users spent on the system, and when they disconnected from the system [113].

8. Auditability: Auditability is highlighted as an important security component in mHealth systems. Therefore, it is important that each organisation in an mHealth system should perform routine security audits to ensure that healthcare data is protected as well as provide policies to comply with international IT standards [57].

9. Reliability: Reliability refers to the ability of a system to provide a consistent intended service most of the time [57].

5.6 Key Mechanisms for Providing Solutions to Manage mHealth Data

As a part of investigating the development of a new security framework, based on the previous work of Mapp et al. [30], there are some possible solutions for managing mHealth data using various mechanisms. These mechanisms are able to deliver the security requirements as parts of an mHealth Taxonomy. The mechanisms are identified as follows:

5.6.1 Encryption as a Service

Encryption is a process to secure information from unauthorised accesses. It changes information which can be read (plaintext) into the form that cannot be read (ciphertext) [114], unless, you have a key that can decrypt the message. HIPAA [115] mandates standards used to secure EHRs and requires a method to be implemented to encrypt and decrypt electronically protected health information. All electronic healthcare data that are created, transmitted in systems, or stored on devices must be encrypted.

The main purpose of encryption is to protect the confidentiality of healthcare data that resides in the system. However, encryption does not protect end-to-end confidentiality nor prevents communication interceptions. Moreover, encryption itself provides only the confidentiality of data but does not provide other security requirements such as integrity, authenticity, or non-repudiation. Therefore, other security mechanisms will be required in mHealth systems to protect healthcare data elsewhere in the system.

5.6.2 Capabilities

A Capability refers to a token that permits authorised users to access certain objects in a system [116]. To develop Capabilities as a solution for managing mHealth data, Internet Protocol version 6 (IPv6) addresses can be modified in to support a Capability system. IPv6 is the latest version of the Internet Protocol (IP) [117]. It provides an identification and a location for every computer, mobile phone, and any other mobile device on networks across the internet through its IP address. The IPv6 protocol also provides several other advantages as it can handle packets more efficiently as well as improves performance and increases security [118].

In an mHealth system, every object and its properties are identified using capabilities. Therefore, it is necessary that capabilities must be carefully managed and are protected from being created or modified in an unauthorised manner.

5.6.3 Storage Management System

This mechanism enables the management of security for each block of data in the Cloud infrastructure using encryption techniques such as AES (Advanced Encryption Standard) and 3-DES (Triple Data Encryption Standard) algorithms to protect the confidentiality of data. Moreover, each block of data is hashed after it has been modified to provide integrity so that data will not be able to be modified by an unauthorised user. To ensure the availability of data, each block may be replicated throughout the Cloud storage structure. Therefore, a coherency protocol within the storage layer is used to synchronise different copies of the block [119].

5.6.4 Digital Filter

In addition to traditional security measures, the use of digital filters is a further advantage in providing more control over who can access healthcare data.

Each healthcare record in the Cloud storage can have a set of filters that is used to prevent certain fields in that record from being accessed in an unauthorised manner. To access a given field, the relevant filter must be removed. This usually requires authorisation from senior personnel. A digital filter can therefore provides authorisation down to the fields of records thus enabling different users with different authorisation privileges to use the system. Moreover, it can also prevent unauthorised accesses, theft, destruction, and DoS attacks [120].

5.6.5 Secure Transport Layer

An examination of network interactions at the local area level clearly indicates that there is a need for more transactional support in the Cloud environment since there are many client/server interactions that use network services [30].

The Transport Control Protocol (TCP) and the User Datagram Protocol (UDP) have been widely used as the main transport protocols. TCP is a connection-oriented protocol, whereas UDP is a connectionless protocol. TCP provides reliable services while UDP provides fast, low latency but unreliable connections. However, recent research in vehicular networks has indicated the need for low latency, reliable, secure transport protocol. As a result, the Simple Lightweight Transport Protocol (SLTP) has been developed to support these issues [121].

SLTP keeps packet processing as simple as possible to reduce latency and provide faster connection setup and takedown times. Moreover, it is designed to be used in many environments including UDP/IP and Raw Ethernet. The details of SLTP functionalities can be explored in [122].

SLTP can be combined with an encryption mechanism to provide secure communications while maintaining fast and reliable connections. Furthermore, SLTP supports the inclusion of Additional Headers which could be used to pass security parameters and certificates. Hence the Transport Layer Security (TLS) protocol can be easily supported using SLTP.

5.6.6 Blockchain

A blockchain is a distributed data system where users share a consistent copy of a database and agree on changes by consensus. The data is represented in the form of blocks, where each block includes a cryptographic signature of the previous block, creating an immutable record [123][124]. A combination of two types of ledger technologies, permissioned ledgers and permission-less ledgers, provides advantages in supporting, recording, and enhancing the administration of patient records [125][126].

The use of blockchain technology in healthcare systems fulfills the security requirement of non-repudiation[127] as well as offers an ability to discover security and privacy violations [128].

5.6.7 Secure Transactional Layer

The Secure Transactional Layer is developed to protect the remote procedure by encoding the transmitted data to achieve data secrecy. The Secure Remote Procedure Call (SRPC) is an inter-process mechanism that is used for communication between clients and servers. It uses a typed system in which the type as well as the value of data passed are explicitly declared. This is used to make sure that security attacks such as buffer overflow attacks can be avoided.

By combining encryption, capabilities, and SRPC, it is possible to provide a secure transactional environment where clients and servers can be authenticated and authorised, and transactions are validated to ensure proper interaction between clients and servers.

5.6.8 Service Management Layer

Since there is a large amount of data being generated in healthcare environments, Cloud storage systems are increasingly being used to store and process healthcare data [129]. EHRs must be securely stored, hence, the challenges of using Cloud services for healthcare environments must be addressed to ensure that patients, doctors, and hospital staff are safe.

There are several challenges to be addressed including a secure execution environment, the best place to run a service at any point in time, and the ability to securely transfer services between Cloud storage. This means that, among other things, it is important that servers are not hosted on unsafe Cloud hardware and Cloud Servers are not corrupted by malicious or badly implemented servers.

As a result, the Service Management Framework (SMF) is developed. It provides a solution for issues of security, deployment, replication, or migration of services on different scales including geographical regional, national, and global contexts.

The Service Management Framework is a new way of deploying and managing services in distributed environments. It allows clients to find services and provides communication endpoints and capabilities which allow a reliable session to be developed. It, therefore, increases the security, efficiency, and management of services, and will be a key part of future IoT systems.

5.7 Developing a New Information Security Framework for mHealth Systems

From the previous studies, several information security frameworks for mHealth devices as well as information security frameworks for Cloud storage have been proposed. However, the major challenge is developing an effective information security framework that will encompass both mHealth devices and Cloud storage to secure mHealth systems. Such a framework would consist of a several coordinated countermeasures to address threats and vulnerabilities, and to protect the assets of an mHealth system.

As part of investigating the development of a new security framework for mHealth systems, based on the previous work of Mapp et al. [30], there are some possible solutions for managing mHealth data using various mechanisms to deliver the security components of mHealth systems. The mechanisms include Encryption as a Service, Capabilities, Storage Management System, Digital Filter, Secure Transport Layer, Blockchain, Secure Transactional Layer, and Service Management Layer. A new security framework is proposed in Figure 5.3 based on the use of these key mechanisms.

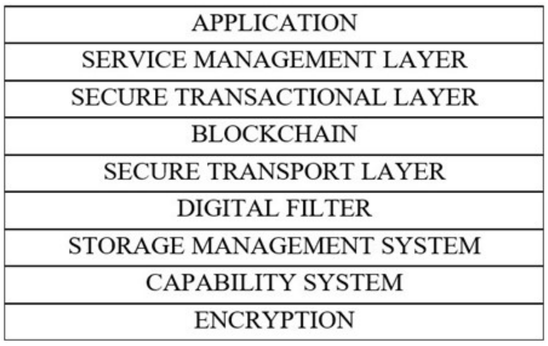


Figure 5.3: A proposed information security framework for an mHealth system

The Application Layer is concerned with the authentication and authorisation of users. Users authenticate themselves through the application on a mobile device. This authentication will lead to the authorisation of the user to access application resources.

The Application Development Layer is part of the Application Layer and is used to authenticate the application and the user to the hypervisor which provides a virtual environment that interacts with the Cloud infrastructure, if accessed to the user's data is required. A hypervisor, also known as a virtual machine manager (VMM), is a hardware virtualisation technique that allows multiple guest operating systems to run on a single host system at the same time. Therefore, the hypervisor is the boundary between the application and the Cloud infrastructure. This layer is also responsible for Presentation Security which encodes and decodes data between the application and the Cloud storage system [30].

The Service Management Layer will define the requirements to run the service at a level that the service provider considers adequate. This layer is responsible for registering, operating and monitoring users, applications, devices, and services residing in the system. The service management layer ensures that resources are working properly and optimally interacting with users and other services.

The Secure Transactional Layer/SRPC will use the SRPC to protect the remote procedure between stakeholders and Cloud server by encoding data transmitted in the system.

In the *The Blockchain System*, healthcare data in mHealth systems will be stored in the form of blocks using a Blockchain mechanism. Each block of data will be encrypted with the strongest possible algorithm and will be stored separately in Cloud storage.

The Secure Transport Layer will use the Simple Lightweight Transport Protocol (SLTP), which is the security mechanism that provides the authentication by using key exchange, to secure the transmission of healthcare data between stakeholders and the Cloud infrastructure.

To deliver additional control by which users will be able to access healthcare data, *Digital Filters* will be applied to healthcare data to prevent certain fields of data from being accessed by unrelated stakeholders.

The Storage Management System will apply encryption techniques, and hash and replicate each block of healthcare data to provide confidentiality, integrity, and availability of data.

Every object in an mHealth system (such as users of mHealth devices, healthcare professionals, mHealth devices, healthcare data) will be managed using *Capabilities* to organise access rights which can ensure that only authorised personnel will be able to access stored healthcare data. Moreover, healthcare data that are managed, stored, and transmitted in an mHealth system will be applied an *Encryption as a Service* to protect the confidentiality.

Figure 5.4 below presents the scenario that the proposed information security framework is applied to an mHealth system where (1) = Encryption as a Service; (2) = Capabilities; (3) = Storage Management System; (4) = Digital filters; (5) = Secure Transport Layer; (6) = Blockchain system; (7) = Secure Transactional Layer; and (8) = Service Management Layer.

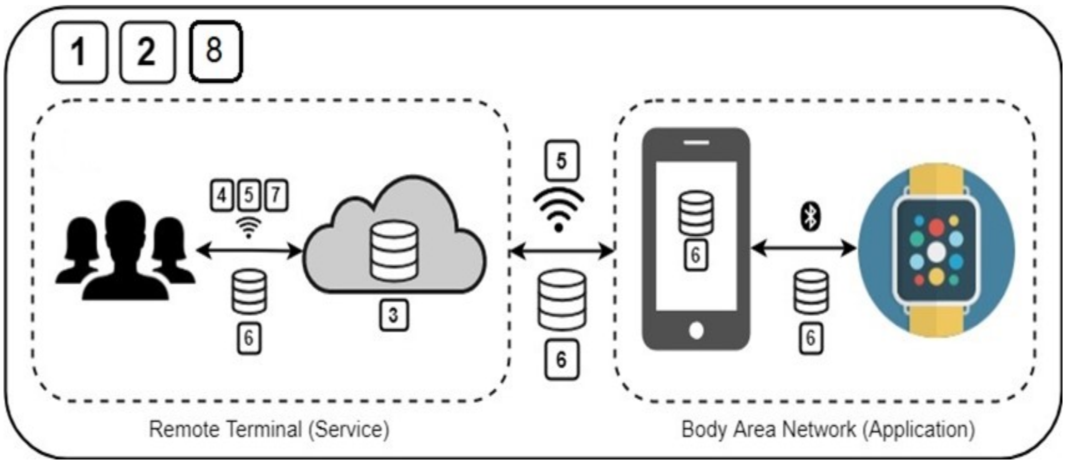


Figure 5.4: Applying security mechanisms to an mHealth system

Table 5.2 below provides the security requirement analysis of each key mechanism presented in Figure 5.4.

Security requirements	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**	**(8)**
Confidentiality	*	*	*		*	*		
Integrity		*	*		*	*		
Availability			*			*		*
Non-repudiation		*				*		
Authentication		*		*	*	*		
Authorisation		*		*	*			
Accountability		*				*		*
Auditability						*		*
Reliability		*			*		*	*

Table 5.2: The security requirement analysis for the new information security framework

5.8 Chapter Summary

Possible solutions for securing an mHealth system using various security mechanisms were discovered. They included (1) Encryption as a Service, (2) Capabilities, (3) Storage management system, (4) Digital filters, (5) Secure transport layer, (6) Blockchain, (7) Secure transactional layer, and (8) Service management layer. The information security framework for mHealth systems (Figure 5.3) was proposed based on the combination of these security mechanisms.

The Encryption as a Service encrypts healthcare data to protect their confidentiality. *The Capabilities* manages access rights of every object in an mHealth system, therefore, it provides various security requirements including confidentiality, integrity, non-repudiation, authentication, authorisation, accountability, and reliability. *The Storage management system* provides security for the storage of healthcare data using encryption techniques to provide confidentiality, hashing to provide integrity, and replication to provide availability. *The Digital filter* provides authentication and authorisation by taking control over who is able to access different fields of healthcare data. *The Secure transport layer* provides secure communications while maintaining fast and reliable connections. *The Blockchain* provides the security requirement of non-repudiation and can also be used to identify security violations. *The Secure transactional layer* delivers secure interactions by encoding data and its value transmitting between clients and servers. *The Service management layer* deploys and manages services that are being provided in an mHealth system. Therefore, the proposed information security framework for mHealth systems provides a full set of security requirements as presented in Table 5.2.

Initially, the information security framework for mHealth systems (Figure 5.3), consists of nine layers including (1) Encryption as a Service, (2) Capabilities, (3) Storage management system, (4) Digital filters, (5) Secure transport layer, (6) Blockchain, (7) Secure transactional layer, (8) Service management layer, and (9) Application, was proposed. Although, the original framework might strongly secure mHealth systems, however, each of security mechanisms are rather complex to implement and it will require significant resources and time which may exceed the Ph.D. timeframe.

As a result, a prototype will be developed in Chapter 6 for the purpose of testing.

Chapter 6

Prototype Design

6.1 Introduction

In this Chapter, a prototype is developed. This prototype consists of four layers including mHealth Secure Storage System, Service Management Layer, Secure Transactional Layer, and Capability System. The main goal of this prototype is to provide an experimental system that contains the necessary functionalities to meet the security requirements that were previously identified.

6.2 The Prototype Design

To develop the information security framework for mHealth systems, the research began with defining the architecture of mHealth systems to gain more knowledge about how healthcare data is processed in the system and what are components of an mHealth system. After that, various information security models were observed to discover essential security requirements that need to be achieved by the framework. Moreover, assets, threats, and vulnerabilities in mHealth systems were identified to gain an insight into different kinds of risks that may pose damage to mHealth systems.

As a result, a detailed analysis of security requirements for mHealth systems' components in Section 5.4 was examined. Security issues in mHealth environments can be distinguished into five subsystems which include human users, devices, digital data, hospital infrastructure, and physical sites. To elaborate a clear structure of mHealth systems, a taxonomy of an mHealth system was developed in Section 5.5. Comprehensive details of mHealth systems including a system architecture (hardware, software, network, and data storage), healthcare data, stakeholder, and security requirements, were represented in a taxonomy of an mHealth system.

The information security framework for mHealth systems was therefore developed to overcome security issues by protecting components of an mHealth system in Section 5.4 and providing a complete set of security requirements that was represented in Section 5.5.4. In theory, the proposed framework (Section 5.7) that consists of many key security mechanisms may be able to provide a secure mHealth environment. Therefore, there is a need to develop a practical prototype to experiment with this proposed framework.

6.3 The Proposed Prototype

At the beginning, the proposed information security framework for mHealth systems consists of nine layers including (1) Application, (2) Service Management Layer, (3) Secure Transactional Layer, (4) Blockchain, (5) Secure Transport Layer, (6) Digital Filter, (7) Storage Management System, (8) Capability System, and (9) Encryption. There is a need to move from the proposed framework to a prototype implementation. Since the proposed framework involves many different security mechanisms, developing a practical prototype for testing that clearly represents every previously mentioned security mechanism will take significant time and effort since each mechanism is rather complex to implement.

As a result, newly developed mechanisms are now introduced as key components of the new prototype (Figure 6.1), namely, Capability System, Secure Transactional Layer, Service Management Layer, and mHealth Secure Storage Application. These mechanisms were developed by the author and are parts of the information security framework that was proposed in Section 5.6. The main purpose of this prototype is to provide an experimental system that contains the necessary functionalities to meet the security requirements that have been identified in Section 5.4 and Section 5.5.4. Although, the prototype may contain much fewer layers than the proposed framework. Therefore, it provides a complete set of requirements for end-to-end security same as the proposed framework, and makes a novel contribution to this research. The prototype's components are explored below:

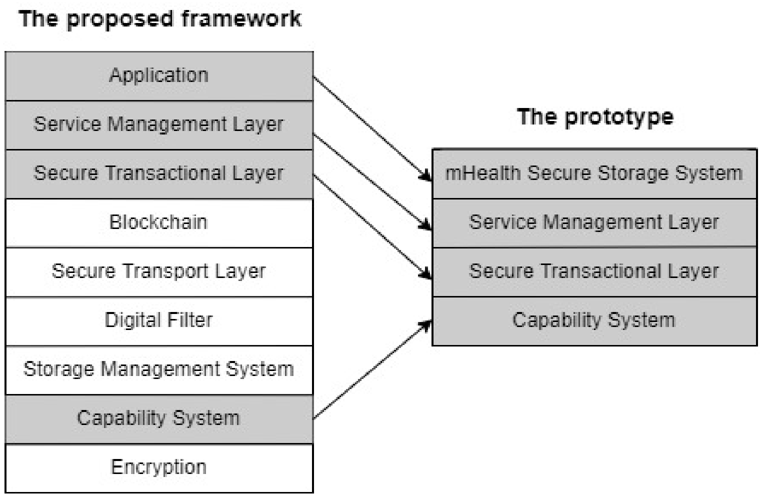


Figure 6.1: Toward the prototype development

• **Capability System**: Every object in an mHealth system (such as user, device, and healthcare record) is managed using capabilities to organise access rights which can ensure that only authorised entities will be able to access each object.

• **Secure Transactional Layer**: A Secure Remote Procedure Call (SRPC) was developed in this work to protect data transmitted between the client and the server by encoding them. An initial prototype of the SRPC was implemented[125] and showed a 10 percent reduction in performance when compared with normal unsafe mechanisms.

This is a small price to pay for such a great security improvement.

- **Service Management Layer**: A Service Management Framework (SMF) is a new approach to manage services in a distributed environment. A simple Service Management Layer (SManL) was developed and integrated into the prototype.

- **mHealth Secure Storage System**: In an mHealth secure storage system, files (e.g., EHRs) can be created, stored, modified, and deleted. The Filesystem in Userspace (FUSE) is used to provide services for controlling how EHRs and other files are accessed, stored, and retrieved. FUSE is connected to the Network Memory Server (NMS) which is network storage. NMS provides basic functionalities in storing and allocating data blocks.

6.4 New Mechanisms in Detail

6.4.1 Capabilities

In mHealth systems, there is a need to protect the confidentiality and privacy of healthcare records since there are several people who may have access to them. Only authorised personnel with access rights depending on their roles are permitted to access these healthcare records [130].

Capabilities provide several benefits to healthcare systems. Capabilities can be used to provide Role-Based Access Control (RBAC) access for users. Capabilities may not be assigned directly to users, instead, they are assigned to roles, and roles are then assigned to users. Therefore, an access right can be identified by a role, based on the job functions of different people in the healthcare system such as doctors, nurses, patients, and researchers.

A type-based capability system which supports role-based access was developed in this work, based on previous work of Mapp et al. [30]. The main components in the capability-based system (as shown in Figure 6.2) include:

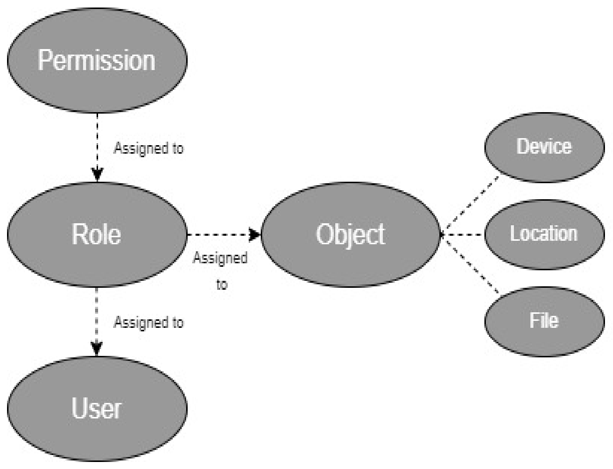


Figure 6.2: Capability-based system components

• **User:** Access rights to objects of a user are based on a designated role.

• **Role:** Roles are identified by job functions.

• **Permission:** Permissions are clarified by the functionality and responsibility of the job.

• **Object:** Objects are entities in the system that require protection. Access right to the object may be directly given to the user or may be associated with the user's role.

Capability Structure

The design of the developed capability-based system is based on the flexible approach of IPv6, which uses both unique ID as well as location as a mechanism to allow communications between objects [131]. This approach improves performance and increases security [132].

Every object and its properties can be identified using capabilities. Therefore, it is necessary that capabilities are managed and protected from being created or modified in an unauthorised manner [30][119]. The format of a capability-based system is shown in Figure 6.3.

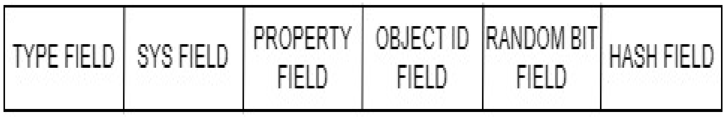


Figure 6.3: Capability format

1. Type Field (8 bits): This field is used to specify the type of object capability that is being used. Types include users, digital assets, facilities, etc. The type field also includes a TIMED type which is used to indicate a timed-based capability that is only valid for a specified period of time after which the system will refuse to grant any rights to the holder of that capability.

In addition, to help administer the system, a special object type known as a Capability List (CL) has been created. The CL is used to group a list of capabilities together. This is explained in more detail below.

2. SYS Field (4 bits): This field is used to help in managing capabilities. The capability related fields are given by four bits. The structure of the SYS FIELD is shown in Figure 6.4:

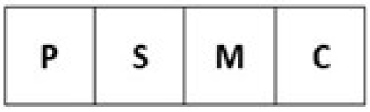


Figure 6.4: SYS field structure

- *The Private or P bit:* This bit is used to restrict the list of people holding the capability. With a private capability, the capability for the object as well as the capability of the subject or the person invoking the object capability must both be presented. Example scenarios include accessing to the hospital main entrance should be a public capability, whereas, accessing to a doctor's office should be a private capability since only authorised personnel is allowed to access it.

- *The System or S bit:* This indicates whether the object involved has been created by the system, or by an application or a user. This means that the capability created by the system cannot be modified or deleted by users or applications. For example, the hospital management creates the hospital concourse capability in the hospital environment. This means that a doctor or a nurse cannot change this capability, only the hospital management system can change it.

- *The Master or M bit:* This bit indicates that the capability was created by a Certificate Authority (CA). This capability is usually created when the object is created. If this bit is not set, it means that this is a proxy capability. Proxy capabilities are derived from master capabilities and cannot be derived from other proxy capabilities. For example, the doctor creates a treatment report file. As the owner of the file, the doctor is given the Master capability for the file which allows him/her to read, write, and delete this file. However, the doctor may want some personnel to be able to read from and write to the file, although, he/she does not want them to delete this file. Hence, the doctor cannot pass them the master capability. Therefore, a proxy capability needs to be created to allow this personnel to read from and write to the file.

- *The Change or C bit:* This bit is used to indicate if this capability is changeable. This means that if this bit is set, the proxy capabilities can be derived from the master capability. In a hospital environment, any capabilities for the main entrance or car parks should not be changed as everyone has to be able to access them.

3. Property Field (12 bits): This field is used to define the properties of the object. This field relates to the properties or functions of the object that the capability represents. For example, the property of a doctor can be subdivided into whether the doctor can access EHR data, order medical tests for patients, discharge a patient, etc. It could also include the rank and speciality of the doctor. For example, whether the doctor is a consultant or medical student and whether he/she has a specialist area such as anaesthesia, paediatrics, surgeon, etc.

4. Object ID (72 bits): This field is used to uniquely identify the object in the system. Location/ID split network addressing is used [118], where the ID is the standard Extended Unique Identifier (EUI) and uniquely identifies the object.

5. Random Bit Field (16 bits): The random bit field provides unforgeability. This field helps to uniquely identify the object. The random bit field is generated after the type field, sys field, property field, and object ID field are created. When proxy certificates are created, a new random field is generated.

6. Hash Field (16 bits): The hash field is used to allow the detection of the tampering of capabilities. When a capability is created, the type field, sys field, property field, and object ID field are first generated, followed by the random bit field. Finally, these fields are used to generate a SHA-1 hash which is placed in the hash field of the capability.

Two issues which have hindered the widespread use of capabilities are the casual tampering and revocation of capabilities. This capability structure has a hash field that prevents causal tampering. Moreover, the random bit field also provides the ability to enable easy revocation of capabilities as this can be done by simply changing the random field and recomputing the capability, hence revoking previous versions [119].

The Capability List (CL)

Lists of capabilities were created as part of the work described in this research. These lists are used to manage people working at an institution such as a hospital. There are three different types of capability lists:

- Common: A common capability list (Public Capability) belongs to all users in the system.

- Role-based: A role-based capability list is assigned to different employees based on their role. For example, doctors are able to access EHRs and medical equipment or can request a blood test result from a laboratory. Hence, these lists of capabilities can be defined using different role-based types such as doctors, nurses, hospital staff, etc.

- Personal: A personal capability list is used to manage the personal items or spaces of users. This includes access to the personal office, personal correspondences (text messages, emails), etc. Hence, this personal capability list will be a list of private capabilities associated with the owner of the object.

Table 6.1 briefly represents how to apply the use of capability list in a real hospital environment. Main persons in hospital environments (e.g., doctor, nurse, IT staff) hold access rights based on their roles. Devices, locations, digital data, and IT infrastructure are assigned the capability list; Common (C), Role-based (R), and Personal (P) and can only access by authorised capability IDs.

Object		**Doctor**	**Nurse**	**Technician**	**IT staff**
Device	Hand sanitiser dispenser	C	C	C	C
	Medical cart	R	R	x	x
	Blood pressure meter	R	R	x	x
	X-Ray machine	R	x	R	x
	Personal computer	P	P	P	P
Location	Main entrance	C	C	C	C
	Parking	C	C	C	C
	Counselling room	R	R	x	x
	Operating theatre	R	R	x	x
	Laboratory	R	x	R	x
	Informatics unit	x	x	x	R
	Personal office	P	P	P	P
Digital data	EHRs	R	R	x	x
	Clinical photograph	R	R	R	x
	X-Ray film	R	R	R	x
	Consent form	R	R	x	x
	Personal correspondence	P	P	P	P
IT infrastructure	Public WIFI	C	C	C	C
	On-premise data storage	x	x	x	R

Table 6.1: Hospital environment capability list

Capabilities - Generating Rules

The capability-based system presented above is very flexible and so additional rules are necessary to ensure proper usage when generating capabilities. These rules are given below:

1. A capability is created for an object when an object is being created for the first time. This is called a master capability and the owner of the object will be designated as the owner of that capability. Master capabilities must be created by a CA that can issue digital certificates related to this capability.

2. If the change bit in the master capability is set, then it is possible to create proxy capabilities from the master capability.

3. Proxy capabilities must have a new random bit field and a new hash field.

4. Proxy capabilities cannot be created from other proxy capabilities. This rule is necessary to prevent the creation of capabilities by unauthorised persons.

5. System capabilities cannot be generated or changed by users. This rule is needed to protect key entities such as operating systems as well as access to system services.

Benefits of Capabilities

This section has shown that capabilities can be used in a flexible manner to provide AAAC in the developed prototype and by extension, for many other environments. For hospital environments, RBAC for humans is sensible and can be easily implemented. However, by requiring all objects to have a capability including devices as well as digital assets such as EHRs, capabilities can be used as a key component of an overarching securing architecture for future healthcare systems.

6.4.2 Secure Transactional Layer

A Secure Transactional Layer was developed in this work. The purpose behind developing the Secure Transactional Layer is to achieve transactional security in the communication between a client and a server by encoding the data type and its value. Capabilities are applied in this layer to provide authentication and authorisation mechanisms. Further, a Secure Remote Procedure Call (SRPC) was developed and implemented into the Secure Transactional Layer to secure data transferred between clients and servers.

SRPC is a technique of inter-process communication between clients and servers to exchange data or execute some instructions. The idea is to use a typed remote procedure call. With this technique, each argument that is passed between a client and a server must have a defined type as well as a value. The types defined by SRPC is shown in Table 6.2:

TYPE	PARAMETER_NO	NUMBER OF BYTES
INT	1	4
U_INT	2	4
SHORT	3	2
U_SHORT	4	2
CHAR	5	1
U_CHAR	6	1
LONG	7	8
U_LONG	8	8
FLOAT	9	4
DOUBLE	10	8
CAPABILITY	11	16
ARRAY	12	SIZE OF ARRAY*SIZE OF PARAMETER TYPE
USER_DEFINED_TYPE	13	VARIABLE

Table 6.2: The types supported by SRPC

As shown in Table 6.2, the basic data types (e.g., INT, SHORT) are represented. The CAPABILITY type is used to increase transactional security by allowing the capability structure to be included directly into the SRPC, thus allowing AAAC in every SRPC call. The ARRAY type is used to represent a collection of similar objects. Finally, the USER DEFINED TYPE is used to allow users to define their secure data passing structures between clients and servers. With this type, the programmer must provide the SRPC routines to encode and decode the USER DEFINED TYPE.

Since SRPC supports an idea of type array, this is an improvement of a traditional RPC where type information is not passed in the RPC call and hence only the interface definition of the call is used to interpret the arguments. This, however, can be easily abused as in the case of buffer flow attacks that plague Web-Servers. SRPC, therefore, enables the endpoint receiving RPC calls to check whether the correct types and values have been sent before servicing the request.

In addition, this approach can detect changes in the data due to human error since it is possible to check that both the type and value passed are correct. This is, therefore, good for configuration systems where human error is quite common.

Table 6.3 shows the various data types associated with patient records.

ID (INT)	Name (CHAR)	Gender (CHAR)	DoB (UCHAR)	Contact_No (UCHAR)	Address (CHAR)	Emergency_Contact_No (UCHAR)
1	John Smith	Male	130291	07648832990	23 Langdon Park, London SE166AP	02080342615
2	Rebecca Davis	Female	030978	07466345613	4 Grove Road, Birmingham B145TX	07988743214
3	Nathan Omar	Male	310895	07588421499	Flat 1, 12 Green close, Oxford OX11AA	07828234556

Table 6.3: Example of a patient record

Table 6.4 shows a use of SRPC to encode a patient record.

Field	Type	Value	Add Type Info	Add Type Value
Patient record	User Defined Type = 13	1 - application specific	No. of entities	7 (Made up of seven fields)
ID	INT =1	1		
Name	ARRAY = 12		No. of entities	10
	CHAR = 5	John Smith (includes space)		
Gender	ARRAY = 12		No. of entities	4
	CHAR = 5	Male		
DoB	ARRAY = 12		No. of entities	6
	UCHAR =6	130291		
Contact_No	ARRAY = 12		No. of entities	11
	UCHAR = 6	07648832990		
Address	ARRAY = 12		No. of entities	32
	CHAR = 5	23 Langdon Park, London SE16 6AP		
Emergency_Contact_No	ARRAY = 12		No. of entities	11
	UCHAR = 6	02080342615		

Table 6.4: Encode a patient record using SRPC

Figure 6.5 below shows the SRPC format.

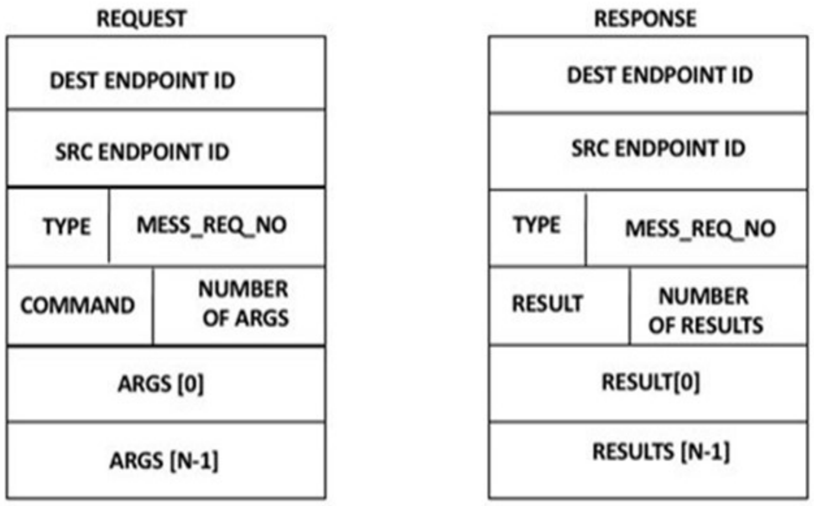


Figure 6.5: SRPC format

The fields of the SRPC format (Figure 6.5) are detailed below:

- DEST ENDPOINT ID (64 bits): The communication endpoint of the destination. This uniquely identifies the communication endpoint of the remote end. This is defined using DEST IPV4/IPV6 ADDRESS, DEST TCP/UDP port. Extra bytes must be set to zero.

- SRC ENDPOINT ID (64 BITS): The communication endpoint of the source. This uniquely identifies the communication endpoint of the local end. This is defined using SRC IPV4/IPV6 ADDRESS, SRC TCP/UDP port.

- TYPE (8 BITS): The type of message: REQUEST = 1, REPLY = 2.

- MESSAGE_REQ_NO (24 BITS): The message sequence message number. This must be the same for the request and subsequent response.

- COMMAND (16 BITS): The command being asked to be executed by the client. This is an application-specific parameter.

- RESULT (16 BITS): The result of executing the command. This again is an application-specific parameter.

- NUMBER OF ARGS: The number of arguments in the request.

- NUMBER OF RESULTS: The number of result parameters in the response.

- ARGS[0] — ARGS[N-1]: The arguments of the request.

- RESULT[0] — RESULTS[N-1]: The result parameters of the response.

The argument and result parameters are specified using SRPC types shown in Table 6.4 above.

Benefits of SRPC

SRPC allows a much more secure transactional environment to be developed which makes sure that clients and servers can interact securely. In addition, the use of the capabilities means that SRPC allows AAAC to be effective not just for people and devices but also for clients and servers.

6.4.3 Service Management Layer

The Service Management Layer (SManL) was developed to manage services by specifying the functions of these services and requirements needed to run them. There is an increasing need to deploy services in different types of networking environments and on many different types of hardware. However, with the deployment of Cloud systems in which servers run in a virtualised environment, multiple services from different domains may share the same hardware system. Furthermore, because of the large amount of data being generated in healthcare environments, Cloud systems are increasingly being used to store and process health data [130]. There is a legal requirement (GDPR) to always keep EHRs safe. Hence, the security challenges of using Cloud services for healthcare environments must be addressed to ensure that patients, hospital staff, and visitors are safe.

The challenges can be articulated as follows:

1. A secure execution environment: there is a need to ensure that services are not hosted on unsafe Cloud hardware and Cloud servers are not corrupted by malicious or badly implemented servers.

2. It is necessary to be able to work out the best place to run a service at any point in time. This may be affected by many factors including location, cost, QoS, and security requirements.

3. There must be the ability to securely transfer services between Clouds. A new security protocol namely the Resource Allocation Security Protocol (RASP) [133] was proposed to secure service migration over Cloud infrastructure. It supports mobile services to ensure that transfer of resources between different Cloud environments is safe.

Service Management Framework (SMF)

To respond to the security challenges highlighted above, a new approach to delivering services is now required in which it is possible to look at issues of security, deployment, replication, or migration of services on a regional, national, or global scale. To achieve this, a new entity called the Service Management Framework (SMF) was proposed. The new environment is shown in Figure 6.6.

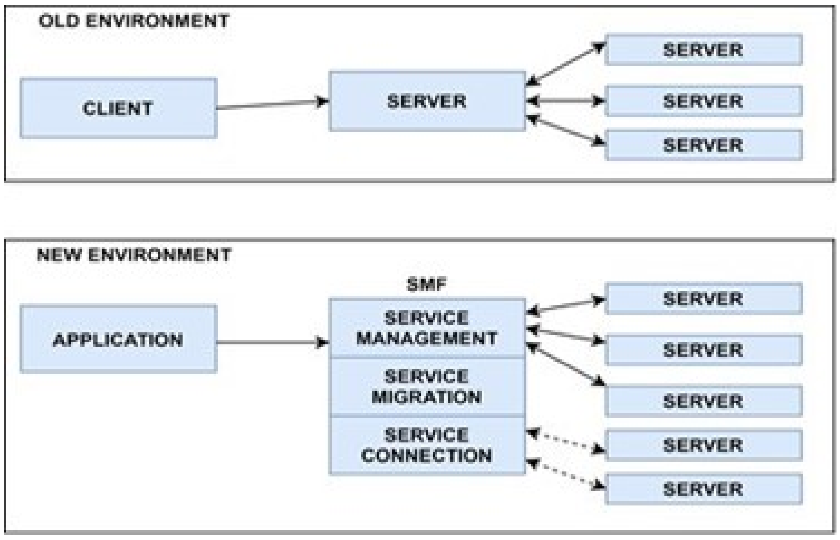


Figure 6.6: Effect of introducing SMF in the client-server environment

Figure 6.7 shows the details and interactions of the Service Management Framework.

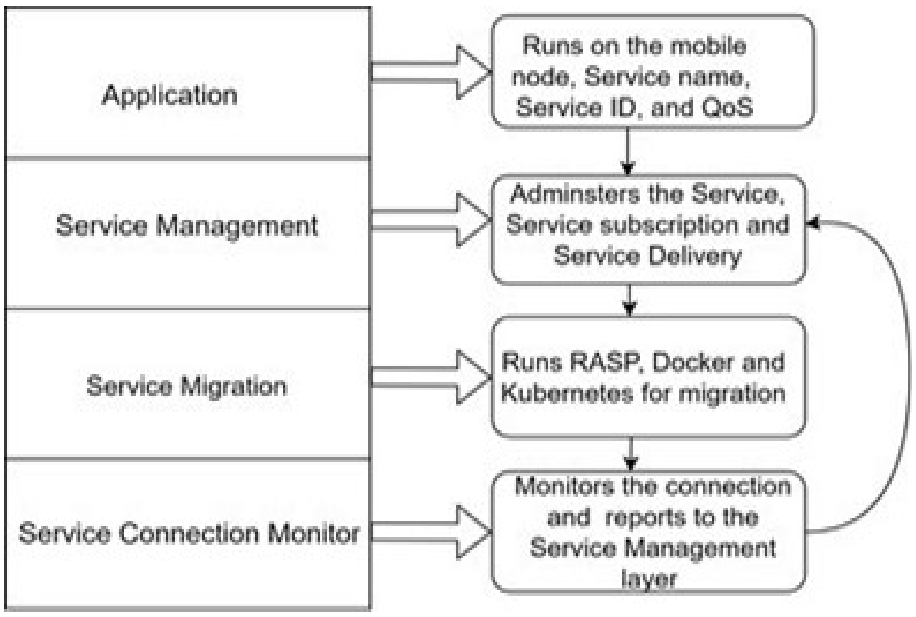

Application
Service Management
Service Migration
Service Connection Monitor
Runs on the mobile node, Service name, Service ID, and QoS
Adminsters the Service, Service subscription and Service Delivery
Runs RASP, Docker and Kubernetes for migration
Monitors the connection and reports to the Service Management layer

Figure 6.7: Details of the Service Management Framework

The SManL ensures an application is assigned to a relevant secure server and is given the correct parameters to securely use that server. SManL, therefore, supports two general interfaces. The first allows a Service Provider to register a service. This is shown in Table 6.5.

Source->Destination	**Type of Message**	**Actions at Destination**
Service Provider ->SManL	Register Service Request [Service name, Service version, Resource requirements (CPU, Memory, Network, Storage), Restriction list, Security level, QoS, Location Restrictions, Maximum Replicas, Actual binary of the service]	SManL first checks to see if the service and version are not already registered. If not, it creates a new service structure and populates this structure with data passed by the service provider. It then creates a unique Server ID and a Service Capability.
SManL->Service Provider	Reply to Register Service Request [Success = 1; Failure = 0; Server ID, Server Capability]	The Service Provider stores the returned parameters

Table 6.5: Registration service protocol for service providers

The second interface allows an application to request a service. SManL then provides the application with the necessary parameters to contact a server that runs the service. This is shown in Table 6.6.

Source->Destination	Type of Message	Actions at Destination
Application ->SManL	Request Service Request [App Node ID, Service name, Service version, QoS Requirements, Node Location, and Network Interfaces]	SManL first looks to see if there is a registered service that meets the request. If a service structure is found, SManL then sees if there already is a server which runs the service close to the application. If there is no server available, then a Cloud System is selected, and a new server is started.
SManL->Application	Reply to Request Service Request [App Node ID, Service name, Service ID, Server Location, QoS Requirements and Server IP address, Service Capability]	The Application uses the Server IP address and Service Capability to contact the server and use the service.

Table 6.6: Request service protocol for applications

Benefits of Service Management Framework

The Service Management Framework is a new way of deploying and managing services in distributed environments. It allows clients to find services and provides communications endpoints and capabilities which allows a reliable session to be developed. It, therefore, increases the security, efficiency, and management of services, and will be a key part of Future Internet.

6.5 mHealth Storage Application

A key part of the new framework is the provision of mHealth applications and services for healthcare systems. In terms of required applications and services, there is the need to develop a secure mHealth storage system (as part of the mHealth application) that provides several functions including creating, storing, modifying, and deleting healthcare records.

6.6 The Prototype Scenario

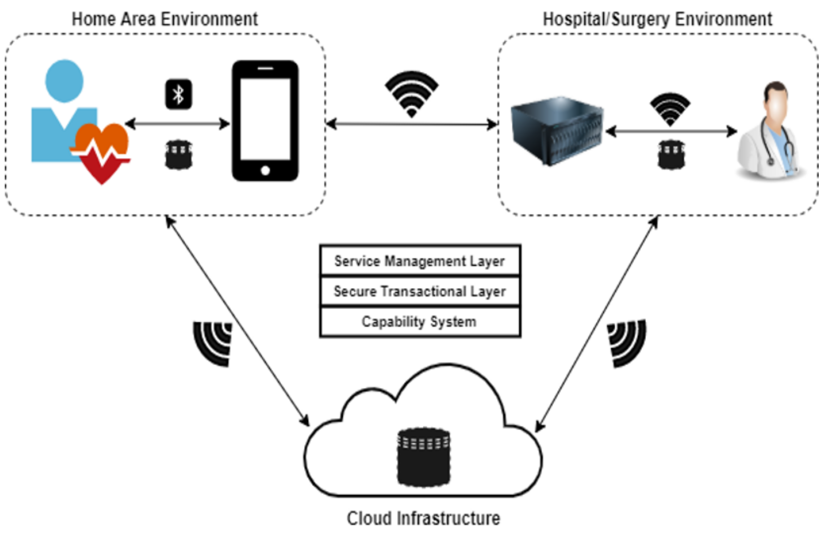


Figure 6.8: A proposed prototype for secure healthcare services

The system being developed is shown in Figure 6.8. Generally, healthcare data, which is classed as sensitive data, are collected from patients through wearable devices. These wearable devices communicate with a mobile device through an authorised mHealth application that has been installed. A mobile device receives collected healthcare data from a wearable device. The healthcare data (received) may be collected on a mobile device database and/or transmitted to be stored on a hospital database and Cloud storage. The Network Memory Server (NMS) is implemented for storage, to control and manage healthcare data residing in the system.

All mHealth devices, clients, and servers must be connected to the Service Management Layer (SManL). To be able to use a service, the mHealth application has to send a service request by giving a service name to the SManL to find a suitable server. The SManL then takes the service name (from the request) and scans through a list of services. If there is a valid service for the request, it will return the Service Structure for that service which contains a list of servers that is currently running that particular service. After that, the SManL chooses a server that contains the requested service and contacts the server. The chosen server accepts a request from the SManL, the SManL then passes an instruction to the mHealth application. The mHealth application can now send a session start request and a service request to the chosen server. The server accepts the requests so that the mHealth application can use the service. Finally, the mHealth application sends an end request to the chosen server to terminate the session.

The client now initiates the request for connecting to the server through SRPC and waits for a response to be returned from the server. Once the connection to the server is successfully established, the client sends encoded instructions to the server where it decodes and processes the data then returns a response to the client.

Capabilities will be used as a main authentication mechanism in the system. Every entity in the system must be represented by a capability. Capabilities manage which entity will be able to gain access to each object in the system as well as identify that the request is on behalf of which entity. Each entity will be represented using a fixed string which statically assigns to different capabilities. In addition, capabilities are passed in the SRPC call.

6.7 Chapter Summary

In this chapter, a prototype implementation was proposed. It was developed as an experimental environment for the proposed information security framework. Newly developed mechanisms including Capability, Secure Remote Procedure Call (SRPC), and Service Management Layer were introduced as main contributions of the research. Moreover, their benefits were explained in detail. Therefore, there is a need to develop an mHealth secure storage as a part of the filesystem and to test that the implemented prototype can be applied into real mHealth environments.

Chapter 7

Implementation, Testing, and Evaluation of an Information Security Framework for mHealth Systems

7.1 Introduction

This chapter explains the implementation, testing, and evaluation of the proposed information security framework for mHealth systems in an actual healthcare environment. The aim of this chapter is to provide details about the implemented prototype that is developed as the experimental system to verify that the proposed information security framework can truly deliver end-to-end security for mHealth systems.

7.2 The Prototype System for mHealth Secure Storage

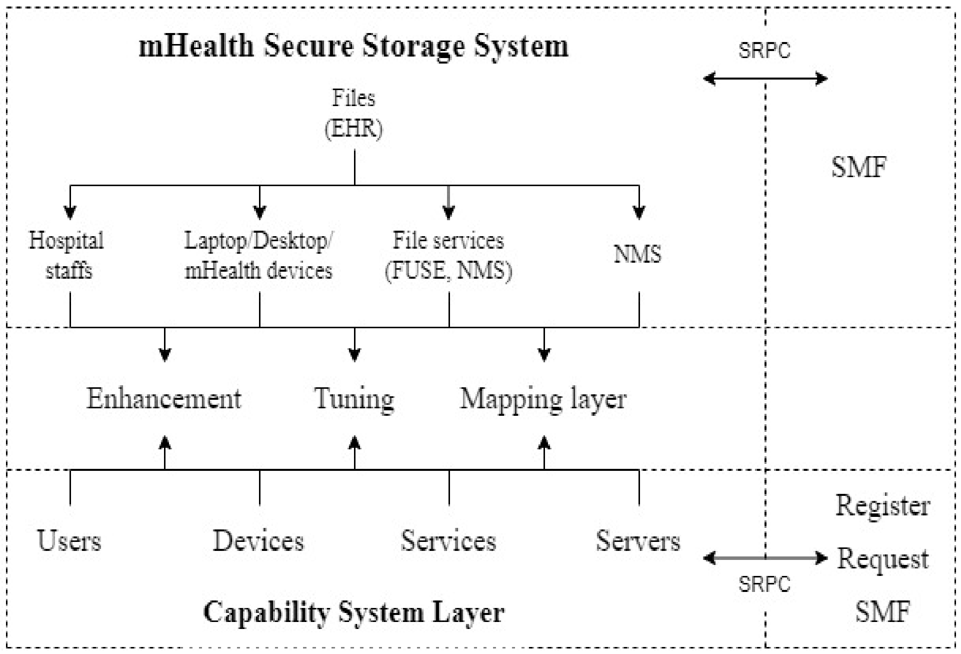


Figure 7.1: Enhancing an mHealth secure storage system

In Capability System Layer, users, devices, and services are created and then registered to the Service Management Layer (SManL). Each of them has its own unique Capabilities which provide access rights to certain objects such as files, devices, physical areas, in the system. Servers are also assigned to services in this layer.

The Basic Capability System Library was written by the ALERT Team and is only a basic system that allows the creation of users, devices, services, and enables servers to be added to implement services. When users, devices, services, and servers are created, only very generic details are assumed. These types are given as CAP_USER, CAP_DEVICE, and CAP_SERVICE as their Capability types respectively.

For a hospital environment, we need to create users that have specific roles such as doctor, nurse, administrator. Hence, the following new Capability types were introduced and given values as shown below:

```
#define CAP_DOCTOR    13    /* capability doctor */

#define CAP_NURSE    14    /* a nurse */

#define CAP_ADMIN    15    /* an admin person */

#define CAP_TECH    16    /* a tech person */

#define CAP_NONTECH    17    /* non-technical retail, etc */

#define CAP_VISITOR    18    /* a visitor */

#define CAP_PATIENT    19    /* a patient */
```

In addition for each type of role described above and where appropriate, rank and speciality fields are also used to subdivide these new types. For example, a user can be a doctor (role), a consultant (rank), and a dermatologist (specialist).

The structure of doctor capability (CAP_DOCTOR) has presented as below.

The structure of CAP_DOCTOR

1. Type field (8 bits) – Doctor

2. Sys field (4 bits)

- P bit: If set, this user is private. → Set

- S bit: If set, this user is created by the system. → Set

- M bit: If set, this user holds a master capability. → Set

- C bit: If set, this user capability is changeable. → Not set

3. Property field (12 bits)

- BIT(0) – If set, a person has access to EHR → Set

- BIT(1) – If set, a person can operate/supervise medical equipment → Set

- BIT(2) - If set, a person can access hospital IT services → Set

- BIT(3) – If set, a person has access to specialised areas in hospital → Set

- BIT(4) – If set, a person can order medical tests and drugs/treatment for patients $\rightarrow$ Set

- BIT(5) – If set, a person can order that a patient is sent to or discharged from hospital/ward $\rightarrow$ Set

- BIT (6)(7) - Rank (2 bits). - 4 levels $\rightarrow$ Medical student, Junior doctor, Senior doctor, and Consultant

- BIT (8)(9)(10)(11) - Specialist (4 bits) – 8 areas $\rightarrow$ Anesthetist, Emergency medicine, Radiologist, Psychiatry, Surgery, Oncologist, Dermatologist, Haematologist

4. Object ID (72 bits)

- National Insurance (NI) number

5. Random bit field (16 bits)

6. Hash field (16 bits)

Since doctors have access rights to files, Table 7.1 shows an example of doctor access rights based on their ranks and specialists.

Role	**Rank**	**Specialist**	**File**
Doctor	Medical student	N/A	r–
	Junior doctor	N/A	rw-
	Senior doctor	Anaesthetist	rw-
		Emergency medicine	rw-
		Padiologist	rw-
		Psychiatry	rw-
		Surgery	rw-
		Oncologist	rw-
		Dermatologist	rw-
		Haematologist	rw-
	Consultant	Anaesthetist	rw-
		Emergency medine	rw-
		Radiologist	rw-
		Psychiatry	rw-
		Surgery	rw-
		Oncologist	rw-
		Dermatologist	rw-
		Haematologist	rw-

Table 7.1: Access rights to files based on role, rank, and specialist

7.2.1 Filesystem structure

The filesystem consists of three main elements including data blocks, superblock, and inode.

1. A block of data in the filesystem is given by:

```
struct block_id{
unsigned int local_blockid;    /* local block id */
unsigned int global_block_id;    /* global block_id */
struct capability_raw;    /* capability for the block for NMS */
unsigned int status;    /* status of the block on client */
unsigned char *data;    /* the actual buffer */
};
```

The status field is defined as follows:

```
#define BMEM    BIT(0)    /* the block is in memory */
#define BND    BIT(1)    /* block contains no data */
#define BMOD    BIT(2)    /* the block has been modified */
#define BGL    BIT(3)    /* the block has been allocated globally
                           i.e., on the NMS */
```

In this file system, blocks are first allocated; locally and then allocated on the NMS if permanent storage is required for the file.

2. Superblock: The superblock stores information about the status of inodes and data blocks which implies availability of inodes or data blocks.

3. Inode: Each file in the system is managed using an inode. In an inode structure, it stores information regarding file objects. The inode structure is given below:

```
struct ux_inode {
    char filename[UX_NAMELEN];    /* filename */
    unsigned int filenamelen;    /* length of file name */
    struct capability_raw mcap;    /* the master capability */
    struct capability_raw rwcap;    /* the read write capability */
    struct capability_raw rocap;    /* the read only capability */
    struct capability_raw xcap;    /* the execute capability */
    struct file_id fid;    /* file object structure */
    int i_type;    /* type:inherit from parent:public,group,etc */
    int i_oaccess;    /* access allowed by other groups */
    int i_inode_access;    /* whether inode access is allowed */
    int i_inode_access_res;    /* restrictions to inode_access */
    struct ux_directory *i_di;    /* if the file is a directory */
    uint32_t i_no;    /* inode number */
    uint32_t i_par_no;    /* the parent inode */
    uint32_t i_status;    /* the status bits */
    uint32_t i_mode;    /* whether file or directory */
    uint32_t i_nlink;    /* number of links to storage systems */
    uint32_t i_atime;    /* last time accessed */
    uint32_t i_mtime;    /* last time modified */
    uint32_t i_ctime;    /* time when status changed */
    int32_t i_uid;    /* uid of owner */
    int32_t i_gid;    /* group owner of file */
uint32_t i_blocks;    /* the number of blocks in the file */

int32_t b_offset;    /* the number of bytes in the last blocks */

int32_t f_size;    /* the size of the file in bytes */

uint32_t i_addr[UX_DIRECT_BLOCKS];    /* the block_id of the individual

                                        blocks */

struct ux_inode *next;    /* inode list */

struct ux_inode *prev;    /* inode list */

};
```

When the file is created, the inode and the capabilities of the file are also created. The capabilities of the file include (1) Master capability, (2) Read/Write capability, (3) Read-only capability, and (4) Execute capability. These capabilities are in the inode as shown above.

In terms of capabilities; the capability type for file is given as CAP_FILE. The structure of file capability (CAP_FILE) is as follows.

The structure of CAP_FILE

1. Type field (8 bits) – File

2. Sys field (4 bits)

- P bit: If set, this file is private. → Set

- S bit: If set, this file is created by the system. → Not set

- M bit: If set, this file holds a master capability. → Not set

- C bit: If set, this file capability is changeable. → Set

3. Property field (12 bits)

- BIT(0): If set, Read → Set

- BIT(1): Read/Write → Set

- BIT(2): Execute → Set

- BIT(3): Delete → Set

4. Object ID (72 bits)

• OBJECT_ID Flags

- BIT(0): If set, the file is a directory.

- BIT(1): If set, the file is an executable file.

• General access type

- 00: Public file

- 01: Private file

- 10: Group file

- 11: EHR

- 32-bit IP address of the device that manages the file

- 28-bit inode number of the file on that device

5. Random bit field (16 bits)

6. Hash field (16 bits)

If the file is a directory, there will be a pointer to a directory structure which is shown below:

```
struct ux_directory{
    int max_entries; /* maximum no.  of entries in this directory */
    int num_entries; /* number of current entries */
    char *file_table; /* so we will go by 10 */
};
```

The file_table contains the information about files in the directory. The directory entry structure is given below:

```
struct ux_dirent{

    uint32_t d_ino;    /* inode number for file */

    uint32_t namelen;    /* length of filename */

    char d_name[52];    /* this name is relative to this directory */

    };
```

The maximum number in a directory is given by max_entries which is set to 20. If more entries are needed, the max_entries will be increased by 20 and a larger file_table is made, and the allocated entries are copied into the new structure.

Every file in the filesystem is related to an inode structure in which the capabilities were assigned to control accesses to each file in the system. In addition, each inode has a type variable which indicated whether which type of file or directory is being accessed. According to the hospital filesystem (Figure 7.2), layout types of files include.

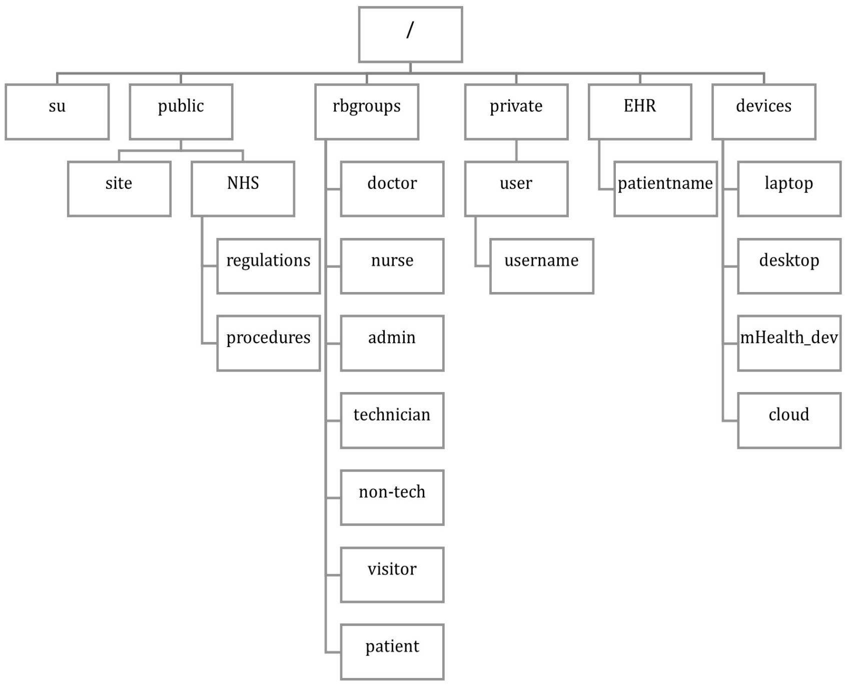


Figure 7.2: The hospital filesystem directory

- **SU**: files belonging to the superuser (CAP_SU)
- **PUBLIC**: public files which can be access by everyone. (CAP_PUBLIC).
- **GROUP**: these flies are groups which are role-based (CAP_DOCTOR, CAP_NURSE, CAP_ADMIN).
- **PRIVATE**: these files belong to the individual users and can only be accessed by individual (CAP_PRIVATE).
- **EHR**: these files are related to patient records and are controlled by the administrators (CAP_ADMIN).
- **DEVICES**: these are files related to devices and are controlled by technicians (CAP_TECH).

These settings are applied to the top directory, but all sub-directories are set to inherit these settings. In order to access a file or directory, the system first checks if you are from the group that control the directories, then you will be allowed to access to the directory. For example, if someone wants to access an EHR file. The system will first check whether they are allowed directly access EHR records. In this system only doctors, nurses and admin staff are allowed to directly access EHR records. The system will then check whether they are from the CAP_ADMIN group, if so, then they would be allowed to access the file because this directory is controlled by CAP_ADMIN. Everyone else would not be initially allowed access. However, doctors and nurses would need access to EHRs. Hence two other mechanisms are introduced. There is another variable called the `oaccess` variable which says what access is allowed for other groups. For the EHR system, this `oaccess` is set to read-only. This means that doctors and nurses can read any EHR, but they cannot change the content based on this access. In order to have to allow a doctor looking after the patient to change the EHR, an inode access approach is used. Each user has three sets of Inode access tables which include:

1. Public inode table: All users will have a pointer to this table to be able to access public files in the system.

2. Group inode table: Each group will have their own inode table related to their roles and ranks.

3. Private inode table: Access to data that is only available for the user (e.g., email, text).

Hence if the doctor wants to read-write access to the EHR of the patient that he/she is looking after, the read-write capability for the file is placed in the personal inode_access node table for the doctor. Thus, the doctor or nurse looking after the patient can access the file. The mHealth Secure Storage System is shown in Figure 7.3.

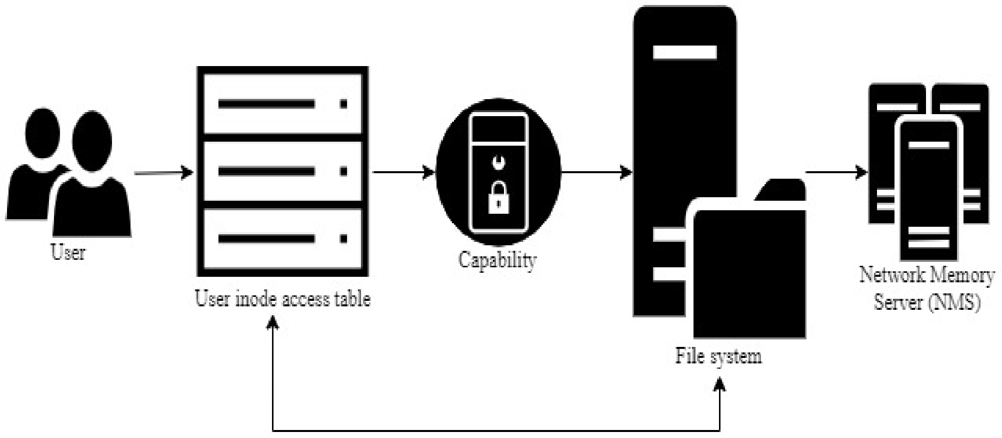


Figure 7.3: mHealth Secure Storage System

7.2.2 Storage mechanisms

In general, most data is stored on a hard disk or a solid-state drive (SSD). However, there is now an interest in block network storage called Network Memory Server (NMS). The NMS stores blocks of data in secure random memory (RAM) over the network. Hence, it can be used as persistent storage for filesystems. Thus, we need to define the messages between the Filesystem and the NMS. The capability for the block on the NMS can be specified as below.

- TYPE = CAP_BLOCK
- PROPS =

  BIT(0) READ_ONLY

  BIT(1) READ_WRITE

  BIT(2) DELETE_BLOCK
- BLOCK_OBJECT_ID

```
unsigned int local_block_id;    /* block on client */
unsigned int global_block_id;    /* block on NMS */
unsigned char netadmin;    /* see below */
```
- netadmin fields

```
SF = 11 = 3:    /* block can be globally accessible */
M = 0:    /* block is a single entity not multicast */
S = 1:    /* the block must stay on NMS */
INF = 0;    /* any interface can be used */
```

In addition, the messages between the NMS and the filesystem all use the same message format as detailed below:

```
struct nms_rpc {
    unsigned short command;    /* command:GET_BLOCK,READ_BLOCK,
                                  WRITE_BLOCK,DELETE_BLOCK */
    unsigned short reply;    /* the reply to command from the NMS */
    unsigned int local_block_id;    /* this is the block on the
                                       client machine */
    unsigned int clnt_ip_address;    /* the IP address on the client
                                          machine */
    unsigned int global_block_id;    /* the block on the NMS */
    struct capability_raw block_capr;    /* the capability for the
                                              block on NMS */
    char data[1024];    /* data for the block */
};
```

7.3 Key Prototype Functions

To test the functionalities of the prototype, the implemented prototype was developed in C language for the Linux platform and tested in the user space mode.

7.3.1 Secure Management Layer (SmanL/SMF)

The SMF was developed to manage services by specifying the functions of these services and the requirements needed to run them. In this system, only basic SMF functions are required to register users, devices, services, add a server to service, and request for services. The SMF provides the following basic functions.

1. `REGISTER_USER`

This function allows the user to be registered to the system. The system will provide the local user ID as well as the global user ID once the user has been registered. It also collects information of the registered user including name, surname, role, rank, specialist, as well as the capability of the user.

2. `REGISTER_DEVICE`

The function allows the device to be registered to the system. Each device will have a unique local device ID and the global device ID. Information about the device including the name of a device, the name and surname of the device owner, and the device capability are also collected once the device is registered.

`3. REGISTER_SERVICE`

The function allows the service to be registered to the system. Similar to users and devices, the service will be provided the local service ID as well as the global service ID. The system also collects some information about the service including the service name, the description of service, the version of service (i.e., 0 is the latest version), the TCP/UDP port number, and the service capability.

`4. ADD_SERVER`

This function allows the server to be added to the service. It also collects information about the server including the server global ID given by NMS, the server's name, the server location using IPv4 address, the server status, and the maximum load of the server. Moreover, it also stores the information of the service this server is added to. The information includes the local and global ID of service, the service name, the service version, and the service capability.

`5. REQUEST_SERVICE`

When the service is registered, the application must be able to request the service provided by servers. This function allows an application to request a service. After the application sends a service request to SManL, SManL will then provide necessary parameters to the application to contact the server which runs the requested service.

7.3.2 Capabilities

Initially, the capability system was developed to perform only basic functions including creating users, devices, services, and allowed servers to be added to implement services. The capability header file (`capability.h`) is the main file defining various types of capabilities (e.g., `CAP_USER, CAP_FILE, CAP_SERVICE`), and structures of the capability system. The capability system consists of three major structures as below.

1. Raw capability: The raw capability is the basic capability structure (Figure 6.3) that is used in the research. The raw capability gives information about the capability such as the type of capability, and the owner of the capability. If the capability is a proxy capability, the system will point to its master capability. Raw capabilities can be passed around by other entities such as users, devices, and services.

```
struct capability_raw{
    unsigned char type;    /*The type of capability*/
    struct obj_id objid;    /*The owner of capability*/
    unsigned short sys:4, props:12;    /*Sys field and Property field*/
    unsigned short rand;    /*Random bit field*/
    unsigned short hash;    /*Hash field*/
} Capability_R;
```

2. Capabilities place holder (ph): The capability ph is a bigger structure of the raw capability since it provides more details (e.g., status, the owner of the capability, the master capability) about the capability. Unlike raw capabilities, ph capabilities should not be directly used by other entities. The ph capability is created once the capability is created. The capability ph structure is shown below.

```
struct capability_ph {
    struct capability_raw capr;    /* the capability */
    struct obj_id objid;    /* the actual object */
    int reference;    /* reference */
    unsigned char status;    /* valid, active, revoked, expired */
    unsigned char cap_type;    /* user, superuser, apps, OS */
    unsigned char owner_cap_type;    /* owner cap_type */
    unsigned char ref_count_limit;    /* cap sharing limit */
    struct timeval tval_created;    /* time when created */
    struct timeval tval_modified;    /* time when modified */
    struct timeval tval_revoked;    /* time when revoked */
    void *struct_ref;    /* pointer to reference structure */
    void *full_cap;    /* pointer to full capability */
    struct capability_ph *master;    /* pointer to master_cap */
    struct capability_ph *owner;    /* pointer to cap owner*/
    struct capability_ph *next;    /* pointer to next cap */
    struct capability_ph *prev;    /* pointer to previous cap */
} Capability_Ph;
```

3. The full capability: The full capability is used to manage the relationship between the current capability and other capabilities as well as a list of private capability and a list of proxy capability. The full capability is created when the capability is created. The structure of full capability is given below.

```
struct capability_full{
    struct capability_ph caph;
    struct capability_plist*priv_capl; /* private c_list */
    struct capability_list *proxy_active; /* Active proxy c_list */
    struct capability_list *proxy_revoked; /* Rev proxy c_list */
    struct capability_full *next; /* pointer to the next one */
    struct capability_full *prev; /* pointer to the prev one */
} Capability_Full;
```

Capability System Entities

The capability system provides support for high-level entities which includes:

1. **Superuser** (`SU_CAP`): The person with privilege levels who manages a device or a set of devices in the system.

2. **User** (`USER_CAP`): The person who uses the system. In an mHealth environment, users are doctors, nurses, administrators, technicians, non-technicians, patients, and visitors.

3. **Device** (`DEVICE_CAP`): In the system, devices include laptops, desktops, mHealth devices, and servers.

Capability System Functions

The capability system allows users to perform some basic functions described as below.

1. Creating system entities: These functions allow the user to create superuser, users, devices, and service. After superuser, users, devices, and service are created, they will then contact SMF to register these entities. Moreover, the user can also create a server and add to a service.

- Making a superuser. This function must be called first and can only call once.

```
extern int make_superuser ();
```

- Making a user and returning a user pointer structure

```
extern int make_user (struct user_cap **);
```

- Making a device and returning a device pointer structure

```
extern int make_device (struct device_cap **);
```

- Making a service and returning a service pointer structure

```
extern int make_service (struct service_min **);
```

- Making a server for a service and returning the updated service pointer

```
extern int add_server (struct service_min **);
```

2. Displaying the structure of system entities

- Displaying a user structure

```
extern int print_user(struct user_cap *);
```

- Displaying a device structure

```
extern int print_device (struct device_cap *);
```

- Displaying a service structure

```
extern int print_service (struct service_min *);
```

3. Listing users, devices, and services in the system

- Listing users

```
extern int list_users();
```

- Listing devices

```
extern int list_devices();
```

- Listing services

```
extern int list_services();
```

4. Switching functions

- Switching to superuser mode

```
extern int switch_to_supervisor_mode();
```

- Switching to user mode

```
extern int switch_to_user_mode();
```

- Switching to a specific user

```
extern int switch_to_user(int user_no);
```

- Switching to a specific device

```
extern int switch_to_device(int dev_no);
```

- Switching to a previous user

```
extern int switch_to_previous_user();
```

- Switching to a previous device

```
extern int switch_to_previous_device();
```

5. Checking capabilities: Ensuring whether the capability is valid or revoked before any operation is performed on it, and who as well is the owner of this capability. This uses a TCP-like mechanism to ensure that the capability has not been tampered with.

```
extern int check_capability(struct capability_raw *);
```

Capability System Variables

Some key variables are accessible to the user including:

1. The seed for random bit field: It is set by default by the library. It should be changed unless the system is being used for a long time, for example for simulation. It should not be set be to zero.

```
extern unsigned int seed;
```

2. A pointer to the current user

```
extern struct user_cap *currentuser;
```

3. A pointer to the current device

```
extern struct device_cap *currentdevice;
```

The capability system only provides some basic structures for users, devices, servers, and services that are created in the system. For example, when `make_user()` is called, only a generic user is created. Therefore, `*enh` pointer is added as a pointer to the new structure which can be tuned to be specific to the system.

7.3.3 Communicating with the NMS using SRPC

SRPC provides a type-based system that is used to ensure that messages are properly formed. It also prevents common threats such as buffer overflow attacks. Hence the nms_rpc message was encoded using SRPC. Table 7.2 below shows a use of SRPC to encode the nms_rpc message (As shown in Section 7.2.2) .

Field	**Type**
command	U_SHORT = 4
reply	U_SHORT = 4
local_block_id	U_INT = 2
clnt_ip_address	U_INT = 2
global_block_id	U_INT = 2
block_capr	USER_DEFINED_TYPE = 13
data	ARRAY = 12
	CHAR = 5

Table 7.2: Encoding the nms_rpc message using SRPC

7.3.4 Filesystem

The hospital filesystem was developed to test access to the filesystem directory. Every user in the filesystem was provided three sets of an inode access table which were previously discussed in Section 7.2.1. Every file in the filesystem is related to an inode structure in which the capability was assigned to control accesses to each file in the system.

In the filesystem, the top-level directory has a structure as described below.

1. `inode`: The inode of the directory

2. `type`: The directory type (e.g., public, private, EHR)

3. `oaccess`: The access for people outside the group

4. `inode_access`: Identify whether the inode can override the system

5. `inode_access_res`: Identify whether there are any restrictions if overriding

6. `mcap`: The master capability

7. `rwcap`: The Read/Write capability for the file

8. `rcap`: The Read-only capability for the file

9. `capgroup`: The group capability of the top-level directory

10. `subdirectories`: A sub-list of directories

The filesystem provides some basic functions as described below.

`1. make_access_node_table ()`

Creating an inode access table and returning its capability structure

`2. list_filesystem ()`

Displaying information about the directory including name, the directory type, and the access right for people outside the group.

`3. find_tldir ()`

Searching the file directory

`4. check_access ()`

Verifying access rights of each directory

Moreover, each directory in the filesystem was assigned specific access rights based on the role of the user. Access rights include:

1. No access (`NOACCESS = 0`): The directory is not allowed to be access.

2. Read-only access (`ROACCESS = 1`): The directory can be read-only.

3. Read/Write access (`RWACCESS = 2`): The directory can be read and written.

4. All access (`ALACCESS = 3`): The directory can be accessed by everyone.

7.4 Testing and Evaluation

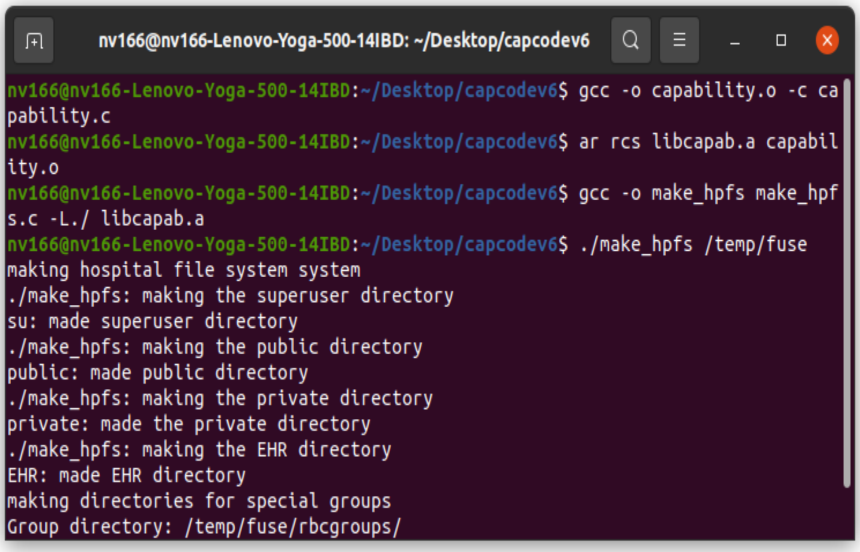
The rational of testing and evaluation was presented in Figure 2.3. The requirements including security requirements and healthcare environment components (Figure 2.2) to be achieved by the prototype were summarised. This test is designed to conduct on both Functional test (i.e., unit test and system test) and Non-functional test (i.e., security test).

In the unit testing, the test was conducted to ensure that each prototype component, including Capabilities, SMF, and SRPC, is functional and works as it was designed to. Moreover, the system must be functional and meets the specified requirements when every component is implemented as a whole. Furthermore, the security test was performed to confirm that the prototype provides a complete set of security requirements, including confidentiality, integrity, availability, authentication, authorisation, accountability, auditability, and reliability.

7.4.1 Creating a hospital filesystem

Firstly, the hospital filesystem is initialised (Figure 7.6). The system has already created the superuser directory, the public directory, the private directory, and the EHR directory. Each of them has access right as below.

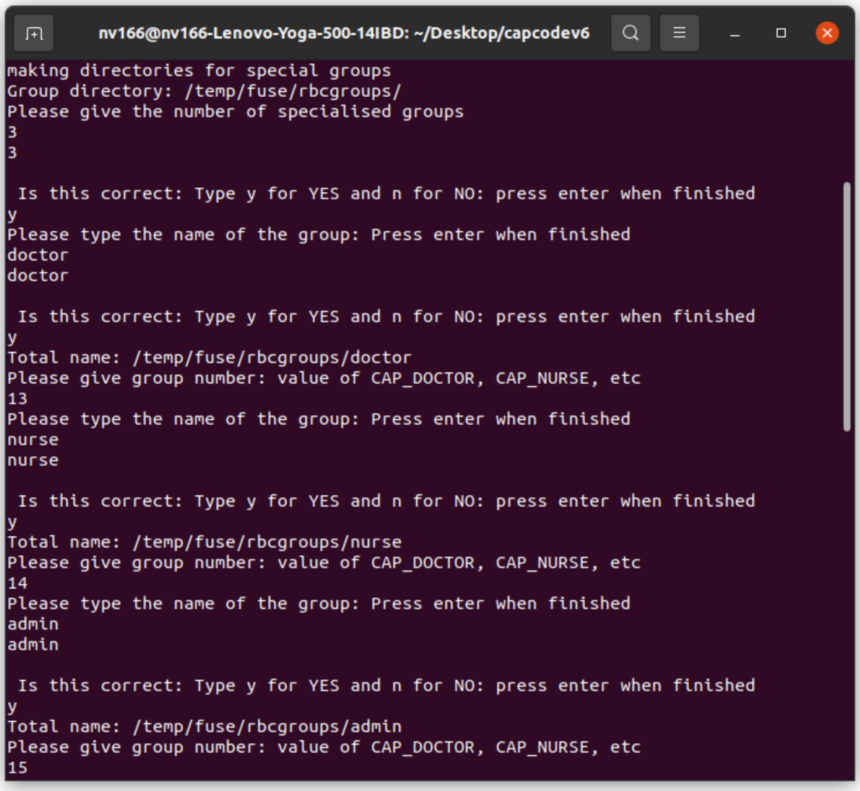
- **The superuser directory**: Only superusers can access this directory.
- **The public directory**: Anyone can access this directory.
- **The private directory**: Only a specific user can access this directory.
- **The EHR directory**: Only administrators can access this directory.



```
nv166@nv166-Lenovo-Yoga-500-14IBD:~/Desktop/capcodev6$ gcc -o capability.o -c ca
pability.c
nv166@nv166-Lenovo-Yoga-500-14IBD:~/Desktop/capcodev6$ ar rcs libcapab.a capabil
ity.o
nv166@nv166-Lenovo-Yoga-500-14IBD:~/Desktop/capcodev6$ gcc -o make_hpfs make_hpf
s.c -L./ libcapab.a
nv166@nv166-Lenovo-Yoga-500-14IBD:~/Desktop/capcodev6$ ./make_hpfs /temp/fuse
making hospital file system system
./make_hpfs: making the superuser directory
su: made superuser directory
./make_hpfs: making the public directory
public: made public directory
./make_hpfs: making the private directory
private: made the private directory
./make_hpfs: making the EHR directory
EHR: made EHR directory
making directories for special groups
Group directory: /temp/fuse/rbcgroups/
```

Figure 7.4: Initialise the filesystem

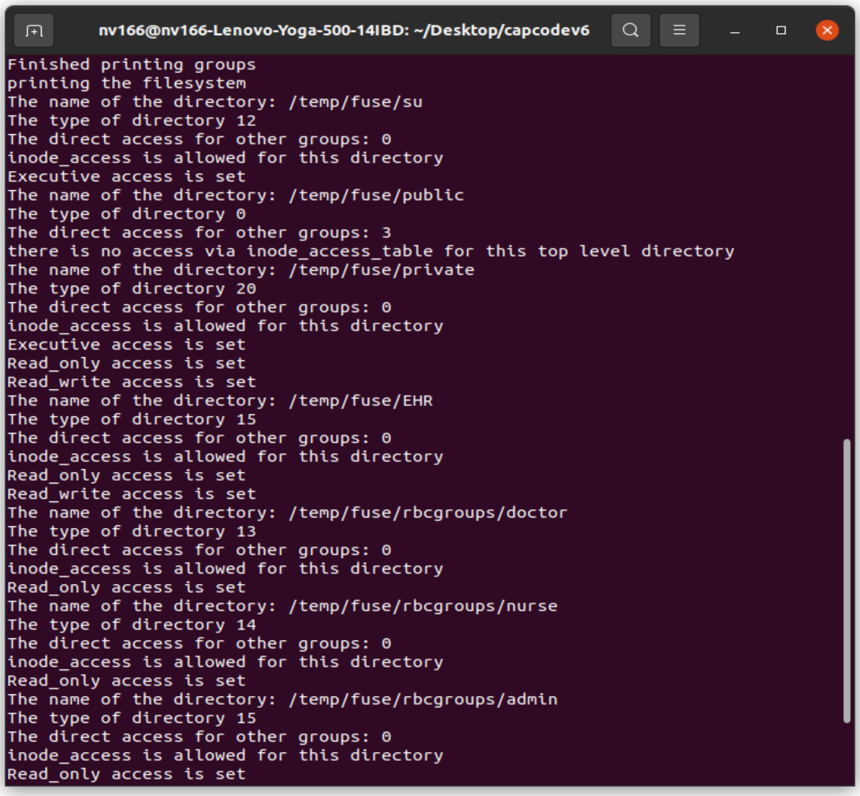
However, the specialised group (rbcgroups) directory can only be accessed by a particular role. Therefore, they have to be specifically created and assigned access rights based on the role of users.



```
making directories for special groups
Group directory: /temp/fuse/rbcgroups/
Please give the number of specialised groups
3
3

 Is this correct: Type y for YES and n for NO: press enter when finished
y
Please type the name of the group: Press enter when finished
doctor
doctor

 Is this correct: Type y for YES and n for NO: press enter when finished
y
Total name: /temp/fuse/rbcgroups/doctor
Please give group number: value of CAP_DOCTOR, CAP_NURSE, etc
13
Please type the name of the group: Press enter when finished
nurse
nurse

 Is this correct: Type y for YES and n for NO: press enter when finished
y
Total name: /temp/fuse/rbcgroups/nurse
Please give group number: value of CAP_DOCTOR, CAP_NURSE, etc
14
Please type the name of the group: Press enter when finished
admin
admin

 Is this correct: Type y for YES and n for NO: press enter when finished
y
Total name: /temp/fuse/rbcgroups/admin
Please give group number: value of CAP_DOCTOR, CAP_NURSE, etc
15
```

Figure 7.5: Creating the rbcgroups directory

In figure 7.5, three rbcgroups directory including doctor, nurse, and admin were created. Each of them was assigned the capability to provide an access right based on the user role. These capability values were declared in the capability header file (i.e., `CAP_DOCTOR = 13, CAP_NURSE = 14, CAP_ADMIN = 15`).



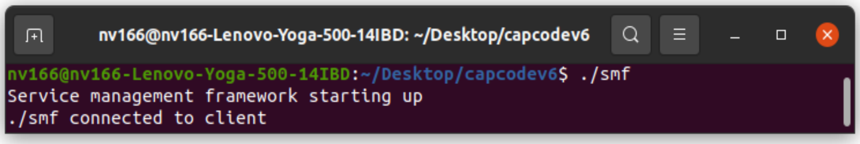
```
Finished printing groups
printing the filesystem
The name of the directory: /temp/fuse/su
The type of directory 12
The direct access for other groups: 0
inode_access is allowed for this directory
Executive access is set
The name of the directory: /temp/fuse/public
The type of directory 0
The direct access for other groups: 3
there is no access via inode_access_table for this top level directory
The name of the directory: /temp/fuse/private
The type of directory 20
The direct access for other groups: 0
inode_access is allowed for this directory
Executive access is set
Read_only access is set
Read_write access is set
The name of the directory: /temp/fuse/EHR
The type of directory 15
The direct access for other groups: 0
inode_access is allowed for this directory
Read_only access is set
Read_write access is set
The name of the directory: /temp/fuse/rbcgroups/doctor
The type of directory 13
The direct access for other groups: 0
inode_access is allowed for this directory
Read_only access is set
The name of the directory: /temp/fuse/rbcgroups/nurse
The type of directory 14
The direct access for other groups: 0
inode_access is allowed for this directory
Read_only access is set
The name of the directory: /temp/fuse/rbcgroups/admin
The type of directory 15
The direct access for other groups: 0
inode_access is allowed for this directory
Read_only access is set
```

Figure 7.6: Displaying filesystem directories

After creating the rbcgroups directory, the system displays the list of directories that were created in the filesystem along with the type of directory, and the access right for other groups. The type of directory is defined by the capability (i.e., The type of directory 20 = `CAP_PRIVATE`).

7.4.2 Creating specific users

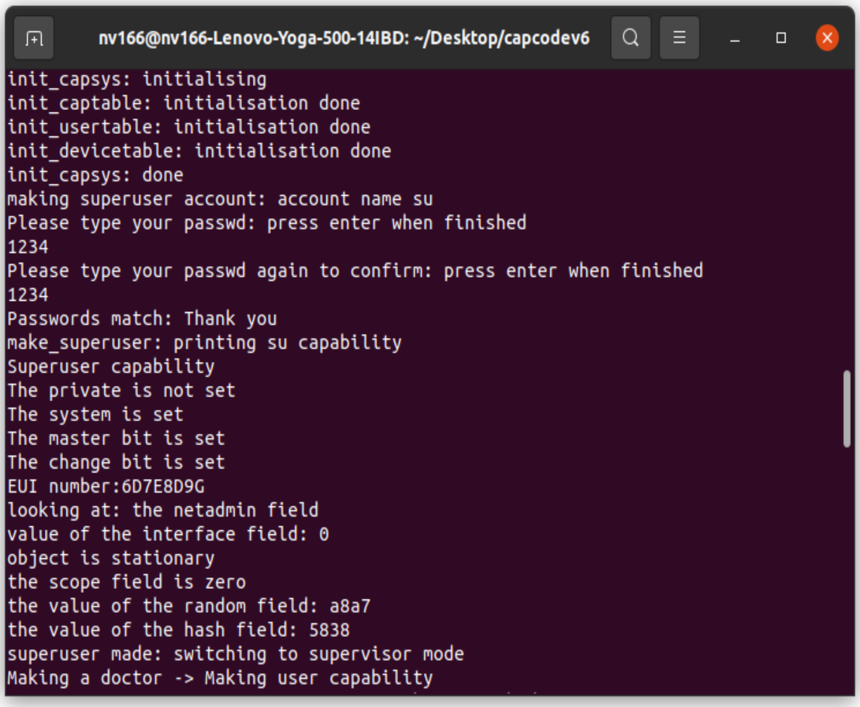
Users in the system are then needed to be created to test access to different directories. Before creating the user, SMF was called to register users (Figure 7.7).



```
nv166@nv166-Lenovo-Yoga-500-14IBD:~/Desktop/capcodev6$ ./smf
Service management framework starting up
./smf connected to client
```

Figure 7.7: Calling SMF

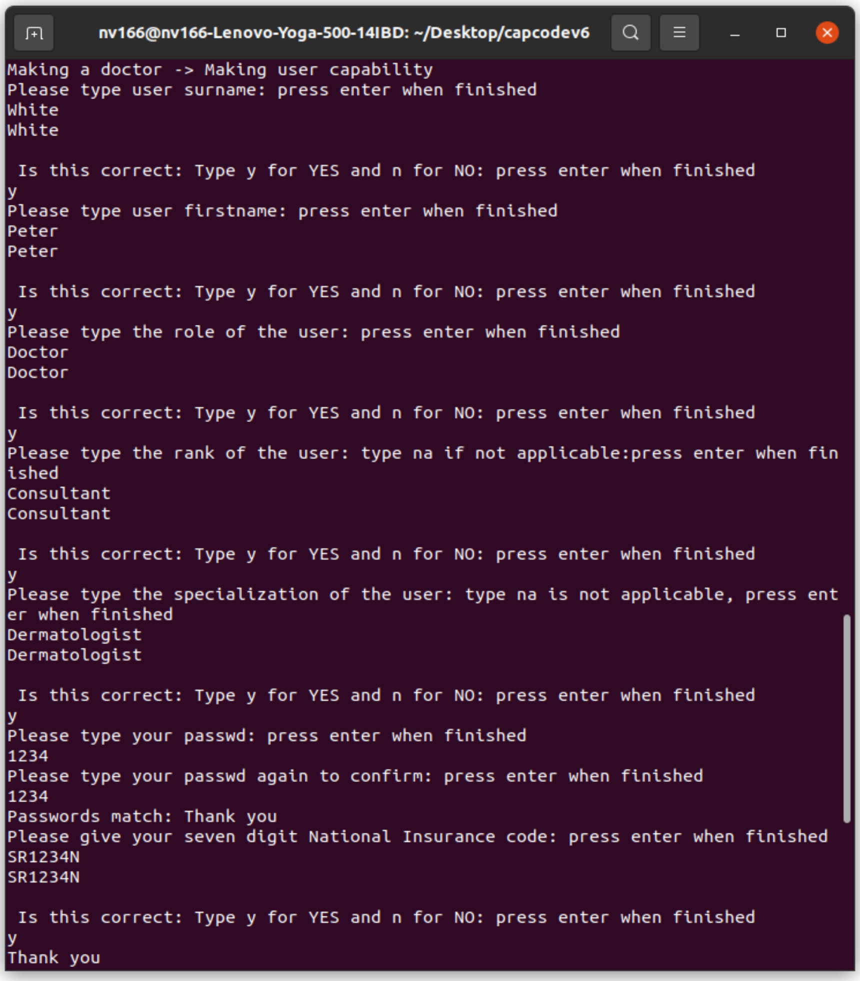
The only person who can create users in the system is the superuser. To create users, the system was switched into the superuser mode. In figure 7.8, the system displayed the superuser capability and began the process of creating users.



```
init_capsys: initialising
init_captable: initialisation done
init_usertable: initialisation done
init_devicetable: initialisation done
init_capsys: done
making superuser account: account name su
Please type your passwd: press enter when finished
1234
Please type your passwd again to confirm: press enter when finished
1234
Passwords match: Thank you
make_superuser: printing su capability
Superuser capability
The private is not set
The system is set
The master bit is set
The change bit is set
EUI number:6D7E8D9G
looking at: the netadmin field
value of the interface field: 0
object is stationary
the scope field is zero
the value of the random field: a8a7
the value of the hash field: 5838
superuser made: switching to supervisor mode
Making a doctor -> Making user capability
```

Figure 7.8: Switching to the superuser mode

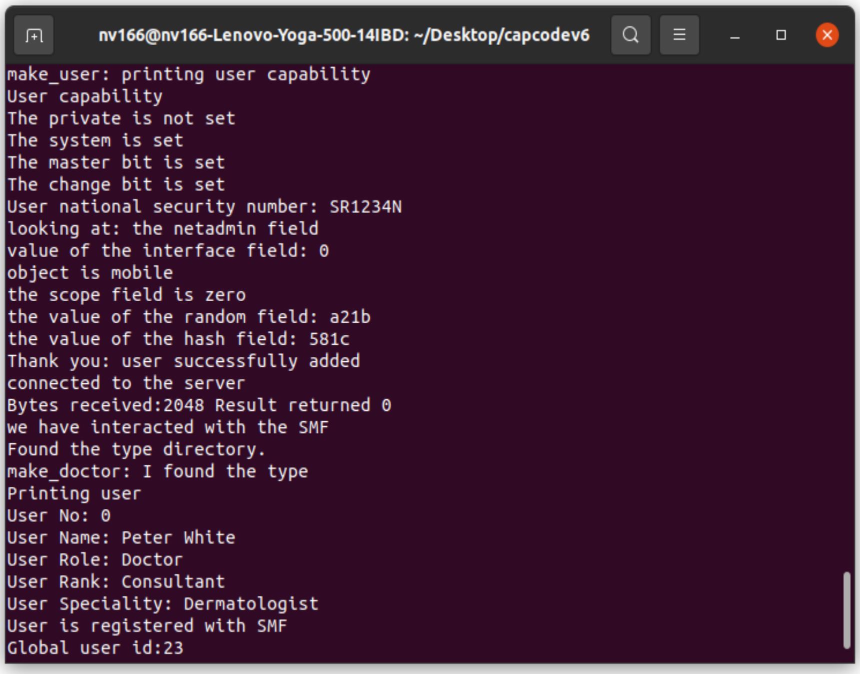
The users and their capabilities were then created. It is necessary to assign a role to each user since it is used as an access right to the filesystem directory. Figure 7.9 shows the process of creating a doctor and its capability.



```
Making a doctor -> Making user capability
Please type user surname: press enter when finished
White
White

 Is this correct: Type y for YES and n for NO: press enter when finished
y
Please type user firstname: press enter when finished
Peter
Peter

 Is this correct: Type y for YES and n for NO: press enter when finished
y
Please type the role of the user: press enter when finished
Doctor
Doctor

 Is this correct: Type y for YES and n for NO: press enter when finished
y
Please type the rank of the user: type na if not applicable:press enter when fin
ished
Consultant
Consultant

 Is this correct: Type y for YES and n for NO: press enter when finished
y
Please type the specialization of the user: type na is not applicable, press ent
er when finished
Dermatologist
Dermatologist

 Is this correct: Type y for YES and n for NO: press enter when finished
y
Please type your passwd: press enter when finished
1234
Please type your passwd again to confirm: press enter when finished
1234
Passwords match: Thank you
Please give your seven digit National Insurance code: press enter when finished
SR1234N
SR1234N

 Is this correct: Type y for YES and n for NO: press enter when finished
y
Thank you
```

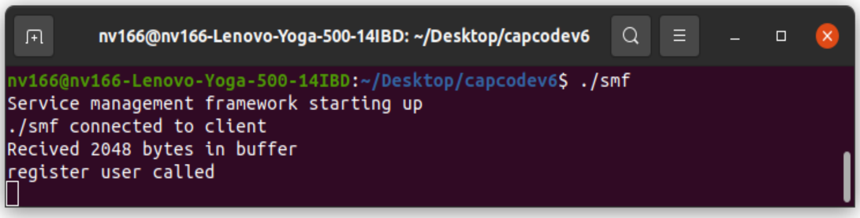
Figure 7.9: Creating a doctor capability

Figure 7.10 below presents the structure of doctor capability. The user was also registered to SMF (as seen in Figure 7.11) after creating.



```
make_user: printing user capability
User capability
The private is not set
The system is set
The master bit is set
The change bit is set
User national security number: SR1234N
looking at: the netadmin field
value of the interface field: 0
object is mobile
the scope field is zero
the value of the random field: a21b
the value of the hash field: 581c
Thank you: user successfully added
connected to the server
Bytes received:2048 Result returned 0
we have interacted with the SMF
Found the type directory.
make_doctor: I found the type
Printing user
User No: 0
User Name: Peter White
User Role: Doctor
User Rank: Consultant
User Speciality: Dermatologist
User is registered with SMF
Global user id:23
```

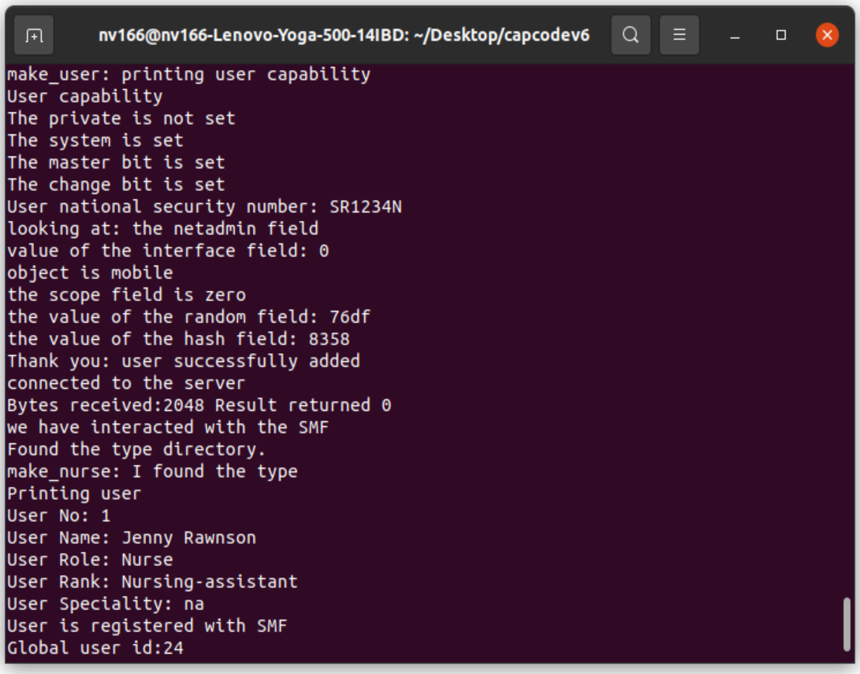
Figure 7.10: The doctor capability structure



```
nv166@nv166-Lenovo-Yoga-500-14IBD:~/Desktop/capcodev6$ ./smf
Service management framework starting up
./smf connected to client
Recived 2048 bytes in buffer
register user called
```

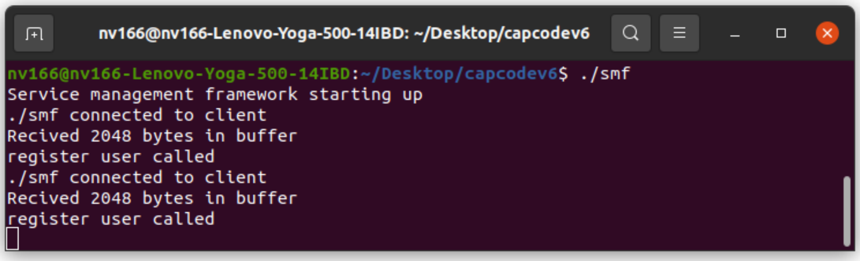
Figure 7.11: Register the doctor to SMF

Figure 7.12 and figure 7.13 show the nurse capability structure and then registered the nurse to SMF.



```
make_user: printing user capability
User capability
The private is not set
The system is set
The master bit is set
The change bit is set
User national security number: SR1234N
looking at: the netadmin field
value of the interface field: 0
object is mobile
the scope field is zero
the value of the random field: 76df
the value of the hash field: 8358
Thank you: user successfully added
connected to the server
Bytes received:2048 Result returned 0
we have interacted with the SMF
Found the type directory.
make_nurse: I found the type
Printing user
User No: 1
User Name: Jenny Rawnson
User Role: Nurse
User Rank: Nursing-assistant
User Speciality: na
User is registered with SMF
Global user id:24
```

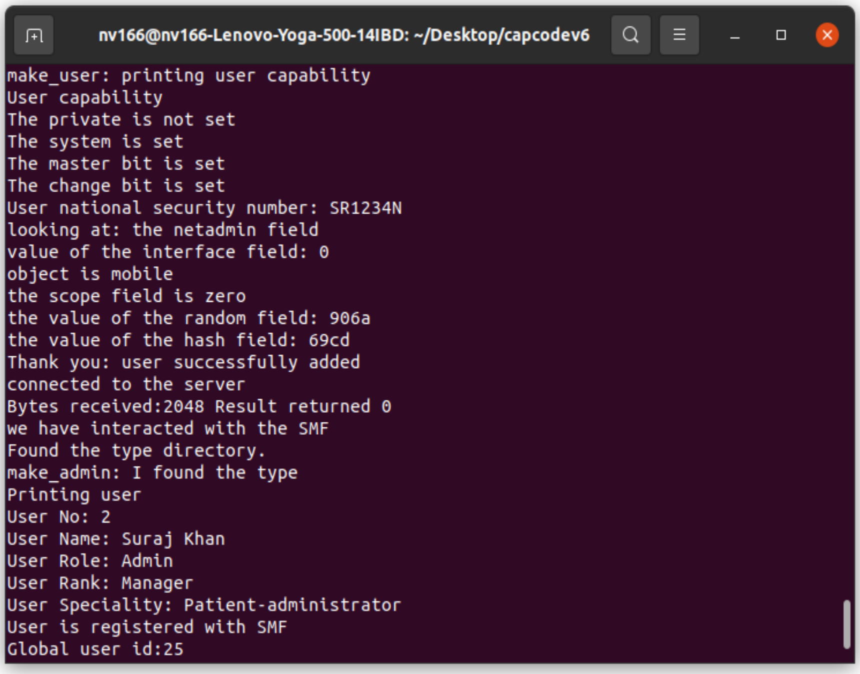
Figure 7.12: The nurse capability structure



```
nv166@nv166-Lenovo-Yoga-500-14IBD:~/Desktop/capcodev6$ ./smf
Service management framework starting up
./smf connected to client
Recived 2048 bytes in buffer
register user called
./smf connected to client
Recived 2048 bytes in buffer
register user called
```

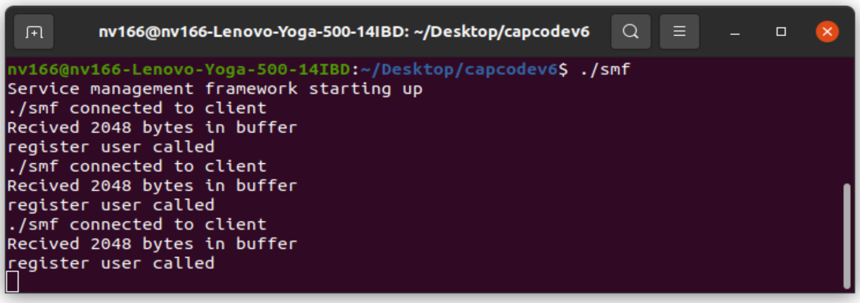
Figure 7.13: Register the nurse to SMF

Figure 7.14 and figure 7.15 show the admin capability structure and then registered the admin to SMF respectively.



```
make_user: printing user capability
User capability
The private is not set
The system is set
The master bit is set
The change bit is set
User national security number: SR1234N
looking at: the netadmin field
value of the interface field: 0
object is mobile
the scope field is zero
the value of the random field: 906a
the value of the hash field: 69cd
Thank you: user successfully added
connected to the server
Bytes received:2048 Result returned 0
we have interacted with the SMF
Found the type directory.
make_admin: I found the type
Printing user
User No: 2
User Name: Suraj Khan
User Role: Admin
User Rank: Manager
User Speciality: Patient-administrator
User is registered with SMF
Global user id:25
```

Figure 7.14: The admin capability structure



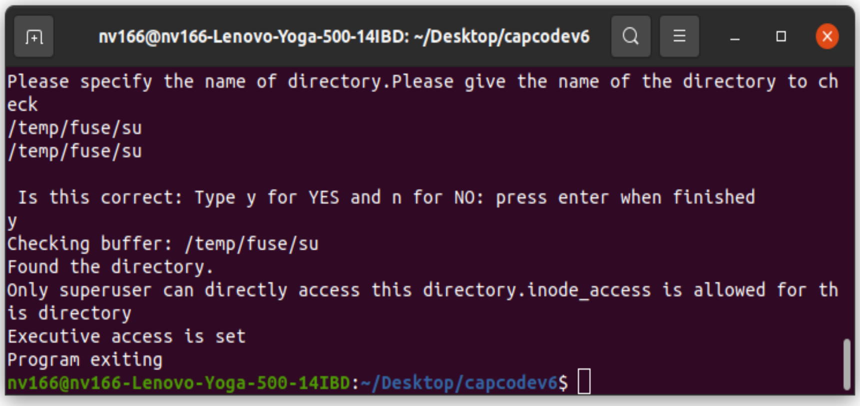
```
nv166@nv166-Lenovo-Yoga-500-14IBD:~/Desktop/capcodev6$ ./smf
Service management framework starting up
./smf connected to client
Recived 2048 bytes in buffer
register user called
./smf connected to client
Recived 2048 bytes in buffer
register user called
./smf connected to client
Recived 2048 bytes in buffer
register user called
```

Figure 7.15: Register the admin to SMF

7.4.3 Testing accesses to the filesystem directory

To achieve authentication and authorisation mechanisms, the filesystem directory was tested to verify that only authorised users can access the designated directories.

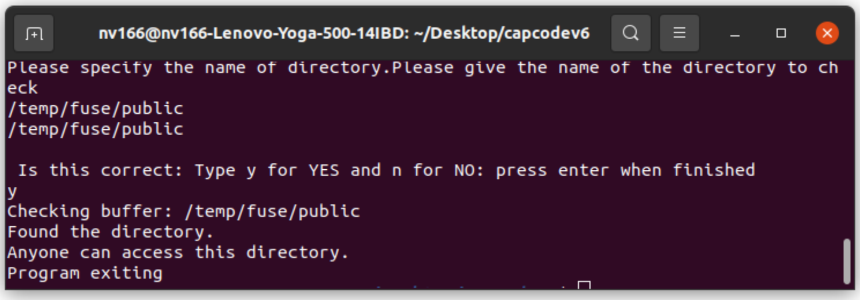
The first directory that is presented below is the superuser directory. The only person who can access this directory is the superuser. However, the `inode_access` allows other people to access the directory but only under the executive mode as seen in Figure 7.16.



```
Please specify the name of directory.Please give the name of the directory to ch
eck
/temp/fuse/su
/temp/fuse/su

 Is this correct: Type y for YES and n for NO: press enter when finished
y
Checking buffer: /temp/fuse/su
Found the directory.
Only superuser can directly access this directory.inode_access is allowed for th
is directory
Executive access is set
Program exiting
nv166@nv166-Lenovo-Yoga-500-14IBD:~/Desktop/capcodev6$
```

Figure 7.16: Access to the superuser directory

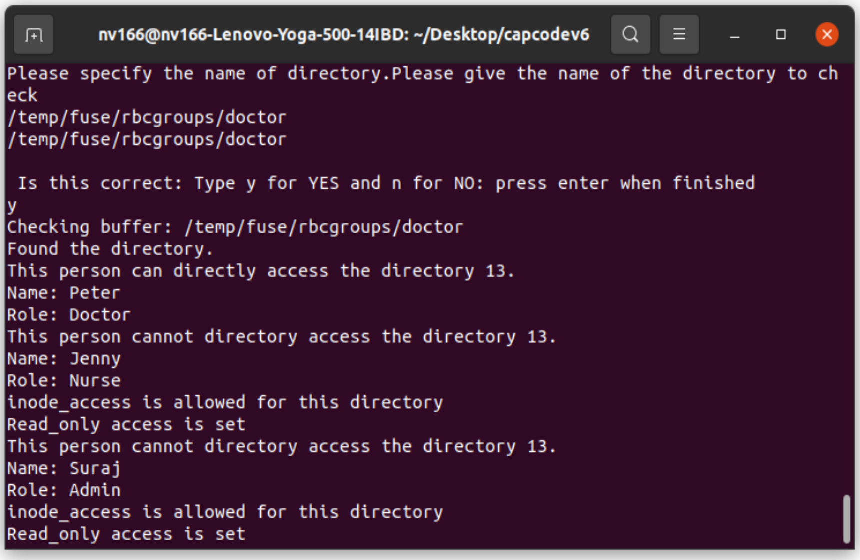
Since the public directory applies the All Access (ALACCESS) right. Therefore, every user in the system can access this directory. Figure 7.17 below shows access to the public directory.



```
Please specify the name of directory.Please give the name of the directory to ch
eck
/temp/fuse/public
/temp/fuse/public

 Is this correct: Type y for YES and n for NO: press enter when finished
y
Checking buffer: /temp/fuse/public
Found the directory.
Anyone can access this directory.
Program exiting
```

Figure 7.17: Access to the public directory

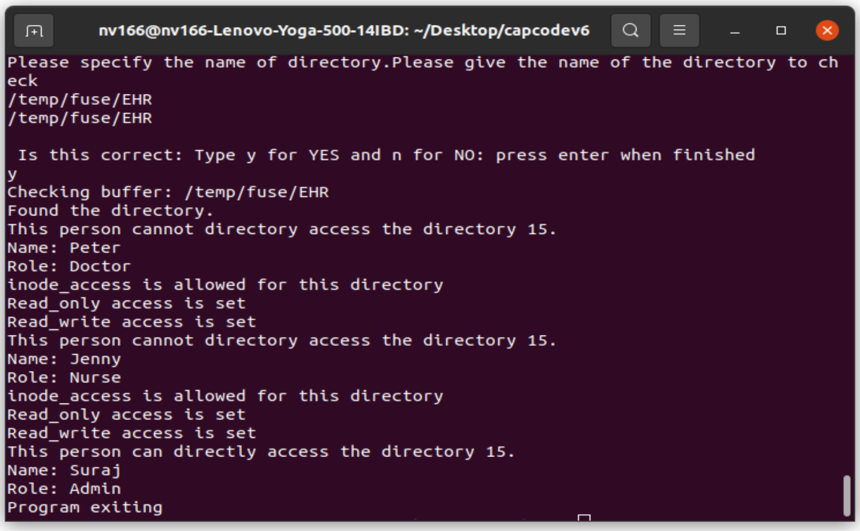
To access the specialised directory, the users need to hold a capability that allows access to a certain directory based on their role. Figure 7.18 shows an access to the rbcgroups/doctor directory. This directory only allows the user who holds the doctor capability (CAP_DOCTOR - 13) to access. Therefore, other users outside the group can also access this directory with a Read-only restriction.



```
Please specify the name of directory.Please give the name of the directory to ch
eck
/temp/fuse/rbcgroups/doctor
/temp/fuse/rbcgroups/doctor

 Is this correct: Type y for YES and n for NO: press enter when finished
y
Checking buffer: /temp/fuse/rbcgroups/doctor
Found the directory.
This person can directly access the directory 13.
Name: Peter
Role: Doctor
This person cannot directory access the directory 13.
Name: Jenny
Role: Nurse
inode_access is allowed for this directory
Read_only access is set
This person cannot directory access the directory 13.
Name: Suraj
Role: Admin
inode_access is allowed for this directory
Read_only access is set
```

Figure 7.18: Access to the rbcgroups/doctor directory

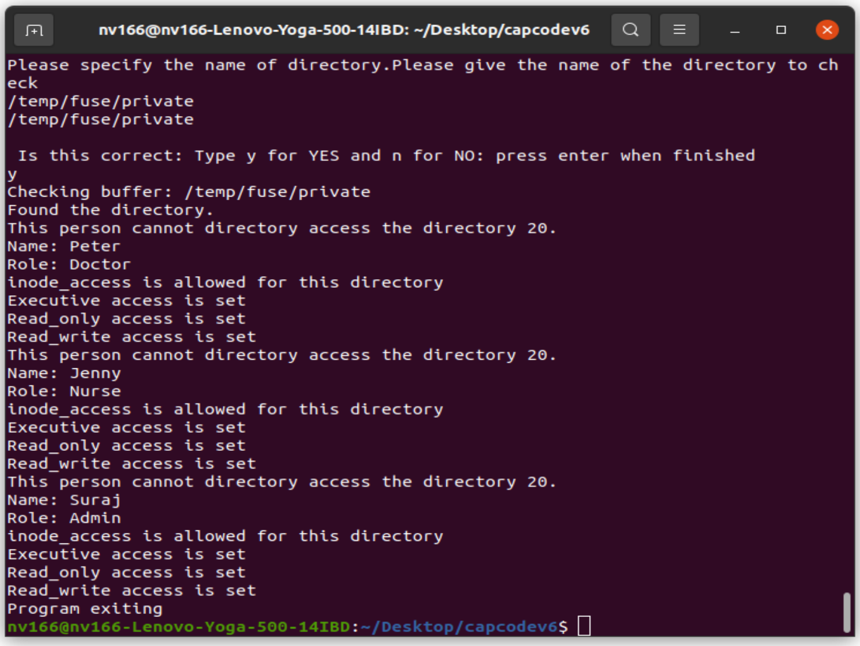
Generally, EHRs in the hospital filesystem are created by the hospital administrators. In this system, therefore, the administrators are the person who have a master capability of EHR and be able to access the EHR directory. Doctors and nurses who are outside the Admin group can also access the EHR directory with Read-only and Read/Write restrictions. Figure 7.19 below shows access to the EHR directory.



```
Please specify the name of directory.Please give the name of the directory to ch
eck
/temp/fuse/EHR
/temp/fuse/EHR

 Is this correct: Type y for YES and n for NO: press enter when finished
y
Checking buffer: /temp/fuse/EHR
Found the directory.
This person cannot directory access the directory 15.
Name: Peter
Role: Doctor
inode_access is allowed for this directory
Read_only access is set
Read_write access is set
This person cannot directory access the directory 15.
Name: Jenny
Role: Nurse
inode_access is allowed for this directory
Read_only access is set
Read_write access is set
This person can directly access the directory 15.
Name: Suraj
Role: Admin
Program exiting
```

Figure 7.19: Access to the EHR directory

Figure 7.20 shows the access rights to the Private directory. Since the private directory only allows a specific user to access it. Therefore, none of the user are able to access it unless their capability is set to provide permission.



```
Please specify the name of directory.Please give the name of the directory to ch
eck
/temp/fuse/private
/temp/fuse/private

 Is this correct: Type y for YES and n for NO: press enter when finished
y
Checking buffer: /temp/fuse/private
Found the directory.
This person cannot directory access the directory 20.
Name: Peter
Role: Doctor
inode_access is allowed for this directory
Executive access is set
Read_only access is set
Read_write access is set
This person cannot directory access the directory 20.
Name: Jenny
Role: Nurse
inode_access is allowed for this directory
Executive access is set
Read_only access is set
Read_write access is set
This person cannot directory access the directory 20.
Name: Suraj
Role: Admin
inode_access is allowed for this directory
Executive access is set
Read_only access is set
Read_write access is set
Program exiting
nv166@nv166-Lenovo-Yoga-500-14IBD:~/Desktop/capcodev6$
```

Figure 7.20: Access to the private directory

7.4.4 Evaluation

Initially, the research began with conducting a literature review by identifying, analysing, and comparing different existing studies about security frameworks. Several security frameworks were developed, however, none of the existing framework provides a full set of security requirements (Confidentiality, Integrity, Availability, Non-repudiation, Authentication, Authorisation, Accountability, Auditability, and Reliability), and all healthcare environment components (User, Devices, Data, IT infrastructure, Physical space access) at the same time.

To be able to test the functionalities of the proposed framework properly, a prototype combining various developed security mechanisms: (1) Capabilities; (2) SRPC; (3) SMF/SManL; and (4) mHealth secure storage, is developed and stands for the purpose of testing. Despite the fact that the prototype may contain fewer numbers of layers, nevertheless, it still meets the security requirements for healthcare environments as discussed in Section 5.2 and Section 5.3.4.

Table 7.3 shows how the implemented framework provides the full set of security requirements for healthcare environments.

Security requirement	**Capabilities**	**SRPC**	**SMF/SmanL**	**mHealth secure storage**
Confidentiality	*			*
Integrity	*			*
Availability			*	*
Non-repudiation	*			
Authentication	*			*
Authorisation	*			*
Accountability	*		*	
Auditability			*	
Reliability	*	*	*	*

Table 7.3: How the implemented framework achieves security requirements

Table 7.4 shows how the implemented framework provides practical security for users, devices, digital data, IT infrastructure, and access to physical space.

Protection	**Capabilities**	**SRPC**	**SMF/SmanL**	**mHealth secure storage**
User of the system	*			
Device and home access	*		*	
Digital data	*	*	*	*
IT infrastructure	*	*	*	
Access to physical space	*			

Table 7.4: How the implements framework achieves security for healthcare environment components

7.5 Chapter Summary

In this chapter, implementation, testing, and evaluation of the proposed information security framework for mHealth systems were described. To be able to test the implemented prototype, the mHealth secure storage system was developed.

The capability system was first developed as a basic system and only provided generic capabilities. Therefore, these basic capability structures were enhanced to be specific to the mHealth system. The test was conducted by verifying the access rights of each user to the filesystem directories based on their roles . The results confirmed that capabilities provided security requirements of authentication and authorisation by allowing authorised users to access the file directories and preventing unauthorised users from accessing them.

Furthermore, the evaluation presented that the implemented prototype that consists of 4 layers: Capability system, SManL/SMF, SRPC, and mHealth secure storage, developed in this research provided essential security requirements required by mHealth systems as well as a practical protection for users, devices, digital data, IT infrastructure, and physical areas that are main assets of mHealth systems.

Chapter 8

Conclusion and Future Work

The final chapter provides an overview of the research. It presents summaries of the novel contributions and their confirmations, and outlines the limitations of the study as well as some guidance on the future research.

8.1 Summary of Work Done

Initially, the research began with conducting a literature review by identifying, analysing, and comparing different sources of existing studies. In Chapter 3, an mHealth system architecture was identified to discover components of mHealth systems and understand how healthcare data are processed in the system. Assets in mHealth systems are healthcare data, mobile devices, IT infrastructure, and data storage. Since sensitive healthcare data is stored and processed in mHealth systems, therefore, mHealth systems are vulnerable to various threats. As a result, it is necessary to develop an information security framework to secure mHealth systems from multiple security challenges. Moreover, various information security models were observed in this chapter to discover essential security requirements for mHealth systems. Confidentiality, Integrity, and Availability (CIA) are three core concepts of security requirements that are required by every system. Due to new threats that emerge in the healthcare environment, the CIA may not be enough to provide a secure end-to-end environment. As a result, it is crucial to apply other security requirements, such as non-repudiation, authentication, and authorisation, into mHealth systems.

In Chapter 4, existing security frameworks [26][27][28][29][30][31][32][33] were investigated comparing similarities and differences. All existing security frameworks highlighted confidentiality, integrity, and availability as common security requirements to form their security frameworks. However, some important security requirements including non-repudiation, accountability, and auditability were still missing from most existing frameworks. Moreover, these existing security frameworks only focused on one or two components of a healthcare environment. Therefore, the results showed that no existing framework caters to all security requirements (confidentiality, integrity, availability, non-repudiation, authentication, authorisation, accountability, auditability, reliability) and protects all healthcare environment components (device, hospital infrastructure, digital data, cloud storage) at the same time.

The identification of assets, threats, and vulnerabilities was identified in Chapter 5. In mHealth systems, assets include mobile device, cloud storage, network connectivity, and data. These assets are exposed to multiple threats that may pose significant harm to them. Most threats which occur attempt to exploit confidentiality, integrity, and availability that are the most common security requirements for mHealth systems. Moreover, a detailed analysis of security requirements for mHealth systems was examined. The result showed that five mHealth subsystems, including (1) Human user, (2) Device, (3) Digital data, (4) Hospital infrastructure, and (5) Physical sites and locations, need to be protected. The taxonomy of an mHealth system was then constructed in an effort to develop a practical information security framework for mHealth systems. It was formed from the mHealth system scenario in Figure 5.1. The structure of mHealth system taxonomy was categorised into four components, including system architecture, healthcare data, stakeholders, and security requirements. Furthermore, various key security mechanisms, including (1) Encryption as a service, (2) Capabilities, (3) Storage management system, (4) Digital filter, (5) Secure transport layer, (6) Blockchain, (7) Secure transactional layer, and (8) Service management layer, (as shown in Section 5.6) were proposed and applied to develop an information security framework that completely specified the required security requirements as presented a detailed analysis of security requirements for mHealth systems (Section 5.4) and the mHealth taxonomy (Section 5.5).

There was a need to move from the framework to a prototype implementation. The implemented prototype was developed in Chapter 6 to provide an experimental system that contains the necessary functionalities to verify that the proposed information security framework for mHealth systems provides the essential set of security requirements as well as protects major components of mHealth systems. Initially, the proposed information security framework consists of nine layers as shown in Section 5.7. Since the proposed framework contains many different security mechanisms, developing a practical prototype to test this proposed framework will require significant time and effort, and may exceed a PhD timeframe. Consequently, this implemented prototype was developed. It consists of four layers developed by the author, namely Capability System, Secure Transactional Layer, Service Management Layer, and mHealth Secure Storage Application.

The Capability System, the Secure Transactional Layer (SRPC), the Service Management Layer (SManL/SMF), and an mHealth secure storage application, which are the main contributions of the research, were developed as parts of the prototype. In Chapter 7, the prototype test was designed to conduct both functional (i.e., unit test and system test) and non-functional (i.e., security test). The hospital filesystem was developed to test the access rights of users with different capabilities. Capabilities can be used flexibly to provide authentication, authorisation, accounting, and control for any environment. SRPC was used to encode data transmitted in the system. It also supported the type array. This is an improvement of a traditional RPC as the type array is not passed in the normal RPC call. SRPC provides a secure transactional environment to ensure that clients and servers can interact securely. The SMF/SManL provided some basic functions to register users, devices, services, and add servers to services. It allows clients to find services and provides communication endpoints and capabilities which allows a reliable session to be developed. Hence, it improved efficiency, security, as well as management of services.

Nevertheless, the prototype may contain fewer numbers of layer compared to the proposed information security framework. Therefore, it was tested and verified that it is efficient and secure, and has also delivered a complete set of security attributes.

8.2 Results of the Research

The result showed that the information security framework for mHealth systems achieved the essential security requirements including Confidentiality, Integrity, Availability, Non-repudiation, Authentication, Authorisation, Accountability, Auditability, and Reliability. Moreover, it also provided a protection to all components of the hospital environment including users of the system, device and home access, IT infrastructure, access to physical space, and most importantly, digital data.

8.3 Contributions to the Research

The findings from this research presented the set of essential requirements for mHealth systems (Section 5.3). Researchers can apply the set of essential security requirements as a basic structure for any secure system. The contributed security mechanisms, including Service Management Framework, Capability System, Secure Remote Procedure Call, were developed to be reusable. Hence, researchers can integrate any type of system by utilising these security mechanisms.

8.4 Contributions to the Field

The information security framework for mHealth systems developed in this research presented opportunities for more insightful and impactful research. Moreover, the findings from this research may be beneficial to various personnel in the mHealth field including mHealth users, healthcare professionals, and policymakers. The contributions to the field, therefore, present in this section.

8.4.1 mHealth users

Since the number of participants involved in mHealth has increased and mHealth may provide both benefits and risks. Several security features require input from end-users. Therefore, it is necessary that end users should have good knowledge of how to use mHealth systems securely. This research can improve awareness and understanding of possible security threats that impose on users' healthcare data. The information security framework for mHealth systems provides an assurance to mHealth users that their healthcare data will be stored securely and protected from end to end.

8.4.2 Healthcare professionals

The research provided a new concept of secure healthcare environments. The information security framework for mHealth systems is adaptable and can be applied to any hospital, surgery, and clinical environment. Capabilities also offer a great benefit in terms of providing role-based access control to healthcare professionals. Capabilities do not only provide access controls regarding who are able to access patient healthcare data, but also who have the right to access certain hospital locations or be able to operate certain devices.

8.4.3 Policymakers

Cyber threats are advancing and increasing at a fast pace. Healthcare data are considered as sensitive data and are likely to be targeted from malicious activities. Therefore, the security of healthcare data should be elevated among national policymaking priorities. The information security framework for mHealth systems provided a conventional framework to explicate how the developed prototype in this research can be implemented into a larger scale of healthcare systems. The research can impact policymakers on raising their concerns on protecting national healthcare data and developing a practical and secure national health system.

8.5 Limitation of the Study

The proposed information security framework for mHealth systems in this research may successfully secure mHealth environments. However, there is still a limitation in the research contributions described in this thesis.

Initially, the proposed information security framework for mHealth systems consisted of many different mechanisms and each of them was complex and requires significant of time to implement. Therefore, developing the proposed information security framework was hardly possible regarding the limitation of Ph.D. timeframe. Consequently, there was a need to develop a prototype that contains less layers but still provided a complete set of security requirements for the purpose of testing.

8.6 Future Work

The information security framework for mHealth systems developed in this research provides benefits to future healthcare environments in terms of delivering security and efficiency. While it opens new insights for research in the security of healthcare environment areas, therefore, it can be extended to many more directions for future research.

Although, the result showed that the implemented prototype improved the security of mHealth systems. However, the test was conducted on the mHealth secure storage system that was developed in this research to provide some basic functionalities of the hospital filesystem. Therefore, this implemented prototype needs to be tested and integrated into small healthcare institutions (e.g., GP surgery, clinic, small hospital), and then large healthcare institutions (e.g., a large NHS hospital).

Moreover, new technologies such as Artificial Intelligence (AI) and Machine Learning (ML) can be introduced as new security mechanisms for the proposed information security framework. Not only the AI/ML can enhance diagnostics, patient care, and clinical decision support across the medical service. It can also be used to analyse the data flow within mHealth systems to detect what are "normal" or "abnormal" behaviours of each user, device, and service in the system so that it can protect mHealth systems from cyberattacks such as ransomware.

8.7 Final Remark

The research aimed to develop an information security framework that encompassed home environments, hospital environments, and Cloud environments to provide end-to-end security for mHealth systems. The result demonstrated that the information security framework for mHealth systems developed in this research achieved the goal by delivering a complete set of security attributes required by mHealth systems.

Bibliography

[1] WHO Global Observatory for eHealth.(2011).mHealth: new horizons for health through mobile technologies: second global survey on eHealth.World Health Organization.[online] Available from: https://apps.who.int/iris/handle/10665/44607 [Accessed: 10 November 2021]

[2] European Commission (2014) GREEN PAPER on mobile Health ("mHealth"). [online] Available from: https://digital-strategy.ec.europa.eu/en/library/green-paper-mobile-health-mhealth [Accessed: 10 November 2021]

[3] Germanakos, P., Mourlas, C., Samaras, G."A Mobile Agent Approach for Ubiquitous and Personalized eHealth Information Systems"Proceedings of the Workshop on 'Personalization for e-Health' of the 10th International Conference on User Modeling (UM'05). Edinburgh, July 29, 2005, pp. 67–70.

[4] European Commission (2020) mHealth solutions relevant for coronavirus pandemic [online] Available from: https://ec.europa.eu/digital-single-market/en/news/mhealth-solutions-relevant-coronavirus-pandemic [Accessed: 30 May 2020]

[5] European Commission (2014) Healthcare in your pocket: unlocking the potential of mHealth. [online] Available from: https://ec.europa.eu/commission/presscorner/detail/en/IP$_14_394[Accessed : 10November2021]$

[6] NHS Digital (N/A) Transparency notice: how we use your personal data [online] Available from: https://digital.nhs.uk/about-nhs-digital/ our-work/keeping-patient-data-safe/gdpr/gdpr-register [Accessed: 3 August 2021]

[7] Strugar, M. (2021) How many people own a smartphone in the UK? [online] Available from: www.cybercrew.uk/blog/how-many-people-own-a- smartphone-in-the-uk/[Accessed: 5 November 2021]

[8] Franklin, R. (2021) 11 surprising mobile health statistics [online] Available from: https://mobius.md/2021/10/25/11-mobile-health-statistics/ [Accessed: 5 November 2021]

[9] Arora, S., Yttri, J., and Nilsen, W. Privacy and Security in Mobile Health (mHealth) Research.Alcohol Research: Current Reviews. 2014;36(1):143-151.

[10] Himss (2014) Health Informatics Defined [online] Available from: http://www.himss.org/health-informatics-defined [Accessed: 22 January 2018]

[11] Bath, P.A. (2008) Health Informatics: Current Issues and Challenges. J. Information Science. 34. 501-518. 10.1177/0165551508092267.

[12] Guide to Master's Programs in IT (N/A) What is health informatics? [online] Available from: http://www.mastersinit.org/faq/what-is-health-informatics [Accessed: 22 January 2018]

[13] Eysenbach, G. What is e-health?.Journal of Medical Internet Research. 2001;3(2):e20. doi:10.2196/jmir.3.2.e20.

[14] Kotz, D., Gunter, C.A., Kumar, S., Weiner, J.P. Privacy and Security in Mobile Health: A Research Agenda. Computer (Long Beach Calif). 2016 June; 49(6): 22-30. Doi: 10.1109/MC.2016.185.

[15] Cruickshank, J., Packman, C., Paxman, J. (2012) Personal Health Records – Putting patients in control. [online] Avail-

able from: www.2020health.org/dms/2020health/downloads/.../2020 PHRreport$_ONLINE.pdf [Accessed : 22 January 2018]$

[16] HIPAA (2009) [online] The definition of Electronic Health Record. Available from: http://www.hipaa.com/the-definition-of-electronic-health-record/ [Accessed: 22 January 2018]

[17] Avancha, S., Baxi, A. and Kotz, D. (2012) Privacy in mobile technology for personal healthcare. ACM Comput. Surv. 45, 1, Article 2 (November 2012), 54 pages.

[18] Vodafone Global Enterprise (2013) Evaluating mHealth Barriers: Privacy and Regulation. [online] Available from: http://mhealthregulatorycoalition.org/wp-content/uploads/2013/01/ VodafoneGlobalEnterprise-mHealth-Insights-Guide-Evaluating-mHealth-Adoption-Privacy-and-Regulation.pdf [Accessed: 10 September 2018]

[19] Adesina, A.O., Agbele, K.K., Februarie, R., Abidoye, A.P., Nyongesa, H.O. (2011) Ensuring the security and privacy of information in mobile health-care communication systems. S Afr J Sci. 2011;107(9/10), Art. 508, 7 pages. Doi:10.4102/sajs.v107i9/10.508

[20] Cambridge University Press (N/A) http://dictionary.cambridge.org/ dictionary/english/technology [Accessed: 21 September 2017]

[21] Science Buddies (N/A) https://www.sciencebuddies.org/science-fair-projects/engineering-design-process/engineering-design-compare-scientific-method [Accessed: 22 September 2017]

[22] Ridley, D. (2012). The Literature Review: A Step-by-Step Guide for Students. 2nd Edition. United Kingdom: Sage Publications.

[23] Hart, C. (1998). Doing a literature review: Releasing the social science research imagination. London: Sage.

[24] Blaxter, L., Hughes, C., Tight, M. (2006). How to Research. 3rd Edition. Buckingham: Open University Press.

[25] Rudestam, K. Newton, R. (1992). Surviving your dissertation. London: Sage Publications.

[26] Firesmith, D. “Specifying Reusable Security Requirements” Journal of Object Technology. Vol.3, no.1, pp.61-75, 2004.

[27] Takabi, H., Joshi, J.B.D., Ahn, G.J. “SecureCloud: Towards a comprehensive security framework for cloud computing environments”. International Computer Software and Applications Conference, 2010, pp.393-398.

[28] Brock, M. Goscinski, A. A. “Toward a Framework for Cloud Security” in Lecture Notes in Computer Science, vol 6082, Springer Berlin Heidelberg, 2010, pp.254-263.

[29] Zissis, D. and Lekkas, D. “Addressing cloud computing security issues”. Future Generations Computer Systems. 28(2012). P.583-592.

[30] Mapp, G., Aiash, M., Ondiege, B., Clarke, M. (2014) Exploring a New Security Framework for Cloud Storage Using Capabilities. In: 2014 IEEE 8th Symposium on Service Oriented System Engineering (SOSE). Oxford: IEEE, P. 484-489

[31] Pirbhulal, S., Samuel, O.W., Wu, W., Sangaiah, K., and Li, G. (2019). A Joint Resource-aware and Medical Data Security Framework for Wearable Healthcare Systems. Future Generation Computer Systems, Vol.95, pp.382-391.

[32] Rathee, G., Sharma, A., Saini, H., Kumar, R., and Iqbal, R. (2020). A hybrid framework for multimedia data processing in IoT-healthcare using blockchain technology. Multimedia Tools and Applications. 79(15-16). 9711-9733.

[33] Yayah, F. (2017) “A Security Framework to Protect Data in Cloud Storage”, PhD Thesis. University of Southampton. Southampton.

[34] Gejibo, S., Mancini, F., Mughal, K.A., Valvik, R.A., Klungsoyr, J. (2012) Secure Data Storage for Java ME-Based Mobile Data Collection Systems. In: 2012 IEEE 14th International Conference on e-Health Networking, Applications and Services (Healthcom 2012). Beijing: IEEE, P.498-501.

[35] Jung, C. (2011) Mobile Data Collection Systems: A review of the current state of the field. [online] Available from: https://humanitarian-nomad.org/wp-content/uploads/2013/03/NOMAD-MDC-Research.pdf (Accessed: June 16th, 2016).

[36] Gejibo, S.H., Mancini, F., Mughal, K.A., Valvik, A.B., Klungsoyr, J. Secure data storage for mobile data collection systems. In Proceeding of the International Conference on Management of Emergent Digital EcoSystem (MEDES'12). New York: ACM, P.131-144.

[37] Gejibo, S.H. (2015) Towards a Secure Framework for mHealth. PhD thesis. University of Bergen.

[38] Gao, W., Emaminejad, S. Nyein, H.Y., Challa, S., Chen, K., Peck, A. Fully integrated wearable sensor arrays for multiplexed in situ perspiration analysis. Nature 2016;529(7584):509-14.

[39] Innovatemedtec (N/A) Medical Sensors and Wearable, what are the applications? [online] Available from: https://innovatemedtec.com/digital-health/sensors-and-wearables [Accessed: 23 January 2018]

[40] Hiremath, S., Yang, G., Mankodiva, K. (2014) Wearable Internet of Things: Concept, architectural components and promises for person-centered healthcare. In: 2014 EAI 4th International Conference on Wireless Mobile Communication and Healthcare (Mobihealth). Athens: IEEE, P.304-307.

[41] Padhy, R.P., Patra, M.R., Satapathy, S.C. (2011) Cloud Computing: Security Issues and Research Challenges. International Journal of Computer Science and Information Technology Security (IJCSITS), Volume 1, P.136-146.

[42] Mell, P., and Grance, T. The NIST definition of cloud computing. Commun ACM 2010;53(6):50.

[43] Cifuentes, Y., Beltran, L., and Ramirez, L. "Analysis of Security Vulnerabilities for Mobile Health Applications" International Journal of Electrical, Computer, Energetic, Electronic and Communication Engineering. Vol, 9, No.9, 2015.

[44] Martin, K. (2012) Everyday Cryptography. United States of America: Oxford University Press Inc.

[45] McCumber, J., "Information Systems Security: A Comprehensive Model," in: Proceeding of the 14th National Computer Security Conference, NIST, Baltimore, MD, 1991.

[46] McCumber, J. (2004) Assessing and Managing Security Risk in IT Systems: A Structured Methodology. United States of America: CRC Press.

[47] Hafiz, M. Johnson, R.E. (2006) Security Patterns and their Classifications Schemes. [online] Available from: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1075.362rep=rep1type=pdf (Accessed: 10 November 2021)

[48] Pender-Bey, G. (N/A) The ParkerianHexad. Master of Science in Information Security, Lewis University

[49] Bosworth, S., Kabay, M.E., and Whyne, E. (2009) Computer Security Handbook. 5th Edition. New Jersey: John Wiley Sons, Inc.

[50] Ypetkova (2012) Basic Information and Network Security Objectives– CIA triad and ParkerianHexad. [online] Available from: http://cyberseecure.com/2012/07/16/basic-information-and-network-security-objectives-cia-triad-and-parkerian-hexad/ (Accessed: 3 July 2016)

[51] Cherdantseva, Y. and Hilton, J. 2013, "A Reference Model of Information Assurance Security". Proceedings of IEEE 2013 8th International Conference on Availability, Reliability and Security (ARES), Regensburg – Germany.

[52] Scarfone, K. Souppaya, M. (2013) Guidelines for Managing the Security of Mobile Devices in the Enterprise. [online] Available from: http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-124r1.pdf [Accessed: 15 January 2017]

[53] Gardner, R.W., Garera, S., Pagano, M.W., Green, M., Rubin, A.D. (2009) Securing Medical Records on Smart Phones. In: 2009 16th ACM Conference on Computer and Communications Security (CCS). Chicago: ACM, pp.31-40.

[54] Schneider, F.B. (N/A) Something You Know, Have, or Are [online] Available from: https://www.cs.cornell.edu/courses/cs513/2005fa/NNLauthPeople.html [Accessed: 18 January 2017]

[55] ICD Security Solutions (2012) Access Control Continued: biometrics and other forms of access authorization [online] Available from: https://www.icdsecurity.com/2014/10/20/access-control-continued-biometrics-and-other-forms-of-access-authorization/ [Accessed: 18 January 2017]

[56] Luxton, D.D., Kayl, R.A., Mishkind, M.C. (2012) mHealth Data Security: The Need for HIPAA Compliant Standardization. In: Telemedicine e-Health, Volume 18, Issue 4, P.284.

[57] Yahya, F., Walters, R.J., Wills, G.B. (2016) Goal-Based Security Components for Cloud Storage Security Framework: A Preliminary Study. In: 2016 International Conference on Cyber Security and Protection of Digital Services (Cyber Security). London: IEEE, P.1-5

[58] IBM (N/A) Remote Procedure Call [online] Available from:

https://www.ibm.com/docs/en/aix/ 7.2?topic=concepts-remote-procedure-call [Accessed: 6 November 2021]

[59] IBM (N/A) RPC Model [online] Available from: https://www.ibm.com/support/knowledgecenter /en/ ssw$_aix_71/com.ibm.aix.progcomc/rpc_mod.htm$ 7$June$2018]

[60] Dcerpc.org (N/A) DCE Remote Procedure Call [online] Available from: https://www.dcerpc.org/ documentation/rpc-porting.pdf [Accessed: 12 June 2018]

[61] Microsoft (N/A) The RPC Name Service Database [online] Available from: https://msdn.microsoft. com/ en-us/library/windows/desktop/aa378865(v=vs.85).aspx [Accessed: 16 June 2018]

[62] Iden, J. Eikebrokk, T.R. (2013) Implementing IT Service Management: A systematic literature review. International Journal of Information Management, 33 (3), 512-523.

[63] Kempter, S. (2018) Availability Management [online] Available from: https://wiki.en.it-processmaps.com/index.php/Availability$_Management[Accessed$: 1$July$2018]

[64] Dennis, J.B., and Horn, E.C.V. (1966). Programming semantics for multiprogrammed computations. Communications of the ACM, 9(3):143-155.

[65] Wilkes, M.V. and Needham R.M. "The Cambridge CAP Computer and its Operating System," in Operating and Programming Series. Elsevier North Holland, 1979.

[66] Felton, D. (2019) "What is TrustZone." [Online]. Available from: https://www.trustonic.com/technical-articles/what-is-trustzone/ [Accessed: 9 November 2021]

[67] Tanenbaum, A.S. Sharp, G.J. (1991) "The amoeba distributed operating system."

[Online]. Available from: www.cs.vu.nl/pub/amoeba/Intro.pdf [Accessed: 9 November 2021]

[68] Fireball Software Distribution (N/A) [online] Available from: http://fsd-amoeba.sourceforge.net/ amoeba.html [Accessed: 9 November 2021]

[69] Sandhu, R., Ferraiola, D., and Kuhn, R. "The NIST model for role-based access control: toward a unified standard" in RBAC Proceedings of the 5th ACM workshop on Role-based access control, pp: 47-63, Berlin, Germany, July 26-28, 2000.

[70] Gupta, M. (2019) Blockchain for Dummies. 3rd IBM Limited Edition. United States of America: John Wiley Sons Inc.

[71] Java T Point (N/A) History of Blockchain [online] Available from: https://www.javatpoint.com/history-of-blockchain: :text=The20blockchain20technology20was20described,not 20be20backdated20or20tampered [Accessed: 9 November 2021]

[72] Iredale, G. (2020) History of Blockchain Technology: A Detailed Guide [online] Available from: https://101blockchains.com/history-of-blockchain-timeline/ [Accessed: 9 November 2021]

[73] Lewis, T.L. Wyatt, J.C. (2014) mHealth and Mobile Medical Apps: A Framework to Assess Risk and Promote Safer Use. Journal of Medical Internet Research. 16(9), e210.

[74] Vacca, J.R. (2012) Computer and Information Security Handbook, 2nd Edition, Elsevier, The United States of America.

[75] International Organization for Standardization (2009) ISO 31000 Risk Management [online] Available from: https://www.iso.org/files/live/sites/isoorg/files/archive/pdf /en/ iso$_3$1000$_for_smes.pdf[Accessed: 15March$2017]

[76] El-Abed, M., Giot, R., Hemery, B., Schwartzmann, J, Rosenberger, C. (2012) Towards the Security Evaluation of Biometric Authentication Systems. IACSIT International Journal of Engineering and Technology. 4(3), pp.315-320.

[77] CSA (2016) The Treacherous 12 Cloud Computing Top Threats in 2016 [online] Available from: https://downloads.cloudsecurityalliance.org/assets/research/top-threats/Treacherous-12$_Cloud - Computing_Top - Threats.pdf[Accessed : 17March2017]$

[78] SANS (2005) Security Vulnerabilities and Wireless LAN Technology [online] Available from: https://uk.sans.org/reading-room/whitepapers/access/security-vulnerabilities-wireless-lan-technology-1629 [Accessed: 20 March 2017]

[79] Akshaya, S., Ganapathy, K. (2016) A Study on the Vulnerabilities of Spoofing Attacks. Journal of Chemical and Pharmaceutical Sciences. 6(8), 29-34.

[80] Shingade, R.M., Mane, A.V. (2014) Detection of Spoofing attackers in wireless network. International Journal of Innovative Research in Advanced Engineering. 1(4), 132-135.

[81] Singh, R.R. Tomar, D.S. (2015) Network Forensics: Detection and Analysis of Stealth Port Scanning Attack. International Journal of Computer Networks and Communication Security. 3(2), 33-42.

[82] Bairwa, S., Mewara, B. Gajrani, J. (2014) Vulnerability Scanner: A Proactive Approach to Assess Web Application Security. International Journal on Computational Sciences Applications (IJCSA). 4(1), 113-124.

[83] Loukas, G. Oke, G. (2010) Protection Against Denial of Service Attacks. The Computer Journal. 53(7). 1020-1037.

[84] Elleithy, K.M., Blagovic, D., Cheng, W. Sideleau, P. (2005) Denial of Service At-

tack Techniques: Analysis, Implementation and Comparison. Journal of Systemics, Cybernetics and Informatics. 3(1), 66-71.

[85] Patrikakis, C.Z., Masikos, M. Zouraraki, O. (2004) Distributed Denial of Service Attacks. The Internet Protocol Journal. 7(4). 13-35.

[86] Techopedia (N/A) Eavesdropping [online] Available from: https://www.techopedia.com/ definition/13612/eavesdropping [Accessed: 12 April 2017]

[87] Grover, K., Lim, A., Yang, Q. (2014) Jamming and anti-jamming techniques in wireless networks: a survey. International Journal of Ad Hoc and Ubiquitous Computing. 17(4). 197-215.

[88] Sufyan, N., Saqib, N.A., Zia, M (2013) Detection of jamming attacks in 802.11b wireless networks. EURASIP Journal on Wireless Communications and Networking. Doi:10.1186/1687-1499-2013-208.

[89] Vonnegut, S. (2016) OWASP Mobile Top Ten: Avoiding the Most Common Mobile Vulnerabilities [online] Available from: https://www.checkmarx.com/2016/06/10/owasp-mobile-top-ten-avoiding-common-mobile-vulnerabilities/ [Accessed: 12 April 2017]

[90] Cooney, M. (2012) 10 common mobile security problems to attack [Online] Available from: http://www.pcworld.com/article/2010278/10-common-mobile-security-problems-to-attack.html [Accessed: 13 April 2017]

[91] Bamiah, M.A. Brohi, S.N. (2011) Seven Deadly Threats and Vulnerabilities in Cloud Computing. International Journal of Advanced Engineering Sciences and Technologies. 9(1). 87-90.

[92] Lukan, D. (2014) The top cloud computing threats and vulnerabilities in an enterprise environment [online] Available from: https://www.cloudcomputing-

news.net/news/2014/nov/21/top-cloud-computing-threats-and-vulnerabilities-enterprise-environment/ [Accessed: 13 April 2017]

[93] Tsidulko, J. (2016) The 10 Biggest Cloud Outages of 2016 [online] Available from: http://www.crn.com/slide-shows/cloud/300081477/the-10-biggest-cloud-outages-of-2016-so-far.htm/pgno/0/10 [Accessed: 13 April 2017]

[94] NIST (2016) Guide to Bluetooth Security [online] Available from: http://csrc.nist.gov /publications/drafts/800-121/sp800$_1$21$_r$2$_d$$raft.pdf[Accessed : 16April$2017]

[95] Vishali, R. (2014) Security in Wireless Local Area Networks. International Journal of Computer Science and Information Technology Research. 2(2). 472-483.

[96] Tahir, M.N. (2007) “C-RBAC: contextual role-based access control model”. Ubiquitous Computing and Communication Journal. 2(3): 6774.

[97] Barkley, J. (1995) “Implementing Role-Based Access Control Using Object Technology”. In Proceedings of the First ACM Workshop on Role-Based Access Control (RBAC), pp. 93-98.

[98] NHS Digital (N/A) Transparency notice: how we use your personal data [online] Available from: https://digital.nhs.uk/about-nhs-digital/ourwork/keeping-patient-data-safe/gdpr/gdpr-register [Accessed: 3 August 2021]

[99] U.S. Department of Health and Human Services (2013) Summary of the HIPPA Privacy Rule [online] Available from: https://www.hhs.gov/hipaa/for-professionals/privacy/lawsregulations/index.html [Accessed: 3 August 2021]

[100] Aver, H. (2021) Ransomware attacks on healthcare [online] Available from: https://www.kaspersky.co.uk/blog/ransomware-vshealthcare/22670/ [Accessed: 6 August 2021]

[101] Center for Internet Security (N/A) Ransomware: In the Healthcare Sector [online] Available from: https://www.cisecurity.org/blog/ransomware-inthe-healthcare-sector/ [Accessed: 6 August 2021]

[102] Ashu, M.R., Zafar S. (2021) DDoS Attacks Impact on Data Transfer in IOT-MANET-Based E-Healthcare for Tackling COVID-19. In: Khanna A., Gupta D., Polkowski Z., Bhattacharyya S., Castillo O. (eds) Data´ Analytics and Management. Lecture Notes on Data Engineering and Communications Technologies, vol 54. Springer, Singapore.

[103] Sami, I., Asif, M., Ahmad, M.B., and Ullah, R. (2018) DoS/DDoS Detection for E-Healthcare in Internet of Things. International Journal of Advanced Computer Science and Applications. 2(1): 297-300.

[104] Hunter, A. (N/A) Taxonomies [online] Available from: http://www0.cs.ucl.ac.uk/staff/a.hunter/ tradepress/tax.html [Accessed: 25 January 2017]

[105] Noy. N. F, McGuinness, D. L. (2001) Ontology Development 101: A Guide to Creating Your First Ontology [online] Available from: http://protege.stanford.edu/publications/ontology$_{d}evelopment/ontology101.pdf[Accessed:$ $19January$2017]

[106] U.S. Food Drug Administration (2015) Implants and Prosthetics [online] Available from: http://www.fda.gov/MedicalDevices/ProductsandMedicalProcedures/ImplantsandProsthetics [Accessed: 10 January 2017]

[107] Medical Device and Diagnostic Industry (2013) Body Hackers Implant Homemade Health Monitor [online] Available from: http://www.mddionline.com/blog/devicetalk/body-hackers-implant-homemade-health-monitor [Accessed: 10 January 2017]

[108] Karulf, E. (2008) Body Area Networks (BAN) [online] Available from:

http://www.cse.wustl.edu/ jain/cse574-08/ftp/ban/index.html [Accessed: 10 January 2017]

[109] Jovanov, E. (2005) Wireless technology and system integration in body area networks for m-health applications. In: 2005 27th Annual International Conference of the IEEE Engineering in Medicine and Biology Science (EMBS). Shanghai: IEEE, pp. 7158-7160

[110] Solent NHS Trust (2020) Anonymisation of Data (Pseudonymisation) Policy and Procedure [online] Available from: https://www.solent.nhs.uk/media/1262/pseudonymisation-policy.pdf [Accessed: 12 November 2021]

[111] World Health Organization (2013) Transforming and scaling up health professionals' education and training [online] Available from: https://www.ncbi.nlm.nih.gov/books/NBK298953/ pdf/Bookshelf$_NBK298953.pdf[Accessed: 12November2021]

[112] Kang, J, Adibi, S. (2015) A Review of Security Protocols in mHealth Wireless Body Area Networks (WBAN). "The series of Communications in Computer and Information Science", Volume 523, P.61-83.

[113] Convery, S. (2007) Network Authentication, Authorization, and Accounting, "The Internet Protocol Journal", Volume 10, pp.2-11.

[114] Jaikaran, C. (2016) Encryption: Frequently Asked Questions [online] Available from: https://fas.org/sgp/crs/misc/R44642.pdf [Accessed: 8 August 2021]

[115] HIPAA Security Series (2007) Security Standards: Technical Safeguards [online] Available from: https://www.hhs.gov/sites/default/files/ocr/privacy/hipaa/administrative/ securityrule/techsafeguards.pdf [Accessed: 8 August 2021]

[116] Dennis, J.B., and Horn, E.C.V. (1966). Programming semantics for multiprogrammed computations. Communications of the ACM, 9(3):143155.

[117] Hermann, S., and Fabian, B. (2014) A Comparison of Internet Protocol (IPv6) Security Guidelines. Journal of Future Internet, 6(1): 1-60.

[118] Shaw, K., and Fruhlinger, J. (2020) "What is IPv6, and why aren't we there yet?" [Online] Available from: https://www.networkworld.com/article/3254575/what-is-ipv6-andwhy-aren-t-we- there-yet.html [Accessed: 2 August 2021]

[119] Ondiege, B., Clarke, M., and Mapp, G. (2017) Exploring a New Security Framework for Remote Patient Monitoring Devices. Journal of Computers. 6(1): 11.

[120] Cisco Press (2010) Developing Network Security Strategies [online] Available from: https://www.ciscopress.com/articles/article.asp?p=1626588seqNum=2 [Accessed: 10 August 2021]

[121] Ezenwigbo, O.A., Paranthaman, V.V., Trestian, R., Mapp, G., and Sardis, F. (2018) Exploring a New Transport Protocol for Vehicular Networks. In 2018 the 5th International Conference on Internet of Things: Systems, Management and Security (IoTSMS). pp.287-294.

[122] Mapp, g. (2017) The Simple Lightweight Transport Protocol (SLTP) for Low Latency Environments [online] Available from: https://moam.info/the-simple-lightweight-transport-protocol-sltp-for-low-latency5c593f7a097c47fa378b45ff.html [Accessed: 9 August 2021]

[123] TayloyWessing (N/A) How secure is blockchain? [online] Available from: https://www.taylorwessing.com /download/article-how-secure-isblock-chain.html [Accessed: 10 August 2021]

[124] Korolov, M. (2016) The blockchain is now being hyped as the solution to all inefficient information processing systems [online] Available from: http://www.csoonline.com/article/ 3050557/security/is-theblockchain-good-for-security.html [Accessed: 10 August 2021]

[125] Mithchell, I. and Hara, S. Securing eHealth and mHealth: moving from frameworks to prototypes: Presented at the Health IT Workshop, Middlesex University, London, 7th-8th November 2019.

[126] ENISA (2017) Distributed Ledger Technology and Cyber Security – Improving information security in the financial sector [online] Available from: https://www.enisa.europa.eu/publications /blockchainsecurity [Accessed: 10 August 2021]

[127] Pair, S. (2015) The Secure Blockchain is Bitcoin's Biggest Asset [online] Available from: https://www.infosecurity-magazine.com/opinions/thesecure-Blockchain-is-bitcoins/ [Accessed: 10 August 2021]

[128] Mapp G, George C, Mitchell I, Hara S, Vithanwattana N, Jusob F and Samuels A: Securing eHealth and mHealth: moving from frameworks to prototypes: Presented at the Health IT Workshop, Middlesex University, London, 7th-8th November 2019.

[129] Ramirez, J., Ezenwigbo, O.A., Karthick, G., Trestian, R., and Mapp, G. (2020) A New Service Management Framework for Vehicular Networks. In 23rd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN). pp. 162-164.

[130] Carvalho, M., and Bandiera-Paiva, P. (2018) Health Information System Role-Based Access Control Current Security Trends. Journal of Healthcare Engineering. (3):1-8.

[131] Hermann, S., and Fabian, B. (2014) A Comparison of Internet Protocol (IPv6) Security Guidelines. Journal of Future Internet, 6(1):1-60.

[132] Shaw, K., and Fruhlinger, J. (2020) What is IPv6, and why aren't we there yet? [Online] Available from: https://www.networkworld.com/article/3254575/what-is-ipv6-and-why-aren-t-we-there-yet.html [Accessed: 2 August 2021]

[133] Karthick, G., Mapp, G., Kammuller, F. and Aiash, M. (2021) Modeling and verify-

ing a resource allocation algorithm for secure service migration for commercial cloud systems. Computational Intelligence. https://doi.org/10.1111/coin.12421