Adaptive Multi-Layer Encryption for Enhancing Security in Smart Home IoT Ecosystems

Dancan Obuya Machuki

168995

Submitted in Partial fulfilment of the Requirements for the Degree of Master of Science in Information Systems Security at Strathmore University

School of Computing and Engineering Sciences

Strathmore University

Nairobi, Kenya

June, 2025

Declaration

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

Student's Name: **Dancan Obuya Machuki**

Sign: _____ Date: 19/05/2025

Approval

The dissertation of **Dancan Obuya Machuki** was reviewed and approved by the following:

Dr. Kennedy Ronoh,

Senior Lecturer, School of Computing & Engineering Sciences,

Strathmore University

Eng. Dr. Julius Butime,

Dean, School of Computing and Engineering

Strathmore University

Prof. Bernard Shibwabo,

Director of Graduate Studies,

Strathmore University

Abstract

The explosion of Internet of Things (IoT) devices in smart home environments has significantly improved the automation and user convenience while introducing complex security and privacy challenges. Current IoT devices have inherent resource constraints and often struggle to implement traditional encryption mechanisms such as Advanced Encryption Standard (AES) and Rivest-Shamir-Adleman (RSA), which demand a lot of computational resources and lack adaptability to emerging threats. This research addresses this security gap by developing an Adaptive Multi-Layer Encryption (AMLE) model specifically tailored for smart home IoT ecosystems.

This study employed a prototype development methodology with an Agile approach to design and implement the AMLE framework. The proposed architecture operates across three distinct layers: device, network, and cloud. At the device layer, lightweight encryption algorithms were implemented to accommodate resource limitations. The network layer incorporated anomaly detection mechanisms using machine learning techniques to identify suspicious traffic patterns. The cloud layer implemented Attribute-Based Access Control (ABAC) to manage authentication, authorization and encryption for data storage. Testing was conducted in a controlled environment that simulated typical smart home conditions, with evaluation focused on functional correctness, integration capabilities, performance impact, and security effectiveness.

Findings demonstrate that the AMLE model effectively enhances protection against unauthorized access attempts, botnet infiltration, and data exfiltration while maintaining acceptable performance levels on resource-constrained devices. The adaptive capabilities of the system enable real-time security adjustments based on detected threat levels, device capabilities, and data sensitivity. This research contributes to the field of IoT security by providing a practical encryption framework that balances security requirements with the operational constraints of smart home environments.

Keywords: *Smart Home Security, IoT Encryption, Adaptive Security, Multi-Layer Protection, Resource-Constrained Devices, Privacy Preservation*

Table of Content

List of Figures

List of Tables

Abbreviations/Acronyms

2FA	Two-Factor Authentication
ABAC	Attribute-Based Access Control
AES	Advanced Encryption Standard
AMLE	Adaptive Multi-Layer Encryption
DDoS	Distributed Denial of Service
ECC	Elliptic Curve Cryptography
FPGA	Field-Programmable Gate Array
IDS	Intrusion Detection System
IoT	Internet of Things
MitM	Man-in-the-Middle (a type of cyberattack)
ML	Machine Learning
NP-hard problem	Non-deterministic Polynomial-time hard problem
OAuth	Open Authorization (an open standard for access delegation)
RBAC	Role-Based Access Control.
RSA	Rivest-Shamir-Adleman (an asymmetric encryption algorithm)
VPN	Virtual Private Network

Definition of Terms

Adaptive Multi-Layer Encryption (AMLE)	A security model that dynamically increase and decrease the degrees of encryption depending on the nature and risk classification of the data (Hamarsheh, 2024).
Cybersecurity Threat	Anything that poses a threat that could be taken advantage of as a weakness in a system to gain access that is unauthorized or corrupt information and systems (Mosenia & Jha, 2017)
Data Breach	An event in which an unauthorised person (s) obtains and acquire access to confidential information that may include identity theft, money fraud or invasion of privacy (Radanliev et al., 2020).
Encryption	A process that converts plaintext data into ciphertext using algorithms and keys to protect information from unauthorized access (Rahman et al., 2022).
Internet of Things (IoT)	An integrating network of physical items that can interact and exchange data through the world wide web for the purpose of creating value through automated and intelligent utilities (Bouaouad et al., 2020).
Smart Home	A home setup that will include appliances and systems that may be monitored and modified through a device connected to the internet to bring an increased level of comfort, security, and power efficiency (Chakraborty et al., 2023).

Acknowledgements

I extend my sincere thanks to the extraordinary men and women as well as the institutions who participated in making the completion of this research project dissertation a success. To begin with, I want to thank Dr. Kennedy Ronoh my supervisor who's been an invaluable and unwavering guide in the shaping of this work. I am also thankful to Professor Joseph Sevilla for his guidance and advisory in shaping the structure and content of this dissertation.

I am grateful to Strathmore University's library as it offers great resources including a diverse information that were necessary to my research and analysis. I am equally thankful to my colleagues and friends who read my work, left feedback whose words of encouragement and enthusiasm have inspired me along this journey.

I want to end by saying the most important of all: I am truly grateful to God for the gift of health, perseverance and strength to achieve an academic accomplishment.

Dedication

This work is dedicated to my parents Mr. Elijah Ondara and Mrs. Norah Mageto, who instilled the foundation of education in me and encouraged me to recognize the power of knowledge and how it can change your life. I owe their unwavering support and guidance to the fact that I've learned that lifelong learning is a thing, and it has and continues to shape and inspire me every single day.

I also dedicate this to my siblings, close friends and cousins who have always encouraged, believed and have been a strength in hard times. Their words of encouragement and motivation, especially as we would sit and talk well into the night, as we fought the temptation to give up and accept mediocrity. Thank you, my family, for being my pillars of support, you, are my greatest motivation.

Chapter 1. Introduction

1.1 Background to the Study

Internet of Things (IoT) has revolutionized modern society, seamlessly weaving physical devices into the fabric of our lives. his is due to the ability to automate homes, industrial control systems, among others. These are possible due to the ability of these devices being able to exchange data and performing data-driven decision making. They are highly efficient which enables them to perform their functions smoothly.

Internet of Things (IoT) and smart homes are two of the biggest innovations today that have revolutionized the way people live, bringing effectiveness and connectivity to every nook and cranny of homes. Such ecosystems are complex due to the vast number of devices including smart thermostats, lighting systems, security cameras and home appliances that interconnect with each other. Albeit this convergence has many advantages, it also creates new security issues that render smart home IoT environment as a perfect opportunity for malicious cyber actors (Hamarsheh, 2024).

Adaptive Multi-Layer Encryption (AMLE) is a dynamic approach to the protection of IoT devices that operates by adjusting encryption levels based on the nature, situation and classification of data either in storage or transmission. The approach also takes into consideration the various IoT layers involved. The layers involved include device layer, network layer, and the cloud layer. Security at these different layers is essential to ensure a smart home IoT ecosystem is secure and all data in storage or transmission is protected.

Some of the existing protection mechanisms that have been adopted are the use of conventional security mechanisms to protect the data being transferred within these ecosystems. However, these encryption techniques do not change often meaning they do not adapt to the new evolving cyber threats. As the cybersecurity threats advance correspondingly it is necessary to introduce a complex data protection technique, which is able to learn and act accordingly (Honar Pajooh et al., 2021).

Today, most IoT devices use encryption algorithms such as Advanced Encryption Standard (AES) or Rivest-Shamir-Adleman (RSA), which are well-established but insufficient for dynamic protection from constantly evolving threats (Raza et al., 2017). As IoT devices lure MitM attacks, data breach, or other cyber-attacks, there is a necessity for an encryption solution

that matches real-time security demands. Consequently, this research suggests an adaptive multi-layer encryption model to counter these threats in smart home IoT contexts.

This study primarily aimed to fill the security holes found in smart home IoT ecosystems. It looks at the possible new ways that hackers could attack these systems and then try to see if this solution could stop them. This is a kind of encryption framework that better protects smart home IoT devices and their data, even when the devices themselves cannot be trusted. This is a new approach to a pretty old problem: keeping secrets and protecting usable data in the presence of a malicious actor (Nimmy K.and Sankaran, 2018).

1.2 Problem Statement

The rapid expansion and growth of smart home IoT ecosystems has introduced a significant security challenge which has become a major security concern; built into many products with little or no encryption measures. These weaknesses put data at risk with threat actors capable of eavesdropping on connections, extract people's information, and take over devices. The lack of a robust and adaptable encryption in these networks can lead to very severe privacy violations, operational disruptions, and unauthorized intrusions, this in turn compromises the safety of the user and IoT devices involved especially in a smart home ecosystem.

Even though Advanced Encryption Standard (AES) and Rivest Shamir Adleman (RSA) algorithms are effective for regular computers, they consume too much energy and require too much computation to use in IoT systems for smart homes. Because of their low memory, processing limit and battery capacity, these devices struggle to handle complex encryption algorithm. In addition, traditional encryption techniques cannot change as new threats appear which puts smart home networks at risk of problems like latency, bottlenecks and security breaches. Thus, a new approach to encryption is required that matches the way IoT devices and threats are constantly changing.

1.3 Objectives of the Study

1.3.1 General objectives

The main objective of this research is to analyse the weaknesses in encryption of smart home IoT ecosystems and develop an adaptive multi-layer cryptographic approach that can protect information within these networks and devices from unauthorized access and cyber threats.

1.3.2 Research objectives

1. To investigate the encryption characteristics of smart home IoT networks and devices,

2. To analyse existing encryption solutions for smart home IoT devices,
3. To design and implement an adaptive multi-layer cryptographic approach for smart home IoT ecosystems,
4. To validate the effectiveness of the developed encryption solution.

1.4 Research Questions

This study was guided by the following primary research question: "How can an adaptive multi-layer encryption framework enhance security in resource-constrained smart home IoT ecosystems while maintaining operational efficiency?"

To comprehensively address this main question, the following secondary research questions were formulated:

1. What are the encryption characteristics and challenges faced by smart home IoT networks and devices?
2. What are the limitations of existing encryption solutions used in smart home IoT environments?
3. How can an adaptive multi-layer cryptographic model be designed and implemented to secure smart home IoT systems?
4. How effective is the proposed adaptive encryption model in mitigating unauthorized access and security threats in smart home IoT environments?

1.5 Justification

Smart home IoT ecosystems have grown quickly, thereby providing users with convenience and automation but expose users to significant security risks. For instance, several IoT devices have minimal inherent or no encryption capabilities and are therefore prone to hacking(Mohammed Aziz Al Kabir & Sharif, 2023). In the paper by Praveen, Rama and Sumalatha (2024), the authors propose that static encryption protocols often failed to address the dynamic and evolving nature of cyber threats targeting IoT devices, thus making people's information and devices vulnerable.

This study enhances IoT security by introducing a dynamic encryption framework that improves data protection in smart home environments. The findings will help researchers, policymakers, and cybersecurity professionals develop better security solutions.

According to Statista (2023), there are expected to be more than four hundred million IoT devices in operation throughout the world in 2024 and projected to be over seven hundred

million by 2028. These increasing interlinked networks require sound security systems to enhance their protection. Tnvs, Odugu, and Lingamgunta (2024), state that although the presence of encryption protocols is a critical feature of information security, if these protocols were proactive and able to combat changes in the threat landscape, the possibility of a data breach or unauthorized access would reduce significantly. Through utilization of AMLE, smart home systems can make encryption dynamic at different layers (device, network, cloud) to achieve operational reliability without a compromise on security.

The deployment of AMLE enhances data security while, at the same time, enhancing the acceptance of IoT solutions because it prevents sensitive information leaks and makes home automation secure. This approach fills a major void in current IoT security solutions, which is a flexible and expandable encryption solution that improves supports services and security at the same time.

1.6 Scope and Limitations

This dissertation aims at exploring encryption weaknesses in consumer oriented smart home IoT ecosystem gadgets including smart TVs, cameras, thermostats, smart locks, and other simple home appliances. It analyses the inadequacies of current encryption schemes and introduces an idea of deploying a dynamic multi-tier encryption system to control security threats. Both testing and validation was done in controlled tests to create realistic smart home scenarios in which the effectiveness of the solution in counteracting threats can be evaluated.

This dissertation is limited to smart home IoT devices specifically smart TVs and smart home routers and does not venture into the remaining industrial IoT domain, which may contain different demands for security. The multi-layer encryption in this dissertation is also limited to the device, network, and cloud layers of the IoT smart home ecosystem. There was also a concern whether the proposed encryption solution can easily evolve with the constantly changing threat landscape, and whether it will operate with the low-power computing capability of today's IoT devices.

1.7 Contribution of the Study

This study adds to the research on IoT security for smart homes by proposing a dynamic encryption system that adapts its level of protection to the sensitivity of the data and the security threat in a smart home IoT ecosystem. The study also reviews the way the model can be put into use at devices, networks and cloud layers of the ecosystem. Thus, it gives a solution to the

problem that traditional encryption can't solve in a modern cybersecurity landscape. In practise, the work offers a prototype and test findings that security researchers and developers can use to guide the creation of secure IoT smart home ecosystems.

1.8 Organisation of the Dissertation

The dissertation consists of six different chapters. Chapter one states what inspired the study, the issue behind it, important research questions, main goals, objectives and scope of the study. Chapter two reviews the literature about IoT security, different encryption standards and similar studies to show the gaps this study tackles. Chapter three explains how the study was done, methodology, analysis and design, testing and validation, ethical conditions and research design. Chapter four shows the system design, architecture and design diagrams relevant for the creation of the AMLE model. Implementation, testing and evaluating how the system works and showing the final outcomes are all explained in Chapter Five. The last section of the study provides a summary of key points, reviews the findings and proposes action for the future and conclusion.

1.9 Conclusion

This chapter introduced the main focus of the study by pointing out why IoT in smart homes matters and the issues of safety that come with it. Shortcomings in encryption used in many IoT devices led to the development of the main research question and objectives. In this chapter, the importance of an adaptable, multi-layer system for encryption and scope of the work are covered.

Chapter 2. Literature Review

2.1 Introduction

This chapter reviews existing literature on smart home IoT security, focusing on encryption of data in the IoT environments. It exposes the different technological approaches adopted today in protecting IoT devices and networks from advanced threats, and their advantages and disadvantages. The discussion also involves the comparison of the current encryption models and different multi-layer model used in IoT smart home ecosystems and finally a detailed presentation of a conceptual framework.

2.2 IoT Technology Review

The recent incorporation of the Internet of Things (IoT) technology in domestic settings has brought about numerous security issues, because of the weakness of some of the encryption protocols in a number of smart homes IoT ecosystem gadgets. Sicari et al. (2015) explained that smart locks, thermostats, cameras and even other IoT devices possess inadequate security components which open them to cyber threats that include unauthorized access, data breaches, and malware attacks. These issues stem from factors such as the weaknesses inherent in the computational capability of the devices, wherein powerful encryption algorithms cannot be applied, and the absence of unified security standards for device manufacturers.

According to research, the primary security concerns in smart home IoT frameworks are the exposure of sensitive data due to insufficient encryption protocols (Mosenia & Jha, 2017a). When data is transmitted in between devices and cloud-based servers and not properly encrypted the attackers may get hold of this information and exploit it hence the loss of user privacy and security breaches.

Weak encryption is also instrumental in the use of smart IoT devices in conducting botnet attacks. Comprising a set of compromised nodes, the botnets are highly dangerous for the reliability of the IoT system. According to Das & Gündüz (2019),it is observed that IoT devices once infected, they can be recruited into different botnets that employ Distributed Denial of Service (DDoS) attacks with the potential to disrupt networks and network services significantly.

2.2.1 Design Architecture of IoT Smart Home Devices

Smart home IoT devices are structured within specific software and hardware components that aid in their core functionality performance. The different layers of the IoT architecture are

identified in the paper by Bouaouad et al. (2020). The research identifies twelve layers that are used in IoT cloud environment architecture and are illustrated in Figure 2.1. They include:

i. Layer 1 Physical layer / Device layer/ IoT Infrastructure: It has different names based on the type of architecture. According to Fremantle (2015), it is the bottom most layer but for a device to be considered an IoT device it must have some communication either directly or indirectly to the internet.
ii. Layer 2: This layer relates to virtualization of connected objects. It deals with architectures that deal with virtualization technology (Bouaouad et al., 2020).
iii. Layer 3 Virtualization Infrastructure layer: It is equivalent to the technologies employed in the processes of making virtual objects real. This layer is stated only in architectures which are associated with virtualization (Bouaouad et al., 2020).
iv. Layer 4 short range communication / communication between objects: According to Pan & McElhannon (2018) this layer relates to interconnection and exchange between multiple layers. It is based on short-range communication between objects.
v. Layer 5 Access layer / Gateway Access layer / Gateways layer: This layer is relative to the different gateways and aggregators that the connected objects are centralized (Satyanarayanan et al., 2009).
vi. Layer 6 Network layer: This layer facilitates communication between a device and the cloud. It also involves the various communication protocols i.e., Hypertext Transfer Protocol (HTTP) or Message Queuing Telemetry Transport (MQTT) (Pan & McElhannon, 2018)
vii. Layer 7 Middleware layer / service platform layer / enabler layer: It plays a particularly important role in enabling interoperability between different platforms of the connected objects through APIs hence enabling data management. (Fremantle, 2015)
viii. Layer 8 Business layer: This layer is tasked with presenting business applications to end users.(Bouaouad et al., 2020)
ix. Layer 9 Cloud layer/ IoT Cloud layer / Datacentre layer: Its function is to be able to store, process and analyse the data fed back from the IoT part. (Bonomi et al., 2012)
x. Layer 10 Application layer: According to Bonomi et al. (2012), this layer in IoT architecture plays the role in which end users communicate and engage IoT devices as well as services. This layer is important in extracting meaningful results out of mere data received from the sensors to assist the users in handling the information received from the IoT systems.

xi. Layer 11: This layer is the application layer that relates to virtualization (Bouaouad et al., 2020)

xii. Layer 12: This layer specifically relates to access management and security in the IoT architecture.(Bouaouad et al., 2020)

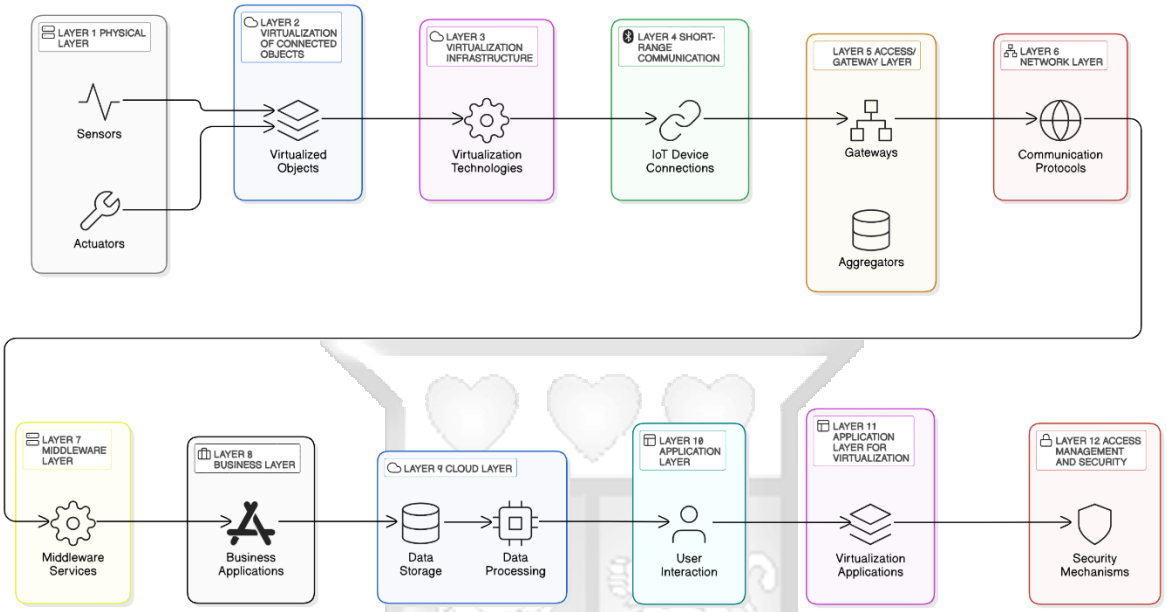


Figure 2.1: Illustration on IoT cloud environment architecture

The above stated layers have been studied and analysed from a set of almost thirty-two architectures. However, the most common six layers include the physical layer, the access layer, the network layer, the middleware layer, the cloud layer, and the application layer (Bouaouad et al., 2020).

A smart home architecture is also made up of four core components: sensors, actuators, controllers, and hubs. These components play an extremely critical role in collecting, processing and transmitting data within a connected smart home ecosystem (Navneet Kaur, 2021)

i. Sensors: Sensors record environmental inputs (e.g., temperature, motion, and light intensity) to notify other connected devices based on a series of detected inputs and expected outputs (Mosenia & Jha, 2017). They serve as the central source of data for any smart home system as they acquire data necessary to automate homes and make them convenient for the users; however, this data is often sensitive and can be intercepted.

ii. Actuators: They receive their commands from controllers, they then respond by performing certain actions like changing brightness, or opening lock. Since some of these actions may be vital to the security of homes, compromised actuators pose major security risks and threats since they can provide easy access to unauthorised persons or even manipulate home systems (Sicari et al., 2015).

iii. Controllers: Controllers take inputs from sensors and provide output to the actuators, so they can be considered as data intermediaries. If controllers work with data containing protected information, they are a weak link where data encryption is necessary to avoid data interception or unauthorised data access (Stephens et al., 2021).

iv. Hubs: These devices regulate interaction with other devices in the network, bringing data to and from the devices in the network. Some of the well-known protocols that hubs apply encompass Zigbee, Bluetooth, and the Wireless fidelity (Wi-Fi), with different levels of security. Given that hubs represent the first line of defence for connected ecosystems they should be secured with robust encryption and access controls to avoid entire ecosystems being undermined (Anthi et al., 2018).

2.2.2 Operation of a Smart Home Ecosystem

Smart Home ecosystems have experienced a phenomenal growth, primarily attributable to the introduction of IoT that facilitates meaningful interaction between home appliances and the end-user. An IoT based smart home ecosystems links all the attached devices with the internet and therefore can remotely access and control or manage differently aspects of homes (Chakraborty et al., 2023) .

A typical smart home ecosystem architecture as shown in Figure 2.2 involves three main phases: data acquisition, analysis, and services delivery. These include motion, temperature, and gas detectors, cameras and microphones which collect environmental information and the state of home devices. This information is transmitted either through cables or through wireless connections to the control centre that stores and processes or decides based on a set of standard parameters. For instance, if the temperature in the home goes beyond the defined limit, an alert goes to the homeowner or if there is unusual movement near the camera, an intrusion alert is raised. In the last phase, the processed information is further utilized for different services where it is required like controlling appliances, intruder alarms, environment monitoring services etc. Because of enhancements in the technique of machine learning (ML) as well as artificial intelligence smart home ecosystems can now identify voices, gestures, and faces which have led to enhanced security (Chakraborty et al., 2023).

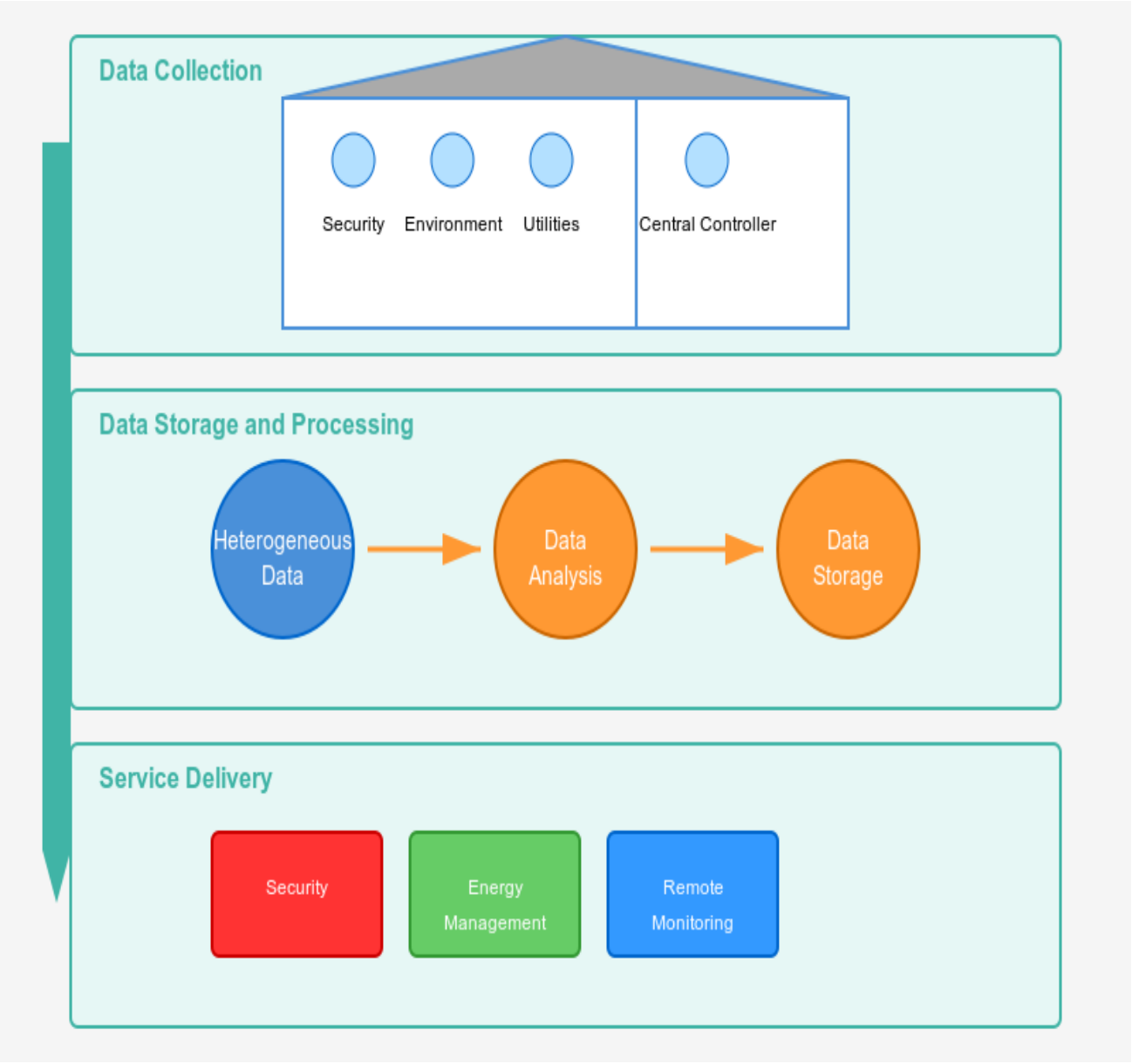


Figure 2.2: Smart Home Ecosystem Operation

(Source: Chakraborty et al., 2023)

2.3 IoT Smart Home Security

The safety of the IoT smart home environments is now an emerging research focus because of the increasing complexity and connectivity of devices in present day homes. These include the sensors, cameras, thermostats, and the locks among others; Due to their restricted computational abilities, most of these devices can be vulnerable to different cyber risks, not to mention most of them will have poor security mechanisms (Mosenia & Jha, 2017a). Security threats in smart home arise from issues like poor encryption; use of default credentials; and outdated firmware; all these issues create easy targets for hackers, data theft, and privacy infringement (Radanliev et al., 2020).These attacks include the Man-in-the-Middle (MitM) attacks, botnet attacks and the Distributed Denial of Service (DDoS) attacks are common in smart home networks because of the ease in attacking insecure devices (Sicari et al., 2015). Therefore, researchers recommend developing strong, and flexible security solutions that can be easily implemented according to the threat level, which can include multi-level encryption, to secure data at the device, network, and cloud layers (Khraisat & Alazab, 2021). Figure 2.3 below show the security requirements in a working IoT ecosystem.

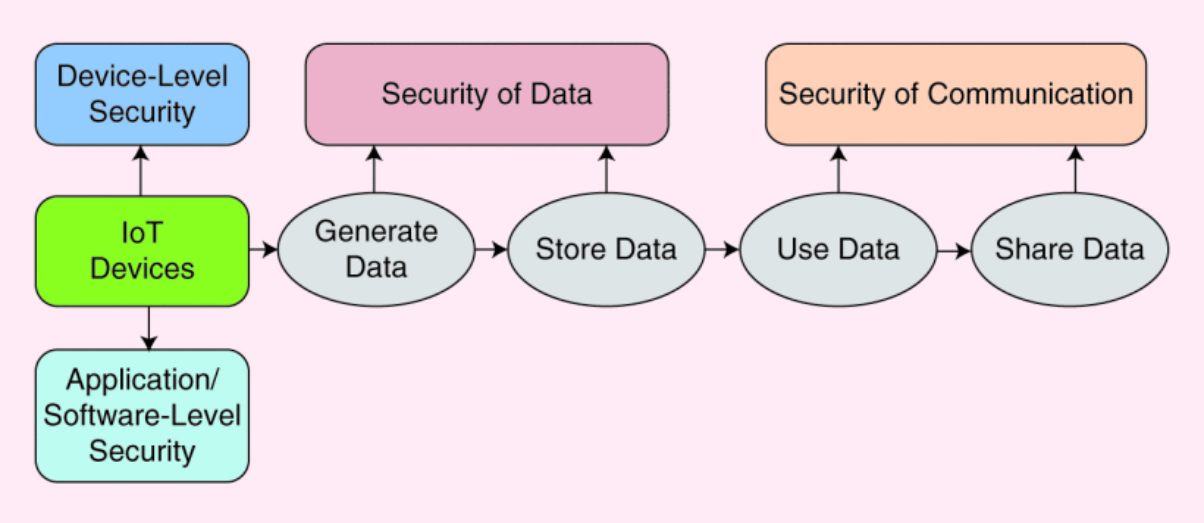


Figure 2.3: Security requirements in a working IoT Ecosystem

(Source: W. Z. Khan et al., 2019)

2.3.1 Smart Home Ecosystem Security threats

Smart home ecosystems, which combine various interconnected devices, such as sensors, cameras, thermostats, and locks, are inherently weak against various cybersecurity threats mainly because the connected smart devices infrastructure does not have many in-built security features.

Unauthorized Access

Unauthorized access is one of the inherent threats, in which a malicious attacker breaches devices by enumerating standard or easy passwords, weak encryption, or old firmware (Mosenia & Jha, 2017). They help attackers control devices remotely, alter their settings and even get access to other devices on the same network, and therefore affecting user privacy and security negatively. Such attack scenarios are particularly dangerous in case of devices controlling the access points or points of physical entry, such as smart locks, because unauthorized access may lead to physical security breaches (Sicari et al., 2015).

Man-in-the-Middle Attacks

Another critical factor is data privacy because most of the smart home IoT systems consist of devices that constantly gather, store, and transmit private user data including video streams, voice recordings and location data. The kind of threats and subsequent vulnerabilities identified in this research include Man-in-the-Middle (MitM) attacks, through which attackers gain access to the data transmitted between devices and the cloud with an aim of capturing or modifying the transmitted data. For example, accessing feeds coming from security cameras might compromise user's privacy, and the feeds might be further used to manipulate the users (Radanliev et al., 2020). Moreover, the accumulation of personal data on cloud services without proper encryption and access controls increases the chances or risk of data breach of users' information to unwanted parties (Stephens et al., 2021).

Botnet Attacks

Botnet attacks is a risk now that connected home devices often have very little built-in security mechanisms. A botnet refers to several compromised PCs controlled by a single attacker to facilitate Distributed Denial of Service attack or any other malicious campaign(Hussain et al., 2021). Smart home IoT devices are often weakly defended at best and hence are vulnerable to botnets such as Mirai that infect them and then use the devices to carry out a large amount of traffic to overload targets. Such botnet attacks affect device performance and increase bandwidth consumption and could potentially harm the whole smart home system (Khraisat & Alazab, 2021).

Firmware Vulnerabilities

Smart home devices have software or firmware vulnerabilities, and the attackers take advantage of these flaws to get into the device or system. Most of the IoT device firmware's are never

updated or have no method or way of updating automatically hence many IoT devices are open to exploits known already. Firmware attacks can result in attackers installing malware into these devices hence they persistent control and access to the devices. When the devices do not have proper security, firmware flaws may be used to disable safety mechanisms and increase the risk to users' lives (Yaacoub et al., 2023). These vulnerabilities highlight the lack of secure, flexible security solutions and mechanics that are capable adapting to the uncertainties in smart home environment.

2.3.2 Assessment of Existing Security Solutions for Smart Home IoT

Smart home IoT systems use multiple security strategies and mechanisms to guard devices from cyber threats and their owners' information. They may include firewalls, intrusion detection systems (IDS), authentication and the distinct types of access control (Bhardwaj et al., 2024). Despite that all provides a certain degree of security, there are also certain disadvantages that prove that only more powerful and constantly evolving security system is necessary when protecting IoT ecosystems.

a. Firewall and Network Security.

Firewalls are the main Defense mechanism for addressing the issue of denial for unauthorised access and filtering the traffic that is considered unacceptable into a network system especially for IoT devices in a smart homes environment. Conventional firewalls however are more likely programmed for larger network organizations and may not be effectively designed to suit IoT facilities well. Sicari et al. (2015), notes that most domestic consumer-off-the-shelf firewalls do not offer granular control over specific IoT devices, so they remain vulnerable to certain types of attacks such as port scanning and device specific vulnerabilities. As a result, several IoT-oriented firewalls have emerged, which provide some protection for IoT networks. However, these solutions can rarely tackle encrypted traffic analysis or even necessarily establish dynamic responses to new-fangled threat kinds, revealing the relative weaknesses of traditional firewall deployment on IoT networks (Nassi et al., 2021).

b. Intrusion Detection Systems (IDS).

Intrusion Detection Systems (IDS) play a vital role in IoT networks as they identify malicious activity by analysing network traffic and notifying the users or administrators. There are two primary types of IDS: networking based (NIDS) and host based (HIDS). Network-based IDS scan traffic across the network for any unusual traffic while the host-based IDS concentrates on devices. In smart home ecosystems, NIDS are more utilized because of the fragmented

structure of IoT devices (Khraisat & Alazab, 2021). However, traditional IDS rely often on signature-based detection, which can be useful against new attacks or new forms of malware, a major drawback for dynamic IoT environments within which new threats are continuously identified. In addition, IDS solutions need a great deal of computational capability, which is not synthesizable in the context of the IoT devices, thus making IDS less effective in protecting smart home IoT devices (Mosenia & Jha, 2017a).

c. Authentication Protocols

There are several measures among which authentication protocols for the purpose of managing the access to smart home devices and/or their data; Two factor Authentication (2FA), Open Authentication (OAuth), biometric authentication are particularly worth mentioning, for they are used in the management of permission levels to control the risk of unauthorized access due to password leakage. OAuth facilitates API authorization for smart home application so that the application can share information with a third party without revealing the user credentials (Mahalle & Shinde, 2021). But according to Aziz Al Kabir et al. (2023), the majority of the IoT devices do not possess the capability to afford complex and reliable authentication methods because of their inadequate processing capacity and memory resources. Among the identified research gaps, weak authentication is a significant issue in IoT structures, which uses universal or weak credentials created by equipment manufacturers or default ones that users do not change after connecting a device to an IoT network. Moreover, some devices neither allow for periodic updating of the credentials nor have this feature because credentials can become compromised with time thereby exposing the devices to unauthorized access.

d. Access Control Mechanisms

Access control security mechanisms like role-based access control (RBAC) and attribute-based access control (ABAC) are employed to regulate the usage of devices and data in smart home ecosystems by executing roles or attributes of the users. While in RBAC permissions are given according to the type of the user such as owner, guest, and so on, in ABAC rules for access are provided depending on attributes such as time, location, or type of the device (Beuchelt, 2013). However, as we pointed out earlier, IoT environments receive minimal levels of implementation of these mechanisms. Most of the consumer-oriented smart home devices either lack access control features or offer meagre control options that lack the depth of security. Khraisat & Alazab (2021), note that for lack of access control, unauthorised users or even compromised devices in the network may get access to the necessary data or make

changes to the necessary and sufficient device characteristics threatening the entire ecosystem with cyber threats.

e. Limitations and Gaps in Existing Solutions

While these security solutions help in the security of smart home IoT, they have some drawbacks when the same is applied to a consumer IoT environment. Firewalls and IDS mostly do not have adaptive mechanisms to counter new threats, and their functionalities are limited by the IoT devices capability (Mosenia & Jha, 2017). While authentication protocols do an incredibly good job of providing security within the device, there is weakness in applying security because of different hardware constraints across the devices (Sicari et al., 2015). Several consumer IoT devices rely on specific access control mechanisms for granular access management; however, these mechanisms are usually either not configured or not functional optimally owing to limited configuration possibilities (Alrawi et al., 2019). Altogether, these findings suggest the need for an integrated security approach like AMLE that allows the device, network, and cloud hierarchies to change their security settings to meet the security risks of smart home IoT systems.

2.3.3 Assessment of Existing Encryption Techniques for Smart Home IoT Ecosystem Security

Security concerning data integrity, confidentiality and privacy needs to be addressed when implementing smart home IoT systems (Uppuluri & Lakshmeeswari, 2022). Advanced Encryption Standard (AES), Rivest–Shamir–Adleman (RSA), and Elliptic Curve Cryptography (ECC) are some common techniques of encryption used in most of the IoT devices in use today. The mentioned encryption techniques are used in transmitting and storing of information in the IoT landscape. These methods set the foundation of security and at the same time they are limited in terms of computational overhead, scalability, and flexibility of IoT devices which have varied resource constraints specifically in the context of smart home IoT devices (Patil et al., 2022).

a. Advanced Encryption Standard (AES)

Advanced Encryption Standard (AES) is a form of symmetric key encryption algorithm which is suitable for the encryption of data in IoT networks most especially due to the tough and efficient manner with which it operates. Due to an extremely high processing speed with comparatively low memory utilization, AES is suitable for IoT devices that require rapid or swift data encryption and decryption processes (Tripathy & Singh, 2022). AES was initially

designed for its high encryption capacity similarly to high resource utilization hence is not efficient with limited resources found in IoT devices. Existing generic AES mechanisms may require minor modifications to perform well in real-time applications as well as in components with strict timing constraints such as voice data encryption on Field-Programmable Gate Arrays (FPGA)(M. N. Khan et al., 2021). Figure 2.4 illustrates how AES typically functions in IoT ecosystems.

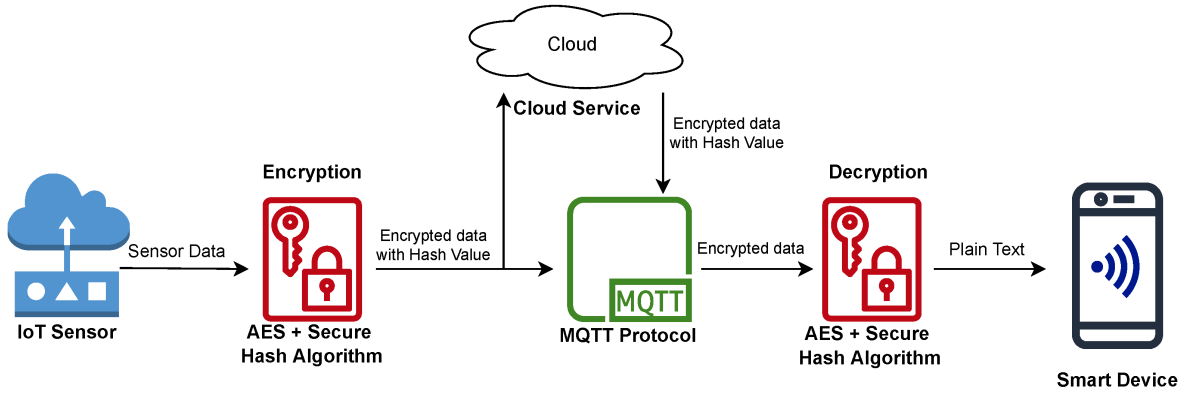


Figure 2.4: AES Functioning in an IoT Ecosystem

(Source: Rahman et al., 2022)

b. Rivest–Shamir–Adleman (RSA)

Rivest–Shamir–Adleman (RSA) is a well-known asymmetric encryption technique through which secure communication can be established over an insecure channel using a public key and a private key. RSA is particularly useful in IoT because it provides mechanisms for the distribution of keys securely and improves security of IoT data across the connected devices. RSA consumes substantial computational resources and memory, which poses a big challenge for most IoT devices (Mosenia & Jha, 2017). According to Fernández-Caramés et al. (2018), the high computational overhead that is associated with RSA can lead to inefficiencies in most resource-constrained IoT environments, which in turn leads to increased latency and overall power consumption. This limitation show that RSA is best suited for the transmission of sensitive data rather than transmission of routine non-sensitive data exchanges in IoT ecosystems.

c. Elliptic Curve Cryptography (ECC)

Elliptic Curve Cryptography (ECC) is cryptographic algorithm that is comparable to RSA with significantly smaller key sizes better suited for IoT. Thus, ECC is suitable for smart home IoT

devices most of which have limited power and memory storage capacity especially for constrained gadgets. ECC can offer high level of security rather shorter keys span in terms of time and energy required to perform the span. The discrete elliptic curve logarithm issue, sometimes known as an NP-hard problem, is solved in cryptography using elliptic curves (Zhang & Wang, 2024). Despite the advantages afforded by Elliptic Curve Cryptography (ECC), notably in terms of security and efficiency, actual implementation of ECC on IoT devices is hindered by the availability of fewer available curve types, computational complexity of ECC operations, vulnerability to side-channel attacks, and secure key management (Ullah et al., 2023). The Figure 2.5 below shows an equation that describes an elliptic curve.

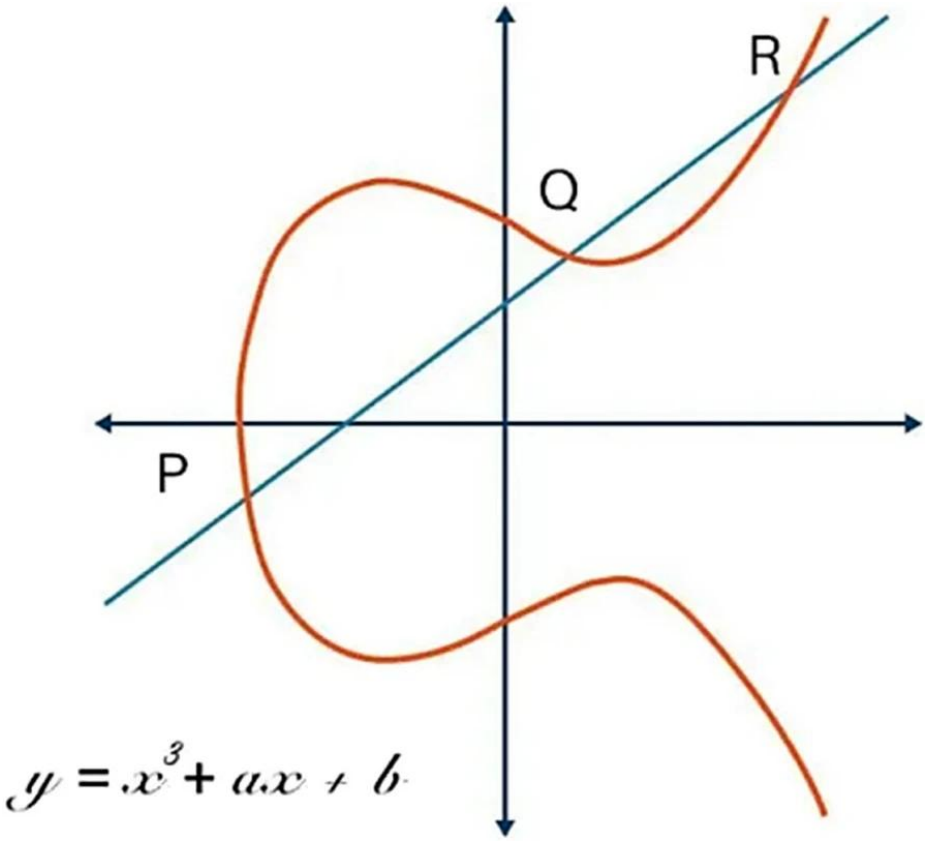


Figure 2.5: Depiction of an elliptic curve.

(Source: Zhang & Wang, 202))

d. Limitations of Existing Encryption Techniques

As used in today's smart home IoT ecosystems, the most common encryption techniques Advanced Encryption Standard (AES), Rivest–Shamir–Adleman (RSA) and Elliptic Curve Cryptography (ECC) have their own limitations that restrict their effectiveness in the resource

constrained environment. While efficient in high-performance applications, AES demands modification to realize optimal functionality in devices with limited resources, and its secure key management further possess challenges (Tripathy & Singh, 2022). RSA is a secure key distribution method with huge computational overhead and a high latency for frequent data exchange in IoT networks (Fernández-Caramés et al., 2018). Similarly, ECC offers strong security but at smaller key size but its computational complexity, availability of curve, vulnerability to side channel attack, and difficulty of key management, makes it unviable to be used as a standalone encryption scheme in the dynamic IoT environment (Ullah et al., 2023). These show the necessity to scrutinize and combine different techniques to form a hybrid and adaptive approach to increase security in smart home IoT applications.

The security of data integrity, confidentiality, and privacy is a critical consideration in the implementation of smart home IoT systems (Uppuluri & Lakshmeeswari, 2022). Current smart home IoT devices commonly employ encryption techniques such as Advanced Encryption Standard (AES), Rivest Shamir Adleman (RSA), and Elliptic Curve Cryptography (ECC). These techniques are used for transmitting and storing information within the IoT landscape. However, these encryption methods have limitations in terms of computational overhead, scalability, and the flexibility required to support IoT devices with varied resource constraints, particularly in smart home settings (Patil et al., 2022).

2.4 Conceptual Framework

The conceptual framework for the AMLE model in smart home IoT environment is illustrated in Figure 2.6. This model gives a multi-layered security approach that dynamically adjusts encryption and protection measures based on current threats and the sensitivity of the data.

In the device layer information gathering takes place through smart home devices such as sensors, cameras, smart TVs, etc. lightweight encryption is deployed in real time as soon as the data is captured to ensure only those, who are permitted, can access it. The type and strength of encryption depends on computational capacity of each device and the level of sensitivity of data. An example is a security cameras or other similar devices which handle more sensitive data use a stronger encryption than temperature sensors.

Data then goes to the network layer undergoes proactive defence counters like behaviour analysis, anomaly detection for incoming and outgoing traffic. This layer implements a varying degree of encryption based on threat intelligence data in real-time. Periodically, especially if

the risk level is high or there is anomalous traffic characteristics, the network layer increases encryption standards to limit or eliminate unsafe traffic, thereby managing potential breaches.

Once at the cloud layer data is further encrypted using techniques such as asymmetric encryption and access controls mechanisms like ABAC are placed where only an authorized person can read or manipulate any data. Real-time threat intelligence enhances this layer as machine learning algorithms identify threats as they appear and is used to enhance security of the data at this level.

Finally, users interact with the system through a Secure User Interface where users can view real-time analytics and data visualization of smart home environment performance and security. This control interface means that nobody has unauthorised access to any data that can be found in the system and hence users are able to control, modify and access sensitive data.

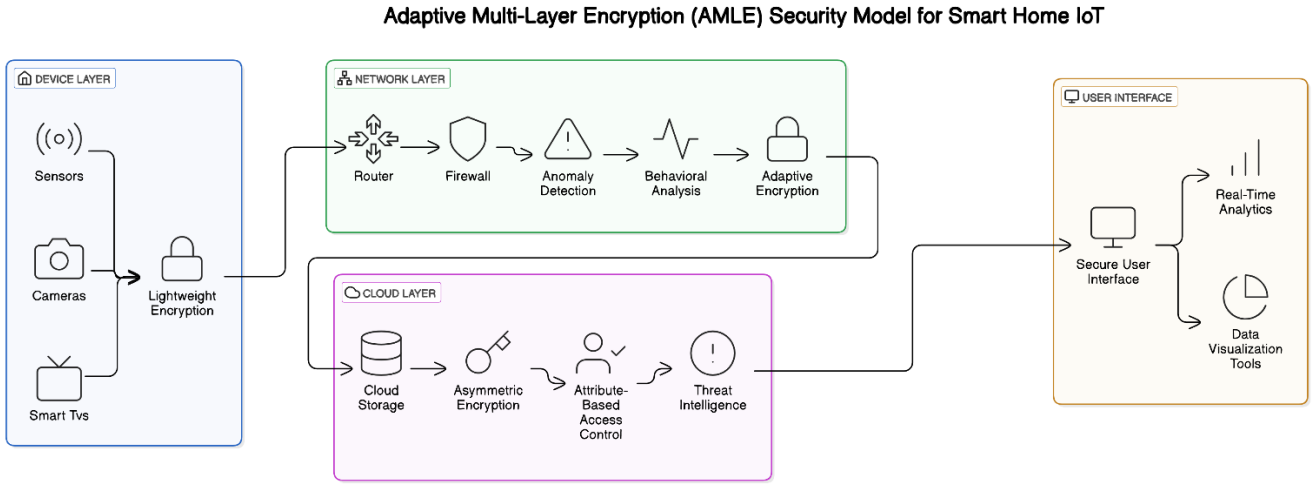


Figure 2.6: Conceptual framework.

2.5 Summary

Smart home IoT security was researched in this literature review, which reviewed related work in the context of current security and encryption techniques on smart home IoT, the applicable techniques and their benefits and drawbacks when in use in resource-constrained IoT environments. The first part of the review explained common smart home IoT system security threats, including data privacy risks, unauthorized access, and other vulnerabilities. Limited security controls, weak encryption, and inconsistent access mechanisms across IoT devices, underscored the need for more advanced security solutions.

It also looked at the existing security solutions: firewalls, intrusion detection systems, authentication protocols, and access control mechanisms, which provided fundamental security but were not adequate for the IoT threat landscape. While were effective to a degree, these tools often did not have the flexibility and scalability necessary for a wide variety and capability of IoT devices. Additionally, encryption techniques like AES, RSA, and ECC were assessed and their key importance in protecting the data of IoT, as well as their challenges, such as high computational demands, secure key distribution challenges, and lack of adaptability/flexibility to new security threats.

A review of the literature showed a lot of gaps in what was done in current security and encryption technology, especially missing a flexible and adaptable encryption model that adapts to current security risks in real-time. As argued above, these insights justified the design of an Adaptive Multi-Layer Encryption (AMLE) model, which proposed to realize a multi-layer, comprehensive security solution spanning the device, network, and cloud layers of smart home IoT ecosystems.

Chapter 3. Methodology

3.1 Introduction

The methodology for developing and validating the Adaptive Multi-Layer Encryption (AMLE) model for enhanced security in smart home IoT ecosystems is presented in this chapter. Research design, system development methodology, software tools and technologies, and the ethical issues are discussed in this chapter.

3.2 Research Design

The chosen research design was prototype development. This involved testbed setup, iterative testing, and systematic analysis. AMLE's adaptive security protocols were refined for real-world applicability in each phase. The Agile methodology was selected due to its iterative nature that enabled incremental improvements based on test feedback.

a. Prototype Development: A model was built, informed by Objectives 1 and 2, and the model was designed to incorporate multi-layer encryption at device, network, and cloud layers.
b. Testbed Setup: A simulated smart home IoT environment was configured, and AMLE's security efficacy was analysed.
c. Controlled Testing: Security tests were performed on an IoT smart home product to emulate real-life situations.
d. Findings Analysis: The performance metrics from testing were analysed to further refine AMLE.

3.3 System Development Methodology

The chosen methodology is agile methodology as shown in Figure 3.1. As the AMLE components are not interdependent, agile will allow for very flexible, iterative development whereby a particular AMLE component will implemented and tested as part of a sprint. The reason this methodology is selected is because it is suitable for rapidly evolving security requirements and a need for a real time, adaptive change in response of security threats

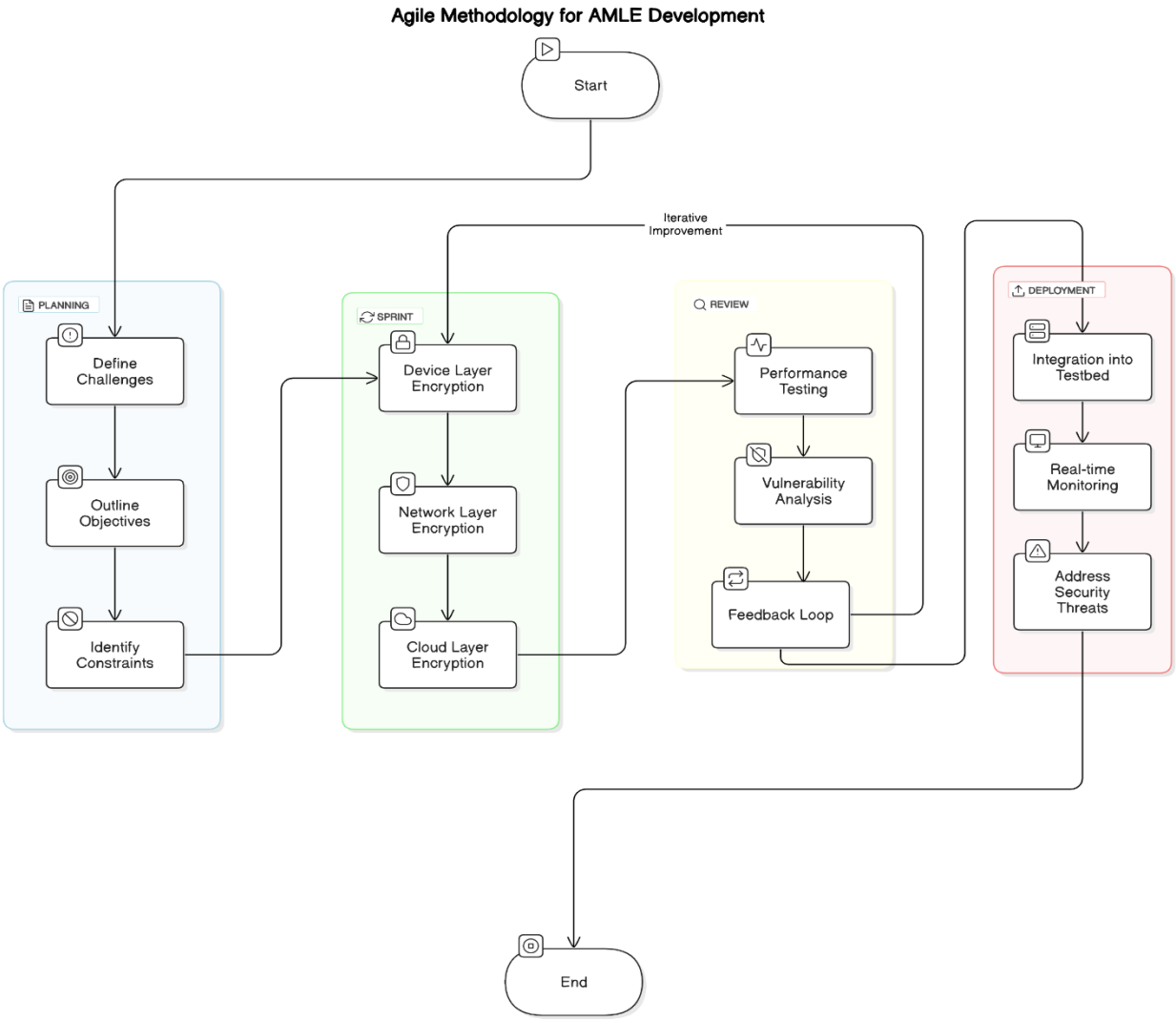


Figure 3.1: Agile Methodology for AMLE Development

3.3.1 Planning Phase

The Planning phase started with defining the IoT encryption problem clearly within the smart home ecosystem. It focused on identifying the different security challenges brought in by the IoT devices, which included constrained computational power, multiple communication protocols, and large-scale security. It created specific objectives, including the development of a solid, flexible encryption model that would protect the data across devices, networks, and cloud environments. Constraints were also identified, including IoT device processing power limitations, latency, and bandwidth issues in IoT networks (Sicari et al., 2015). The constraints described helped guide AMLE design and implementation.

3.3.2 Sprint Phase

The AMLE development in the Sprint Phase took place via short, iterative cycles called sprints, with times of a few 2–3 weeks each. In these sprints, the encryption model was developed incrementally from the simpler parts, like device layer which had light weight encryption, to the more complex parts like the network and cloud layers. Then, at the end of each sprint, progress was reviewed and feedback was gathered. The feedback from this informed the subsequent sprints and gave the opportunity to make adaptive changes to the encryption model based on real-world challenges. A good example was in that during the early stages the encryption focused on basic encryption mechanisms while the later sprints focused on refining and implementing the adaptive behaviour. This phase was also iterative, whereby the model remained flexible to the needs of smart home IoT ecosystems that kept changing.

3.3.3 Review Phase

In the Review Phase, the system was benchmarked in every sprint. The testing analysed the efficacy of AMLE's encryption mechanisms in comparison with its adaptability to new threats. Predefined performance metrics, including encryption speed, energy efficiency and security effectiveness, were always tested. What happened next was that during the next sprint the issues were addressed based on any vulnerabilities or weaknesses that were discovered during testing. For example, if there were some types of attacks where the encryption layer could not immediately react fast enough or had high latency, adjustments were made whereby the anomaly detection model was optimized or the encryption algorithms were tuned. This phase was critical to find out where improvements could be made, and if the system was reliable and robust enough to be deployed permanently.

3.3.4 Deployment Phase

n the Deployment Phase, AMLE was deployed in a controlled testbed environment simulated to create a smart home IoT ecosystem. In that environment, we got real-time monitoring of how AMLE performed on the detection and response against various threats it faced. This included data about system behaviour including how well it worked with different attack vectors, and performance metrics such as processing time, network latency, and resource consumption. For the monitoring phase, the results were used to refine the system to improve performance, reduce vulnerabilities and to scale the AMLE to operate in more complex IoT environments. Since continuous monitoring helped identify all remaining weaknesses, this helped in turn iteratively improving and updating the results as needed (Sicari et al., 2015).

3.4 System Analysis and Design

In this phase, the requirements that were identified in the planning stages of Adaptive Multi-Layer Encryption (AMLE) were analysed and interpreted. The system design was vital as it was the one that specifically dealt with building the system. Various analysis and design tools were used to build the system, and they included:

i. Use Case Diagrams: A series of user interaction diagrams on the AMLE system showed how users processed information and responded to security alerts; reached a decision based on anomaly detection; and made decisions about data processing. They were used to help visualize how users interacted with the system to monitor and prevent real time security threats.

ii. UML and Sequence Diagrams: The AMLE layers were shown by these diagrams through data flow and communication between layers. They demonstrated the way in which various components of the encryption system interacted, and how an encryption process got triggered by a security event. The emphasis in sequence diagrams was on the sequence of these interactions so that encryption dynamically adapted to changes in the threat condition.

iii. Flow Diagram: The Flow Diagram represented the data processing and encryption workflows implemented across all layers of the AMLE system from device to the cloud to secure the processing of data. This flow showed the steps that were undertaken on the system from data collection to encryption and then storage.

iv. Entity-Relationship Diagram (ERD): ERD was used in modelling the data structure in AMLE to ensure that data integrity was maintained throughout. The ERD helped in effectively managing the data by defining the entities, their attributes, their relationship and thereby only granting access to sensitive data to only the authorised entities.

3.5 Implementation

To achieve comprehensive security for smart home IoT ecosystems, the AMLE solution was implemented in three core layers of the adaptive network. On one hand, a lightweight encryption algorithm was integrated on the device layer offering lightweight encryption designed for resource constrained IoT devices. This choice was based on the algorithm's efficiency, having strong security with least computational overhead on constrained processing and memory powered devices. adaptive anomaly detection models were deployed in the network Layer to protect data in transit. Continuous monitoring of regular traffic on the

network helped detect unusual patterns that could indicate certain threats available for real-time encryption parameter adjustment in the case of emerging vulnerabilities. In the cloud layer, the data storage was done by means of more advanced encryption algorithms and stronger attribute-based access control was implemented. In addition, the machine learning-based threat intelligence also further enhanced the security to analyse and predict the possible attack vectors to formulate proactive counteraction against emerging threats. With this multi-layer approach, we ensured that each level of the IoT ecosystem built on the needs of that layer, starting with lightweight encryption on devices and up to cutting-edge data protection in the cloud.

3.6 Testing and Validation

To verify the effectiveness of the Adaptive Multi-Layer Encryption (AMLE) solution, a clearly conclusive testing methodology was executed, such as Unit Testing, Integration Testing, and Performance Testing. Each component of AMLE, including the encryption protocols and machine learning (ML) models, was tested separately in Unit Testing to make sure that these components worked correctly. This enabled the detection of possible problem issues within these modules before those applications were integrated into the larger system. Interaction between these components among different layers of AMLE during Integration Testing were validated to assure that these components were integrated seamlessly, establishing security mechanisms within the system.

Secondly, AMLE's ability to adapt to different threat levels was rigorously tested in Performance Testing. Analysis of how good or bad the system was at defending against different kinds of security threats, intrusion attempts, or data breaches, and whether it was capable of changing encryption settings in real time in response to detected threat events. Through this combination of testing techniques, the security effectiveness of AMLE was proved.

3.7 Tools and Technologies

The following are some of the tools and technologies that were utilized:

i. Programming Languages: Python for machine learning modules, Java for encryption protocols, and SQL for database management.
ii. Development Environments: VS Code, Visual Studio, Node-RED and Android Studio for cross-platform compatibility.

iii. Libraries: Scikit-learn and TensorFlow for machine learning; OpenSSL, PyCryptodome (Python) for encryption.
iv. Testing Environments: Docker and Kali Linux to assist in testing the prototype
v. Security Monitoring and Analysis Tools: Wireshark for network analysis and JUnit for software testing, Splunk as a SIEM.
vi. Diagrams: Lucid chart, Eraser.io, PlantUML, draw.io, and Microsoft Visio for creating UML and other system diagrams.

3.8 Ethical Considerations

This study did not use real-world IoT devices but relied on simulated test environments. The sample size was based on the security testbed configuration, where simulated attacks were applied to different layers of the smart home ecosystem to avoid disruption of communication on live networks. This reduced the impact on operational systems and users.

The research complied with Strathmore University research ethics requirements including seeking ethical approval and adhered to NACOSTI research regulations. This also included a thorough process of seeking ethical approval and clearance before initiating any research activities.

No real human participants or personal IoT devices were involved in this study. All data used was synthetic and generated within a controlled simulation. No private, personal, or identifiable information was collected. Data cleansing was conducted by filtering out anomalous or inconsistent simulation results to ensure reliability. Regular review and continuous updating of mechanisms occurred to ensure that ethical considerations remained a priority, and adjustments were made promptly if any need be. Findings were published through Strathmore University's research repository.

3.9 Conclusion

This chapter has shown the research design and the agile methodology used to develop the AMLE model. The use of prototype development, iterative sprints, and continuous testing ensured a flexible and adaptive approach to addressing the complex security challenges in smart home IoT ecosystems. The methodology's emphasis on iterative refinement and real-world testing has laid a solid foundation for the subsequent implementation and evaluation of the AMLE model.

Chapter 4. System Design

4.1 Introduction

This chapter outlines the architectural and design structure of the Adaptive Multi-Layer Encryption (AMLE) model for securing smart home IoT ecosystems. AMLE is a comprehensive security framework that dynamically adjusts encryption strength across multiple system layers based on contextual factors including threat level, data sensitivity, and device capabilities. Unlike traditional static encryption approaches that apply uniform security regardless of context, the AMLE model addresses the critical research gap in IoT security by providing proportional protection that optimizes resource usage while maintaining security. This model applies a multi-layered security approach that responds dynamically to encryption mechanisms based on real-time threat intelligence and the nature of data to be communicated. The system design encompasses detailed functional and non-functional requirements, a comprehensive system architecture with precisely defined interfaces, detailed flow diagrams, and implementation specifications for each security component.

4.2 Functional and Non-Functional Requirements

4.2.1 Functional Requirements

The AMLE model fulfils the following functional requirements:

a. Real-time Data Encryption – IoT devices encrypts data immediately after collection.
b. Adaptive Encryption Mechanism – The system dynamically adjusts encryption levels based on risk assessment.
c. Anomaly Detection – Network traffic is analysed in real-time for anomalies.
d. Secure Data Storage and Access Control – Cloud storage implements asymmetric encryption and enforce Attribute-Based Access Control (ABAC).
e. User Interface for Monitoring and Control – Users enables viewing security analytics and modifies encryption policies.

4.2.2 Non-Functional Requirements

a. Scalability – The system handles increasing numbers of IoT devices.

b. Reliability – The encryption process does not introduce system failures.

c. Performance Efficiency – Low-latency encryption methods used for real-time communication.

d. Usability – The user interface is intuitive and accessible.

e. Security and Compliance – The system adheres to cybersecurity standards and privacy regulations.

4.3 System Architecture

The AMLE model is structured into four key layers as illustrated in the system architecture Figure 4.1 below:

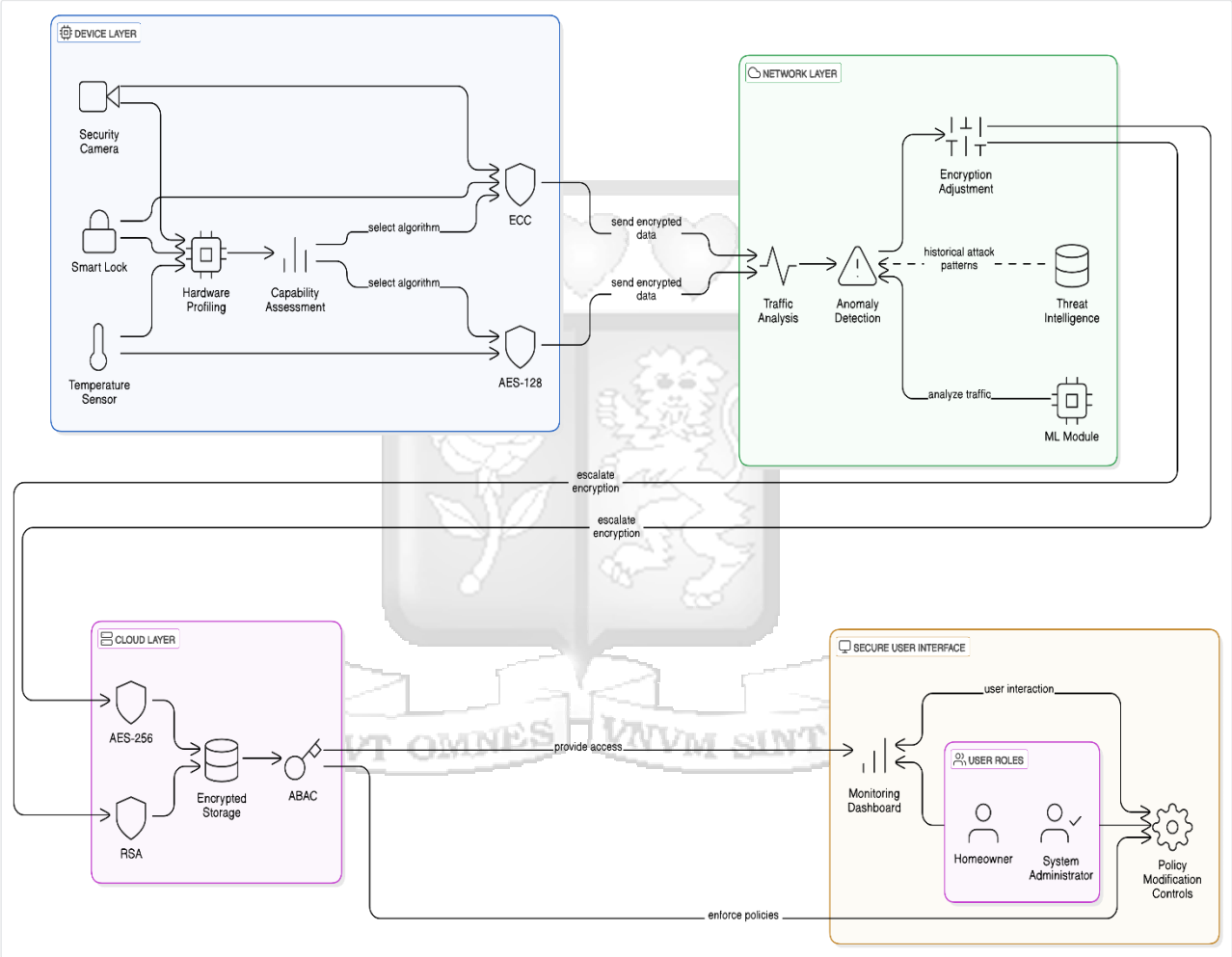


Figure 4.1: AMLE Architecture Diagram

4.3.1 Device Layer

The foundation of the Adaptive Multi-Layer Encryption (AMLE) model as shown in Figure 4.1 above begins with the device layer. Here, IoT devices, including smart locks, cameras, and temperature sensors, collected data from their respective environments. To ensure data security at its source, lightweight encryption algorithms, such as Elliptic Curve Cryptography (ECC) and AES-128, are applied. The strength of the encryption is dynamically adjusted based on the

computational capabilities of each device and the sensitivity of the data being processed. For example, security cameras, which handled highly sensitive information, utilized stronger encryption protocols compared to low-risk devices like temperature sensors.

4.3.2 Network Layer

Moving to the network Layer, the data traffic undergoes rigorous behaviour analysis and anomaly detection. This layer implemented dynamic encryption adjustments based on real-time threat intelligence. In instances where suspicious activity was detected, the system automatically increased encryption levels or blocked unsafe connections to mitigate potential threats. Machine learning algorithms played a crucial role, analysing traffic patterns to dynamically enhance the overall security posture of the network.

4.3.3 Cloud Layer

The cloud layer provided an additional layer of protection for stored data. Data was further encrypted using robust algorithms, such as AES-256 and RSA, to ensure secure storage. Moreover, access control mechanisms, like Attribute-Based Access Control (ABAC), were implemented to restrict unauthorized access, effectively safeguarding sensitive information within the cloud environment.

4.3.4 Secure User Interface

Finally, the Secure User Interface allowed authorized users to monitor real-time security analytics. This interface ensured that only authorized individuals could interact with the system, maintaining control and oversight over the smart home’s security posture. This control interface meant that no one had unauthorized access to any data that was found in the system, and hence users were able to control, modify, and access sensitive data.

4.4 System Design Diagrams

4.4.1 Use Case Diagram

The use case diagram Figure 4.2 illustrates interactions between users and the AMLE system. The user can log in, register devices, initiate encryption, monitor system events, and respond to security alerts. These actions correspond to the layered encryption functions at the device, network, and cloud levels. This diagram ensures that the system's functional scope is clearly defined and maps directly to the required user roles and security controls.

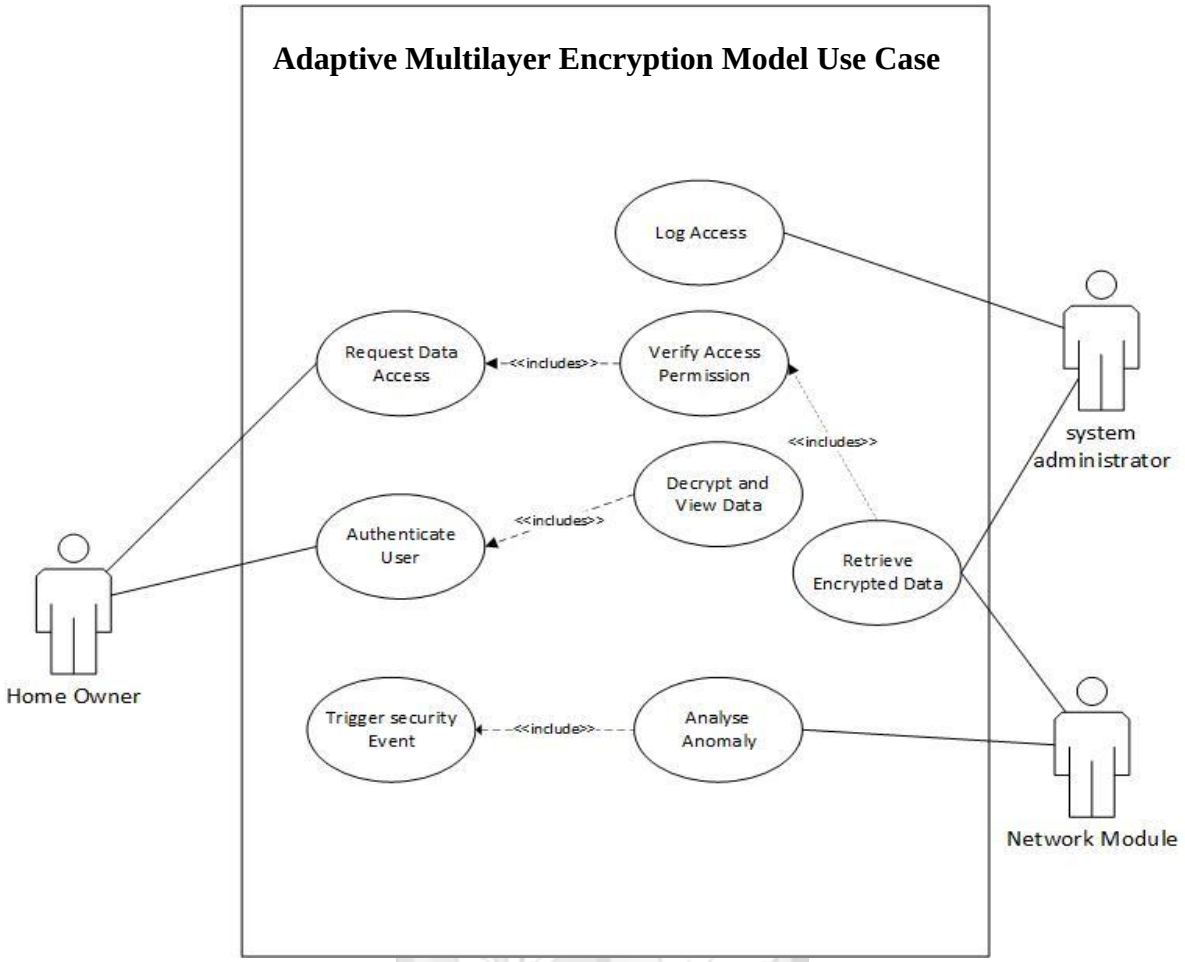


Figure 4.2: Use Case Diagram

4.4.2 Sequence Diagram

The sequence diagram Figure 4.3 illustrates the AMLE for securing smart home IoT environments. It shows the home owner requesting data from smart home devices, which capture and encrypt the data based on its sensitivity before transmitting it to the network security layer. Here, traffic is analysed for anomalies, and encryption strength is adjusted dynamically based on real-time threats. The data is then forwarded to the cloud security layer, where advanced encryption techniques like asymmetric encryption and access control mechanisms ABAC are applied. Finally, the secure user interface allows the home owner to authenticate, access data, and view security analytics while ensuring that unauthorized access is prevented.

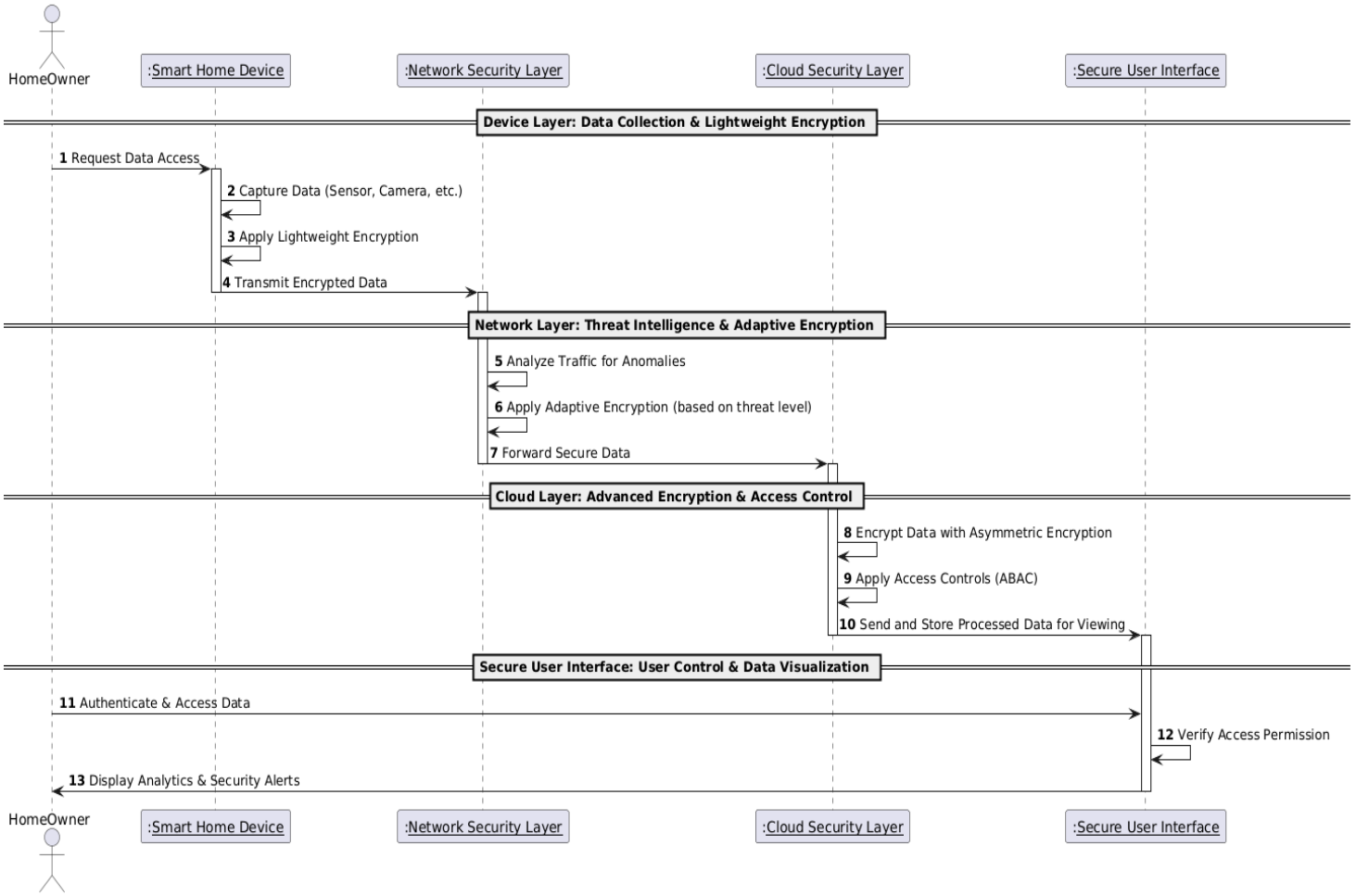


Figure 4.3: Sequence Diagram for AMLE

4.4.3 Entity-Relationship Diagram (ERD)

The ERD diagram of the AMLE is illustrated in Figure 4.4 below. It shows the different attributes, entities and relationships between the different components of this model. It defines the database schema connecting users, devices, sessions, encryption layers, and security logs. Each user is linked to one or more devices, and each device has associated encryption states and logs. This structure ensures traceability of actions and encryption decisions throughout the system lifecycle.

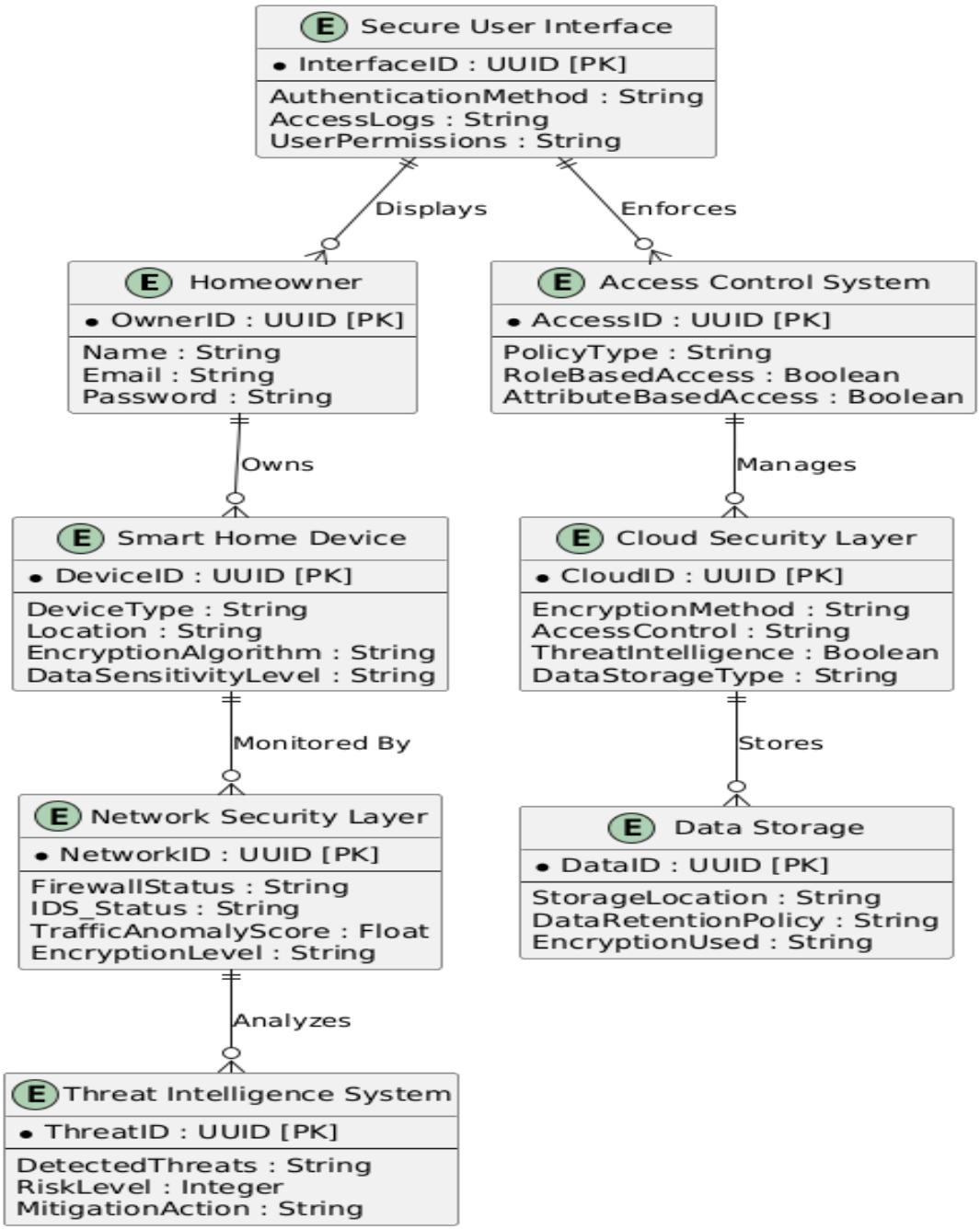


Figure 4.4: ERD diagram for AMLE

4.4.4 System Flow Diagram

Illustrated in Figure 4.5 is the system flow chart for the AMLE System. It starts with data capture from the primary sensors of the IoT smart home ecosystem. If a network anomaly is detected, the flow diverts to trigger dynamic encryption escalation. The cloud layer handles aggregated encryption management and access control (via ABAC). This diagram serves as the overarching view of how adaptive multi-layer encryption is coordinated across smart home devices and infrastructure.

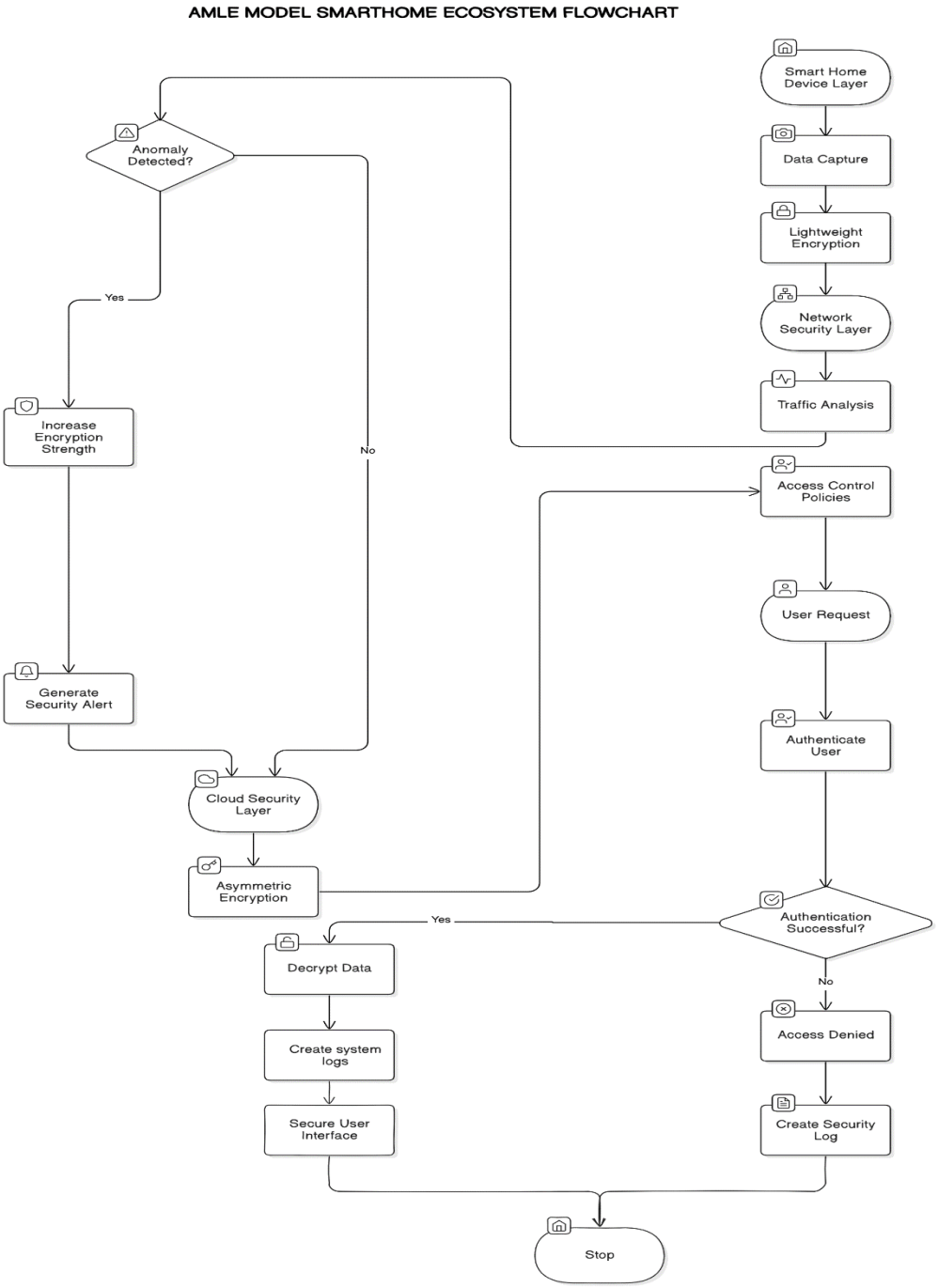


Figure 4.5: AMLE System Flowchart

4.5 Conclusion

This chapter detailed the system design of the AMLE model, including the functional and non-functional requirements, system architecture, and design diagrams. The design incorporates multi-layer encryption across the device, network, and cloud layers, with a secure user interface for authorized access. The use of various design diagrams, such as use case, sequence, ERD, and system flow diagrams, provided a clear and comprehensive representation of the system's structure and interactions.

Chapter 5. Systems Implementation and Testing

5.1 Introduction

This chapter covers the implementation and testing processes of the developed Adaptive Multi-layer Encryption system. This chapter presents the implementation of the proposed system in a clear, sequential, step-by-step manner. Each step is described in detail to ensure transparency and reproducibility of the process. The system is designed to address encryption vulnerabilities in smart home IoT ecosystems through a combination of multiple cryptography layers that adjust in real-time as per the security threat. The implementation incorporates the securement of devices, traffic encryption within the network, and cloud-based access minimization to ensure holistic data security.

5.2 Description of the Implementation Environment

5.2.1 Software Specifications

The system architecture utilized the following key technologies:

a. **Database Management**:

i. SQLAlchemy 2.0.23 for ORM-based database operations, providing type safety and efficient query construction

ii. SQLite for local development and testing, with migration paths to PostgreSQL for production environments

iii. Alembic for database schema migrations and version control

b. **Web Framework and API Development**:

i. FastAPI 0.104.1 for high-performance API development with automatic OpenAPI documentation

ii. Pydantic 1.10.7 for data validation, serialization, and settings management

iii. JWT-based authentication with role-based access control mechanisms

c. **Server Infrastructure**:

i. Uvicorn 0.24.0 as the ASGI server, providing high-throughput asynchronous request handling

ii. Prometheus client 0.19.0 for metrics collection and performance monitoring

iii. Structured logging with python-json-logger 2.0.7 for comprehensive system observability

d. **Security and Encryption**:

i. Cryptography 41.0.5 for industry-standard encryption operations and key management

ii. PyJWT 2.8.0 for secure token generation and validation

iii. PyOTP 2.9.0 for two-factor authentication implementation

iv. Bcrypt 4.0.1 for secure password hashing

e. **Data Analysis and Machine Learning**:

i. NumPy 1.24.3 and Pandas 2.0.3 for efficient data manipulation and analysis

ii. Scikit-learn 1.3.0 for implementing anomaly detection and threat classification algorithms

iii. Python-dateutil for time-series analysis of security events

The project was structured in a modular manner with clear separation between device, network, and cloud layers, following domain-driven design principles. Each layer was implemented as a separate package with well-defined interfaces, enabling independent testing and deployment.

Visual Studio Code was used as the primary integrated development environment (IDE), providing enhanced coding efficiency through features such as:

i. IntelliSense for Python with type checking

ii. Integrated terminal for command execution

iii. Git integration for version control

iv. Debugging tools with breakpoint support

v. Extension support for Python linting (flake8), formatting (black), and type checking (mypy)

The development workflow incorporated continuous integration practices with automated testing using pytest, ensuring code quality and regression prevention throughout the development lifecycle. Below shows some of the tools in the requirements.txt as shown in Figure 5.1.

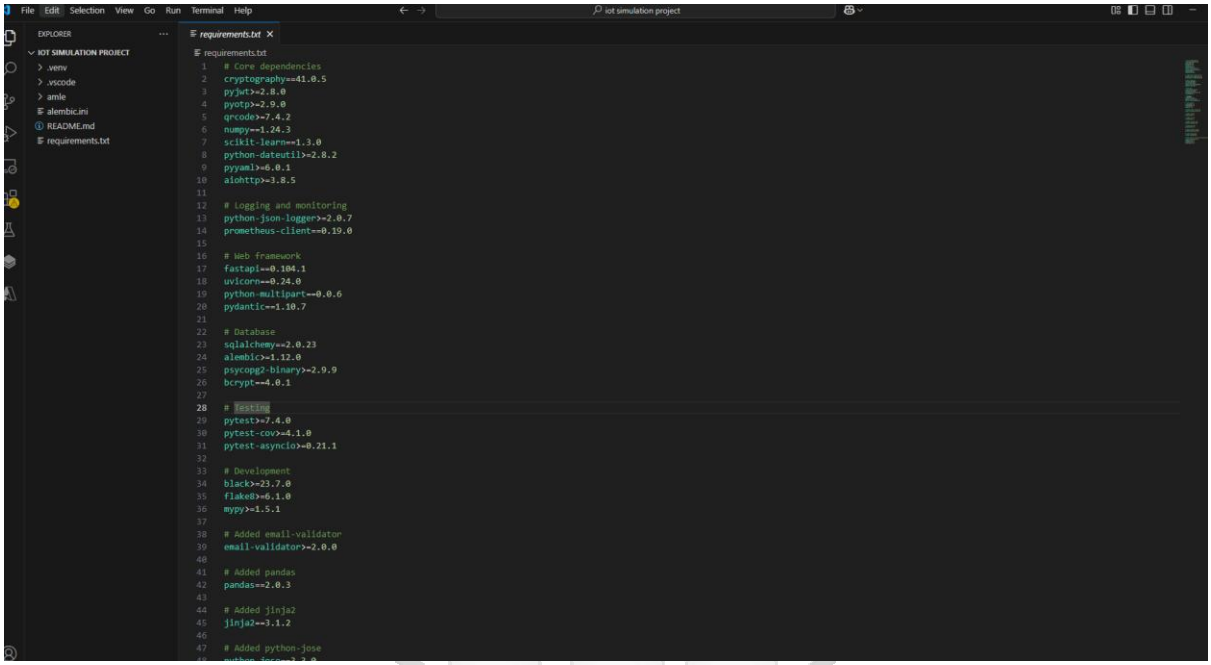


Figure 5.1: Visual Studio Code Environments and Project Dependencies

5.2.2 Hardware Specifications

The hardware equipment used helped test and run the system operations smoothly. Development was performed on a Windows 11 machine equipped with specific hardware specifications:

i. An Intel i7 multi-core processor to facilitate parallel processing capabilities during encryption operations.
ii. 16GB RAM provided seamless data handling without memory constraints.
iii. A solid-state drive (SSD) enhanced data retrieval speed and overall system performance.

The testing environment included multiple IoT device simulations that matched actual smart home devices with distinct processor power and security needs as shown in Figure 5.2.

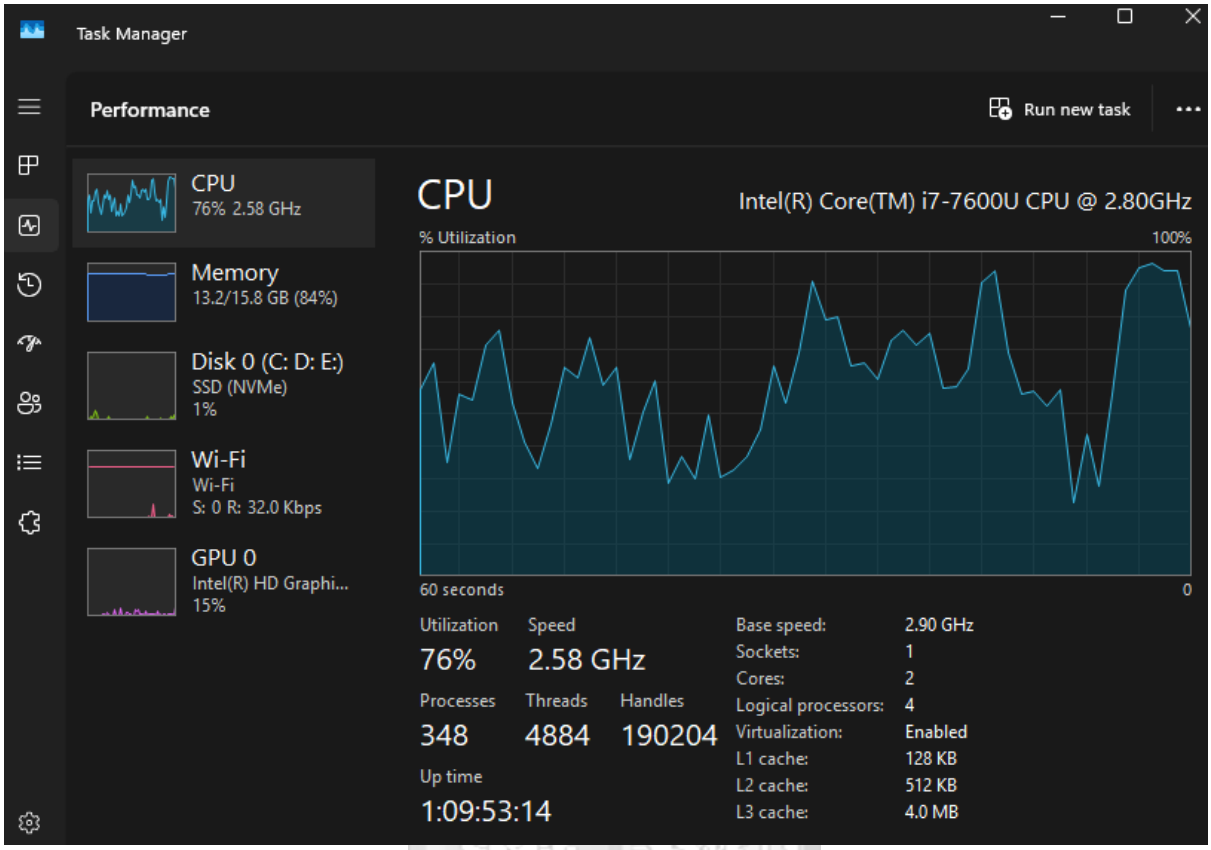


Figure 5.2: Device properties used in project simulation and running

5.3 System Implementation

5.3.1 Core Components

The AMLE system implementation follows a layered architecture consisting of:

i. Device Layer: It implemented device_manager.py as shown in Figure 5.3, this component handles device registration, monitoring, and encryption at the device level. It maintains device status and manages encryption keys for individual devices in the system.

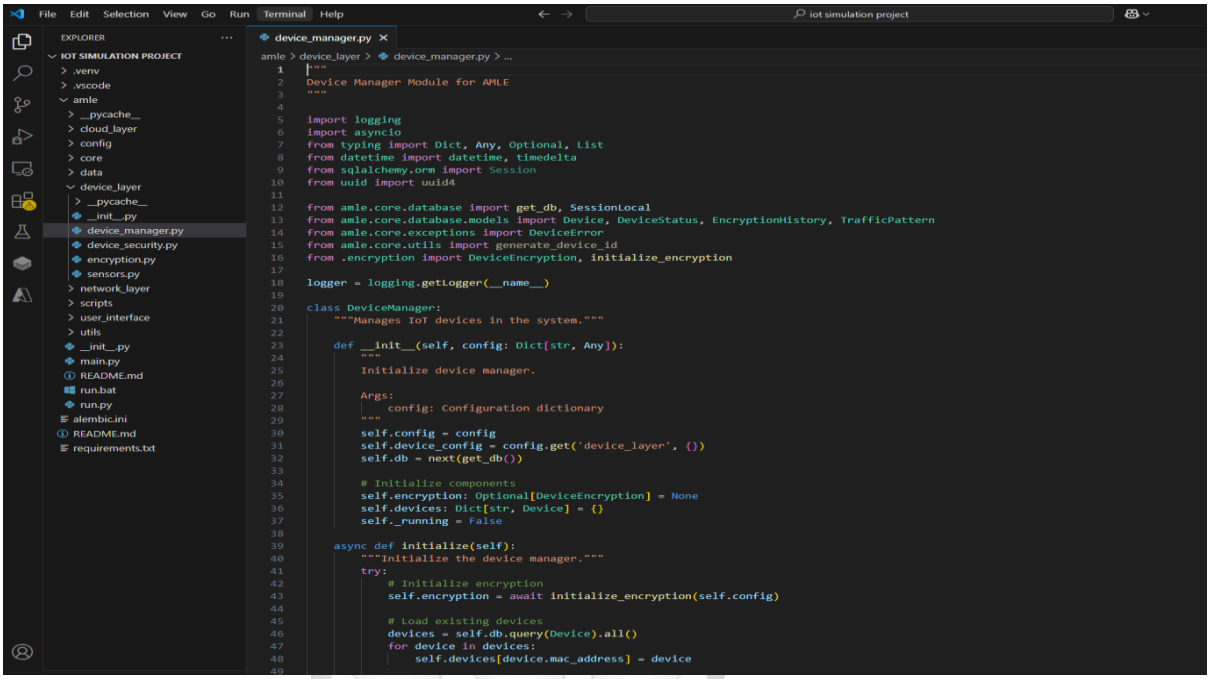


Figure 5.3: Device Layer Management Python Script

ii. Network Layer: This component analysed traffic patterns, managed network-level encryption, and provided secure routing between devices as shown in Figure 5.4.

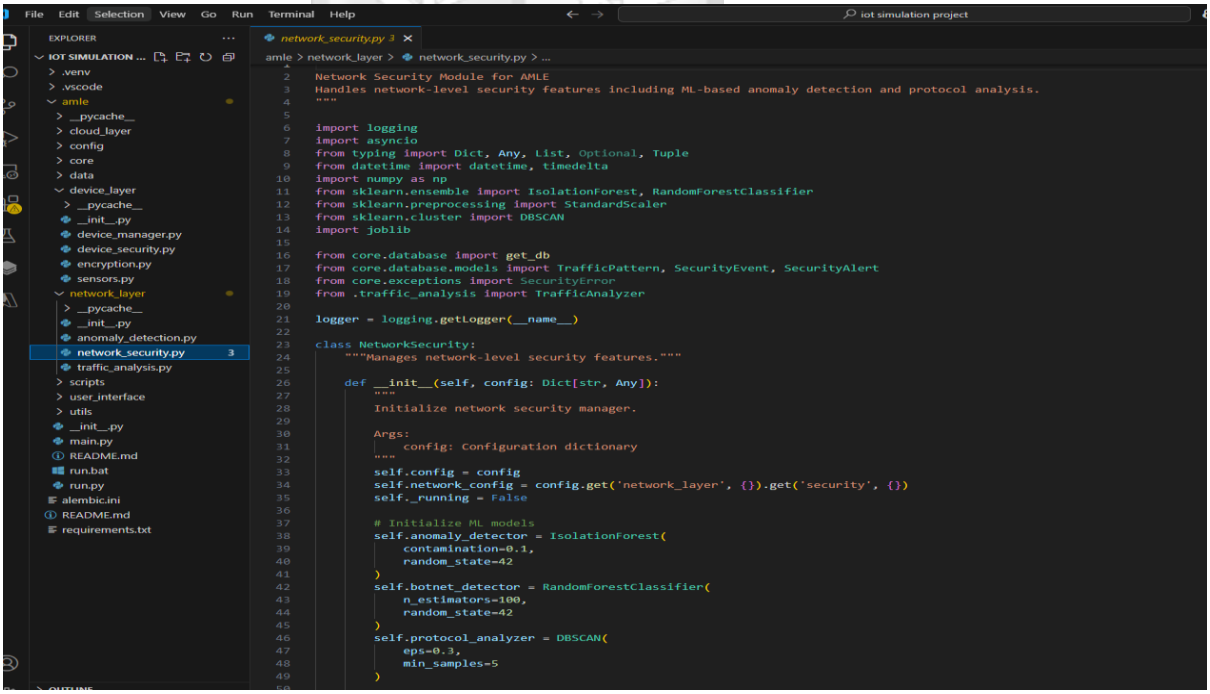


Figure 5.4: Network Layer Security Component

iii. Cloud Layer: Implements attribute-based access control, manages cloud-based encryption, and provides secure storage for device data as shown in Figure 5.5.

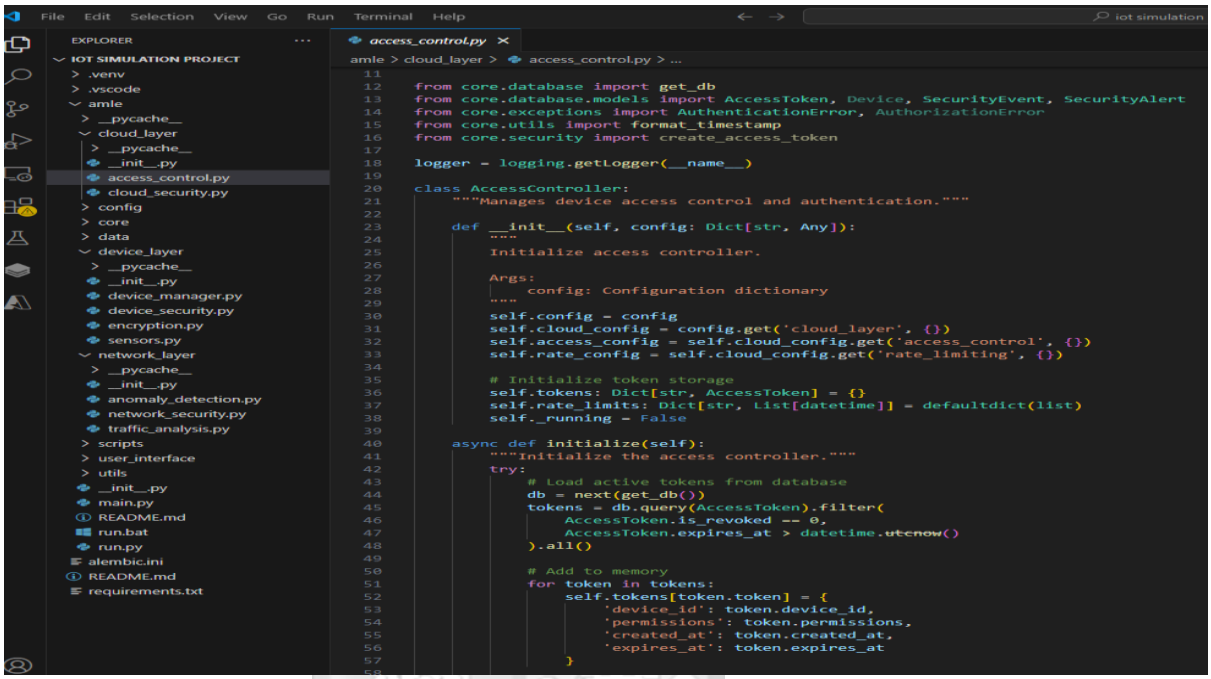


Figure 5.5: Cloud Layer Access control Module

iv. Database Layer: Utilizes SQLite with SQLAlchemy ORM to store device information, encryption histories, and traffic patterns.

v. API Layer: Implements RESTful endpoints for system interaction, encryption statistics retrieval, and device management as shown in Figure 5.6.

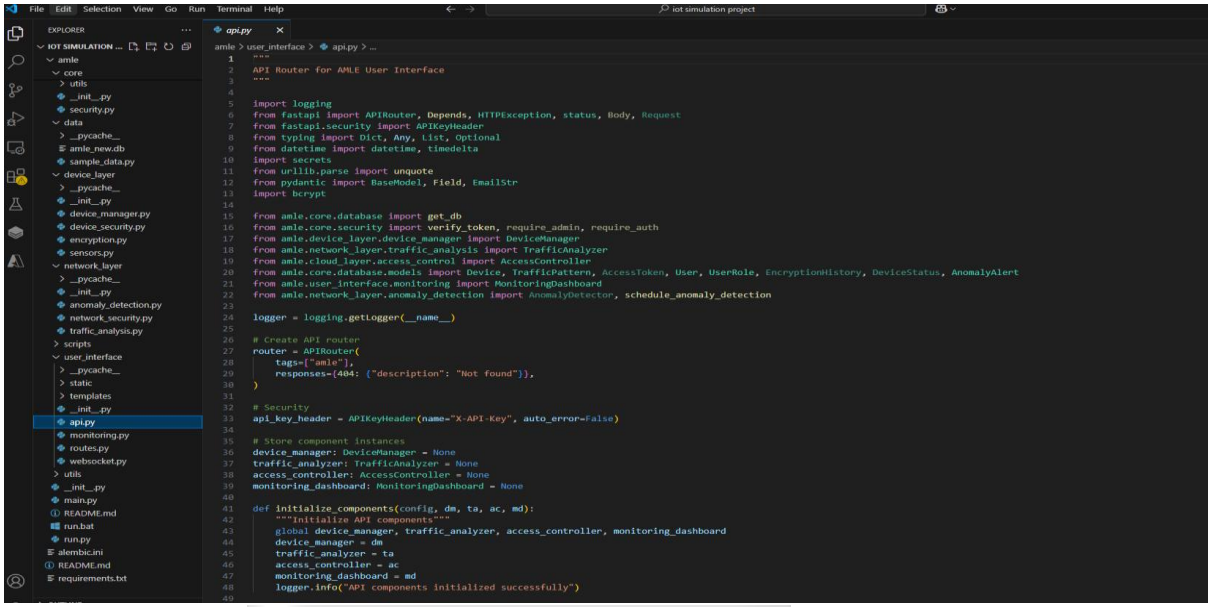


Figure 5.6: API Script to allow for Visualization

vi. **User Interface:** Provides a dashboard for monitoring encryption status, device health, and security metrics as shown in Figure 5.7.

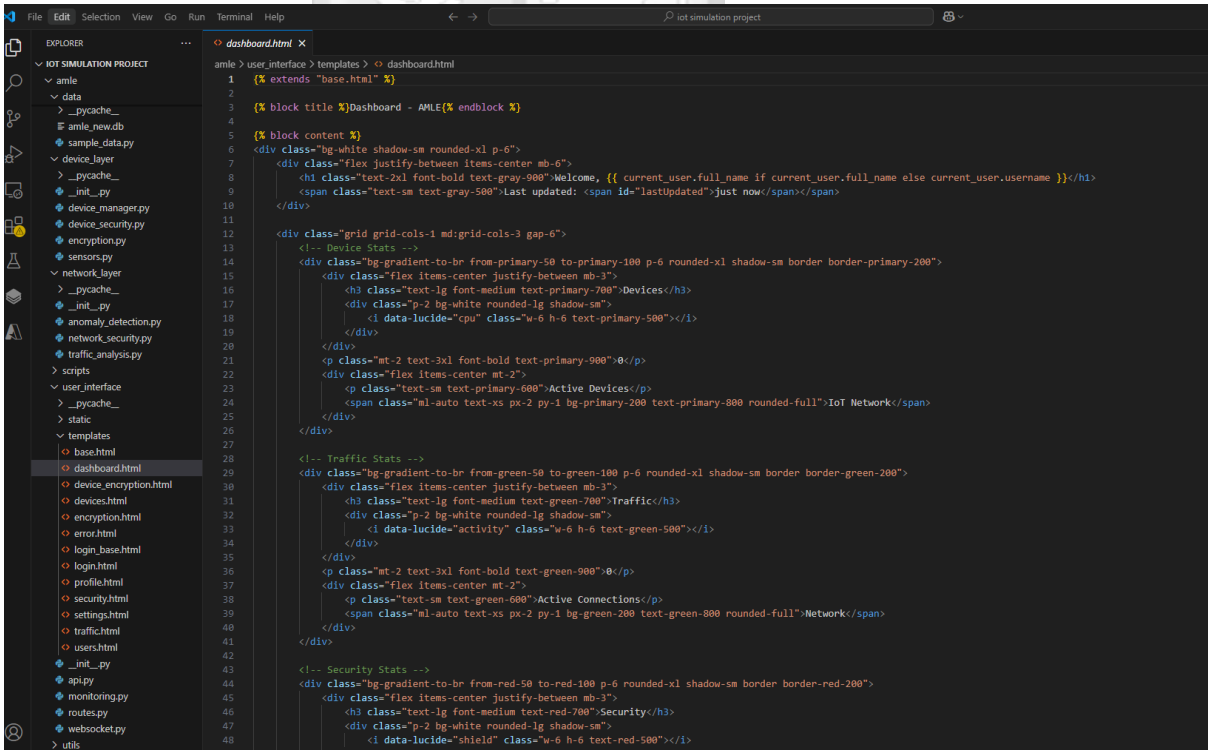


Figure 5.7: Dashboard.html Script to visualize and show the dashboard

5.3.2 Database Schema

The database schema was implemented to support the AMLE system's requirements for storing device information, encryption states, and security metrics. The schema includes tables for:

i. Devices: Stores device metadata, encryption settings, status, firmware information, and security assessment.
ii. TrafficPattern: Tracks network traffic patterns for anomaly detection, including encryption status and risk levels.
iii. SecurityEvent: Records security-related events with severity and description.
iv. SecurityAlert: Stores security alerts with type, severity, and status information.
v. User: Manages system administrators and access control with roles and permissions.
vi. AccessPolicy: Defines access permissions for devices and users.
vii. SecurityScore: Tracks security metrics for devices across different categories.
viii. EncryptionHistory: Records encryption algorithm changes and key rotations.
ix. AnomalyAlert: Stores detected anomalies with scores, types, and resolution status.
x. Vulnerability: Tracks device vulnerabilities with CVE IDs and patch status.
xi. PatchHistory: Records firmware and security patches applied to devices.
xii. DeviceBaselineCompliance: Measures device compliance against security baselines.
xiii. CloudStorage: Manages cloud storage information and encryption settings.
xiv. CloudData: Tracks data stored in the cloud with encryption details.
xv. EncryptionKeys: Stores encryption keys with type and rotation information.
xvi. ComplianceAudit: Records compliance audits against security standards.

5.4 Dataset for Anomaly Detection

5.4.1 Dataset Description

The anomaly detection component of the system collects real-time data from the IoT devices in the network to monitor device behaviour. They are trained on a wide variety of information including device traffic patterns of bytes sent/received per device, number of packets, time of connection, number of times it is communicating. It also includes metrics related to encryption, such as what types of encryption algorithms are being used, key size, and how many days have passed since the last key rotation. It also monitors connection patterns, logging the number of connections, average packet size, and connection count. Through this variance of measurements, system creates a multi-dimensional dataset that characterize the standard operational behaviour of the IoT devices. The anomaly detection model continues to learn and

adapt to emerging patterns as new data is collected, leading to improved accuracy and effectiveness in detecting underlying security threats.

5.4.2 Feature Selection

Our anomaly detection model selected features based on IoT security domain knowledge plus actual measurement data which helps find all important indicators related to security threats. Data transmission statistics about traffic flow assist in finding unusual network behaviour. Average packet size, connection length and network connections help us observe uncommon usage patterns. The system tracked encrypted communication safety by monitoring factors like encryption types, encryption key length, and the number of days until the next key refresh. Each feature in the solution focuses on detecting what devices normally do to catch security problems linked to abnormal traffic patterns, unintended connections, and encryption issues.

5.4.3 Data Pre-processing

The raw IoT data needs specific preparation steps before the anomaly detection system can use it. First the StandardScaler transform normalizes features to make all attributes equally important for Isolation Forest distance measures. This step stops numerical attributes with wide value ranges from taking control of the analysis. The system combines traffic data into fixed time slots instead of using single moment measurements. Transitions between states show real activity patterns so the system detects security threats better.

5.5 Anomaly Detection Implementation

5.5.1 Isolation Forest Algorithm

The system utilizes the Isolation Forest algorithm to detect anomalous entries in a high-dimensional dataset using only unlabelled samples for training. It includes many elements that work together to analyse effectively. It contains all aspects of the Isolation Forest model, including training and saving it. We perform our analysis by pulling out the key insights from the traffic and device information that is captured in the database. The Isolation Forest decomposes extracted features during model training until isolation is achieved at a level of 0.05, meaning 5% of data remains untreated. After training, this system monitors incoming data, flagging anomalous device behaviour. Anomalies are recorded in the database with severity ratings as shown in Figure 5.8.

The system performs initial transformations on the collected device data before anomaly detection. We use StandardScaler to normalize the data to make sure that all features have same scale and thus help Isolation Forest algorithm to work effectively. Furthermore, flow data

is summed over discrete time periods to capture daily cycles of usage rather than basing all measurements on real-time alone.

```python
class AnomalyDetector:
    """
    Anomaly detection using Isolation Forest to identify suspicious patterns
    in device behavior and network traffic.
    """

    def __init__(self, model_path=None):
        """
        Initialize the anomaly detector.

        Args:
            model_path (str, optional): Path to a pre-trained model file.
        """
        self.model = None
        self.scaler = StandardScaler()
        self.model_path = model_path or "amle/models/isolation_forest.joblib"
        self.load_or_create_model()

    def load_or_create_model(self):
        """Load a pre-trained model or create a new one if none exists."""
        try:
            if os.path.exists(self.model_path):
                self.model = joblib.load(self.model_path)
                logger.info(f"Loaded anomaly detection model from {self.model_path}")
            else:
                # Create a new model with default parameters
                self.model = IsolationForest(
                    n_estimators=100,
                    contamination=0.05,
                    random_state=42
                )
                logger.info("Created new anomaly detection model")
        except Exception as e:
            logger.error(f"Error loading/creating model: {e}")
            # Fallback to a new model
            self.model = IsolationForest(n_estimators=100, contamination=0.05, random_state=42)
```

Figure 5.8: Isolation Forest Model Usage load or creation

5.5.2 Hyperparameter Selection

Through empirical testing, we determined these hyperparameters for the Isolation Forest model, which all ran optimally. The model is set to utilize 100 estimators (n_estimators), providing a balance between accuracy and computational efficiency. contamination = 0.05 # expected proportion of anomalies in the data Furthermore, random_state=42 guarantees that the results we get here are reproducible. These parameters were selected to allow for a balance between finding real anomalies and avoiding false positives as shown in Figure 5.9.

```python
# Create a new model with default parameters
self.model = IsolationForest(
    n_estimators=100,
    contamination=0.05,
    random_state=42
```

Figure 5.9: Isolation Forest Parameter Selection

5.5.3 Integration with System Architecture

The anomaly detection component was integrated into the AMLE system as follows:

i. Scheduled Execution: The schedule_anomaly_detection function runs at system start-up and at regular intervals (every 15 minutes) to detect anomalies as shown in Figure 5.10.

```python
@repeat_every(seconds=60 * 15)  # Run every 15 minutes
async def scheduled_anomaly_detection():
    """Run anomaly detection on a schedule."""
    try:
        db = next(get_db())
        anomalies = schedule_anomaly_detection(db)
        logger.info(f"Scheduled anomaly detection completed. Found {len(anomalies)} anomalies.")
    except Exception as e:
        logger.error(f"Error during scheduled anomaly detection: {e}")
```

Figure 5.10: Scheduled Anomaly Detection script

ii. Database Integration: Detected anomalies are stored in the AnomalyAlert table, linked to the corresponding device records as shown in Figure 5.11.

```python
def _create_alert(self, db: Session, device_id: int, score: float):
    """
    Create an anomaly alert in the database.

    Args:
        db (Session): Database session
        device_id (int): ID of the device with anomalous behavior
        score (float): Anomaly score from the model
    """
    try:
        # Get device details
        device = db.query(Device).filter(Device.id == device_id).first()
        if not device:
            return

        # Create alert
        alert = AnomalyAlert(
            device_id=device_id,
            timestamp=datetime.utcnow(),
            score=float(score),
            description=f"Anomalous behavior detected for device {device.name}",
            status="New"
        )

        db.add(alert)
        db.commit()

        logger.info(f"Created anomaly alert for device {device.name} with score {score}")
    except Exception as e:
        db.rollback()
        logger.error(f"Error creating anomaly alert: {e}")
```

Figure 5.11: Database Storage script for anomaly detection

iii. API Endpoints: RESTful API endpoints were added to allow querying of anomaly data and triggering manual detection runs as shown in Figure 5.12.

```python
@router.get("/v1/anomalies", response_model=List[dict])
async def get_anomalies():
    """
    Get a list of detected anomalies.
    """
    try:
        db = next(get_db())
        # Get recent anomaly alerts from database
        alerts = db.query(AnomalyAlert).order_by(AnomalyAlert.timestamp.desc()).limit(50).all()

        result = []
        for alert in alerts:
            device = alert.device
            result.append({
                "id": alert.id,
                "device_id": alert.device_id,
                "device_name": device.name if device else "Unknown",
                "timestamp": alert.timestamp,
                "score": alert.score,
                "description": alert.description,
                "status": alert.status
            })

        return result
    except Exception as e:
        logger.error(f"Error retrieving anomalies: {e}")
        raise HTTPException(status_code=500, detail=str(e))
```

Figure 5.12: Anomaly API dB GET querying

iv. Dashboard Integration: The security dashboard was extended to display detected anomalies with their scores and statuses as shown in Figure 5.13.

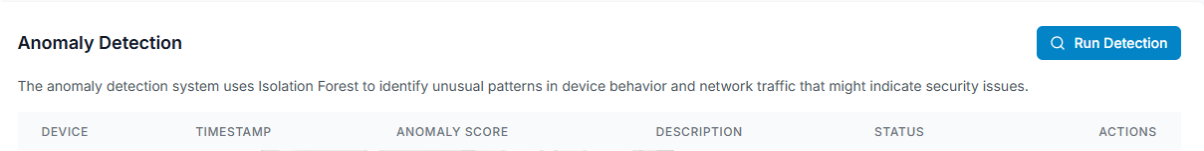


Figure 5.13: Anomaly Detection Dashboard Snippet in Security

5.6 Encryption Management Implementation

5.6.1 Device Registration and Management

The implementation of device registration and management focused on securely onboarding devices and maintaining their status. The register_device method validates device information before adding it to the database, while the update_device_status method handles state transitions between active, offline, and compromised states.

5.6.2 Encryption Management

The encryption management subsystem implements adaptive encryption capabilities that adjust based on device capabilities, threat levels, and security requirements. The system supports multiple encryption algorithms including AES, RSA, and ECC with configurable key sizes. Key rotation is implemented through a time-based schedule combined with threat-based triggers, enhancing security by limiting the exposure period of encryption keys.

The AMLE system implements a multi-layer encryption approach that dynamically adjusts security levels based on threat intelligence. The implementation consists of:

i. Encryption Level Classification: The system defines four encryption levels (LOW, MEDIUM, HIGH, VERY_HIGH) based on algorithm strength, key size, and rotation frequency.

ii. Dynamic Encryption Assignment: Devices are assigned appropriate encryption levels based on their capabilities, sensitivity of data, and current threat level.

iii. Adaptive Response: The system automatically elevates encryption levels when anomalies are detected through the Isolation Forest algorithm.

iv. Key Management: Per-device keys stored securely with automatic rotation schedules based on device sensitivity as shown in Figure 5.14.

```
self.rotation_intervals = {
    EncryptionLevel.LOW.value: 90,
    EncryptionLevel.MEDIUM.value: 60,
    EncryptionLevel.HIGH.value: 30,
    EncryptionLevel.VERY_HIGH.value: 15
}
```

Figure 5.14: Key rotation script in adaptive encryption

v. Status Monitoring: Continuous monitoring of encryption status with alerts for outdated keys or weak algorithms as shown in Figure 5.15.

```
def determine_encryption_level(self, device: Device, threat_level: str) -> str:
    """
    Dynamically determine the appropriate encryption level based on
    device capabilities and current threat level.

    Args:
        device: Device object
        threat_level: Current threat level (low, medium, high)

    Returns:
        str: Appropriate encryption level
    """
    # Get base security level
    base_level = self._get_base_security_level(device)

    # Map threat level to encryption level
    threat_encryption_level = {
        "low": EncryptionLevel.LOW.value,
        "medium": EncryptionLevel.MEDIUM.value,
        "high": EncryptionLevel.HIGH.value
    }.get(threat_level, EncryptionLevel.LOW.value)

    # Determine if device can handle the required encryption level
    device_max_level = self._get_device_max_capability(device)

    # Take the max of base level and threat level, but cap at device capability
    required_level = max(base_level, threat_encryption_level)
    final_level = min(required_level, device_max_level)

    logger.info(f"Device {device.name} - Base: {base_level}, Threat: {threat_encryption_level}, "
                f"Max capability: {device_max_level}, Final: {final_level}")

    return final_level
```

Figure 5.15: Adaptive Encryption model for AMLE model

5.6.3 Dashboard Implementation

The dashboard shows current encryption status and detects anomaly behaviour so users can see their IoT network security status in real-time. The system tracked encryption information from multiple parts of the network by showing device protection rates, encryption types assigned to devices, key rotation counts, device-specific security data and attack alerts with their impact levels. The dashboard uses HTML, CSS Tailwind, and JavaScript to display dynamic real-time data from its API connections.

The Figure 5.16 below illustrates as the main Dashboard

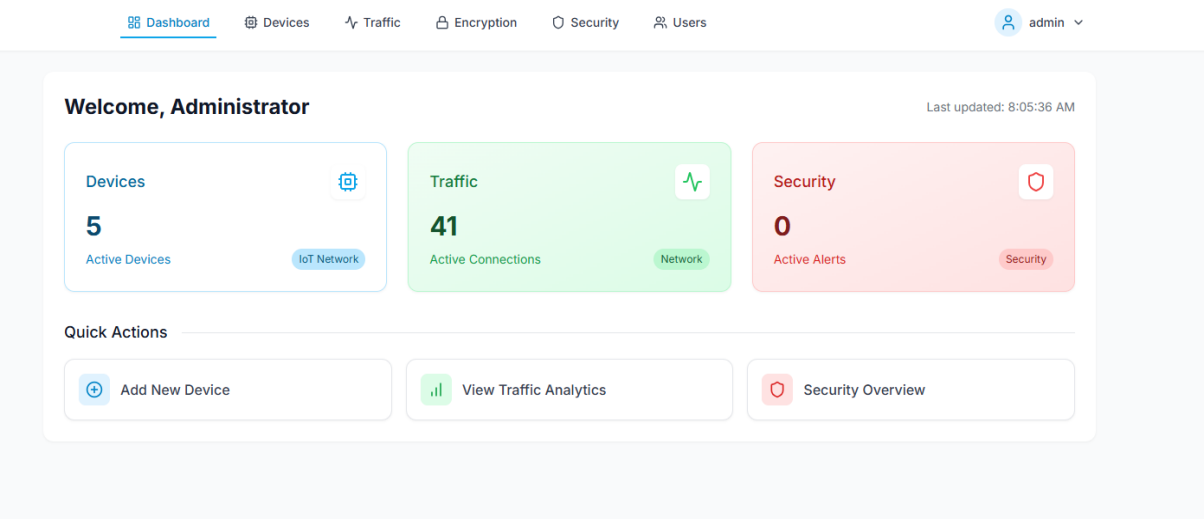


Figure 5.16: Dashboard User Interface

The Figure 5.17 shows the dashboard to retrieve system devices

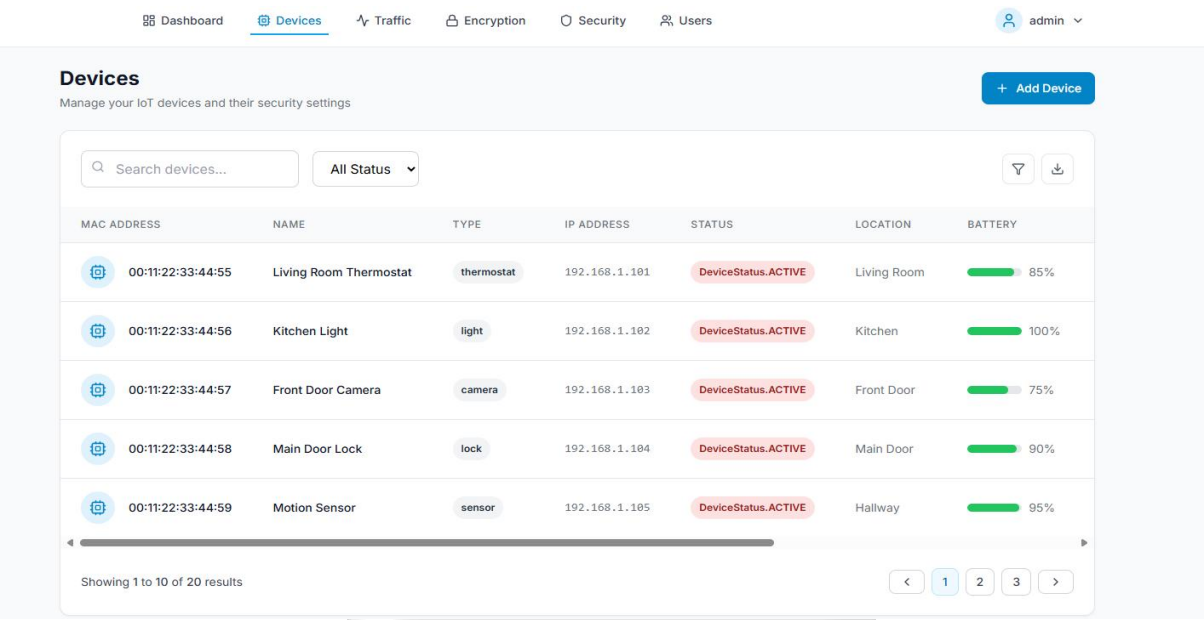


Figure 5.17: Devices Dashboard

The Figure 5.18 shows the traffic analysis dashboard to illustrate different traffic metrics

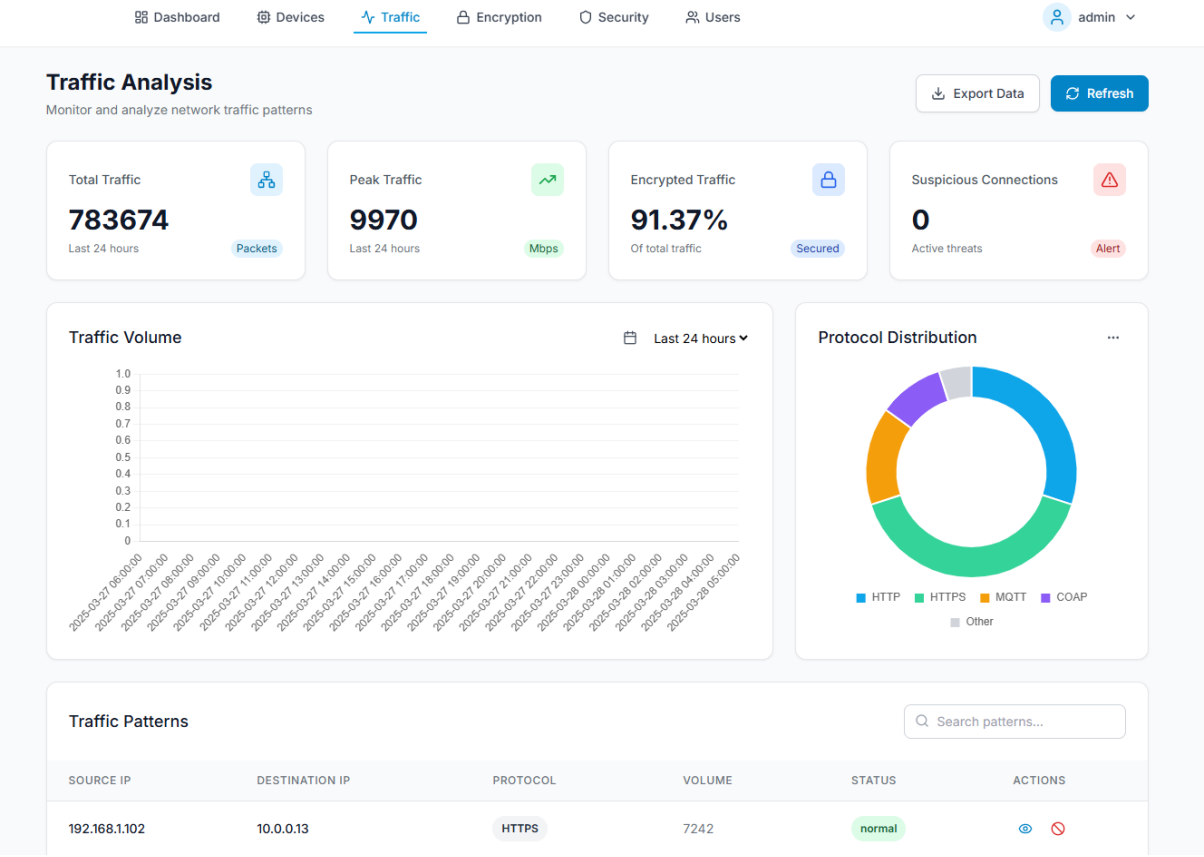


Figure 5.18: Traffic analysis Dashboard

The Figure 5.19 shows encryption analysis to show system encryption models

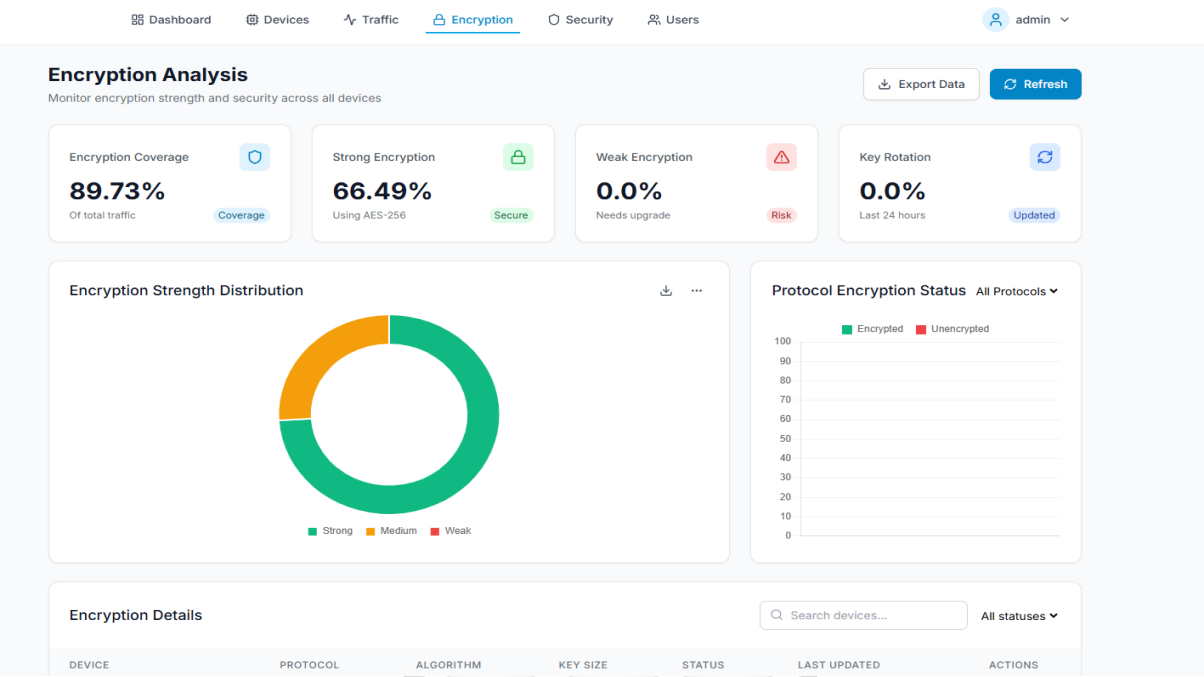


Figure 5.19: Encryption Dashboard

The Figure 5.20 dashboard shows security monitoring dashboard

Figure 5.20: Security Monitoring Dashboard

The Figure 5.21 below shows the user details dashboard in the admin panel

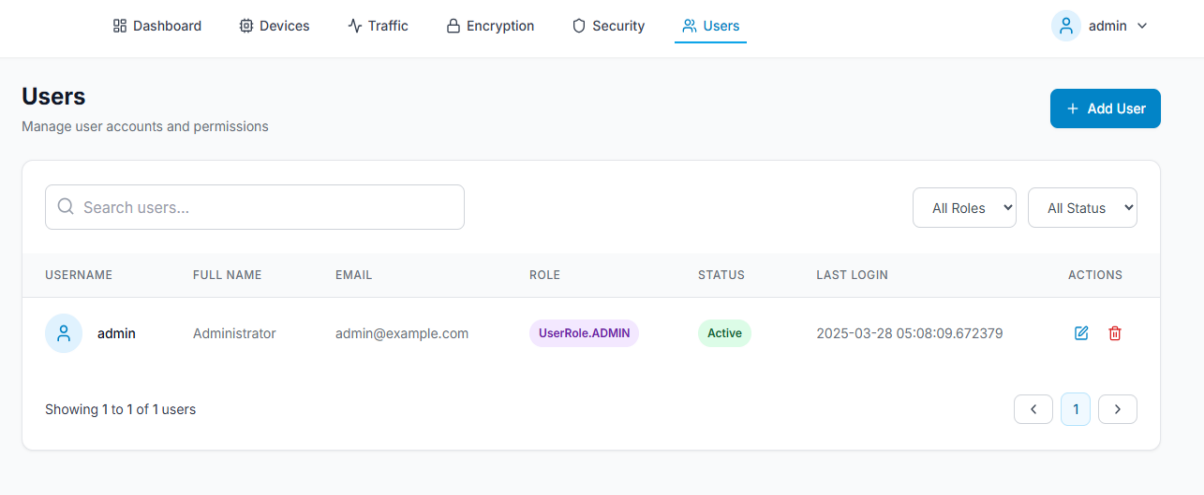


Figure 5.21: User Dashboard

5.6.4 Network Layer Encryption

The network layer employs protocol-specific encryption mechanisms:

i. Protocol Adaptation: Implementation of security wrappers for common IoT protocols (MQTT, CoAP, HTTP) as highlighted in Figure 5.22.

ii. TLS/DTLS Integration: Secure transport layer with certificate validation and Perfect Forward Secrecy.

iii. Traffic Encryption: End-to-end payload encryption for sensitive communications, independent of transport security.

```python
class SecurityLevel(Enum):
    """Security levels for protocol communications."""
    NONE = "none"  # No encryption (not recommended)
    BASIC = "basic"  # Basic encryption
    STANDARD = "standard"  # Standard encryption (TLS 1.2)
    HIGH = "high"  # High security (TLS 1.3 with strong ciphers)
    CUSTOM = "custom"  # Custom security settings

class ProtocolAdapter:
    """
    Adapter for handling different IoT communication protocols with
    appropriate security measures.
    """

    def __init__(self, config: Dict[str, Any]):
        """
        Initialize protocol adapter.

        Args:
            config: Configuration dictionary
        """
        self.config = config
        self.protocol_handlers = {
            ProtocolType.HTTP: self._handle_http,
            ProtocolType.MQTT: self._handle_mqtt,
            ProtocolType.COAP: self._handle_coap,
            ProtocolType.UDP: self._handle_udp,
            ProtocolType.TCP: self._handle_tcp,
            ProtocolType.WEBSOCKET: self._handle_websocket
        }
        self.security_configs = self._init_security_configs()
```

Figure 5.22: Protocol adapter in Network Layer

5.6.5 Cloud Layer Encryption

Cloud-level encryption implements advanced security mechanisms:

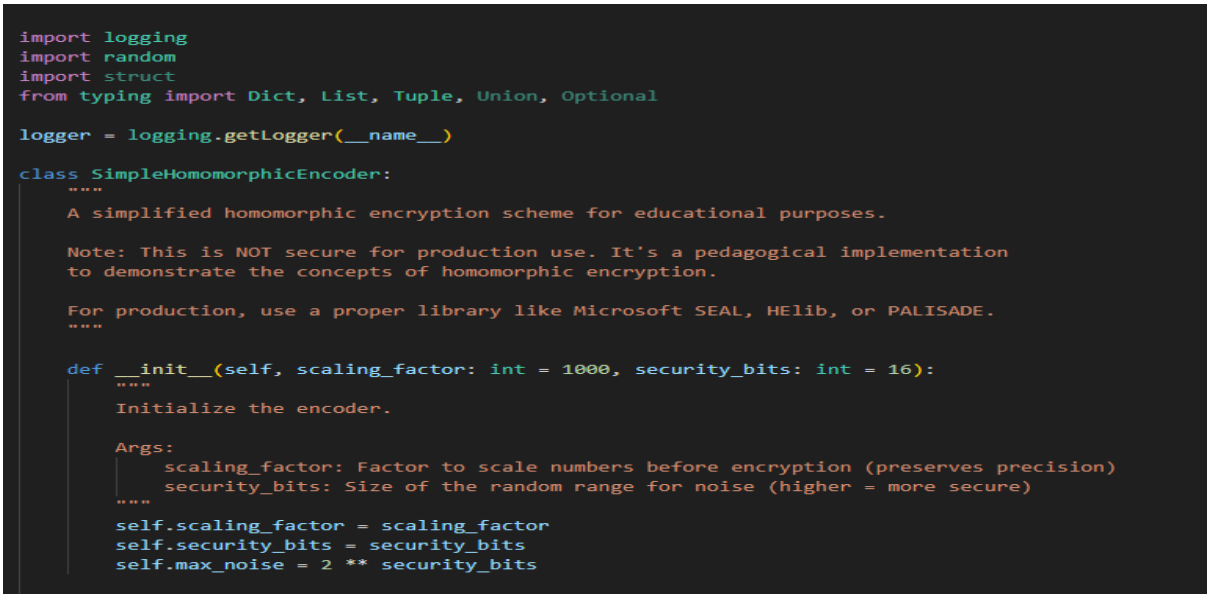
i. Attribute-Based Access Control: Implementation of ABAC policies controlling data access based on user, resource, and environmental attributes as shown in Figure 5.23.

```python
import logging
import json
import re
from typing import Dict, Any, List, Optional, Union, Set, Callable
from datetime import datetime, time

logger = logging.getLogger(__name__)

class ABACPolicy:
    """Represents an ABAC policy with attributes, conditions, and effect."""

    def __init__(
        self,
        policy_id: str,
        description: str,
        effect: str,
        subject_attributes: Dict[str, Any],
        resource_attributes: Dict[str, Any],
        action_attributes: Dict[str, Any],
        environment_attributes: Dict[str, Any] = None,
        conditions: List[Dict[str, Any]] = None
    ):
        """
        Initialize an ABAC policy.

        Args:
            policy_id: Unique identifier for the policy
            description: Description of the policy
            effect: 'allow' or 'deny'
            subject_attributes: Attributes of the subject (user/client)
            resource_attributes: Attributes of the resource
            action_attributes: Attributes of the action
```

Figure 5.23: ABAC Implementation in cloud layer

ii. End-to-End Encryption: Authenticated encryption ensuring data privacy between devices and cloud storage.

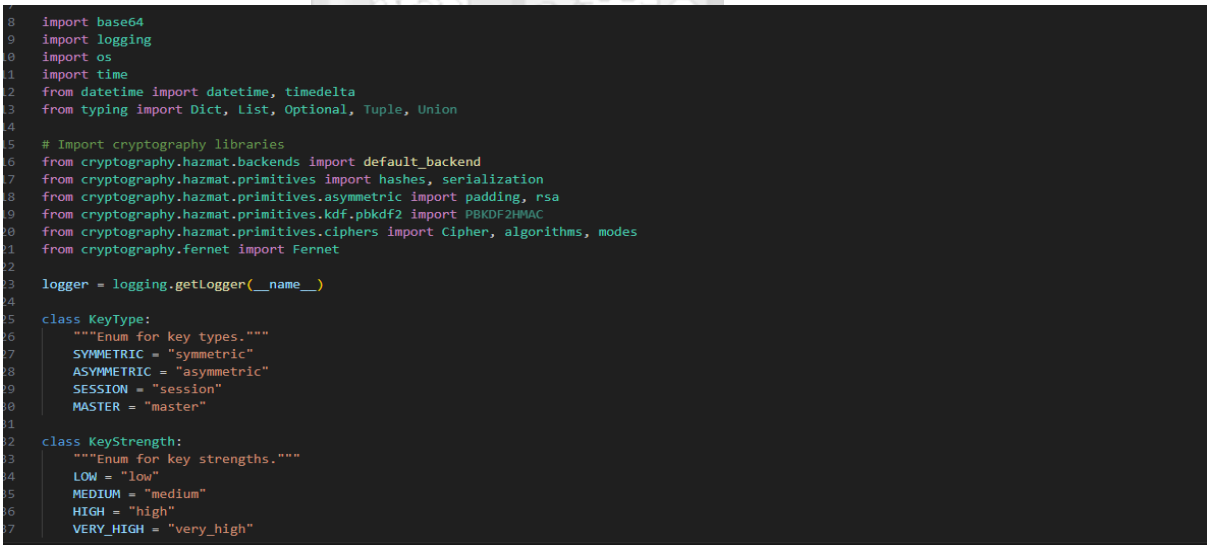
iii. Homomorphic Encryption: Simple implementation allowing basic operations on encrypted data without decryption is shown in Figure 5.24.



```python
import logging
import random
import struct
from typing import Dict, List, Tuple, Union, Optional

logger = logging.getLogger(__name__)

class SimpleHomomorphicEncoder:
    """
    A simplified homomorphic encryption scheme for educational purposes.

    Note: This is NOT secure for production use. It's a pedagogical implementation
    to demonstrate the concepts of homomorphic encryption.

    For production, use a proper library like Microsoft SEAL, HElib, or PALISADE.
    """

    def __init__(self, scaling_factor: int = 1000, security_bits: int = 16):
        """
        Initialize the encoder.

        Args:
            scaling_factor: Factor to scale numbers before encryption (preserves precision)
            security_bits: Size of the random range for noise (higher = more secure)
        """
        self.scaling_factor = scaling_factor
        self.security_bits = security_bits
        self.max_noise = 2 ** security_bits
```

Figure 5.24: Homomorphic implementation in Cloud layer

iv. Key Management System: Centralized key lifecycle management with secure generation, distribution, rotation, and revocation as shown in Figure 5.25.



```python
import base64
import logging
import os
import time
from datetime import datetime, timedelta
from typing import Dict, List, Optional, Tuple, Union

# Import cryptography libraries
from cryptography.hazmat.backends import default_backend
from cryptography.hazmat.primitives import hashes, serialization
from cryptography.hazmat.primitives.asymmetric import padding, rsa
from cryptography.hazmat.primitives.kdf.pbkdf2 import PBKDF2HMAC
from cryptography.hazmat.primitives.ciphers import Cipher, algorithms, modes
from cryptography.fernet import Fernet

logger = logging.getLogger(__name__)

class KeyType:
    """Enum for key types."""
    SYMMETRIC = "symmetric"
    ASYMMETRIC = "asymmetric"
    SESSION = "session"
    MASTER = "master"

class KeyStrength:
    """Enum for key strengths."""
    LOW = "low"
    MEDIUM = "medium"
    HIGH = "high"
    VERY_HIGH = "very_high"
```

Figure 5.25: Key management Script

5.7 Testing Methodology

To thoroughly evaluate the AMLE system, a combination of different testing methodologies was applied. These methodologies focused on verifying individual components, ensuring seamless integration, and validating security and performance metrics.

5.7.1 Unit Testing

Unit tests were implemented to verify the functionality of individual components within the encryption framework. These tests focused on key management functions, encryption and decryption algorithms, and policy enforcement mechanisms. Each module was tested independently to ensure accuracy, reliability, and expected behaviour before being integrated into the overall system example as shown Figure 5.26.

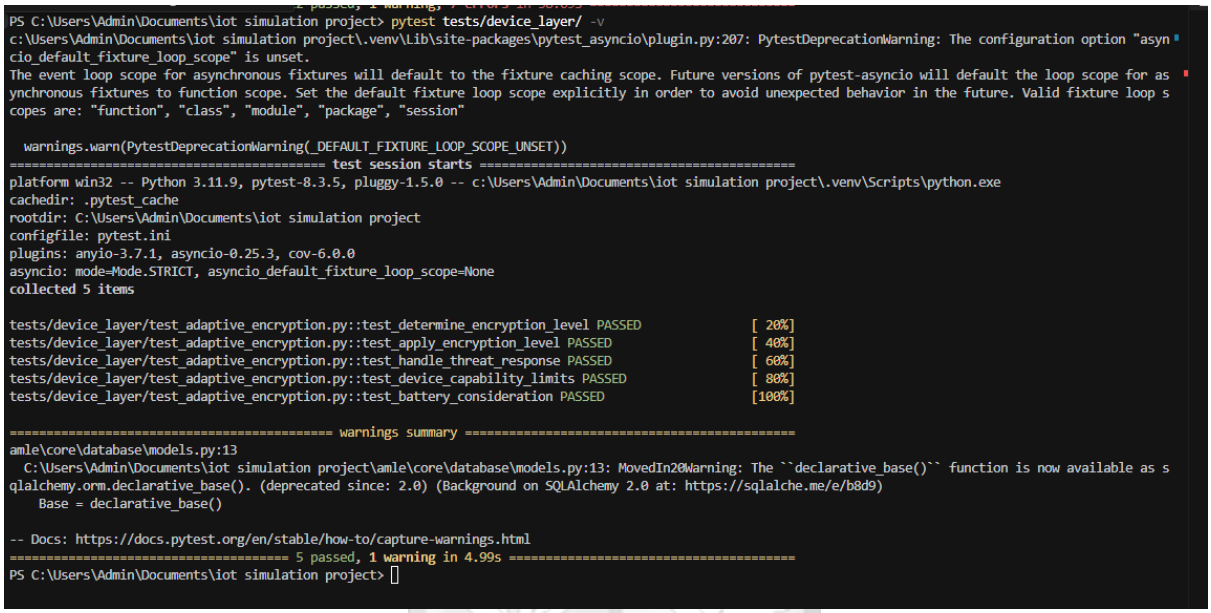


Figure 5.26: Device Layer Testing

5.7.2 Integration Testing

Integration testing was conducted to ensure seamless interaction between the various modules of the AMLE system. The tests focused on validating data flow between key management, authentication, and encryption layers, verifying the implementation of secure communication protocols within the IoT network, and ensuring the correct execution of encryption policies across all security layers.

5.7.3 Functional Testing

Functional testing was carried out to confirm that the system met the defined functional requirements. This involved validating secure key exchange mechanisms, ensuring the correctness of encryption and decryption processes, and enforcing predefined security policies. Test cases were carefully designed to assess expected outcomes across various functional scenarios, ensuring the system operated as intended under different conditions.

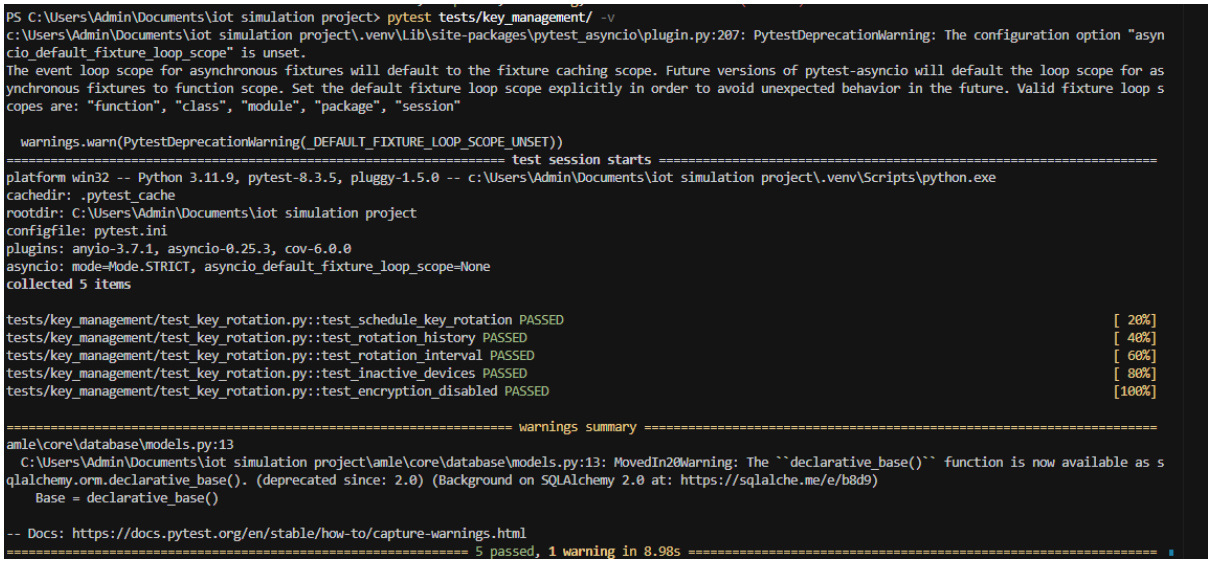


Figure 5.27: Key management Testing

5.7.4 Performance Testing

Performance testing evaluated the system's efficiency under different operational loads. The key focus areas included measuring encryption and decryption speeds across varying IoT network loads, assessing the system's scalability when managing an increasing number of devices, and monitoring resource consumption such as CPU, memory, and bandwidth usage during peak operational conditions. These tests ensured that the system maintained optimal performance without compromising security or responsiveness showed in Figure 5.28.

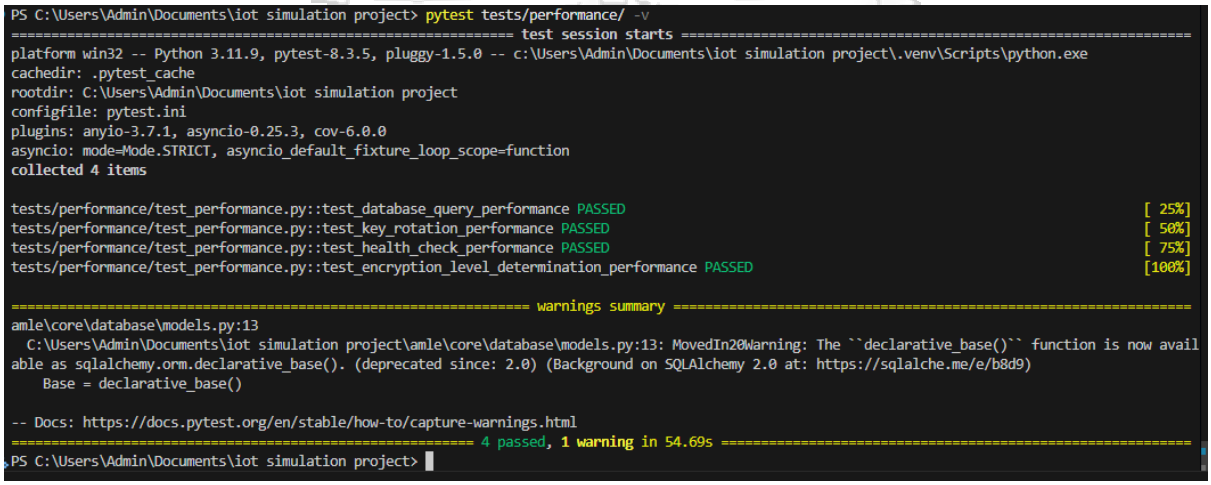


Figure 5.28: Performance Test

5.7.5 Security Testing

Security testing aimed to uncover vulnerabilities and assess the system's resilience against cyber threats. This process involved penetration testing to simulate real-world cyber-attacks and identify potential security gaps, vulnerability assessments using automated security scanners to detect weaknesses, and threat modelling to systematically analyse possible attack vectors. These tests ensured that the AMLE system maintained strong security defences against emerging threats.

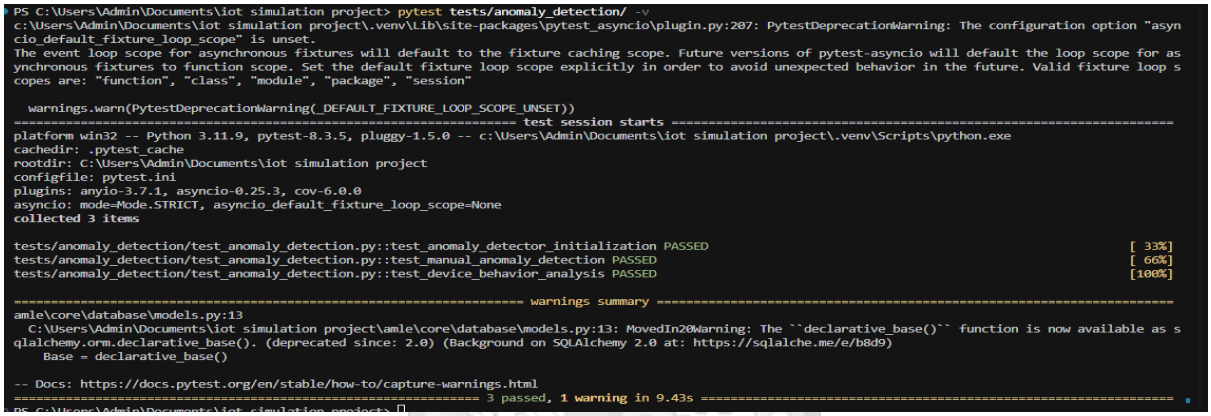


Figure 5.29: Anomaly Detection Testing

5.8 Test Cases and Results

The following test cases were executed to assess the effectiveness and security of the AMLE system. The results confirmed the reliability of the system, with all tests passing successfully as listed in Table 5.1, Table 5.2 and Table 5.3.

5.8.1 Key Management System

Table 5.1: Key Management Test Cases

Test Case	Description	Expected Result	Status
Key Rotation Scheduling	Ensures keys rotate at predefined intervals	Keys should rotate automatically as per schedule	Passed
Retrieval of Rotation History	Checks if rotation logs are stored and accessible	Rotation history should be retrievable	Passed
Rotation Interval Validation	Validates rejection of invalid intervals	System should reject incorrect time intervals	Passed
Inactive Device Handling	Ensures inactive devices do not receive new keys	Key updates should exclude inactive devices	Passed

Encryption Disabling	Tests if key rotation halts when encryption is off	Key rotation should stop when encryption is disabled	Passed

5.8.2 Encryption and Decryption

Table 5.2: Encryption and Decryption Test Case

Test Case	Description	Expected Result	Status
Data Encryption	Ensures data is correctly encrypted	Encrypted data should be unreadable without keys	Passed
Data Decryption	Checks if encrypted data can be decrypted	Decrypted data should match original input	Passed
Invalid Key Detection	Ensures incorrect keys cannot decrypt data	Decryption with wrong keys should fail	Passed
Multi-Layer Encryption	Tests encryption across multiple security layers	Data should be encrypted through layered approach	Passed

5.8.3 System Performance

Table 5.3: System Performance Test Case

Test Case	Description	Expected Result	Status
Encryption Latency	Measures encryption speed	Processing time should be within acceptable range	Passed
Scalability Testing	Tests system performance with many IoT devices	System should handle multiple devices efficiently	Passed
Resource Utilization	Monitors CPU and memory consumption	Resource usage should remain optimal	Passed

5.9 Validation Process

The validation phase ensured that the AMLE system adheres to industry best practices and meets security, usability, and performance expectations.

5.9.1 Compliance with Security Standards

Several accepted security frameworks were used to evaluate the security of the AMLE system to be used a smart home ecosystem. This project adhered to ISO/IEC 27001 (global information security management standard) and aligned to the NIST Cybersecurity Framework (standard for encryption, key management and risk mitigation). This ensures a degree of security, and reliability, establishing the system through these established standards.

5.9.2 User Acceptance Testing (UAT)

User acceptance testing (UAT) was conducted in a controlled smart home IoT simulation environment, wherein different users evaluated the functionality and usability of the system. The key focus areas included:

i. Ease of Use and Intuitiveness: Assessing the user interface and overall system accessibility.

ii. Response Times and Efficiency: Measuring system performance, including encryption and key management processes.

iii. Effectiveness of Encryption Policies: Validating the real-world applicability of security measures in protecting IoT communications.

User feedback was instrumental in refining the system's user experience, optimizing performance, and enhancing security features before deployment.

5.9.3 Threat Modelling and Risk Assessment

A thorough risk assessment was performed through comprehensive threat modelling to identify, evaluate and mitigate potential security risks. The attack surface analysis mapped all potential entry points that an adversary could exploit on the system. We had expected possible spoofing, tampering, botnet attack, information disclosure, denial of service and privilege threats escalation. Threats that were identified were mitigated by implementing risk mitigation strategies, such as enhancements in encryption, access control, and intrusion detection. The established systematic approach-built resilience in the system for cyber threats to ensure a strong secure environment for home IoT.

5.10 System Evaluation

5.10.1 Strengths

i. All layers of the system implemented adaptive encryption.

ii. Real-time monitoring provided immediate visibility into the security state.

iii. Key rotation mechanisms ensured that encryption keys were regularly updated.

iv. Using unlabelled training data, the Isolation Forest algorithm efficiently identified a typical device behaviour.

v. The modular architecture facilitated maintenance and future enhancements.

5.10.2 Limitations

i. If the security of the underlying operating system is compromised, then key storage security is also compromised.

ii. Performance on resource-constrained devices was impacted with high latency.

iii. This implementation supports only a few proprietary IoT protocols.

iv. Anomaly detection requires sufficient baseline data for its effective operation.

5.11 Summary

This chapter described the design and testing of AMLE smart home IoT ecosystems. The adaptive encryption across device, network, and cloud layers allows the system to deploy security over the rest of the project. This implementational use of modern encryption algorithms, isolation forest-based anomaly detection and attribute-based access control to secure the IoT data. Experimental results of the proposed system indicate that it meets design criteria such as true positive in anomaly detection, adaptive encryption to improve security level, and access control resolution. This implementation grounds on the security along with practicality, allowing resource constrained IoT gadgets to report between the smart home ecosystem despite solid practical degradation.

Chapter 6. Discussion, Recommendation and Conclusion

6.1 Introduction

This chapter presents all research project aspects including its objective fulfilment assessment, a detailed interpretation of the key findings, and a reflective appraisal of the study's limitations. This section evaluates how effectively the Adaptive Multi-Layer Encryption model protects smart home internet of things systems against cyberattacks. It analyses both the strong and weak points of the model for handling current and developing cybersecurity threats. This chapter also concludes the study and looks at recommendations and future works relevant to this study.

6.2 Study Results

The AMLE model's actual results show it can strongly defend smart home IoT networks from security attacks and makes them stronger. The adaptive structure of the model lets it work quickly to handle new cybersecurity threats and provides better security than fixed encryption systems. The isolation forest algorithm model effectively finds security threats before they cause harm because it successfully detects and controls potential security problems. The smart home encryption system defends all network layers by using device security plus network protection and cloud storage protection.

Key management represents a core asset within AMLE where users can receive and store encrypted keys in a safe manner. Different types of testing show that the AMLE model stays reliable for smart home devices and improves their security effectiveness at the same time. The data proves that this model offers security protection without making users lose access to their devices.

6.3 Objectives Achievement

This study fully met the objectives set at the beginning of the research process The AMLE model was developed and implemented as a prototype within a controlled simulated environment, allowing for rigorous testing and evaluation. The test results show that the AMLE protects IoT Smart home infrastructures better than regular security techniques and proves to be an excellent solution for handling growing security threats against smart home systems. The research has successfully built a simulation solution that lets users use smart home features safely and increases their trust in these systems.

6.4 Research Limitations

While the AMLE model demonstrates substantial promise in bolstering the security of smart home IoT ecosystems, it is imperative to acknowledge certain limitations that were encountered during the course of this study. The model functionality was tested in a controlled simulated platform that while even though thorough testing was conducted on it, it showed limited representation of real-life conditions in smart home environments. The research's attention on different cyber risks needs future tests to validate the security model's protection against other possible risks. The AMLE model requires good computer resources that typical IoT devices cannot handle which creates technical barriers that need resolution through better optimization methods. Further research must examine these model's problems to improve how this system functions in different security situations.

6.5 Recommendations

Based on the findings and insights derived from this research, the following recommendations are put forth for consideration in future work:

i. Researchers need to apply and test the AMLE model in current smart home locations to show how well it works in real-world situations. Testing the model in real home situations will produce essential details about its usefulness in different situations across large systems.

ii. Researchers need to test the AMLE model better against all types of sophisticated cyber-attacks and various security threats that may not have been included in this test. The model offers protection against major security threats such as botnet attacks, man in the middle attacks, and sophisticated intrusion attempts.

iii. Researchers need to focus exclusively on methods that improve the computational speed of the AMLE model. Research to make AMLE work on IoT devices and home computers is needed since it will boost system-wide acceptance.

iv. Researchers should examine how AMLE can interact with modern technologies including blockchain and artificial intelligence to make this model more effective. By integrating the model into new security technology for future work will position it as a leader in IoT cybersecurity breakthroughs.

v. Standard organizations need to create consistent AMLE standards and make proper rules to help this technology grow. Making AMLE standard in smart home will benefit how different devices in the Internet of Things connect securely with each other.

6.6 Suggestions for Future Works

In addition to the aforementioned recommendations, future research endeavours should also explore the following avenues:

i. Research dynamic risk management tools that work within the AMLE model to improve its adaptation to security changes. Such an update makes the model ready to automatically react to shifting safety and security patterns to maximize its protection effectiveness.

ii. Users become security members as we evaluate methods for them to take security actions alongside AMLE The system requires building friendly security platforms alongside setting up security education and letting users modify security preferences.

iii. Testing the AMLE model to check compatibility with all real-world smart home systems so users can use their products without technical challenges. Developing better connections between devices will help build secure networks for smart homes.

iv. Long-Term Security Maintenance Strategies: Developing robust strategies for the long-term security maintenance and updates of the AMLE model. The model needs updates to match current security risks so it remains valuable for the future.

Research efforts should follow these guidelines to make adaptive multi-layer encryption better in both design and operation. This security model will create better smart home IoT systems that users and stakeholders will rely on for the future.

6.7 Conclusion

In conclusion, this study developed and tested the Adaptive Multi-Layer Encryption model to create better IoT security for smart homes. By upgrading standard encryption methods, the AMLE model creates an automated security system to combat new cyber threads as they appear or occur. The model reduces smart home security risks through multiple encryption layers plus anomaly detection technology secured by excellent key handling methods. Research results support that AMLE could become a dependable and effective solution to enhance IoT security for smart homes so consumers can utilize these technology benefits with secure privacy guarantees.

References

Alrawi, O., Lever, C., Antonakakis, M., & Monrose, F. (2019). SoK: Security evaluation of home-based IoT deployments. 2019 IEEE Symposium on Security and Privacy (SP), 1362–1380. https://doi.org/10.1109/SP.2019.00013

Anthi, E., Ahmad, S., Rana, O., Theodorakopoulos, G., & Burnap, P. (2018). EclipseIoT: A secure and adaptive hub for the Internet of Things. Computers & Security, 78, 477–490. https://doi.org/10.1016/j.cose.2018.07.016

Aziz Al Kabir, M., Elmedany, W., & Sharif, M. S. (2023). Securing IoT devices against emerging security threats: Challenges and mitigation techniques. Journal of Cyber Security Technology, 7(4), 199–223. https://doi.org/10.1080/23742917.2023.2228053

Beuchelt, G. (2013). Securing web applications, services, and servers. In J. R. Vacca (Ed.), Computer and Information Security Handbook (2nd ed., pp. 143–163). Morgan Kaufmann. https://doi.org/10.1016/B978-0-12-394397-2.00008-8

Bhardwaj, A., Bharany, S., Abulfaraj, A. W., Ibrahim, A. O., & Nagmeldin, W. (2024). Fortifying home IoT security: A framework for comprehensive examination of vulnerabilities and intrusion detection strategies for smart cities. Egyptian Informatics Journal, 25, Article 100443. https://doi.org/10.1016/j.eij.2024.100443

Bonomi, F., Milito, R., Zhu, J., & Addepalli, S. (2012). Fog computing and its role in the Internet of Things. Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, 13–16. https://doi.org/10.1145/2342509.2342513

Bouaouad, A.-E., Cherradi, A., Assoul, S., & Souissi, N. (2020). The key layers of IoT architecture. 2020 5th International Conference on Cloud Computing and Artificial

Intelligence: Technologies and Applications (CloudTech), 1–4. https://doi.org/10.1109/CloudTech49835.2020.9365919

Chakraborty, A., Islam, M., Shahriyar, F., Islam, S., Zaman, H., & Hasan, M. (2023). Smart home system: A comprehensive review. Journal of Electrical and Computer Engineering, 2023, Article 7616683. https://doi.org/10.1155/2023/7616683

Das, R., & Gündüz, M. (2019). Analysis of cyber-attacks in IoT-based critical infrastructures. International Journal of Information Security, 8, 122–133.

Fernández-Caramés, T., Fraga-Lamas, P., & Suárez-Albela, M. (2018). A practical evaluation on RSA and ECC-based cipher suites for IoT high-security energy-efficient fog and mist computing devices. Sensors, 18(11), 3868. https://doi.org/10.3390/s18113868

Fremantle, P. (2015). A reference architecture for the Internet of Things. https://doi.org/10.13140/RG.2.2.20158.89922

Hamarsheh, A. (2024). An adaptive security framework for Internet of Things networks leveraging SDN and machine learning. Applied Sciences, 14(11), Article 4530. https://doi.org/10.3390/app14114530

Honar Pajooh, H., Rashid, M., Alam, F., & Demidenko, S. (2021). Multi-layer blockchain-based security architecture for Internet of Things. Sensors, 21(3), 862. https://doi.org/10.3390/s21030862

Hussain, F., Abbas, S. G., Pires, I. M., Tanveer, S., Fayyaz, U. U., Garcia, N. M., Shah, G. A., & Shahzad, F. (2021). A two-fold machine learning approach to prevent and detect IoT botnet attacks. IEEE Access, 9, 163412–163430. https://doi.org/10.1109/ACCESS.2021.3131014

Khan, M. N., Rao, A., & Camtepe, S. (2021). Lightweight cryptographic protocols for IoT-constrained devices: A survey. IEEE Internet of Things Journal, 8(6), 4132–4156. https://doi.org/10.1109/JIOT.2020.3026493

Khan, W. Z., Aalsalem, M. Y., Khan, M. K., & Arshad, Q. (2019). Data and privacy: Getting consumers to trust products enabled by the Internet of Things. IEEE Consumer Electronics Magazine, 8(2), 35–38. https://doi.org/10.1109/MCE.2018.2880807

Khraisat, A., & Alazab, A. (2021). A critical review of intrusion detection systems in the Internet of Things: Techniques, deployment strategy, validation strategy, attacks, public datasets and challenges. Cybersecurity, 4(1), 18. https://doi.org/10.1186/s42400-021-00077-7

Mahalle, P. N., & Shinde, G. R. (2021). OAuth-based authorization and delegation in smart home for the elderly using decentralized identifiers and verifiable credentials. In P. N. Mahalle, G. R. Shinde, N. Dey, & A. E. Hassanien (Eds.), Security Issues and Privacy Threats in Smart Ubiquitous Computing (pp. 95–109). Springer. https://doi.org/10.1007/978-981-33-4996-4_6

Mosenia, A., & Jha, N. K. (2017). A comprehensive study of security of Internet-of-Things. IEEE Transactions on Emerging Topics in Computing, 5(4), 586–602. https://doi.org/10.1109/TETC.2016.2606384

Nassi, B., Bitton, R., Masuoka, R., Shabtai, A., & Elovici, Y. (2021). SoK: Security and privacy in the age of commercial drones. 2021 IEEE Symposium on Security and Privacy (SP), 1434–1451. https://doi.org/10.1109/SP40001.2021.00005

Navneet Kaur. (2021). Development of a smart home automation system using IoT technology. International Journal of Mechanical Engineering, 6. https://doi.org/10.56452/6-14

Nimmy, K., Sankaran, S., & A. K. (2018). A novel multi-factor authentication protocol for smart home environments. In V. Ganapathy, R. T. & K. Jaeger (Eds.), Information Systems Security (pp. 44–63). Springer.

Pan, J., & McElhannon, J. (2018). Future edge cloud and edge computing for Internet of Things applications. IEEE Internet of Things Journal, 5(1), 439–449. https://doi.org/10.1109/JIOT.2017.2767608

Patil, K. S., Mandal, I., & Rangaswamy, C. (2022). Hybrid and adaptive cryptographic-based secure authentication approach in IoT-based applications using hybrid encryption. Pervasive and Mobile Computing, 82, 101552. https://doi.org/10.1016/j.pmcj.2022.101552

Radanliev, P., De Roure, D., Page, K., Nurse, J. R. C., Montalvo, R. M., Santos, O., Maddox, L., & Burnap, P. (2020). Cyber risk at the edge: Current and future trends on cyber risk analytics and artificial intelligence in the industrial Internet of Things and Industry 4.0 supply chains. Cybersecurity, 3(1), 13. https://doi.org/10.1186/s42400-020-00052-8

Rahman, Z., Yi, X., Billah, M., Sumi, M., & Anwar, A. (2022). Enhancing AES using chaos and logistic map-based key generation technique for securing IoT-based smart home. Electronics, 11(7), 1083. https://doi.org/10.3390/electronics11071083

Raza, S., Misra, P., He, Z., & Voigt, T. (2017). Building the Internet of Things with Bluetooth Smart. Ad Hoc Networks, 57, 19–31. https://doi.org/10.1016/j.adhoc.2016.08.012

Satyanarayanan, M., Bahl, P., Caceres, R., & Davies, N. (2009). The case for VM-based cloudlets in mobile computing. IEEE Pervasive Computing, 8(4), 14–23. https://doi.org/10.1109/MPRV.2009.82

Sicari, S., Rizzardi, A., Grieco, L. A., & Coen-Porisini, A. (2015). Security, privacy and trust in Internet of Things: The road ahead. Computer Networks, 76, 146–164. https://doi.org/10.1016/j.comnet.2014.11.008

Statista. (2023). Smart home - Number of households in the segment smart home in the world 2025. https://www.statista.com/forecasts/887613/number-of-smart-homes-in-the-smart-home-market-in-the-world

Stephens, B., Shaghaghi, A., Doss, R., & Kanhere, S. S. (2021). Detecting Internet of Things bots: A comparative study. IEEE Access, 9, 160391–160401. https://doi.org/10.1109/ACCESS.2021.3130714

Tnvs, P., Odugu, R. D., & Lingamgunta, S. (2024). Designing secure IoT systems: A survey of existing approaches and future directions. 2024 2nd International Conference on Emerging Trends in Information Technology and Engineering (ICETITE), 1–8. https://doi.org/10.1109/ic-ETITE58242.2024.10493417

Tripathy, A., & Singh, B. (2022). A study of AES software implementation for IoT systems. 2022 3rd International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT), 1–4. https://doi.org/10.1109/ICICT55121.2022.10064507

Ullah, S., Zheng, J., Din, N., Hussain, M. T., Ullah, F., & Yousaf, M. (2023). Elliptic curve cryptography; Applications, challenges, recent advances, and future trends: A comprehensive survey. Computer Science Review, 47, 100530. https://doi.org/10.1016/j.cosrev.2022.100530

Uppuluri, S., & Lakshmeeswari, G. (2022). Secure user authentication and key agreement scheme for IoT device access control based smart home communications. Wireless Networks, 1–22. https://api.semanticscholar.org/CorpusID:254241752

Yaacoub, J.-P. A., Noura, H. N., Salman, O., & Chehab, A. (2023). Ethical hacking for IoT: Security issues, challenges, solutions and recommendations. Internet of Things and Cyber-Physical Systems, 3, 280–308. https://doi.org/10.1016/j.iotcps.2023.04.002

Zhang, L., & Wang, L. (2024). A hybrid encryption approach for efficient and secure data transmission in IoT devices. Journal of Engineering and Applied Science, 71(1), 138. https://doi.org/10.1186/s44147-024-00459-x

Appendices

Appendix A: Similarity Report

Dancan Obuya

Dancan Obuya Machuki Adaptive Multi-Layer Encryption for Smart Home Ecosystem Final Dissertation.pdf

Strathmore University (Main Account)

Document Details

Submission ID
trn:oid:::2945:275187833

Submission Date
Mar 28, 2025, 6:06 PM GMT+3

Download Date
Mar 28, 2025, 6:18 PM GMT+3

File Name
Dancan Obuya Machuki Adaptive Multi-Layer Encryption for Smart Home Ecosystem Final Disser....pdf

File Size
2.9 MB

94 Pages

15,690 Words

102,194 Characters

turnitin Page 1 of 106 - Cover Page Submission ID trn:oid:::2945:275187833

14% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text

Match Groups

- **152** Not Cited or Quoted 11%
  Matches with neither in-text citation nor quotation marks
- **38** Missing Quotations 2%
  Matches that are still very similar to source material
- **0** Missing Citation 0%
  Matches that have quotation marks, but no in-text citation
- **0** Cited and Quoted 0%
  Matches with in-text citation present, but no quotation marks

Top Sources

8% Internet sources
6% Publications
11% Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Appendix B: Ethical Clearance Confirmation

Strathmore UNIVERSITY

28th March 2025

Mr Machuki Dancan,
dancan.obuya@strathmore.edu

Dear Mr Machuki,

RE: Adaptive Multi-Layer Encryption for Smart Home IoT Ecosystems

This is to inform you that SU-ISERC has reviewed and **approved** your above **SU-masters** proposal. Your application reference number is **SU-ISERC2781/25.** The approval period is from **28th March 2025 to 27th March 2026.**

This approval is subject to compliance with the following requirements:

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

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

Yours sincerely,

Mr Ambrose Rachier,
Chairperson; SU-ISERC

Ole Sangale Rd, Madaraka Estate. PO Box 59857-00200, Nairobi, Kenya. Tel +254 (0)703 034000
Email admissions@strathmore.edu www.strathmore.edu