

**Detecting Malware Attack in Mobile Phone using Intrusion Detection and Prevention
System (IDPS)**

BY

Leow Yu Hong

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

**BACHELOR OF INFORMATION TECHNOLOGY (HONOURS) COMMUNICATIONS
AND NETWORKING**

Faculty of Information and Communication Technology

(Kampar Campus)

JUNE 2024

REPORT STATUS DECLARATION FORM

Title: Detecting Malware Attack in Mobile Phone using Intrusion Detection and Prevention System (IDPS)

Academic Session: JUN 2024

I LEOW YU HONG

(CAPITAL LETTER)

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

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

Verified by,



(Author's signature)



(Supervisor's signature)

Address:

No 6, Jalan Melur Impian 3,
Taman Melur Impian, 28300
Triang, Pahang

_ Dr Abdulrahman Aminu Ghali _
Supervisor's name

Date: 11 Sept 2024

Date: 11 Sept 2024

Universiti Tunku Abdul Rahman			
Form Title : Sample of Submission Sheet for FYP/Dissertation/Thesis			
Form Number: FM-IAD-004	Rev No.: 0	Effective Date: 21 JUNE 2011	Page No.: 1 of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY
UNIVERSITI TUNKU ABDUL RAHMAN

Date: 11 Sept 2024

SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS

It is hereby certified that **LEOW YU HONG** (ID No: **20ACB05558**) has completed this final year project/ dissertation/ thesis* entitled “*Detecting Malware Attack in Mobile Phone using Intrusion Detection and Prevention System (IDPS)*” under the supervision of Dr. Abdulrahman Aminu Ghali (Supervisor) from the Department of Faculty of Information and Communication Technology (FICT).

I understand that University will upload softcopy of my final year project / dissertation/ thesis* in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.


Yours truly,



(Leow Yu Hong)

DECLARATION OF ORIGINALITY

I declare that this report entitled “**Detecting Malware Attack in Mobile Phone using Intrusion Detection and Prevention System (IDPS)**” is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : _____ 

Name : Leow Yu Hong

Date : 11 Sept 2024

ACKNOWLEDGEMENTS

I would like to express my gratitude to my supervisor, Dr. Abdulrahman Aminu Ghali, from the bottom of my heart for giving me this amazing chance to work on a cybersecurity project. It's the first step I'm taking to pursue a career in cybersecurity. I am so appreciative of you.

Furthermore, I would want to express my appreciation to my parents and family for their continuing love, support, and encouragement along this journey.

ABSTRACT

This project centers on cybersecurity, with a specific focus on detecting and preventing adware through the use of Intrusion Detection and Prevention Systems (IDPS) on Android mobile devices. The project integrates both Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS) to strengthen defenses against adware attacks using the IDPS approach. Multiple techniques are employed, such as signature-based adware detection, machine learning model detection, and network-based detection. In the signature-based method, adware is identified by comparing it with a database of known adware signatures. For adware not found in the database, detection is handled through machine learning models or network-based approaches. Several malware attributes are analyzed, including file name, size, type, and API calls. The research data covers the period from 2019 to 2023, with some data from earlier years. Thanks to the diverse detection methods used by the IDS, such as signature-based detection and machine learning models, we were able to detect both known and previously unknown adware in our initial tests. However, false positives can arise due to configuration errors or low-accuracy model development. Our quarantine system stops specific application processes to prevent further malware infection. Regular updates to the signature database are crucial for effectively detecting and stopping threats. By integrating IDS and IPS, we can significantly improve our success rate in preventing malware attacks, as each system compensates for the other's weaknesses and enhances overall detection.

TABLE OF CONTENTS

TITLE PAGE	i
REPORT STATUS DECLARATION FORM	ii
FYP THESIS SUBMISSION FORM	iii
DECLARATION OF ORIGINALITY	iv
ACKNOWLEDGEMENTS	v
ABSTRACT	vi
TABLE OF CONTENTS	vii
LIST OF FIGURES	x
LIST OF TABLES	xii
LIST OF ABBREVIATIONS	xiii
CHAPTER 1 INTRODUCTION	1
1.1 Problem Statement and Motivation	3
1.2 Objectives	6
1.3 Project Scope and Direction	6
1.4 Contributions	6
1.5 Report Organization	7
1.6 Summary	7
CHAPTER 2 LITERATURE REVIEW	8
2.1 Previous Works on Mobile Malware Detection	8
2.1.1 Static Signature-based Detection	8
2.1.2 Behavior Signature-based Detection	9
2.1.3 Static Anomalies-based Detection	10
2.1.4 Dynamic Anomaly-based Detection	11
2.1.5 Machine learning Approaches	12
2.2 Strength and Weakness	13
2.3 Summary	15
CHAPTER 3 SYSTEM METHODOLOGY/APPROACH	16
3.1 System Design Diagram/Equation	16

3.1.1	System Architecture Diagram	16
3.1.2	Use Case Diagram and Description	17
3.1.3	Activity Diagram	18
3.2	Summary	19
CHAPTER 4	SYSTEM DESIGN	20
4.1	System Block Diagram	20
4.2	System Components Specifications	21
4.3	Circuits and Components Design	22
4.4	System Components Interaction Operations	27
4.5	Summary	27
CHAPTER 5	SYSTEM IMPLEMENTATION	28
5.1	Hardware Setup	28
5.2	Software Setup	28
5.3	Setting and Configuration	29
5.4	System Operation (with Screenshot)	30
5.5	Implementation Issues and Challenges	35
5.6	Concluding Remark	36
CHAPTER 6	SYSTEM EVALUATION AND DISCUSSION	37
6.1	System Testing and Performance Metrics	37
6.2	Testing Setup and Result	38
6.3	Project Challenges	40
6.4	Objectives Evaluation	41
6.5	Concluding Remark	42
CHAPTER 7	CONCLUSION AND RECOMMENDATION	43
7.1	Conclusion	43
7.2	Recommendation	43
REFERENCES		45
APPENDIX		47

WEEKLY LOG	47
POSTER	53
PLAGIARISM CHECK RESULT	54
FYP2 CHECKLIST	57

LIST OF FIGURES

Figure Number	Title	Page
Figure 1.1	Intrusion Prevention System vs Intrusion Detection System	2
Figure 1.2	Attack Distribution by Software Type Used	3
Figure 1.3	Trojan Installers for Mobile Banking Were Found	5
Figure 2.1	Methods for Mobile Malware Detection	8
Figure 2.2	Signature-based Static Detection	9
Figure 2.3	Signature-based Behavior Detection	10
Figure 2.4	Anomaly-based Static Identification	11
Figure 2.5	Dynamic Detection of Anomalies	11
Figure 2.6	Malware detection powered by machine learning	12
Figure 3.1	System Architecture Diagram	16
Figure 3.2	Use Case Diagram	17
Figure 3.3	Activity Diagram for Malware IDPS	18
Figure 4.1	System Block Diagram of Malware IDPS	20
Figure 4.2	System Flow of Malware IDPS	23
Figure 4.3	Logical Flow of Malware IDPS	25
Figure 4.4	Logical Flow of Malware IDPS	26
Figure 5.1	Scanning Options	31
Figure 5.2	Quick Scan Result	31
Figure 5.3	Specific Scan Result	31
Figure 5.4	Malicious Detected	31
Figure 5.5	Default Page	32
Figure 5.6	Normal Web Browse	32
Figure 5.7	Malicious Web Browse	32
Figure 5.8	Blocked Website	32
Figure 5.9	Downloading File	33
Figure 5.10	File detected as Safe	33
Figure 5.11	Downloading File	33
Figure 5.12	File Detected as Malicious	33
Figure 5.13	Quarantine File Lists	34
Figure 5.14	Remove File Permanently	34

Figure 6.1	Malicious URL Testing Accuracy Chart	39
Figure 6.2	Download File Detection Testing Accuracy Chart	39
Figure 6.3	Resource Utilization Testing Chart	40

LIST OF TABLES

Table Number	Title	Page
Table 1.1	Specifications of Development System	28
Table 2.1	Malicious URL Detection Testing Result	37
Table 2.2	Download File Detection Testing Result	37
Table 2.3	Resource Utilization Testing Result	38

LIST OF ABBREVIATIONS

<i>IDPS</i>	Intrusion Detection and Prevention System
<i>IDS</i>	Intrusion Detection System
<i>IPS</i>	Intrusion Prevention System

Chapter 1:

Introduction

Malware poses a significant danger to cybersecurity as it is one of the most common methods for gaining unauthorized access. Malware comes in many different forms, including as worms, ransomware, Trojan horses, spyware, and adware. Malware brings distinct threats of its own. Because of these parallels, spyware and adware are frequently discussed in the same context. On mobile devices, spyware often gathers user information—such as location, keystrokes, browser history, and online activity—without the user's knowledge. Following theft, phishing and other fraudulent activities frequently use this information. Spyware may also monitor online activities for commercial and promotional objectives. It is concerning to note that throughout the previous eight years, there have been an estimated 7.7 billion malware occurrences, peaking at 10.5 billion in 2018 and reaching 5.5 billion attacks in 2022 [1]. As of early, the manufacturing sector has been the main focus of malware assaults, accounting for 24.8% of all malware attacks in 2022. Next on the list, the finance and insurance industries were impacted by 18.9% of attacks in the same year. Affected industries also include retail, consumer services, energy, professional services, and education [2]. Furthermore, educational and scientific institutions see the highest frequency of malware occurrences—2,314 on average each week—followed by government and military institutions, which report 1,661 events weekly [3].

Malware attacks have become one of the major concerns in today's world, since smartphones have become indispensable. Malicious software affected 1,661,743 mobile devices in 2022 alone [7]. These malware installers are pieces of software designed especially to infect mobile devices with malicious apps. The majority of these installers come from dubious sources as opposed to reliable ones like the App Store or Google Play Store. Malicious installers are not the only thing that have attracted fraudsters' attention; mobile devices, which are often utilized for internet banking, are also appealing targets. According to cybersecurity reports, the number of new mobile banking Trojans found in 2022 was 196,476; in comparison, the number was 69,777 in 2019, 156,710 in 2020, and 97,661 in 2021. This indicates that malware infections will noticeably rise in 2022 [7]. These problems were made worse by the startling increase in malware, adware, and undesired software attacks that occurred in the first three quarters of

2023. In the first quarter alone, an astounding 4,948,522 cases were documented; in the second quarter, 5,704,599 attacks were reported. The crisis took a dramatic turn for the worse during the third quarter, with 8,346,169 attacks reported [18]. Figure 1.1 Intrusion Prevention System vs Intrusion Detection System, which displays pertinent data, appears below.

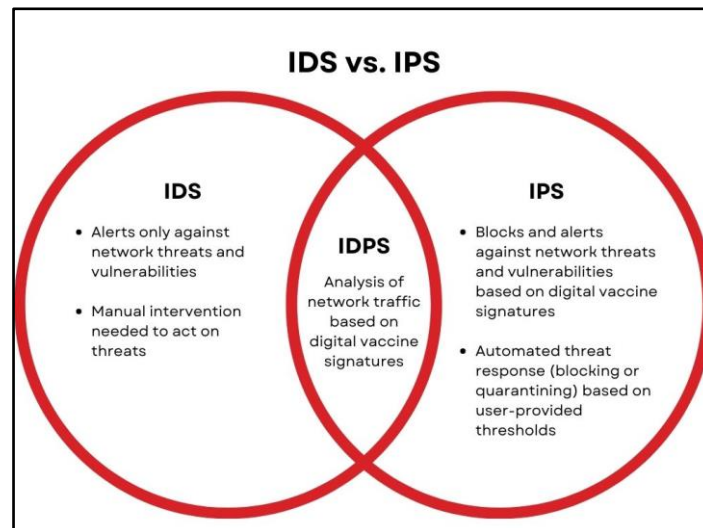


Figure 1.1 Intrusion Prevention System vs Intrusion Detection System

Intrusion detection and prevention systems, or IDPS, are essential security tools for spotting and thwarting malware attacks. A mobile phone detection system can be seen in Figure 1.1. This system can be used to monitor activities, analyse behaviours, identify potential threats, and take necessary action to stop attacks that are discovered. Encouraging data is displayed in Figure 1.2, Attack Distribution by Software Type Used.

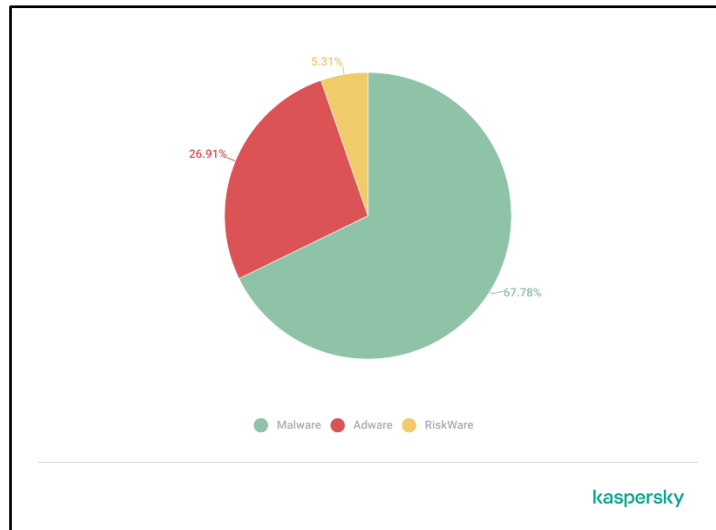


Figure 1.2 Attack Distribution by Software Type Used

It is critical to address this issue because, as Figure 1.2 illustrates, there has been a discernible increase in malware attacks directed towards mobile devices. To protect mobile devices from dangerous threats, it is essential to implement a malware detection system.

1.1 Problem Statement and Motivation

Many issues can occur when malware compromises a mobile device. The theft of login information, including passwords and usernames, is among the most frequent results. Online privacy data indicates that about 21% of users have had unwanted access to their accounts, such as social media and email [8]. This suggests that 5 percent of internet users have been the victim of account hacking. Furthermore, when malware infects mobile devices, sensitive data—such as bank account information, personal details, and family photos—is highly vulnerable to theft. Over 10% of internet users have experienced identity theft including credit cards and bank information [8]. Due to the widespread usage of mobile devices for online banking, which makes user data more easily accessible to hackers, these devices are especially susceptible to data breaches. Another common problem associated with malware infestations is financial loss, which is frequently brought on by downloading malicious files or installing dubious third-party software. These malicious apps have the ability to carry out unlawful financial transactions and steal data from the device, which can result in large financial losses. For instance, to schedule cleaning services, two Malaysian ladies downloaded an app offered by a cleaning agency [9]. The program was utilized by scammers to obtain their banking

information, which cost them RM40,000 in total. The quantity of malicious installation packages found on mobile devices increased steadily over time, as reported by Statista, from 276,319 in Q4 2022 to 438,962 in Q3 2023 [16]. Spying and surveillance, in particular the use of spyware as a tool for surveillance, is a problem that is frequently ignored. Under the user's awareness, spyware frequently installs itself surreptitiously on mobile devices and gathers private data, including passwords for banking and social media accounts, usernames, and financial information. Until their smartphone slows down or they get a warning from antivirus or anti-malware software, consumers are often unaware that several spyware programs are concealed within apps that appear legal. One noteworthy instance concerned the program built by mSpy, which gave the impression of being a legitimate parental monitoring app. Paradoxically, in May 2015, mSpy was compromised, demonstrating that even software that appears to be "safe" can be exploited [10].

Unwanted advertisements have gained notoriety in the digital age, especially when they are delivered using Adware. Because adware uses a lot of background resources, it can cause a user's device to run much slower by constantly bombarding it with intrusive adverts. In more serious situations, sexual content may be displayed by Adware. A quarter of all malware instances worldwide in 2022 were caused by adware [5][7]. Mobile malware cases increased dramatically during the fourth quarter of 2022, with Adware accounting for 38.36% of newly identified cases. In the first quarter of 2023, this percentage decreased somewhat to 34.76% [4]. Adware continued to be the most dangerous threat, impacting 62.65% of all mobile users attacked in Q2 2023, even though Kaspersky's newly discovered Adware cases decreased from 34.79% in Q1 2023 to 22.69% in Q2 2023 [17].

When using a mobile phone, everyone looks for a secure environment because these devices store a lot of sensitive and personal information. Attacks by malware on mobile devices have increased, especially when users do private tasks like paying bills and transacting with banks. As digital banking systems expand quickly, people are adopting online banking due to its ease, efficiency, portability, and perceived security. Attackers consequently concentrate more on banking systems, which causes people to inadvertently download malicious software from untrusted sources, giving hackers access to their usernames and passwords. Figure 1.3, "Trojan Installers for Mobile Banking Were Found," provides essential information.

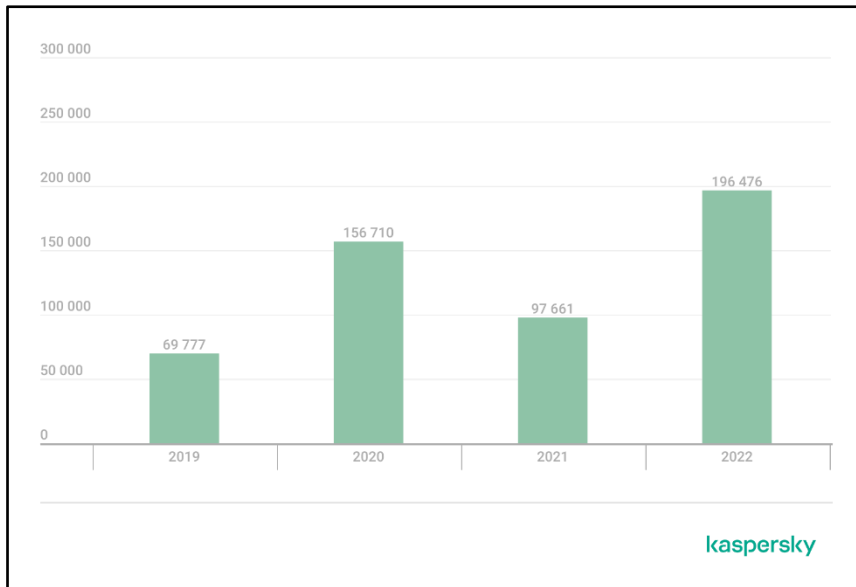


Figure 1.3 Trojan Installers for Mobile Banking Were Found

There were found over 100,000 Trojans related to mobile banking in 2022, a notable increase from 2019 [7] as illustrated in Figure 1.3. Malaysian banks have increased their vigilance in response to multiple incidences during the COVID-19 outbreak where people were duped by fake banking websites or applications. In 2022, an elderly woman suffered a financial loss of RM476,100 as a result of a bogus investing app [15]. Given the frequency of these types of incidents, Malaysia has an urgent need for effective malware detection and prevention systems. There are numerous methods for identifying and preventing malware, such as hybrid signature-based detection, behavioural signature-based detection, and static signature-based detection. Techniques based on static, dynamic, or hybrid signatures can also be used for anomaly detection. Regrettably, these solutions don't take proactive measures to stop problems before they start; instead, they just concentrate on detection. As a result of their heavy human configuration and maintenance requirements, these systems are vulnerable to false positives and may find it difficult to combat new malware threats. This project suggests creating an Intrusion Detection and Prevention System (IDPS), which can stop malicious software and malware in addition to detecting them, in order to address these issues.

1.2 Research Objectives

This final year project aims to accomplish three key objectives:

- To build a strong intrusion detection and prevention system
- To recognize and identify malware
- To look on ways to prevent mobile malware

1.3 Project Scope and Direction

By employing a thorough approach that prioritizes the identification and stopping of malware intrusions as well as the prevention of unwanted activity on mobile devices, this project seeks to improve mobile device security. The monitoring methods used will assist in spotting illicit activity and malicious conduct that could point to the existence of malware. By comparing known assaults to pre-existing malware attack signatures, the system will identify and prevent them through the use of static signature-based detection. By identifying malware based on its action, the system will be able to stop its propagation with the help of behavioural signature detection. Network-based assaults that take advantage of network vulnerabilities can be stopped by the system by examining attack patterns and keeping an eye on any suspicious activity associated with these patterns. Furthermore, as new threats are identified, machine learning models will be integrated to continuously detect and update malware signatures. The system will be enhanced using signature-based detection algorithms to make it easier to identify and remove known malware from mobile devices.

1.4 Contributions

Our study has made the following primary contributions, which we highlight in this section:

- Experiments to validate behaviour-based and signature-based methods for malware prevention and detection were planned.
- Through the recognition of well-established attack patterns, intrusion detection systems are able to discover malware signatures.
- Network packet analysis is used to find malware network behaviour and patterns of malicious activity.
- Machine learning models that have been trained on large amounts of data have the ability to recognize malware that has encrypted signature headers and anticipate potentially dangerous attacks.

- To stop additional harm, a quarantine mechanism is put in place to isolate particular files or IP addresses that these systems find.
- Whenever new malware is discovered, automatic updates to malware signatures are carried out, improving the system's attack pattern database.

1.5 Report Organization

The ensuing chapters go over the specifics of this study. The Literature Review is presented in Chapter 2. The Malware IDPS system's methodology and approach are described in Chapter 3. The Malware IDPS System Design is covered in Chapter 4. Furthermore, Chapter 6 discusses the System Evaluation and Discussion, whereas Chapter 5 concentrates on the System Implementation. Chapter 7 offers the Final Year Project II of the Malware IDPS's Conclusion.

1.6 Summary

This chapter provides an overview of the environment surrounding cybersecurity risks, namely malware and how it affects mobile devices. It draws attention to many types of malware, including worms, ransomware, Trojan horses, and spyware, emphasizing how they can harm users by stealing their personal data. It also describes the project's objective, highlighting the necessity of malware detection and prevention systems in light of the rise in malware attacks on mobile devices. The study goals, which include creating an efficient Intrusion Detection and Prevention System (IDPS) for mobile malware detection, are presented in the chapter's conclusion.

Chapter 2:

Literature Review

Prior to analysing existing solutions, a comprehensive assessment of pertinent research is carried out, providing a basis for comprehending the current status of the area.

2.1 Previous Works on Mobile Malware Detection

Mobile malware has emerged as one of the most significant hazards in our globally interconnected world due to the swift progress made in mobile technology and communication. Researchers have looked into a number of methods for efficiently detecting mobile viruses. An overview of pertinent data is provided in below Figure 2.1, Methods for Mobile Malware Detection.

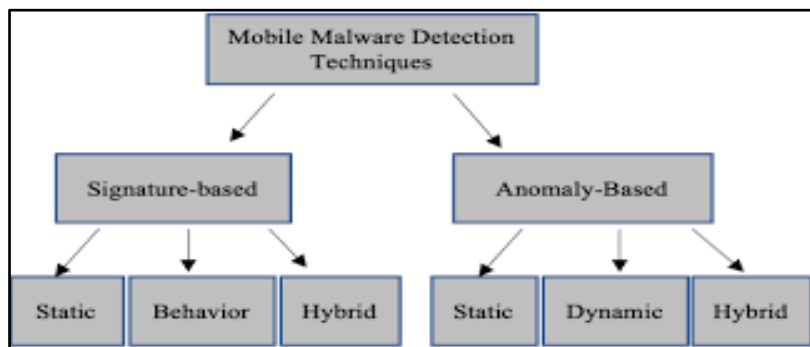


Figure 2.1 Methods for Mobile Malware Detection

An overview of earlier studies on mobile malware detection strategies is shown in Figure 2.1, with particular attention paid to static and dynamic anomaly-based detection methods, as well as behaviour signature-based and static detection methods.

2.1.1 Static Signature-based Detection

Static signature-based malware detection works by identifying malware using a list or database of known malware signatures. This technique compares the target application's attributes to known malware signatures to see if the program has any characteristics. It offers a high degree of accuracy in detecting known malware threats on mobile devices, making it one of the most

effective approaches for doing so. The pertinent data is displayed as shown in below Figure 2.2, Signature-based Static Detection.

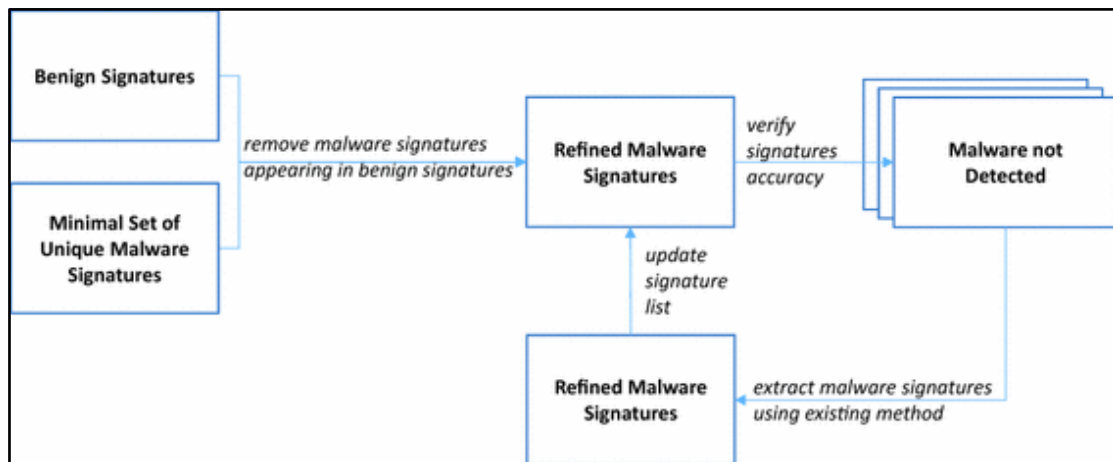


Figure 2.2 Signature-based Static Detection

A low-complexity static signature-based detection technique is shown in Figure 2.2. This method filters out malware signatures that coexist with benign signatures, then testing is done. The malware signature is then updated in the database in case it is later found to not be malicious.

2.1.2 Behavior Signature-based Detection

Like static signature-based detection, behaviour-based signature detection works in a similar way. To determine the type of assault, it keeps an eye on the data that a program gathers from users, how it gathers it, and the techniques it employs while it's running. This method is predicated on an established database of recognized attack patterns. The system that relies on behaviour signatures to detect suspicious activity is activated when the actions of an application come together. In contrast to static detection, this approach necessitates the execution of the application in order to watch its behaviour and identify possible dangers. Relevant data is displayed in below Figure 2.3 Signature-based Behaviour Detection.

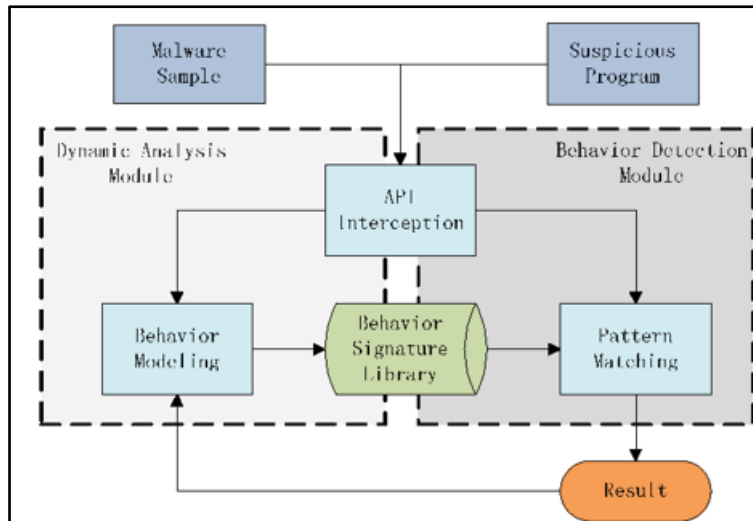


Figure 2.3 Signature-based Behaviour Detection

The use of API interception to gather application data is shown in Figure 2.3. Dynamic Analysis and Behaviour Detection modules are then used to examine the data further. Together, these modules assess the gathered information in order to detect any potentially harmful activity or suspect activity based on how the application interacts with the system.

2.1.3 Static Anomalies-based Detection

Through the examination of source code, static analysis enables the detection of certain code fragments, questionable features, and behavioural patterns in potentially harmful mobile applications without the requirement for execution. Compared to dynamic anomaly-based detection, this offers a significant advantage because it does not require the malicious payload to be activated. On the other hand, this method's increased risk of false positives is a major disadvantage.

Static analysis examines potentially hazardous applications using a number of detection patterns. To examine the application's code for questionable elements, these include opcode frequency distributions, control flow graphs, syntactic library calls, byte-sequence n-grams, string signatures, and control flow graphs. Relevant data is displayed in Figure 2.4 Anomaly-based Static Identification below.

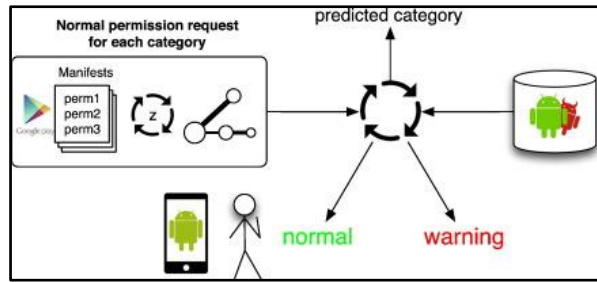


Figure 2.4 Anomaly-based Static Identification

The Anomaly-based Static Identification approach is shown in Figure 2.4. It compares fresh or incoming permission requests to a set of normal or usual requests for each category. This technique aids in the detection of possible abnormalities that can point to malevolent activity by highlighting departures from the accepted standard.

2.1.4 Dynamic Anomaly-based Detection

On the other hand, malware is identified by dynamic anomaly-based detection, which depends on the functioning of mobile applications. Accurate models of typical behaviour are constructed in the training phase and are subsequently employed in the detection phase to identify deviations. While this approach is quite good at detecting malware on mobile devices, it takes a lot of time and resources, which makes it difficult to scale. Relevant data is displayed in Figure 2.5 Dynamic Detection of Anomalies below.

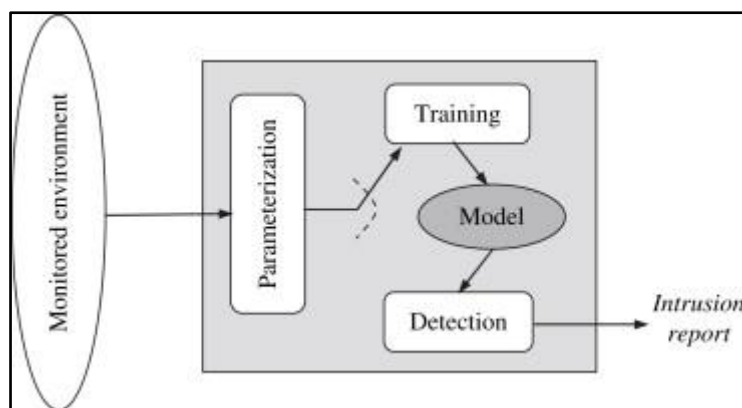


Figure 2.5 Dynamic Detection of Anomalies

The Dynamic Detection of Anomalies model, trained on data from typical behaviour patterns, is shown in Figure 2.5. The model is used for real-time detection following this training phase, where it looks for departures from the accepted norm in order to spot possible anomalies or malicious activities.

2.1.5 Machine Learning Approaches

Machine learning methods like association rules, decision trees, and random forests can also be used to detect mobile malware. These techniques make it possible to categorize malware samples and mark questionable ones for additional examination. The use of complex evasion strategies by some malware samples, including obtaining kernel-mode capabilities without utilizing the system call interface, poses a serious obstacle to machine learning-based categorization and further muddies the waters. Relevant data is displayed in Figure 2.6 Malware detection powered by machine learning below.

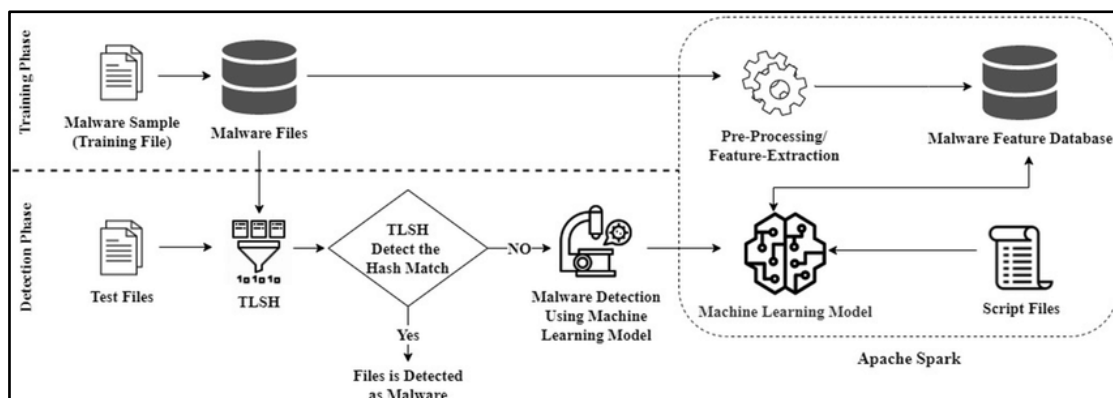


Figure 2.6 Malware detection powered by machine learning

The process of utilizing machine learning to detect malware is shown in Figure 2.6, wherein a model is trained on known malware samples in order to extract important properties. When malware is detected, the system first looks for hash matches using the TLSH technique; if none are discovered, machine learning models are used to assess and categorize possible malware.

2.2 Strengths and Weakness

Static Signature-based Detection

It turns out that Static Signature-based Detection is a useful technique for identifying malware that is currently in circulation online. By matching a prospective threat's digital fingerprint, or signature, to a precompiled database of recognized malware fingerprints, this method detects threats. As soon as a match is discovered, the program instantly flags it as malicious, thereby displaying a virtual "no entry" sign.

There is a drawback to this tactic, too. Despite being incredibly effective, it might overlook more recent malware strains that haven't been seen online yet. It's like when a detective recognizes criminals they've dealt with before with ease, but finds it difficult to identify strangers.

Behavior Signature-based Detection

Signature-based behaviour Instead of relying on pre-established malware signatures, detection makes great use of its database to analyse assault patterns in order to find adware. Because it compares the behaviour patterns of newly discovered adware strains to those already defined in the database, this method is especially useful for flagging unknown threats based on actions that are comparable.

It's crucial to recognize this method's limits, though. Its incapacity to identify attacks that deviate from the established attack patterns is a significant flaw. As a result, the risks that Behaviour Signature-based Detection can identify are restricted to those that are specifically listed in its database [11].

Static Anomalies-based Detection

This method's ability to identify adware without actually needing the installation of malicious payloads on devices is one of its main advantages. The system can detect any threats before any harm is done thanks to the proactive detection approach. The technique efficiently finds adware early by closely examining the source code of harmful mobile applications and identifying patterns and behaviours that are frequently linked to such infestations.

It's crucial to recognize one possible disadvantage of this strategy, though. A concern

associated with this technique is false positives because it looks at different parts of the source code, including byte sequences, syntactic library calls, and n-grams.

Dynamic Anomaly-based Detection

This approach is very special since it can identify new adware strains with accuracy. When rogue programs that cause harm are encountered, the underlying model can quickly identify anomalies since it has been educated on typical behaviour patterns. The method also performs exceptionally well in detecting zero-day malware, indicating its versatility in detecting threats even in the absence of prior knowledge.

It is imperative to take into account any possible disadvantages associated with the ongoing functioning of the system on devices following deployment. The extended duration of this operation could result in significant resource usage, which would affect the system's overall performance. Careful tuning is needed to maintain optimal efficiency and avoid false positives in order to address this [13].

Machine Learning Approaches

Machine learning has gained popularity as a practical and efficient way for identifying mobile malware since it is adaptable and efficient when used in conjunction with other detection techniques. Because it is so easily scaled and modified to new advancements, it is very popular. These models get more and more proficient at detecting threats as a result of regular upgrades that incorporate newly discovered malware samples.

Still, this strategy carries drawbacks in addition to its benefits. One noteworthy problem is that some malware samples are able to obtain kernel-mode privileges without using the system call interface. These situations make the categorization process more difficult and put the model's accuracy in detection to the test. Given the intricacy of these sophisticated malware activities, it is possible that the model will have difficulty correctly identifying and categorizing these dangerous threats.

2.3 Summary

An extensive review of earlier studies on mobile malware detection methods is given in this chapter. It goes over the main techniques for detecting malware, such as anomaly-based detection and detection based on static and behavioural signatures. The chapter also discusses the benefits and drawbacks of machine learning techniques for identifying mobile viruses. It offers a basis for comprehending the difficulties connected with each detection approach as well as the operation of the existing systems.

Chapter 3:

System Methodology/Approach

3.1 System Design Diagram/Equation

The Intrusion Detection and Prevention System (IDPS) on an Android platform was developed using a system design methodology that is described in this section. A feature-driven approach was employed throughout the development process, with each feature being developed separately before being integrated into the system as a whole. A complete overview of the system architecture, important use cases, and activity processes may be found in the sections that follow.

3.1.1 System Architecture Diagram

The IDPS's general organizational structure is shown in the system architecture diagram. The system has been divided down into different parts, each of which is in charge of a certain function, like file isolation, virus scanning, and online browser monitoring. These elements and how they work together are shown in the diagram below. Below is Figure 3.1 System Architecture Diagram showing relevant data.

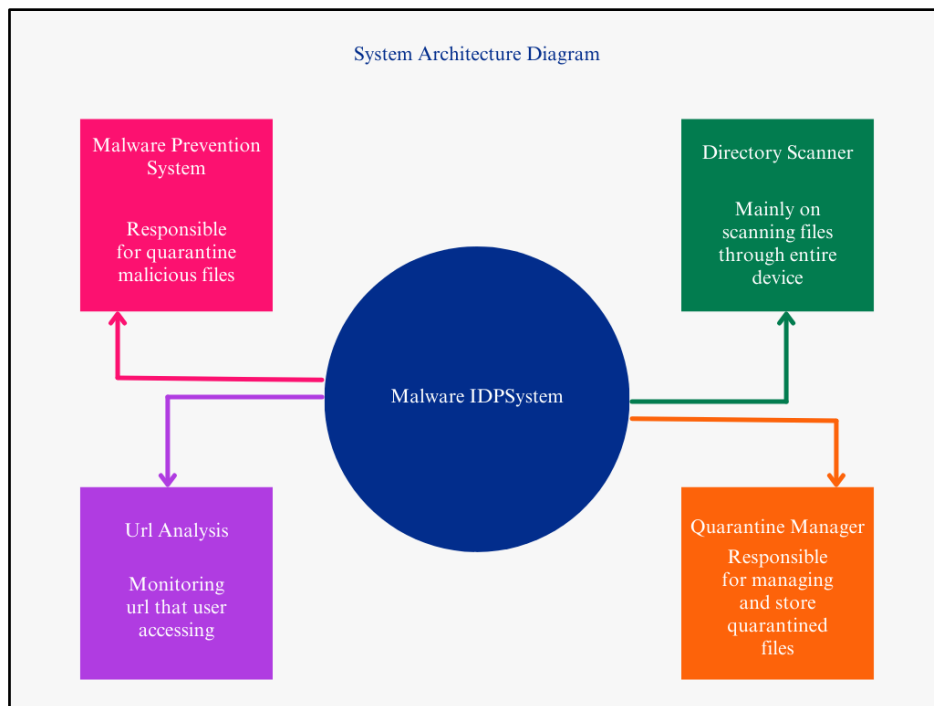


Figure 3.1 System Architecture Diagram

Based on the Figure 3.1, the system is made up of several important parts:

1. MalwarePreventionSystem: In charge of starting scans, maintaining files in quarantine, and handling the whole malware prevention process.
2. FileScanner: Manages the process of looking for potentially malicious files within directories.
3. UrlAnalysis: Keeps track of and examines URLs the user visits in order to identify potentially harmful conduct.
4. QuarantineManager: Oversees the secure storage of identified malicious files off of the user's device by managing their isolation.

3.1.2 Use Case Diagram and Description

The main interactions a user may have with the system, like running malware scans, keeping an eye on online activity, and isolating malware that has been found, is shown in the use case diagram below. Below is Figure 3.2 Use Case Diagram showing relevant data.

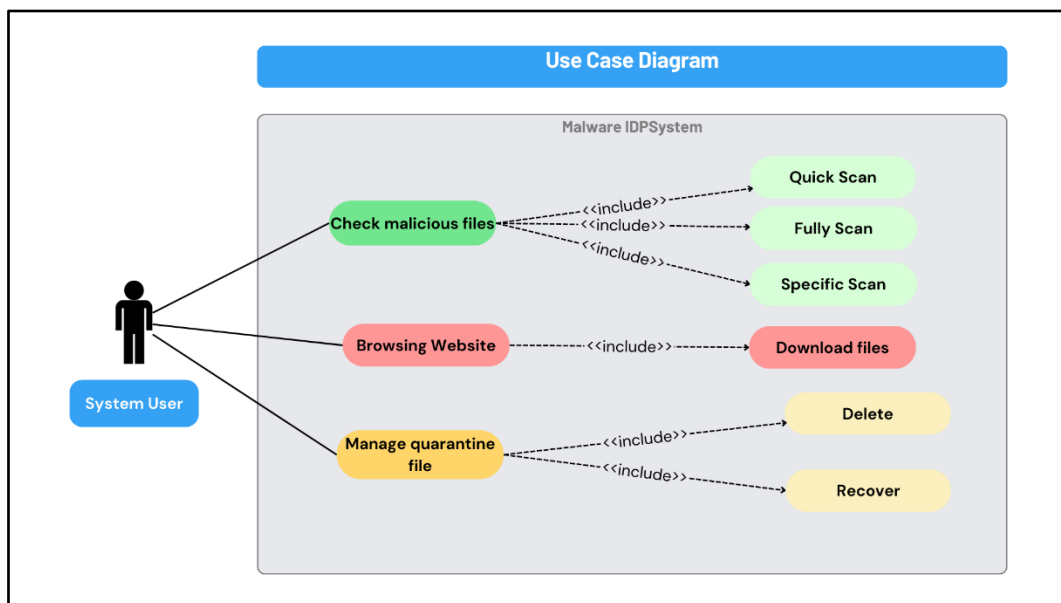


Figure 3.2 Use Case Diagram

Based on the Figure 3.2, here is a description of each use case:

1. Run a Malware Scan: To find any dangerous files on the device, the user launches a scan. The technology finds possible dangers by scanning designated directories.

2. Monitor Web Browsing: The system keeps track of the URLs that users visit and compares them to databases of websites that are known to be harmful. The technology prevents access to the website if it finds a threat.

3. Isolate Malicious Files: To stop malware from damaging the device, the system quarantines the file after detecting it.

3.1.3 Activity Diagram

The processes for several important system functions, including file isolation and virus detection, are shown in the activity diagrams below. These flowcharts illustrate the sequential steps that the system takes to guarantee that the user is in a secure environment. Below is Figure 3.3, an Activity Diagram for Malware Scanning, showing the relevant process.

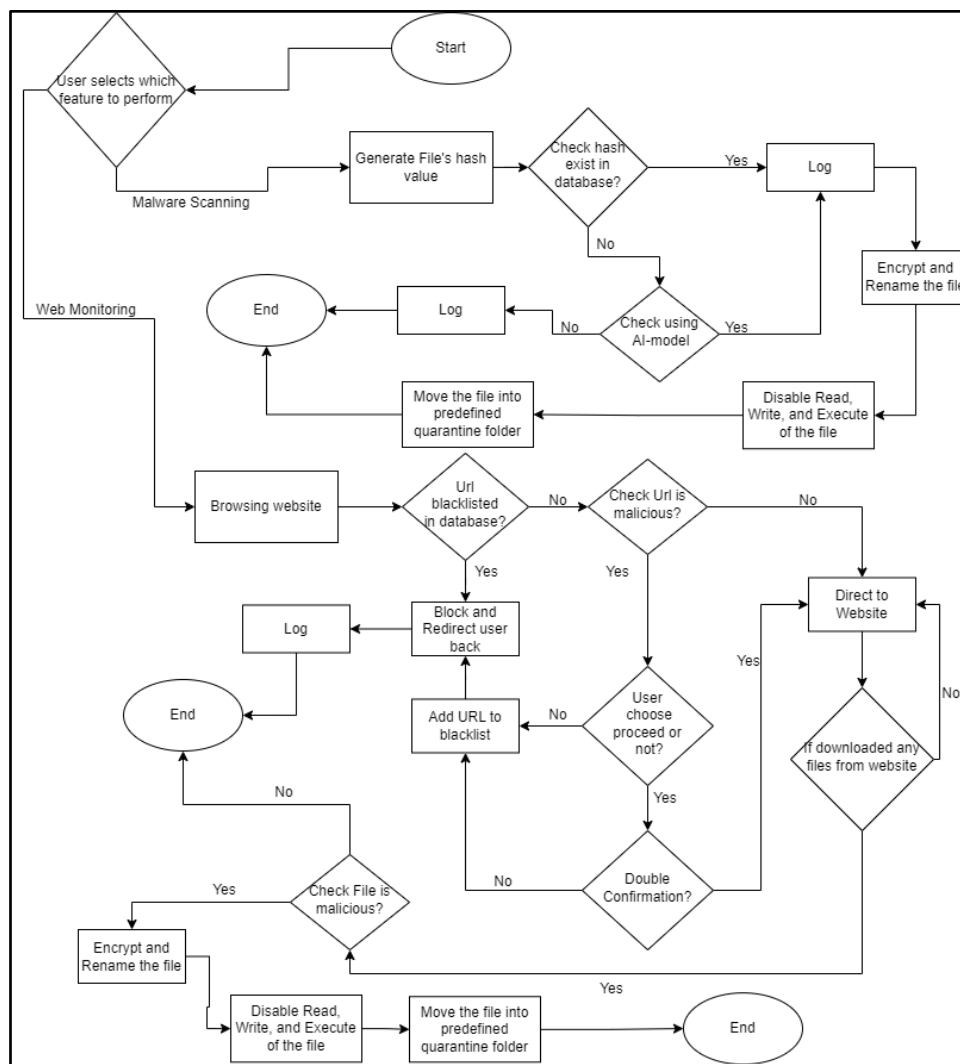


Figure 3.3 Activity Diagram for Malware IDPS

Based on Figure 3.3, it visually represents the steps involved in protecting a computer from malware through scanning and web monitoring. It starts with the system checking each file and website against a known list of threats. If a threat is detected, actions such as quarantining the file or blocking the website are taken to prevent harm. This ensures that the user's device remains safe from malicious attacks without requiring deep technical knowledge.

3.2 Summary

Chapter 3 describes the methodology and approach used in the development of the Malware IDPS system. It includes system architecture and design, as well as key processes such as file scanning, web browsing monitoring, and file quarantine. The use case and activity diagrams demonstrate the system's main functions and how users interact with the system to detect and isolate malware. The chapter also covers the architecture of various system components, including the scanning engine, quarantine manager, and URL analysis module.

Chapter 4:

System Design

4.1 System Block Diagram

A top-down view of the IDPS is given by the system block diagram below, which shows the key parts and how they interact. This diagram represents the architecture of the complete system, illustrating the connections between several modules including the URL analysis, quarantine control, and scanning engine. Below is Figure 4.1, a System Block Diagram of Malware IDPS, showing the relevant process.

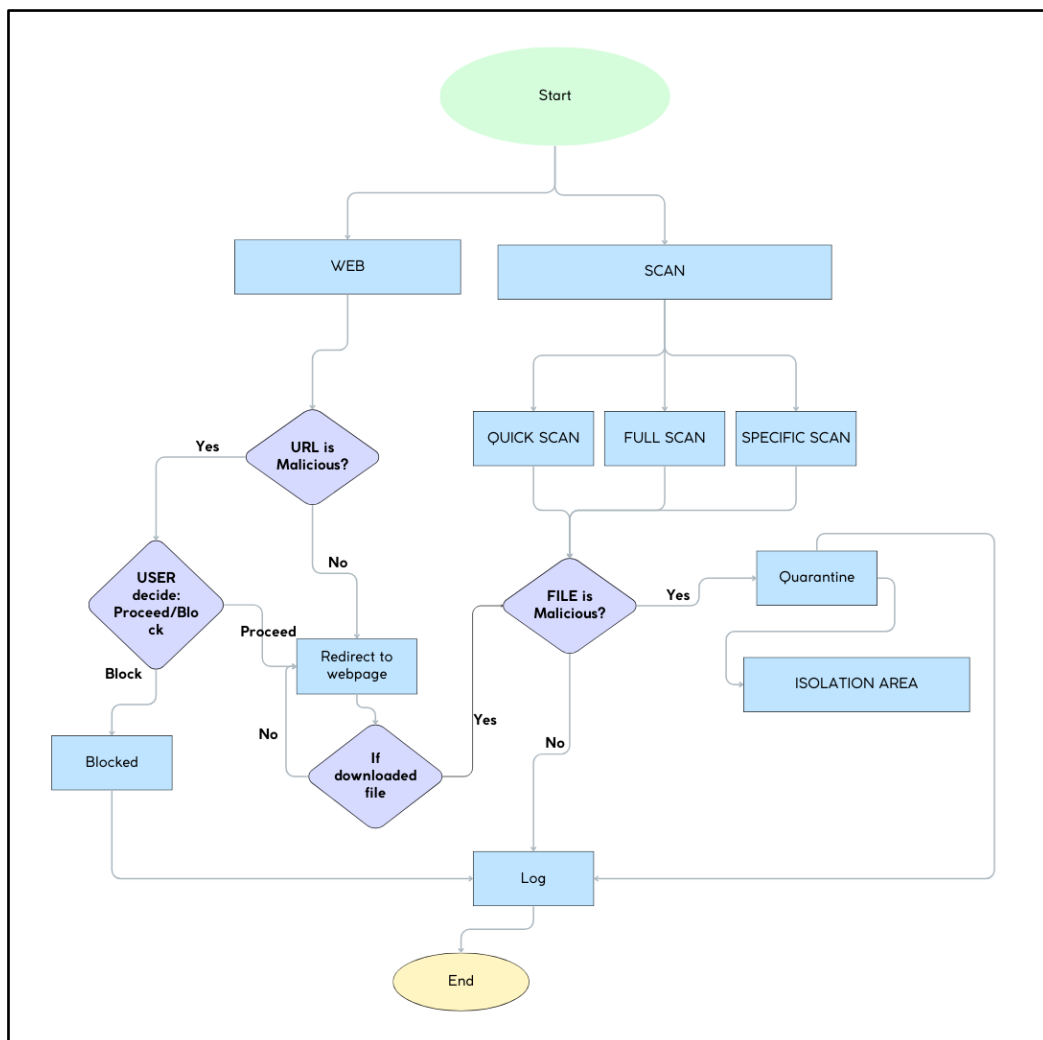


Figure 4.1 System Block Diagram of Malware IDPS

Based on Figure 4.1, every block in the diagram stands for an essential part of the system:

1. Scanning Engine: In charge of starting and overseeing the device's malware scans.
2. Quarantine Manager: Manages the safe storage and isolation of harmful files that have been identified.
3. URL Analysis Module: Checks for any security risks by tracking and analyzing URLs that the user accesses.

4.2 System Components Specifications

The specifications of every system component are described in depth in this section. It covers the database structure, implementation procedures, and descriptions of the classes and methods used in the Android application.

Classes and Methods

The functionality of the system depends on the classes and methods listed below:

1. MalwarePreventionSystem: Manages the system's overall performance, coordinating efforts amongst many parts to identify and stop malware.
2. DirectoryScanner: It searches the device's directories for any malware and takes additional action if any threats are found.
3. UrlAnalysis: Examines user-accessed URLs to find and prohibit dangerous websites.

Database

The system uses a SQLite database to store information about detected malware and analyzed URLs. The following SQL commands are used to create and manage the database.

SQL Commands for Malware Database:

```
CREATE TABLE malware_signatures (  
  id INTEGER PRIMARY KEY AUTOINCREMENT,  
  hash TEXT UNIQUE  
);  
CREATE INDEX index_hash ON malware_signatures (hash);  
INSERT INTO malware_signatures (hash) VALUES (?);  
SELECT hash FROM malware_signatures WHERE hash = ?;  
DELETE FROM malware_signatures WHERE hash = ?;
```

SQL Commands for Hostname Database:

```
CREATE TABLE hostnames (  
  id INTEGER PRIMARY KEY AUTOINCREMENT,  
  hostname TEXT,  
  status TEXT  
);  
INSERT INTO hostnames (hostname, status) VALUES (?, ?);  
SELECT status FROM hostnames WHERE hostname = ?;  
SELECT * FROM hostnames;  
DELETE FROM hostnames WHERE hostname = ?;
```

Implementation Steps

The steps that were taken to create and implement the system are outlined below:

1. Development Environment: Android Studio was used to create the system, and an Android emulator was used for testing.
2. Compilation and Deployment: To enable testing and deployment, the program was uploaded to an Android device after being compiled in Android Studio.

4.3 Circuits and Components Design

As this is a software project, this section focuses on the logical design of the software components rather than physical circuitry. The following flow diagrams illustrate the interactions between different classes and methods, highlighting the logical flow of data through the system. Below is Figure 4.2, a System Flow of Malware IDPS, showing the relevant process.

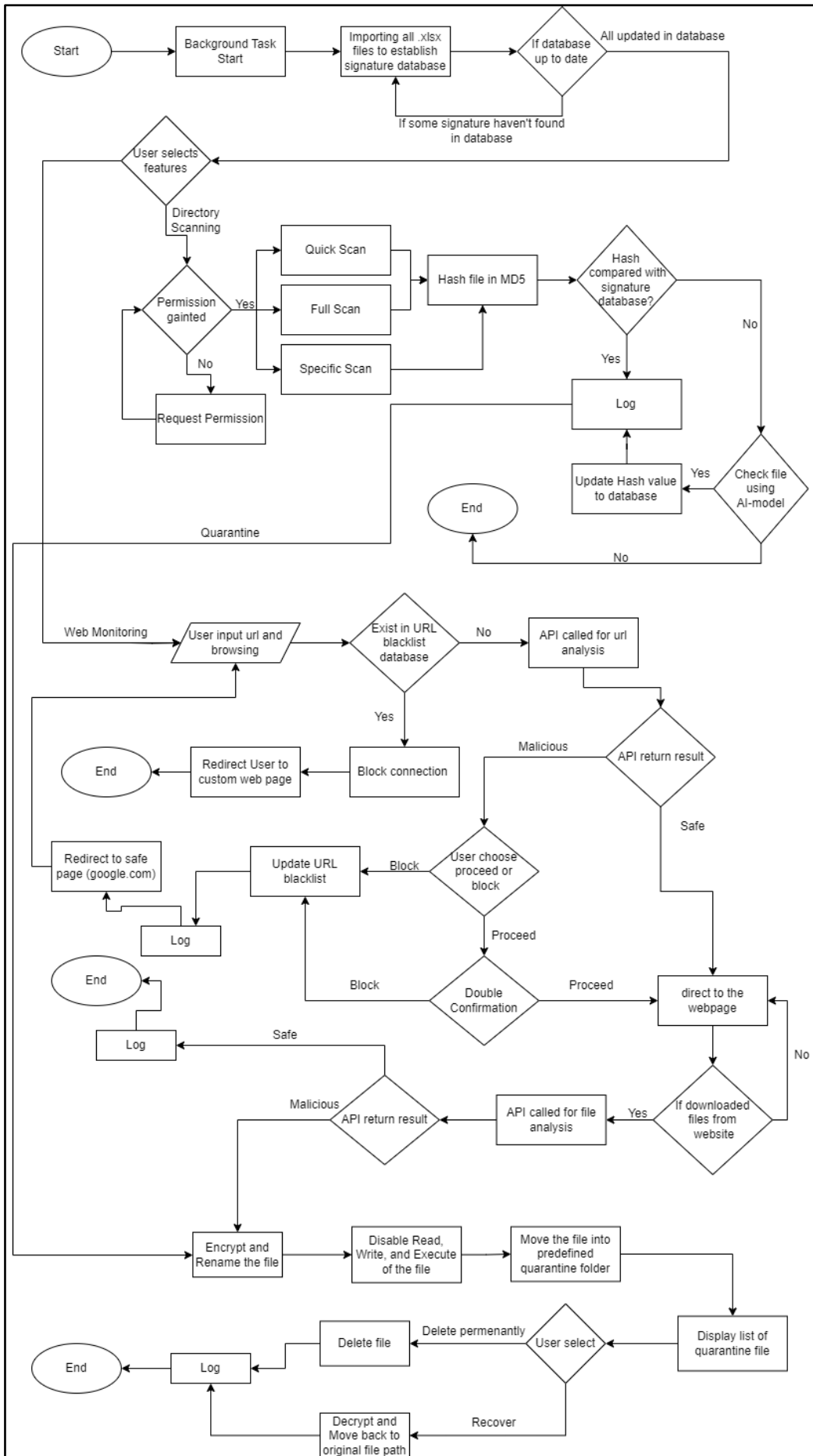


Figure 4.2 System Flow of Malware IDPS

Based on the Figure 4.2, it outlines the process of malware detection, prevention, and file quarantine within the system. The system begins with tasks such as scanning files, updating the signature database, and checking for permissions. When files or URLs are analyzed, the system compares them against known threats, either quarantining or logging malicious content if detected. If URLs are deemed unsafe, users are redirected, and malicious files are isolated for further action. The system allows users to either recover quarantined files or permanently delete them based on the analysis and user selection. This ensures comprehensive protection from malware, keeping both files and web activity secure. Below is Figure 4.3 and Figure 4.4, a Logical System Flow Charts of Malware IDPS, showing the relevant process.

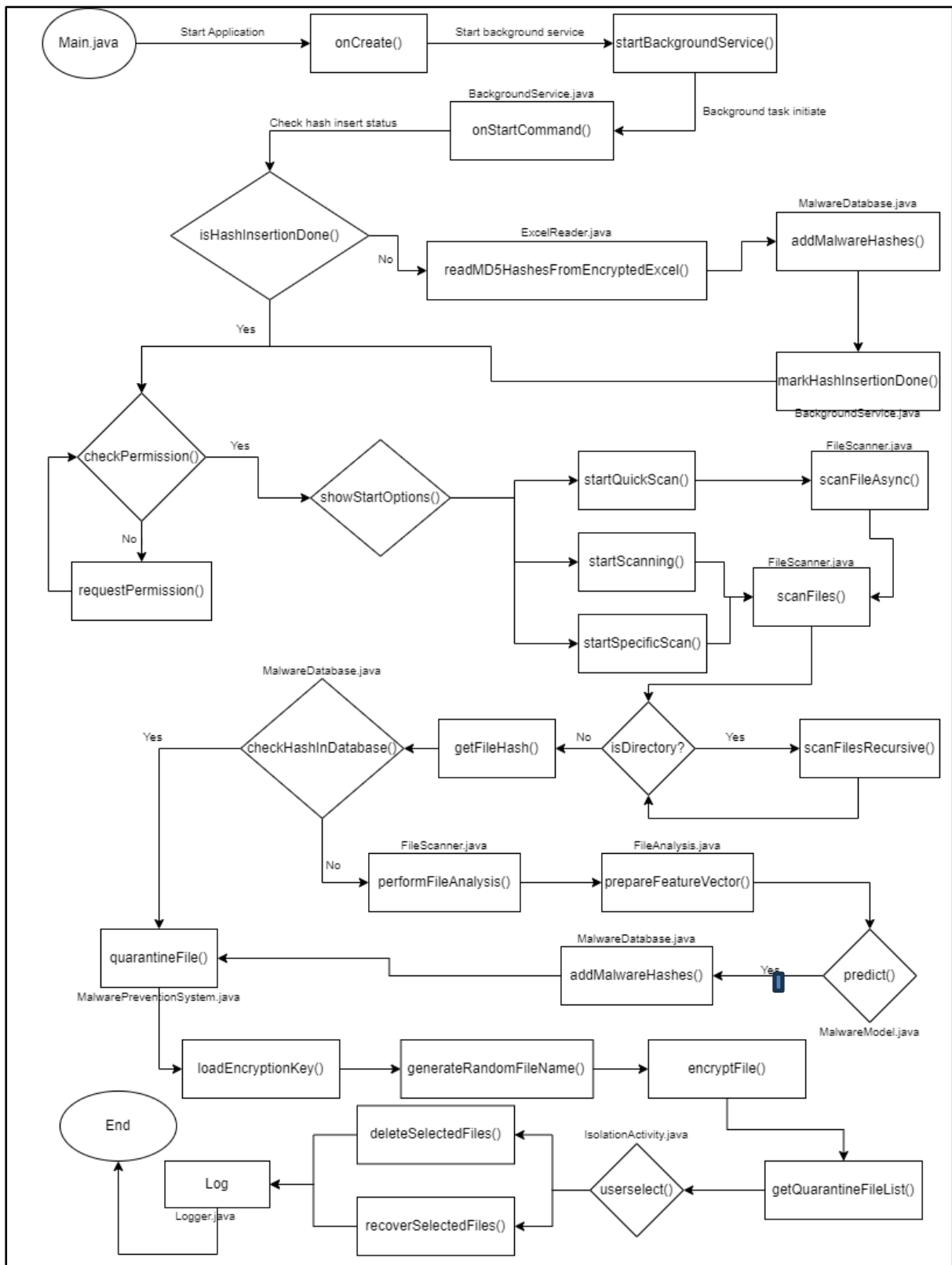


Figure 4.3 Logical Flow of Malware IDPS

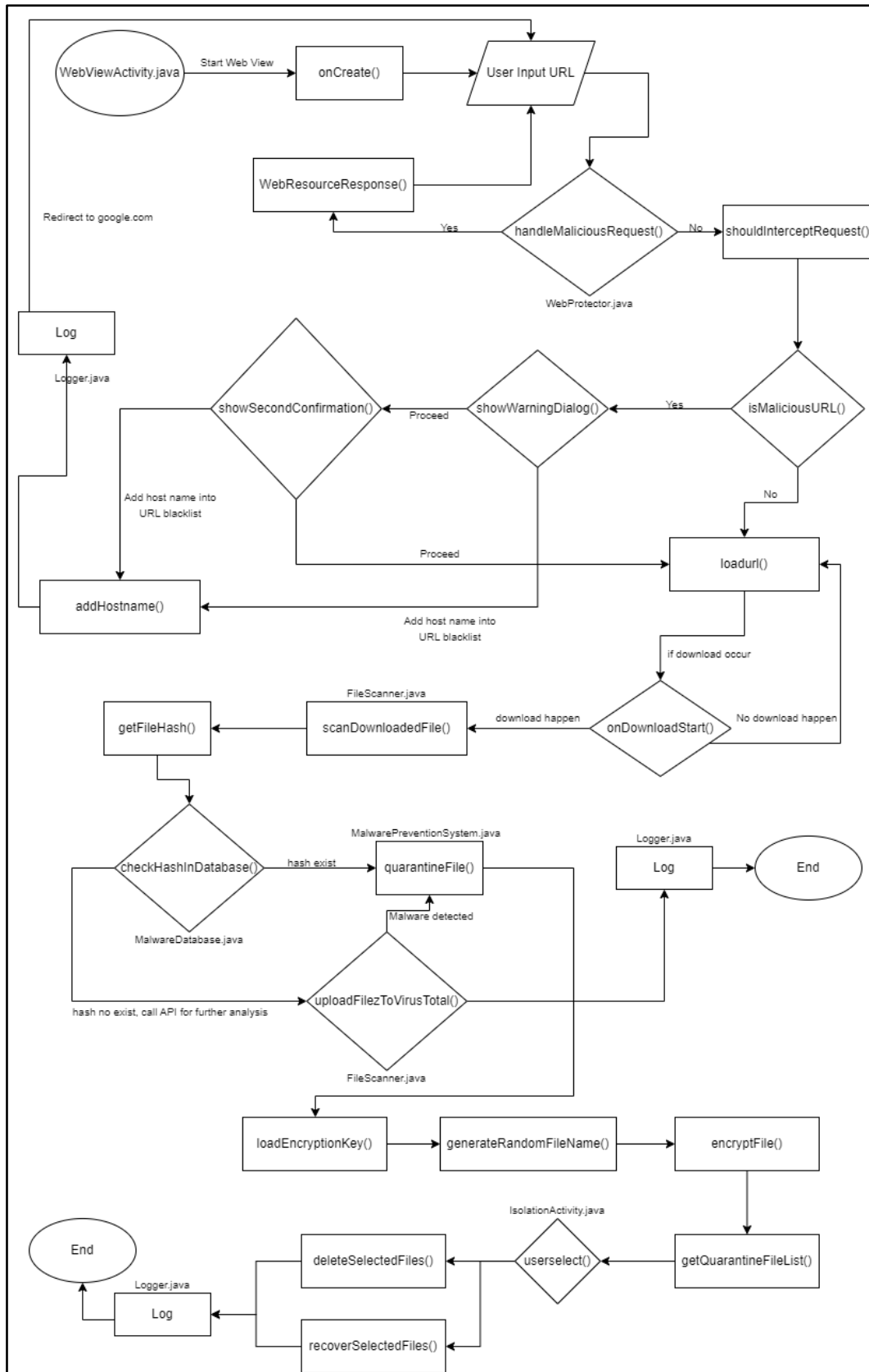


Figure 4.3 Logical Flow of Malware IDPS

4.4 System Components Interaction Operations

Based on the Figure 4.2 and 4.3, the functioning of the system's component parts is explained in this section. The steps that follow describe how important system functions including URL analysis, quarantine management, and virus scanning are carried out:

1. Scanning Engine Interaction: To identify and separate malware, the scanning engine communicates with the quarantine manager and file scanner.
2. Quarantine Management: Malware that has been identified is safely relocated to a quarantine area, where it is kept separate from the main system.
3. URL Analysis Interaction: The URL analysis module keeps track of all online activity, examines each URL, and prevents access to websites that are known to be harmful.

4.5 Summary

This chapter focuses on the detailed design of the Malware IDPS system. It includes system block diagrams and logical flowcharts that depict how different components of the system interact to perform malware detection and prevention. The chapter outlines the core components and explains their roles in ensuring the security of mobile devices by scanning files, updating malware signatures, and quarantining malicious content. It also covers the design and interaction of the system's scanning, quarantine management, and URL analysis features.

Chapter 5:

System Implementation

5.1 Hardware Setup

The hardware components used in this project are a computer and an Android emulator. Python programming is developed on the computer to run within the Android environment, as well as machine learning models to identify malware. Tests and deployments of this malware detection and prevention system are conducted on a mobile device. Below is the Table 1.1, Specifications of System showing relevant data.

Table 1.1 Specifications of System

Description	Specifications
Model	Msi Katana GF66
Processor	Intel Core i5-11400H
Operating System	Android 14
Graphic	NVIDIA GEFORCE 3060 6GB
Memory	8GB DDR4 RAM
Storage	25GB SSD

Based on the Table 1.1, it provides the specifications of the laptop used to run an Android emulator for the project. The laptop is an MSI Katana GF66, equipped with an Intel Core i5-11400H processor, an NVIDIA GEFORCE 3060 graphics card (6GB), 8GB of RAM, and 25GB of SSD storage. The system runs the Android 14 operating system through the Android Studio emulator, allowing for testing and development of mobile applications on a laptop.

5.2 Software Setup

In Malware IDPS, some of the software been used during development and real time detection. Below is the list of software been used by IDPS:

1. Android Studio Koala | 2024.1.1
2. Android Emulator: Pixel 5 API 34
3. Google Safe Browsing API
4. VirusTotal API

5. Jupyter Notebook (AI Model Training Used)

5.3 Setting and Configuration

In this section, we outline the critical setup and configuration steps required to enable key functionalities in the system, such as malware detection and URL analysis. These configurations ensure that the system can access necessary resources like Google Safe Browsing and VirusTotal for analyzing URLs and files, as well as handling specific Android permissions that enable file access and overlay display. Below are the steps for configuring the Google Safe Browsing API, VirusTotal API, and granting Android system permissions such as "All Files Access" and "Display Over Other Apps" to allow the smooth functioning of the application:

Google Safe Browsing API Configuration:

1. Create a Google Cloud Project: Log into your Google Cloud account and access the project dashboard.
2. Create a New Project: Click the "Create Project" button and name the project "Safe-Browsing-API".
3. Navigate to API & Services: Go to the API & Services dashboard within your Google Cloud console.
4. Enable Google Safe Browsing API: Search for 'Google Safe Browsing API' and enable it for your project.
5. Create API Key: In the API & Services dashboard, create credentials by selecting "API key". This key will be used to authenticate your requests to the Safe Browsing API.

VirusTotal API Configuration:

1. Login to VirusTotal Account: Log in to your VirusTotal account or create an account if necessary.
2. Access API Key: Navigate to the "API Key" section in your account settings to retrieve the key required for API integration.

All Files Access Permission (Android):

1. Open Settings: In your Android emulator or physical device, open the "Settings" app.
2. Privacy Section: Scroll down and select "Privacy".
3. Permission Manager: Under the Privacy section, tap on "Permission Manager".

4. Files and Media Permission: Scroll down to "Files and Media" and tap on it.
5. Select Application: In the list of applications, find and select your app (e.g., "Malware IDPSystem").
6. Grant Permission: Tap "Allow access to manage all files" to grant your app permission to manage all files, which is necessary for malware scanning.

Display Over Other Apps Permission (Android):

1. Open Settings: In the Android emulator, open the "Settings" app.
2. Search for Special App Access: Use the search bar in the Settings app to find "Special App Access". Alternatively, navigate to "Apps" > "Special App Access" manually.
3. Display Over Other Apps: Scroll down to find "Display over other apps" and tap on it.
4. Select Your Application: Find your app in the list and tap on it.
5. Enable Permission: Toggle the option for "Allow display over other apps" to enable the app to display information as an overlay, which is crucial for user warnings and notifications.

5.4 System Operation

In this section, the system operation will be demonstrated for each of the main features developed in this system, including the Scanning Features, Web Browsing Monitoring Features, and Malicious Files Quarantine Features, along with their operation results. Below is the operation and demo result for each of the main features developed in this system.

Below are Figure 5.1 and 5.2, which show the Scanning Options and Results (Safe).

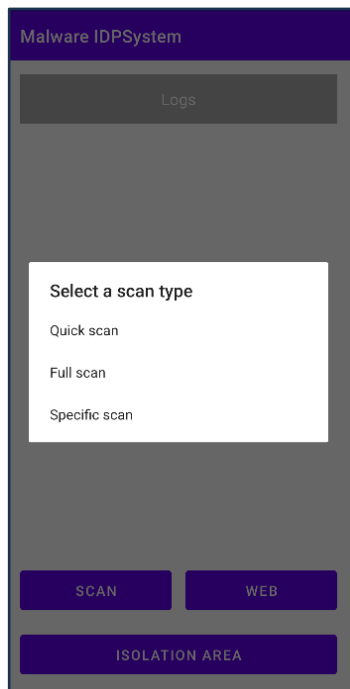


Figure 5.1 Scanning Options

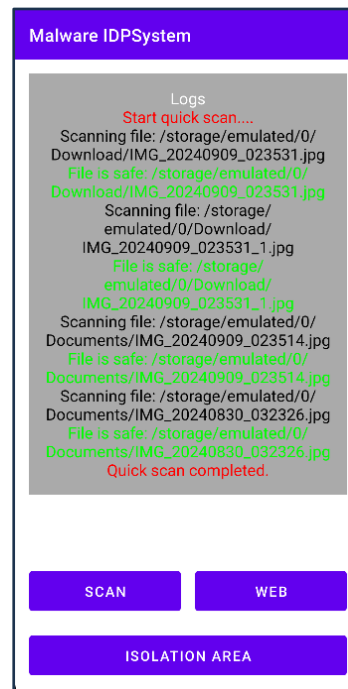


Figure 5.2 Quick Scan Result

Based on the Figure 5.1 and 5.2 above, it shows that the Quick Scan was selected as the scanning method, and the operation result for the Quick Scan has been initiated.

Below are Figures 5.3 and 5.4, which show the Scanning Options and Results (Malicious).

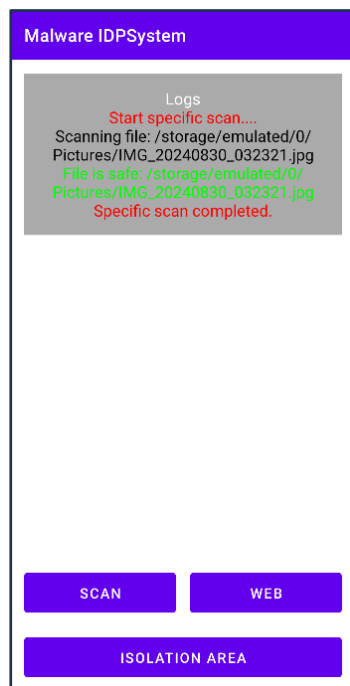


Figure 5.3 Specific Scan Result

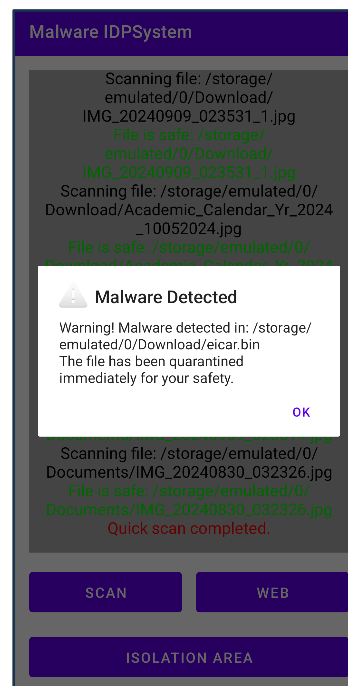


Figure 5.4 Malicious Detected

Based on the Figure 5.3 and 5.4, it shows that when a malicious file is detected, a message will pop up, and the file will be quarantined immediately.

Below are Figure 5.5 and 5.6, which show the Web Monitoring and Results (safe).

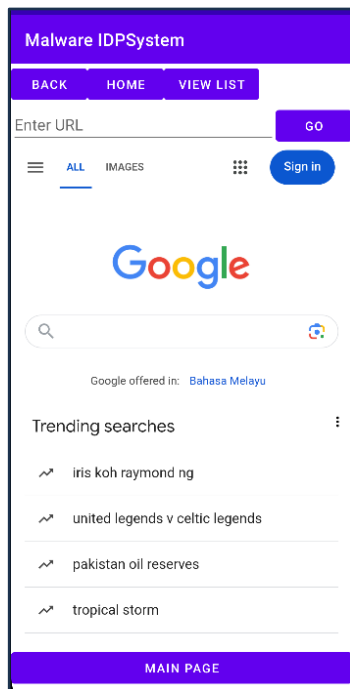


Figure 5.5 Default Page

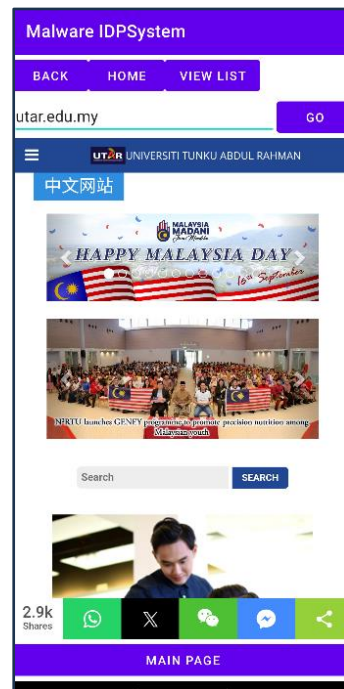


Figure 5.6 Normal Web Browse

Based on the Figure 5.5 and 5.6 shown above, the website is able to show without any blocking when the URL is not detected as malicious URL.

Below are Figure 5.7 and 5.8, which show the Web Monitoring and Results (Malicious).

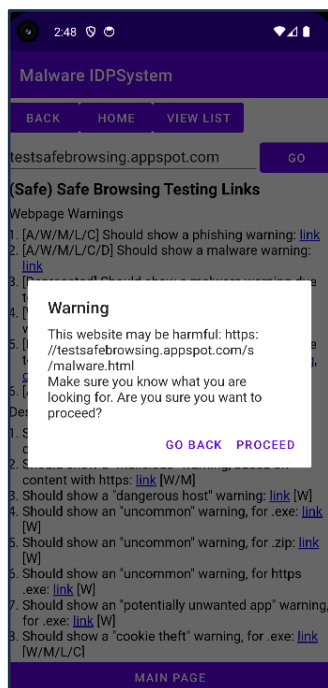


Figure 5.7 Malicious Web Browse

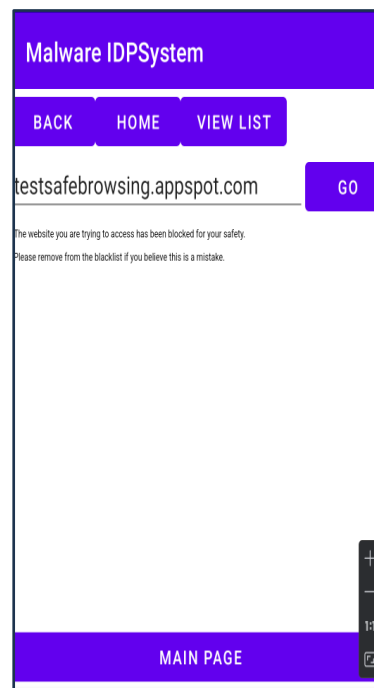


Figure 5.8 Blocked Website

Based on the Figure 5.7 and 5.8, the website is able to show without any blocking when the URL is not detected as malicious URL.

Below are Figure 5.9 and 5.10, which show the Web Download Files and Results (safe).

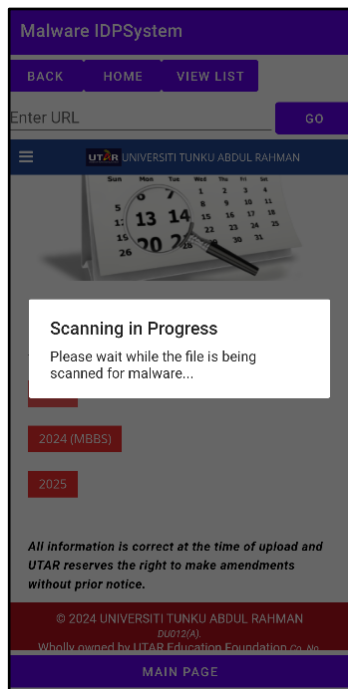


Figure 5.9 Downloading File

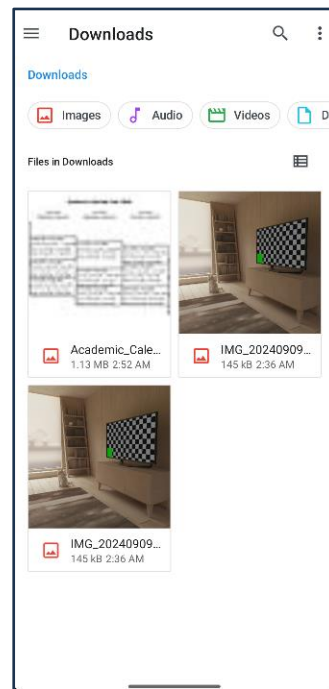


Figure 5.10 File detected as Safe

Based on the Figure 5.9 and 5.10 above, when a file is downloaded during web browsing, it will be scanned immediately. If no issues are found, the user can access the file in the folder without any problems.

Below are Figure 5.11 and 5.12, which show the Web Download Files and Results (malicious).

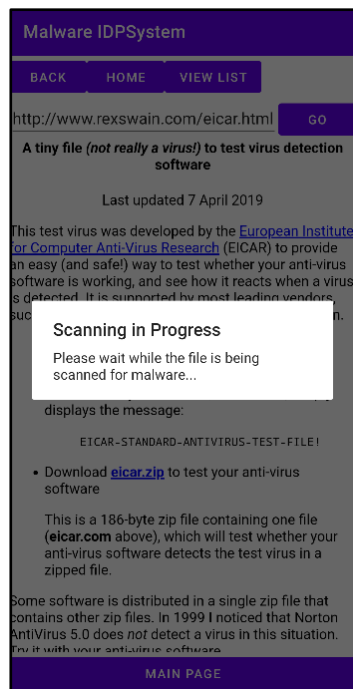


Figure 5.11 Downloading File

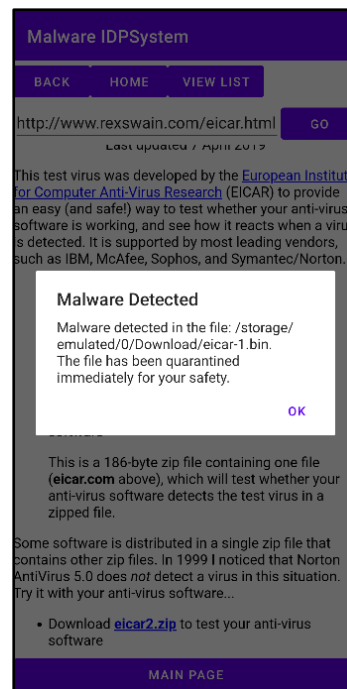


Figure 5.12 File Detected as Malicious

Based on the Figure 5.11 and 5.12 above, when a downloaded file is detected as malicious, it will be quarantined immediately to prevent the user from executing the file on their system.

Below are Figure 5.13 and 5.14, which show the Quarantine Page and Results:

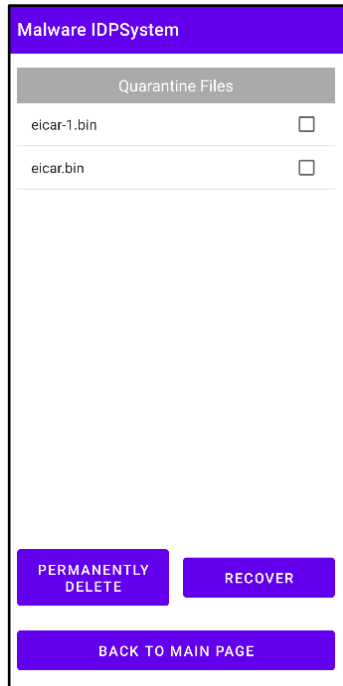


Figure 5.13 Quarantine File List

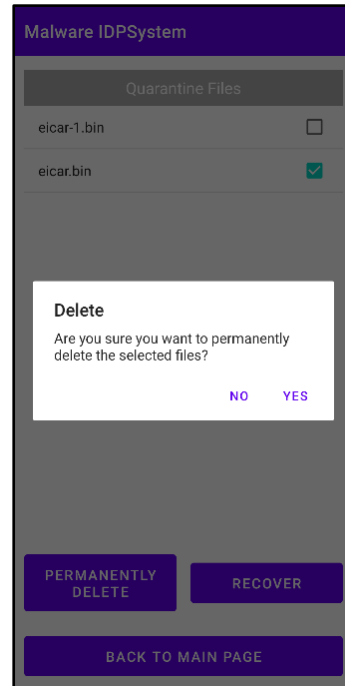


Figure 5.14 Remove File Permanently

Based on the Figure 5.13 and 5.14, the results above show that the user can choose to permanently delete the quarantined file or recover it if it was falsely detected as malicious.

5.5 Implementation Issues and Challenges

Throughout the system's implementation, numerous challenges emerged that necessitated extensive troubleshooting and adjustments. One of the main issues involved the signature detection process, which resulted in both memory leaks and increased latency. As the system handled a large volume of malware signatures, it became clear that the memory management was inefficient, leading to leaks that negatively impacted overall performance. These problems were compounded by latency during file scans, which caused delays in malware detection and prolonged the time required to process large file sets.

Another hurdle was the slow URL loading in the WebProtector feature. This slowdown primarily occurred when the system queried external APIs, like VirusTotal and Google Safe Browsing, to verify URLs. Since the system had to wait for these external responses before determining whether to allow or block a URL, it considerably slowed down browsing, resulting in a poor user experience.

Additionally, Android's limitations on network monitoring created a significant obstacle. The initial plan was to use a VPN service to capture and analyze network packets, identify potential threats, and reroute the traffic back. However, due to constraints in Android's VPN implementation, the system failed to properly reroute traffic post-analysis, making this approach impractical. Consequently, an alternative method using WebView for URL detection and protection was adopted, eliminating the need for direct packet analysis.

Lastly, the system encountered memory and CPU overload when importing large malware signature databases. The sheer volume of signatures, often housed in encrypted .xlsx files, led to excessive resource consumption during the import process. This triggered spikes in both memory and CPU usage, further degrading system performance and requiring optimization to handle large datasets more efficiently.

5.6 Concluding Remark

The implementation process generally proceeded smoothly, though a few challenges were encountered along the way. Thanks to effective troubleshooting and additional effort, the system was developed nearly according to plan. The only area where it diverged from the original design was in network monitoring, where Android's VPN limitations caused complications. However, adopting WebView for URL detection proved to be an effective alternative. All other system components functioned as intended, with some even surpassing initial expectations.

Reflecting on the outcome of the Malware IDPSystem, the development stayed on track and was completed within the expected timeframe. In my view, the system is capable of handling a significant portion of real-world tasks for malware protection, demonstrating its strong potential for practical application.

For future enhancements, a key improvement would be to continuously update the database. While the current system includes hundreds of thousands of hash values, expanding this database over time would improve detection accuracy and further enhance the system's ability to identify malware.

Chapter 6:

System Evaluation And Discussion

6.1 System Testing and Performance Metrics

In this section, multiple tables are given to show the system testing and performance metrics results for this system.

Below is the table of Malicious URL Detection Testing result:

Table 2.1 Malicious URL Detection Testing Result

URL	EXPECTED	ACTUAL	ACCURACY
https://utar.edu.my/	Allow	Allow	Correct
https://testsafebrowsing.appspot.com/	Block	Allow	False Negative
https://testsafebrowsing.appspot.com/malware.html	Block	Block	Correct

Based on the results in Table 2.1, the URL was successfully blocked when the malware.html page was detected as malicious. However, the page containing multiple malicious URLs was not detected or blocked, which could pose a potential risk when a regular user accesses that page.

Below is the table of Download File Detection Testing result:

Table 2.2 Download File Detection Testing Result

FILE NAME	EXPECTED	ACTUAL	ACCURACY
eicar.com	Block	Block	Correct
eicar.zip	Block	Block	Correct
eicar2.zip	Block	Block	Correct
Acedemic_Calendar_Yr_2024_10052024.jpg	Allow	Allow	Correct

As tested, Table 2.2 shows that the download file detection test was successfully executed for both malicious and normal files, including multiple file types such as zip, jpg, and bin files.

Below is the table of Resource Utilization Testing result:

Table 2.3 Resource Utilization Testing Result

OPERATION	CPU	MEMORY (MB)
IDLE	1%	77.9
QUICK SCAN (NO DETECTED)	39%	85.1
QUICK SCAN (DETECTED)	40%	85.8
FULL SCAN (NO DETECTED)	40%	88.9
FULL SCAN (DETECTED)	43%	93.9
SPECIFIC SCAN (NO DETECTED)	24%	89.1
SPECIFIC SCAN (DETECTED)	28%	91.5
START WEB SERVICE	41%	136.5

Based on Table 2.3, the system's resource consumption was planned to remain at a stable level. Tasks such as quick scan, full scan, specific scan, and web service did not overload the CPU or memory, with most tasks keeping CPU usage around 50%, which is considered optimal.

6.2 Testing Setup and Result

The system underwent testing in three main areas: Malicious URL Detection, Download File Detection, and Resource Utilization. The tests involved a combination of real-world and synthetic scenarios on an Android emulator. The setup included enabling network and file access permissions, configuring the Google Safe Browsing and VirusTotal APIs, and providing necessary file access and overlay display permissions in the Android environment.

Malicious URL Detection Testing:

This test evaluated the system's effectiveness in identifying and blocking malicious URLs. It successfully allowed legitimate URLs and blocked some malicious ones as expected. However, one false negative was encountered with a test URL (<https://testsafebrowsing.appspot.com/>), where the system incorrectly allowed a URL that should have been blocked. This result indicates a potential area for improvement in enhancing the accuracy of URL detection. Below is the Figure 6.1 to showing the relevant data.

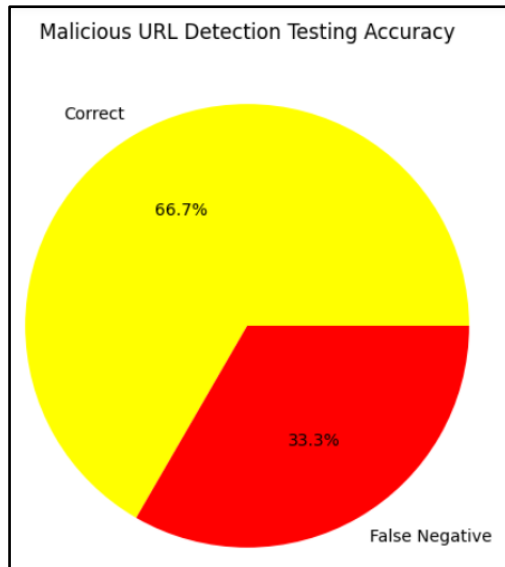


Figure 6.1 Malicious URL Testing Accuracy Chart

Based on Figure 6.1, it shows the accuracy of Malicious URL Testing, including correct results and false negatives. The detected malicious URLs achieved over 66% accuracy, with only around 33% resulting in false negatives, according to the testing results.

Download File Detection Testing:

The system was evaluated using a range of test files, including the well-known EICAR test files. In these tests, all malicious files were successfully blocked, while non-malicious files were allowed without issue. The results were fully in line with expectations, demonstrating a 100% accuracy in detecting and processing both malicious and safe files. Below is Figure 6.2 that shows the relevant data.

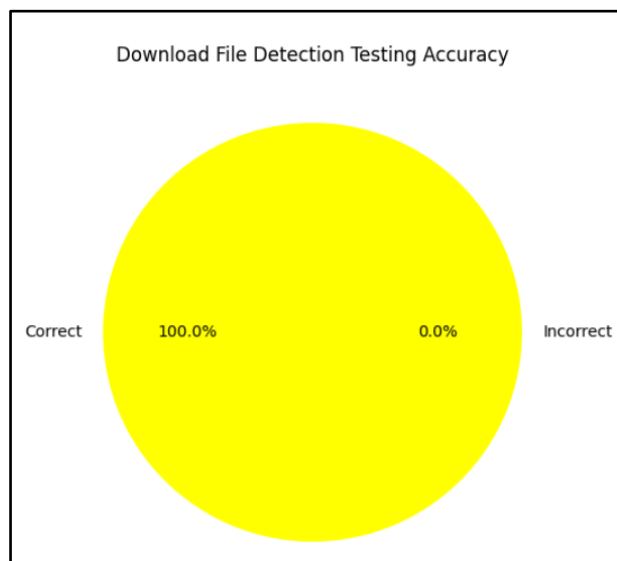


Figure 6.2 Download File Detection Testing Accuracy Chart

Based on the results of Figure 6.2, it shows that downloaded file detection during web browsing achieved significant success in testing, successfully identifying both malicious and normal files.

Resource Utilization Testing:

The resource utilization tests assessed the system's impact on CPU and memory usage during different tasks, including idle states, quick scans, full scans, and web service initialization. CPU usage showed a notable increase during full scans and web service startup, while memory usage remained generally stable, with the exception of a spike observed when the web service was initiated. Overall, the system demonstrated efficient resource management across most tasks. Below is the Figure 6.3 that showing relevant data.

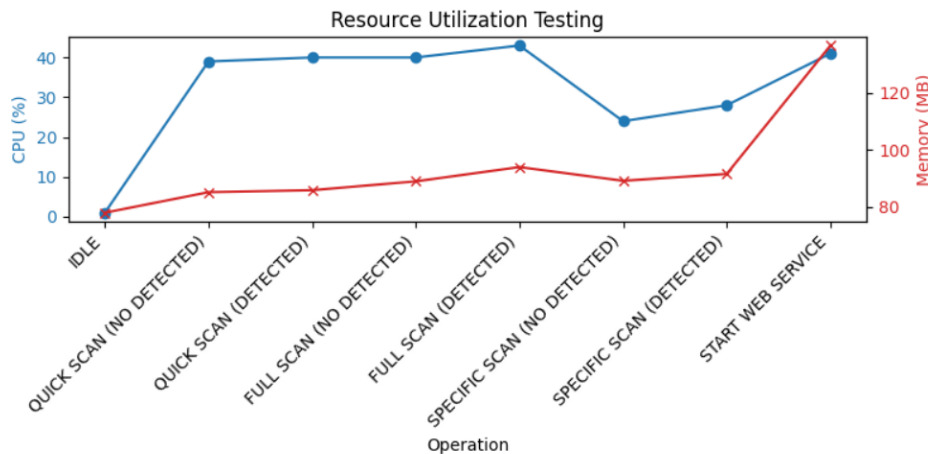


Figure 6.3 Resource Utilization Testing Chart

Based on Figure 6.3, it shows that the system was able to run smoothly with CPU usage remaining around 50% in various scenarios, including IDLE, Scanning, and Web Service. Additionally, memory usage was well managed to prevent app crashes and ensure consistent performance.

6.3 Project Challenges

This project encountered a range of both technical and non-technical challenges. A major technical challenge was monitoring network traffic. Although the system was able to capture traffic successfully, reintegrating it into the network flow proved challenging, resulting in internet outages during emulator testing. Additionally, the system's ability to handle URL

protection was tested by the large volume of URLs transmitted during communication. The need to examine each URL without introducing significant latency posed another major hurdle. Setting up APIs to capture, filter, and verify URL responses was problematic as well, since any errors during the extraction process could lead to failures in malware detection. Importing large datasets also led to memory and CPU overloads, which caused the emulator to crash.

Non-technical challenges included the need to develop a clear and effective implementation plan, as well as choosing the right algorithms and system flow to integrate detection and prevention methods in the Malware IDPS. Time management emerged as a significant issue, with technical problems often requiring more time to resolve than anticipated—sometimes taking over a week to address a single issue.

Had these challenges not been overcome, users might have experienced substantial latency while browsing websites, and the system might have failed to properly detect and block malware from downloads or malicious websites. While the planned VPN-based network monitoring feature could not be successfully implemented, the project adapted by switching to WebView for URL detection and protection. As part of this adaptation, a new feature was added that allows the system to scan all downloaded files during browsing, further enhancing the protection provided to users.

6.4 Objectives Evaluation

Upon reviewing the objectives set during the initial phase of this final year project, it is clear that most have been successfully achieved, though a few challenges remain. The first objective—identifying and detecting malware—was largely met through the implementation of signature-based detection and AI techniques. The system proved effective in detecting known malware, but it struggled with identifying newly emerging malware due to limitations in the detection techniques.

The second objective, which was to develop an effective Intrusion Detection and Prevention System (IDPS), was fulfilled by using various scanning and prevention methods. These methods resulted in a fully functioning system.

The third objective—exploring countermeasures against mobile malware—was achieved through the implementation of a prevention system that quarantines detected files. This system

renames, encrypts, and restricts all read, write, and execute capabilities of the files. However, updating the database signature to match current market standards was limited by the lack of free, up-to-date data available on the internet.

6.5 Concluding Remark

In conclusion, the system developed during this project has demonstrated strong capabilities in detecting and preventing malware. It achieved a high level of accuracy in both file and URL detection. The combination of signature-based detection and AI techniques enabled robust identification of known threats, and the quarantine mechanism effectively protected the system by isolating malicious files. Even under stress—during operations like full scans and web service initialization—the system maintained stable CPU and memory usage.

However, there are still some limitations. The system faces challenges in detecting newly emerging malware, and network monitoring via VPN could not be fully implemented. Additionally, accessing a frequently updated signature database remains a challenge due to the limited availability of free resources. Despite these constraints, the system is largely ready for real-world deployment, with opportunities for future improvements, such as enhancing detection accuracy for new malware and refining the network monitoring functionality. This project highlighted the importance of balancing system performance with comprehensive protection, and the need for regular testing and updates to keep pace with evolving security threats.

Chapter 7:

Conclusion and Recommendation

7.1 Conclusion

This project aimed to strengthen mobile security by developing an Intrusion Detection and Prevention System (IDPS) capable of detecting and preventing malware on Android devices. Throughout the development phase, the system combined various detection techniques, including signature-based and AI-based methods, to offer a comprehensive solution for malware detection. Most of the project's objectives were successfully met, with the system efficiently identifying known malware and quarantining infected files through encryption and isolation. Although some challenges, such as the inability to fully implement network monitoring and difficulty detecting newly emerging malware, were encountered, the system performed well in testing. It maintained stable resource utilization, and detection accuracy was high for file-based malware and certain URLs. The project also provided valuable insights into managing large datasets and integrating external APIs, such as Google Safe Browsing and VirusTotal, for real-time threat detection.

Key lessons from this project include the necessity of keeping detection methods continuously updated to adapt to evolving malware threats, and the importance of optimizing resource usage in mobile environments. Effective time management and troubleshooting were essential in overcoming obstacles, ensuring the development process remained smooth despite its complexities. With further enhancements, especially in improving detection techniques for unknown malware, the system shows strong potential for real-world application in safeguarding mobile devices.

7.2 Recommendation

While the core features of an IDPS for mobile devices were successfully implemented, there remains room for improvement and future development. First, the system could benefit from incorporating more advanced AI models to enhance detection of newly emerging malware, addressing current limitations in signature-based detection. Training the AI model on a larger dataset would improve its predictive accuracy. Second, further stress testing and security penetration tests are recommended to assess the system's robustness under real-world

conditions. This would ensure the system can handle higher malware volumes and network traffic without degrading performance.

Another suggestion is to broaden the system's scope by exploring alternative methods for network monitoring, given the limitations encountered with the VPN-based approach in this project. Enhancing the system's database by continuously integrating up-to-date malware signatures would improve its detection accuracy and overall effectiveness. Collaborating with external cybersecurity databases or research institutions could also provide access to more comprehensive signature data. Finally, improving the user interface and user experience would make the system more user-friendly for non-technical users, increasing its practical value and encouraging wider adoption.

REFERENCES

- [1] A. Petrosyan. “Annual number of malware attacks worldwide from 2015 to 2022.” Satista. <https://www.statista.com/statistics/873097/malware-attacks-per-year-worldwide/> (accessed Aug 24, 2023)
- [2] A. Petrosyan. “Distribution of cyber attacks across worldwide in industries in 2022.” Satista. <https://www.statista.com/statistics/1315805/cyber-attacks-top-industries-worldwide/> (accessed Aug 24, 2023)
- [3] A. Petrosyan. “Average weekly number of malware attacks in organizations worldwide in 2022, by industry.” Satista. <https://www.statista.com/statistics/1377217/average-weekly-number-attacks-global-by-industry/> (accessed Aug 24, 2023)
- [4] A. Kivva. “IT threat evolution Q1 2023. Mobile statistics.” SECURELIST. <https://securelist.com/it-threat-evolution-q1-2023-mobile-statistics/109893/> (accessed Aug 24, 2023)
- [5] A. Petrosyan. “Distribution of mobile malware worldwide in 2022, by type.” Satista. <https://www.statista.com/statistics/653688/distribution-of-mobile-malware-type/> (accessed Aug 24, 2023)
- [6] N. James. “30+ Malware Statistics You Need To Know In 2023.” Astra. <https://www.getastra.com/blog/security-audit/malware-statistics/#:~:text=all%20cyber%20attacks,-,There's%20a%20cyber%20attack%20every%2039%20seconds,of%20malware%20are%20created%20daily.> (accessed Aug. 23, 2023)
- [7] T. Shishkova. “The mobile malware threat landscape in 2022.”. SECURELIST. <https://securelist.com/mobile-threat-report-2022/108844/#:~:text=Distribution%20of%20attacks%20by%20type%20of%20software%20used,-Distribution%20of%20attacks&text=Similarly%20to%20previous%20years%2C%202022,2.38%25%20in%202021%2C%20respectively.> (accessed Aug 2, 2023)
- [8] W. Baig. “Ten Crucial Privacy Statistics That May Surprise You.”. SECURITY TODAY. <https://securitytoday.com/articles/2020/05/18/ten-crucial-privacy-statistics-that-may-surprise-you.aspx>
- [9] V. Nair. “Nearly RM40,000 lost to scammers within minutes of mobile app use.”. TheStar. <https://www.thestar.com.my/metro/metro-news/2022/08/29/nearly-rm40000-lost-to-scammers-within-minutes-of-mobile-app-use>
- [10] Kaspersky. “Mobile Malware.”. kaspersky. <https://usa.kaspersky.com/resource-center/threats/mobile-malware>
- [11] V. KOULIARIDIS, K. BARMPATSA LOU, G. KAMBOURAKIS, and S. CHEN, “A Survey on Mobile Malware Detection Techniques,” *IEICE Transactions on Information and Systems*, vol. E103.D, no. 2, pp. 204–211, Feb. 2020, doi: <https://doi.org/10.1587/transinf.2019ini0003>.

- [12] E. Gandotra, D. Bansal, and S. Sofat, “Malware Analysis and Classification: A Survey,” *Journal of Information Security*, vol. 05, no. 02, pp. 56–64, 2014, doi: <https://doi.org/10.4236/jis.2014.52006>.
- [13] Abdelfattah Amamra, Chamseddine Talhi, and J.-M. Robert, “Smartphone malware detection: From a survey towards taxonomy,” Oct. 2012, doi: <https://doi.org/10.1109/malware.2012.6461012>.
- [14] A. Sharifi, F. F. Zad, F. Farokhmanesh, A. Noorollahi, and J. Sharif, “An Overview of Intrusion Detection and Prevention Systems (IDPS) and Security Issues,” *IOSR Journal of Computer Engineering*, vol. 16, no. 1, pp. 47–52, 2014, doi: <https://doi.org/10.9790/0661-16114752>.
- [15] MalayMail. “In N. Sembilan, housewife loses RM480,000 to investment scam.” MalayMail. <https://www.malaymail.com/news/malaysia/2022/11/22/in-n-sembilan-housewife-loses-rm480000-to-investment-scam/41343>
- [16] A. Petrosyan. “Number of detected malicious installation packages on mobile devices worldwide from 4th quarter 2015 to 3rd quarter 2023.” Statista. <https://www.statista.com/statistics/653680/volume-of-detected-mobile-malware-packages/#:~:text=During%20the%20third%20quarter%20of,the%20first%20quarter%20of%202021.> (accessed March 3, 2024)
- [17] A. Kivva. “IT threat evolution Q2 2023. Mobile statistics.” SECURELIST. <https://securelist.com/it-threat-evolution-q2-2023-mobile-statistics/110427/> (accessed Mar 3, 2024)
- [18] A. Kivva. “IT threat evolution Q3 2023. Mobile statistics.” SECURELIST. <https://securelist.com/it-threat-evolution-q3-2023-mobile-statistics/111224/> (accessed Mar 3, 2024)

APPENDIX

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Jun 2024	Study week no.: 2
Student Name & ID: LEOW YU HONG 2005558	
Supervisor: Dr Abdulrahman Aminu Ghali	
Project Title: Detecting Malware Attack in Mobile Phone using Intrusion Detection and Prevention System (IDPS)	

1. WORK DONE

- Reviewed related literature on mobile malware detection techniques, focusing on signature-based and AI approaches.
- Initial planning and setup of the project environment and tools required for development.

2. WORK TO BE DONE

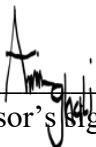
- Start designing system architecture and outlining the functional components of the IDPS.
- Begin exploring potential APIs for real-time malware detection.

3. PROBLEMS ENCOUNTERED

- Stuck in mindset on how to start and where to begin building the system.

4. SELF EVALUATION OF THE PROGRESS

- Despite the initial challenges in mindset, progress was made in setting up the groundwork for the project.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Jun 2024	Study week no.: 4
Student Name & ID: LEOW YU HONG 2005558	
Supervisor: Dr Abdulrahman Aminu Ghali	
Project Title: Detecting Malware Attack in Mobile Phone using Intrusion Detection and Prevention System (IDPS)	

1. WORK DONE

- Designed the system architecture for the IDPS and finalized the components to be used.
- Began working on the algorithm for scanning and detecting malware.

2. WORK TO BE DONE

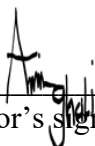
- Continue refining the file search algorithm and test its performance across different devices.
- Explore machine learning models for unknown malware detection.

3. PROBLEMS ENCOUNTERED

- Building a suitable and effective algorithm for searching files in the entire system took longer than expected.

4. SELF EVALUATION OF THE PROGRESS

- Progress has been steady. The architecture design is complete, and an initial algorithm has been created, though refinement is necessary.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Jun 2024	Study week no.: 6
Student Name & ID: LEOW YU HONG 2005558	
Supervisor: Dr Abdulrahman Aminu Ghali	
Project Title: Detecting Malware Attack in Mobile Phone using Intrusion Detection and Prevention System (IDPS)	

1. WORK DONE

- Implemented and tested the file search algorithm within the IDPS.
- Added functionality to retrieve file information after malware is detected.

2. WORK TO BE DONE

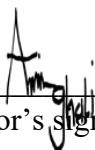
- Integrate the quarantine system to handle detected malware.
- Work on the reporting system to log detected threats and actions taken.

3. PROBLEMS ENCOUNTERED

- Issues with retrieving file information post-detection; the retrieval process caused some delays in system response.

4. SELF EVALUATION OF THE PROGRESS

- Significant progress in the detection and response mechanism. Some delays in information retrieval were encountered but are being worked on.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Jun 2024	Study week no.: 8
Student Name & ID: LEOW YU HONG 2005558	
Supervisor: Dr Abdulrahman Aminu Ghali	
Project Title: Detecting Malware Attack in Mobile Phone using Intrusion Detection and Prevention System (IDPS)	

1. WORK DONE

- Implemented the quarantine mechanism and tested it with several known malware samples.
- Started working on network monitoring using VPN-based methods.

2. WORK TO BE DONE

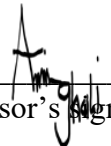
- Refine the quarantine system and continue testing.
- Finalize the network monitoring setup and troubleshoot routing issues.

3. PROBLEMS ENCOUNTERED


- The network monitoring using a VPN-based solution faced routing issues, where traffic could not redirect back correctly.

4. SELF EVALUATION OF THE PROGRESS

- Progress on quarantine functionality is good, but the network monitoring aspect still requires work to resolve routing problems.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Jun 2024	Study week no.: 10
Student Name & ID: LEOW YU HONG 2005558	
Supervisor: Dr Abdulrahman Aminu Ghali	
Project Title: Detecting Malware Attack in Mobile Phone using Intrusion Detection and Prevention System (IDPS)	

1. WORK DONE

- Continued testing and debugging the VPN-based network monitoring system.
- Began optimizing the WebView component to handle a large number of URL requests.

2. WORK TO BE DONE

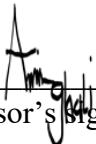
- Improve the performance of WebView to avoid slowdowns when processing large numbers of URLs.
- Complete the implementation of the URL detection and blocking feature.

3. PROBLEMS ENCOUNTERED

- The WebView became slow due to the large volume of URL requests being sent for scanning.

4. SELF EVALUATION OF THE PROGRESS

- While there is steady progress in developing core features, the performance issues with WebView need urgent attention.



Supervisor's signature



Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Jun 2024	Study week no.: 12
Student Name & ID: LEOW YU HONG 2005558	
Supervisor: Dr Abdulrahman Aminu Ghali	
Project Title: Detecting Malware Attack in Mobile Phone using Intrusion Detection and Prevention System (IDPS)	

1. WORK DONE

- Improved WebView performance by optimizing URL handling and request processing.
- Finalized the system evaluation metrics for testing.

2. WORK TO BE DONE

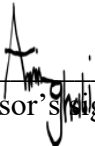
- Conduct final tests on all components, focusing on performance and accuracy.
- Prepare for the final project presentation.

3. PROBLEMS ENCOUNTERED

- Minor performance issues during full scans, but overall system stability has improved.

4. SELF EVALUATION OF THE PROGRESS

- The system is now in its final stages, with most features functioning as intended. Only minor adjustments are required to ensure optimal performance.



Supervisor's signature



Student's signature

Universiti Tunku Abdul Rahman (UTAR)



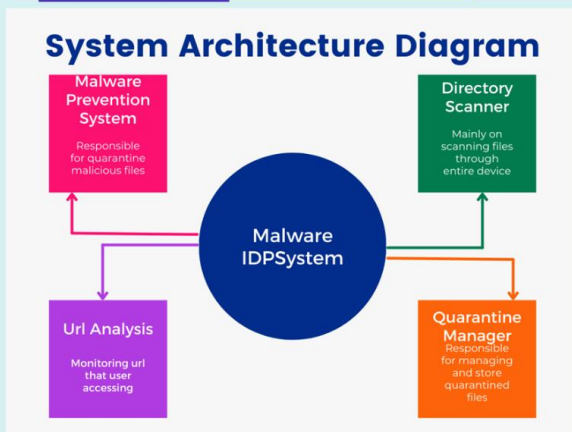
FACULTY OF INFORMATION COMMUNICATION AND TECHNOLOGY

Detecting Malware Attack in Mobile Phone using Intrusion Detection and Prevention System (IDPS)

Introduction

- ▶ This is an android Intrusion Detection and Prevention System used for detecting and preventing against with mobile malware

Methods



There are four methods in our system:

- Malware Prevention System
- Url Analysis
- Directory Scanner
- Quarantine Manager

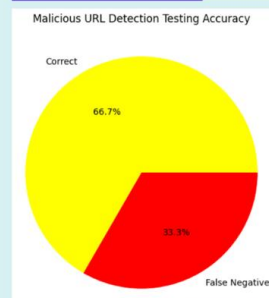
Conclusion

- ▲ IDPS is an useful system used for detecting and preventing against with all attacks that exist. IDPS aims to improve the security of the data and privacy.



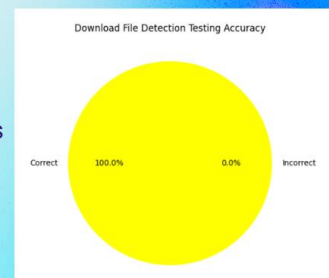
Project Developer: **Leow Yu Hong**
Project Supervisor: **Dr Abdulrahman Aminu Ghali**

Results



Malicious URL Testing Accuracy Chart

Result:
-More than 66% of detection accuracy



Download File Detection Testing Accuracy Chart

Result:
-All of the file was been detected correctly during testing.

PLAGIARISM CHECK RESULT

Detecting Malware Attack in Mobile Phone using Intrusion Detection and Prevention System (IDPS) .docx

ORIGINALITY REPORT

8%	3%	1%	6%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Universiti Tunku Abdul Rahman Student Paper	6%
2	eprints.utar.edu.my Internet Source	1%
3	pure.southwales.ac.uk Internet Source	<1%
4	Submitted to Africa Nazarene University Student Paper	<1%
5	Submitted to Deakin University Student Paper	<1%
6	Submitted to University of Wales Institute, Cardiff Student Paper	<1%
7	lab.cs.orst.edu Internet Source	<1%
8	eprints.uthm.edu.my Internet Source	<1%

www.archyde.com

9	Internet Source	<1 %
10	www.bartleby.com Internet Source	<1 %
11	www.coursehero.com Internet Source	<1 %
12	hdl.handle.net Internet Source	<1 %
13	Vasileios KOULIARIDIS, Konstantia BARMPATSALOU, Georgios KAMBOURAKIS, Shuhong CHEN. "A Survey on Mobile Malware Detection Techniques", IEICE Transactions on Information and Systems, 2020 Publication	<1 %

Exclude quotes Off
Exclude bibliography On

Exclude matches Off

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



FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	LEOW YU HONG
ID Number(s)	20ACB05558
Programme / Course	FICT-CN
Title of Final Year Project	Detecting Malware Attack in Mobile Phone using Intrusion Detection and Prevention System (IDPS)

Similarity	Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)
Overall similarity index: <u>8%</u> % Similarity by source Internet Sources: <u>3</u> % Publications: <u>1</u> % Student Papers: <u>6</u> %	
Number of individual sources listed of more than 3% similarity: <u>0</u>	
Parameters of originality required and limits approved by UTAR are as Follows: (i) Overall similarity index is 20% and below, and (ii) Matching of individual sources listed must be less than 3% each, and (iii) Matching texts in continuous block must not exceed 8 words <i>Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.</i>	

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

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

Signature of Supervisor

Name: Dr Abdulrahman Aminu Ghali

Date: 11 September 2024 _____

Signature of Co-Supervisor

Name: _____

Date: _____



UNIVERSITI TUNKU ABDUL RAHMAN

**FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY
(KAMPAR CAMPUS)**

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	20ACB05558
Student Name	LEOW YU HONG
Supervisor Name	Dr Abdulrahman Aminu Ghali

TICK (✓)	DOCUMENT ITEMS
	Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
✓	Title Page
✓	Signed Report Status Declaration Form
✓	Signed FYP Thesis Submission Form
✓	Signed form of the Declaration of Originality
✓	Acknowledgement
✓	Abstract
✓	Table of Contents
✓	List of Figures (if applicable)
✓	List of Tables (if applicable)
✓	List of Symbols (if applicable)
✓	List of Abbreviations (if applicable)
✓	Chapters / Content
✓	Bibliography (or References)
✓	All references in bibliography are cited in the thesis, especially in the chapter of literature review
✓	Appendices (if applicable)
✓	Weekly Log
✓	Poster
✓	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
✓	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

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

I, the author, have checked and confirmed all the items listed in the table are included in my report.

(Signature of Student)

Date: 11 September 2024