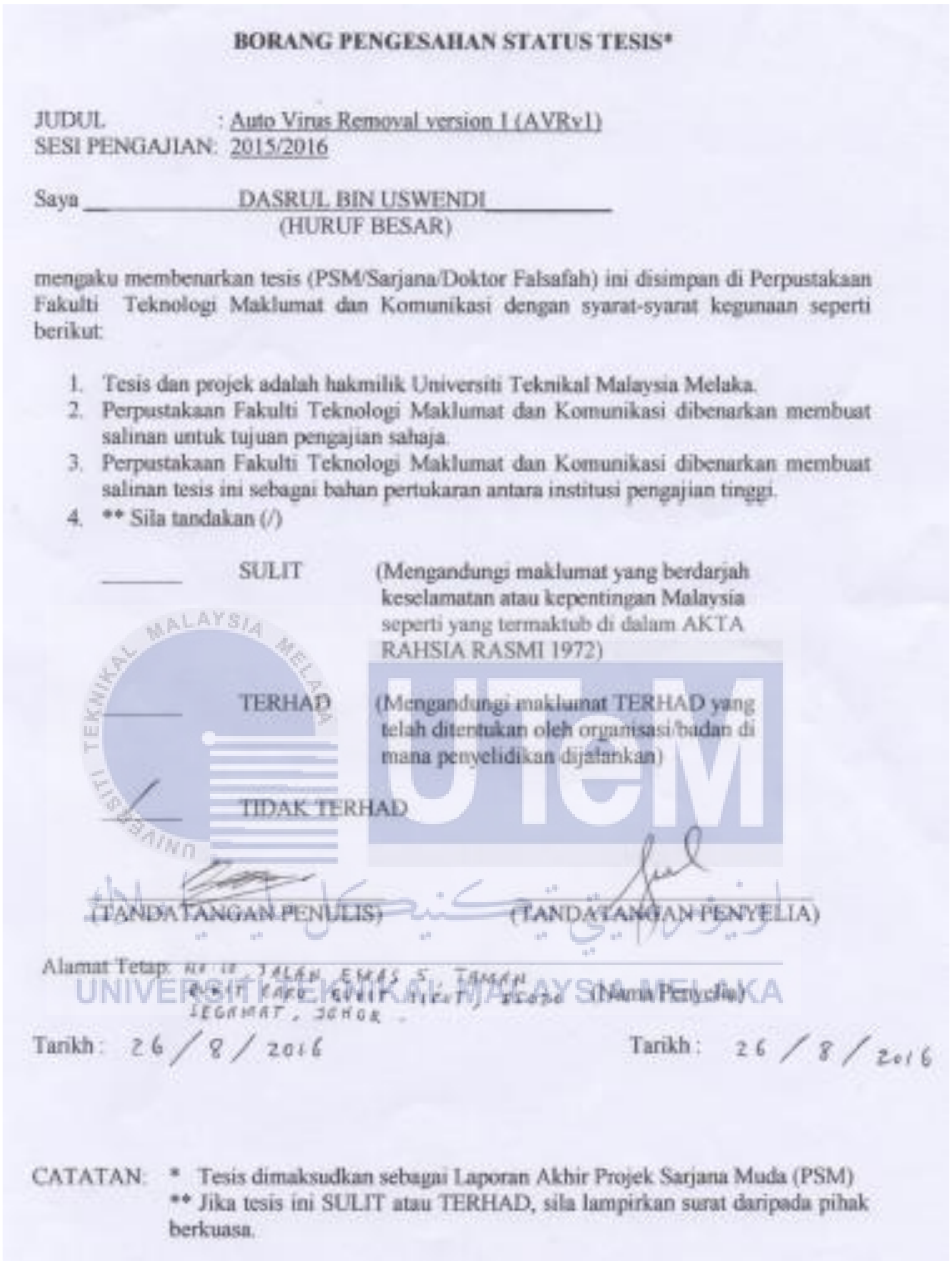
AUTO VIRUS REMOVAL VERSION 1 (AVRV1)

DASRUL BIN USWENDI

UNIVERSITI TEKNIKAL MALAYSIA MELAKA



BORANG PENGESAHAN STATUS TESIS*

JUDUL : Auto Virus Removal version 1 (AVRv1)
SESI PENGAJIAN: 2015/2016

Saya DASRUL BIN USWENDI
(HURUF BESAR)

mengaku membenarkan tesis (PSM/Sarjana/Doktor Falsafah) ini disimpan di Perpustakaan Fakulti Teknologi Maklumat dan Komunikasi dengan syarat-syarat kegunaan seperti berikut:

1. Tesis dan projek adalah hakmilik Universiti Teknikal Malaysia Melaka.
2. Perpustakaan Fakulti Teknologi Maklumat dan Komunikasi dibenarkan membuat salinan untuk tujuan pengajian sahaja.
3. Perpustakaan Fakulti Teknologi Maklumat dan Komunikasi dibenarkan membuat salinan tesis ini sebagai bahan pertukaran antara institusi pengajian tinggi.
4. ** Sila tandakan (/)

______	SULIT	(Mengandungi maklumat yang berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972)
______	TERHAD	(Mengandungi maklumat TERHAD yang telah ditentukan oleh organisasi/badan di mana penyelidikan dijalankan)
✓	TIDAK TERHAD	

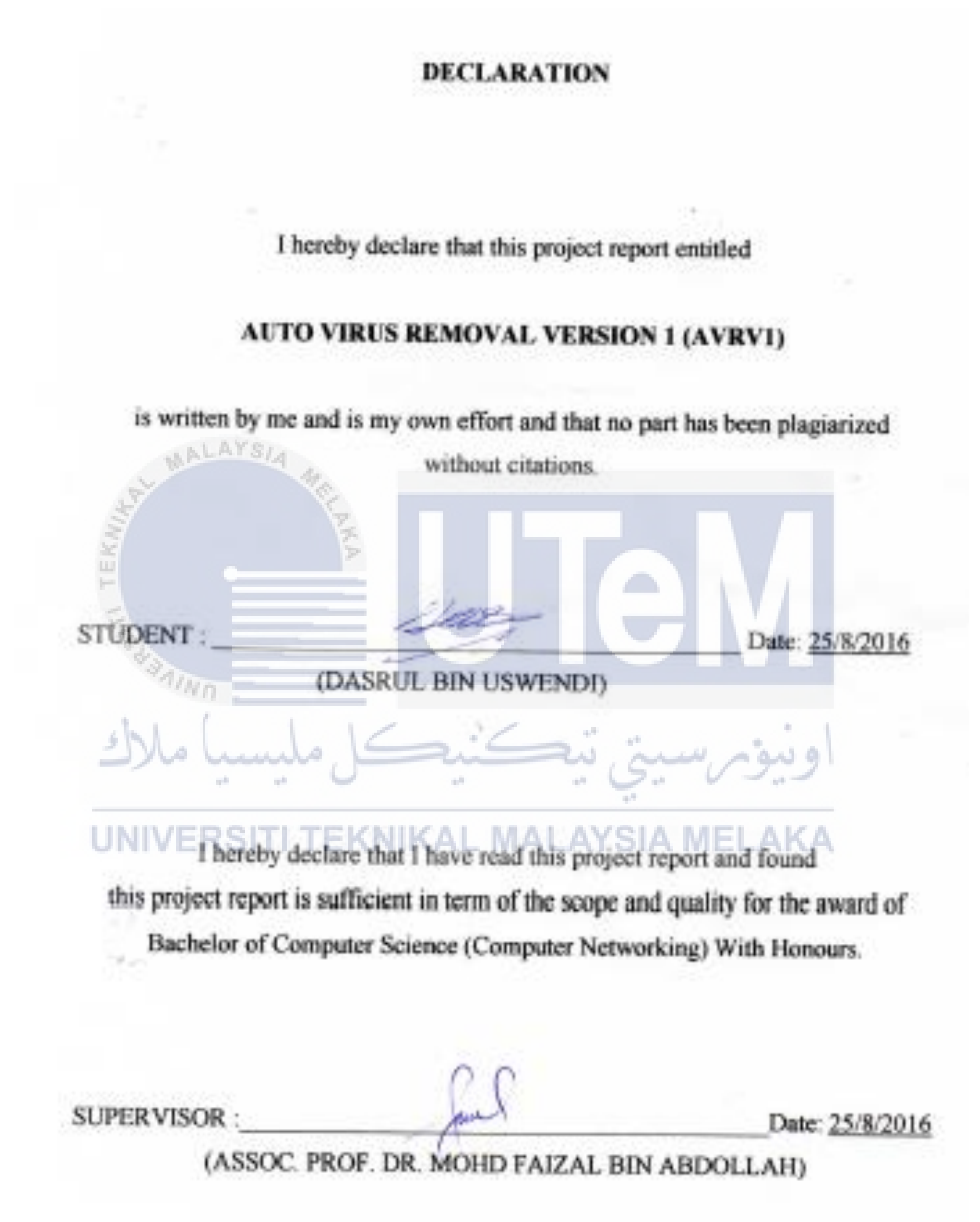
[illegible] [illegible]
(TANDATANGAN PENULIS) (TANDATANGAN PENYELIA)

Alamat Tetap: NO 18, JALAN EMAS 5, TAMAN
BUKIT KAKU, BUKIT [illegible], 85000
SEGAMAT, JOHOR.

(Nama Penyelia)

Tarikh: 26 / 8 / 2016 Tarikh: 26 / 8 / 2016

CATATAN: * Tesis dimaksudkan sebagai Laporan Akhir Projek Sarjana Muda (PSM)
** Jika tesis ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa.



DECLARATION

I hereby declare that this project report entitled

AUTO VIRUS REMOVAL VERSION 1 (AVRV1)

is written by me and is my own effort and that no part has been plagiarized without citations.

STUDENT : [illegible] Date: 25/8/2016

(DASRUL BIN USWENDI)

I hereby declare that I have read this project report and found this project report is sufficient in term of the scope and quality for the award of Bachelor of Computer Science (Computer Networking) With Honours.

SUPERVISOR : [illegible] Date: 25/8/2016

(ASSOC. PROF. DR. MOHD FAIZAL BIN ABDOLLAH)

DEDICATION

This thesis is dedicated to my father, Uswendi bin Dauni, who taught me that the best kind of knowledge to have is that which is learned for its own sake. It is also dedicated to my mother, Wirda binti Mawar, who taught me that even the largest task can be accomplished if it is done one step at a time. Both of them are my inspiration and they are everything in my life.

ACKNOWLEDGEMENTS

First of all, I would like to express my grateful to Allah S.W.T, whom without His guidance I would not have been able to be as I am as today. Second, I would like to express my thankful to Prophet Muhammad S.A.W, who had taught the truth of Islam to human in the world. Third, I would like to give a big thanks to my supervisor, Assoc. Prof. Dr. Mohd Faizal Bin Abdollah for guiding me throughout this project and he never been regretted to choose me as his student. Lastly, I would like to give thanks to all my friends especially to my partner, Mohd Aiman Afnan bin Mohd Yusoff who had helped me to give some ideas for this project. Thanks for everything.

ABSTRACT

Nowadays, there have many various type of malware which making damage to our devices. They can infect to the file or registry in the storage whether user are known or not. With the increasing a number of new malware in year statistic, the total number of new product of anti-viruses also increase dramatically in order to against malware. The difficulty to detect the malware on internal devices or external devices makes people become more interested to make investigation on these malware. There are need to study the malware behaviour first and suggest the potential techniques to use to detect and remove the malware. There have many techniques that used for most existing anti-virus and one of the most popular of virus detection technique is signature-based detection. The Auto Virus Removal version 1 (AVRv1) is the new anti-virus as a final year project that would be developed by using several techniques for detecting the virus which refer to the system of existing anti-virus.

ABSTRAK

Pada zaman kini, terdapat banyak jenis malware dimana ia melakukan kerosakan pada alat penyimpanan data. Ia bole melakukan jangkitan pada fail di dalam alat penyimpanan data sama ada disedari oleh pengguna atau pun tidak. Dengan peningkatan jumlah virus baru mengikut statistik tahunan, jumlah penghasilan anti-virus yang baru juga meningkat secara mendadak untuk melindungi serangan malware. Kesukaran untuk mengesan malware di dalam computer atau pun pada alat penyimpanan data yang lain membuatkan manusia ingin lebih mengetahui tentang perkara ini. Mereka perlu mengetahui tentang pergerakan malware dahulu dan perlu mencadangkan teknik-teknik untuk mengesan virus secara berkesan. Terdapat banyak teknik yang digunakan oleh kebanyakkan anti-virus yang sedia ada dalam mengesan dan membuang virus ini. Salah satu teknik yang popular adalah teknik penyamaan. The Auto Virus Removal version 1 (AVRv1) adalah sebuah anti-virus yang baru dan sebagai satu projek sarjana muda yang akan dibangunkan sebagai sebuah anti-virus yang baru di mana akan menggunakan beberapa teknik mengikut sistem dalam anti-virus yang tersedia ada.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

1.1. Introduction

Any computer can be infected by malware. Malware is a catch-all term for malicious programs, such as viruses, worms, Trojans, and spyware, which are designed to infect and take control of computer. Anti-virus software is designed to protect computer against malware. There have many types of Anti-virus software such as Avira, Avast, Kaspersky, and so on.

In this project, The Auto Virus Removal version 1 (AVRv1) is a virus removal tool that will be developed to detect and remove the virus effectively. The Online Sandbox is used as a requirement to get the output from it. The output from the Online Sandbox is samples of infected file by malware. With these samples, The Auto Virus Removal version 1 (AVRv1) will try to detect the virus from these samples and remove the detected virus as well.

1.2. Problem Statement (PS)

When scanning the virus infected storage with anti-virus software, it will inform to user that no threat detected which was really confusing because not all anti-virus can detect this virus. In addition, somebody who did not know on how to remove this virus, they just formatting the infected storage as last choice. So, the virus with all the important or non-important data will be lost together. Other than that, the details about the techniques on how to use the virus remover tool at default web page are not more user-friendly and uninformative to user.

Table 1.1: Problem Statement

PS	Problem Statement
PS_1	The existing anti-virus software are not really detect the virus on the storage
PS_2	The instruction on how to use the anti-virus software are not informative

The main problem that support to develop the Auto Virus Removal version 1 (AVRv1) is the existing anti-virus software are not really detect the virus on the storage as shown on Table 1.1.

1.3. Project Question (PQ)

The first question referred by the problem statement is on how to detect the virus from the output of the Online Sandbox by using the Auto Virus Removal version 1 (AVRv1). The second one is on what type of virus that can be detected. Another question is on how to remove the virus that has been detected by using the Auto Virus Removal version 1 (AVRv1).

Table 1.2: Project Question

PS	PQ	Project Question
PS$_1$	**PQ$_1$**	How can the Auto Virus Removal version 1 (AVRv1) helps in this project?
PS$_2$	**PQ$_2$**	How can the webpage of the Auto Virus Removal version 1 (AVRv1) provides informative instructions to user?

Based on Table 1.2, the main questions that support to develop the Auto Virus Removal version 1 (AVRv1) are on how can the Auto Virus Removal version 1 (AVRv1) helps in this project and how can the webpage of the Auto Virus Removal version 1 (AVRv1) provides informative instructions to user.

1.4. Project Objectives (PO)

Project objective defines the things that want to achieve. The objectives must be considered based on the problem statement and the project question of this project.

Table 1.3: Summary of Project Objectives

PS	PQ	PO	Project Objective
PS_1	**PQ_1**	**PO_1**	To detect the virus that has been found from the output of the Online Sandbox by using the Auto Virus Removal version 1 (AVRv1)
		PO_2	To remove the virus that has been detected by using the Auto Virus Removal version 1 (AVRv1)
PS_2	**PQ_2**	**PO_3**	To provide an informative instructions of the Auto Virus Removal version 1 (AVRv1) on the webpage

Based on table 1.3, there have three objectives that need to achieve in order to make a successful project.

1.5. Project Scope

Scope of project going to be handled as follows:

1) Focusing on how to detect the virus from the output of the Online Sandbox
2) Focusing on how to remove the detected virus by using Auto Virus Removal version 1 (AVRv1)
3) Analyzing and testing the function of the Auto Virus Removal version 1 (AVRv1) by using the virtual machine, VMware.

1.6. Project Contribution (PC)

Project contribution defines the expected output from this project. This part can be referred to the objectives of this project.

Table 1.4: Summary of Project Contribution

PS	PQ	PO	PC	Project Contribution
PS_1	**PQ_1**	**PO_1**	**PC_1**	Proposed an effectiveness anti-virus for user
		PO_2		
PS_2	**PQ_2**	**PO_3**	**PC_2**	Proposed a user-friendly and informative in web page

In this project contribution, there have two contributions that have been considered of this project as shown on Table 1.4.

1.7. Thesis Organization

Thesis organization defines the summary of each chapter presented in this report of project. There have seven chapters that need to implement and present. All the descriptions of each chapter are shown in below:

Chapter 1: Introduction

This chapter discuss about the beginnings of system development. It will focuses on introduction, background of the project, problem statement of the current project, questions of the project, objective of the project, scope of the project, project significant and report organization

Chapter 2: Literature Review

This chapter discuss about more explanation and details about the project referred on the chapter 1. This chapter need supported with reading materials and conference paper.

Chapter 3: Project Methodology

This chapter discuss about what method are used in this project. This chapter also discuss about the whole organization of this project.

Chapter 4: Analysis and Design

This chapter discuss about the analysis of this project. This chapter also discuss about the design of the tool and the web page of the Auto Virus Removal version 1 (AVRv1). The requirements of hardware and software has been introduced with environment setup,

Chapter 5: Implementation

This chapter discuss about environment setup, sample of virus will collected. The Online Sandbox is needed to get the output from it. The techniques to detect the virus also will be used.

Chapter 6: Testing

This chapter discuss about the testing on detect and remove the virus that got from the output of the Online Sandbox. VMware is needed as beginnings for analyzing and testing the function of the Auto Virus Removal version 1 (AVRv1).

Chapter 7: Project Conclusion

This chapter discuss about all project summarization, project contribution and project limitation will be explained. All the steps that have been made and that have been developed for this project will be listed briefly. In this last chapter also explain on additional work can be done in future.

1.8. Conclusion

In conclusion, it was clearly on what to do as beginning for developing the project. The objectives of this project are the key point to achieve. The next chapter is about literature review. It will cover about model approach, and related work about malware behaviour.

CHAPTER II

LITERATURE REVIEW

2.1 Introduction

In this chapter, it will discuss about the literature review on finding the technique to detect the infected file which implement all materials and resources such several of virus detection techniques and some the existing anti-virus. This chapter will provide a clear understanding based on the objectives that have been considered in chapter 1.

2.2 Related Work

For this project, there are three related work that has been done by a few researcher. There are the Sandbox, Malware, and Anti-Virus.

2.2.1 Malware

Malware is stand of malicious software; it is malicious code, refer to various types of software that can cause problems, damage, and disrupt computer. There have many types of malware such as viruses, worms, spyware, Trojans, hijackers, and adware and many more. (Landage & Wankhade, 2013)

1. Type of Malware

There have many types of malware which are:

a) **Virus:** It is a program or programming code that able to copy itself or initiating its copying to another program, computer boot sector or document. Viruses can be transmitted as attachments to an e-mail note or in a downloaded file, and many more.

b) **Trojan:** It is a program in which pretend itself as an original program to trick a user into running it. It accesses the infected host remotely, steal confidential data, monitor users activity, installing botnet and many more.

c) **Worm:** It is a self-replicating virus that does not alter files but resides in active memory and copies itself. It can steal the data, delete files, crashing a hard drive or create botnets.

d) **Spyware:** It is any technology that spying and monitoring users activity to gain confidential data without their knowledge. Furthermore, it can collect users keystroke, harvesting user's data and many more.

e) **Adware:** It is any software application in which advertising banners are displayed while the program is running. In addition, most are created to serve the purpose of gaining revenue.

Based on all types of malware given, the virus is the one type of malware that would be focused on this project as the input and output data. It is because there have many types virus that infect to the devices and is one of the most common detected of new malware today. The statistic of malware and virus will be clarified on the next sub-chapter.

2. The statistic of Malware

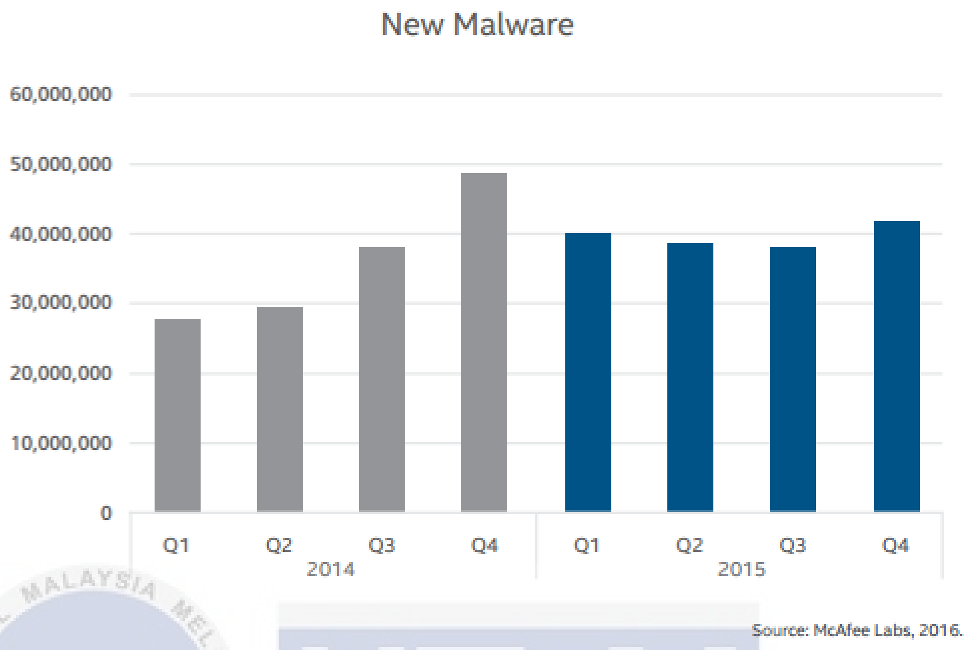


Figure 2.1: The increase in the known number of new malware from 2014 to 2015

Based on Figure 2.1, there have 4 quarters for year 2014 and year 2015. The malware sample counting method has been adjusted to all quarters to increase its accuracy. In year 2015, the number of new malware sample in Q4 was increased with 42 million new malicious. Compare with the number of new malware in Q3 at the same year, the number of new malware in Q4 increase 10%. (McAfee Labs, 2015)

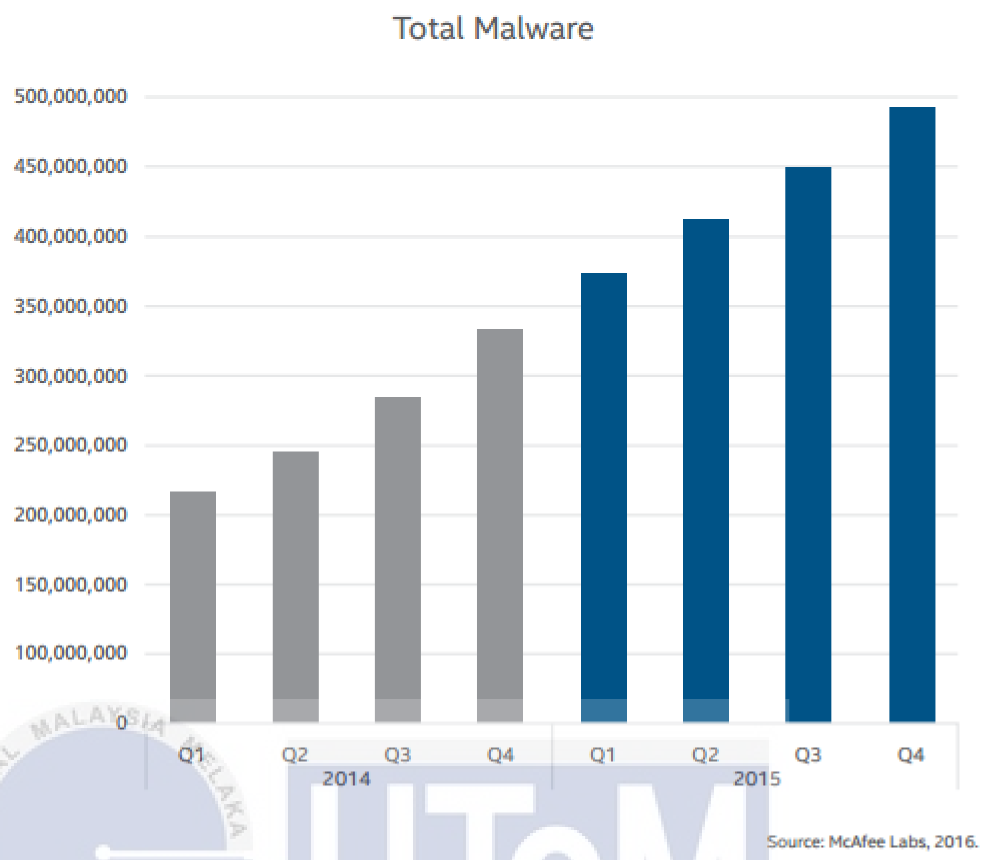


Figure 2.2: The increase in the known number of total malware from 2014 to 2015

The Figure 2.2 shows the total number of malware include the number of new malware for year 2014 until 2015. The total number of malware was increased constantly. The total of identified new malware was increase dramatically for every year. (McAfee Labs, 2015)

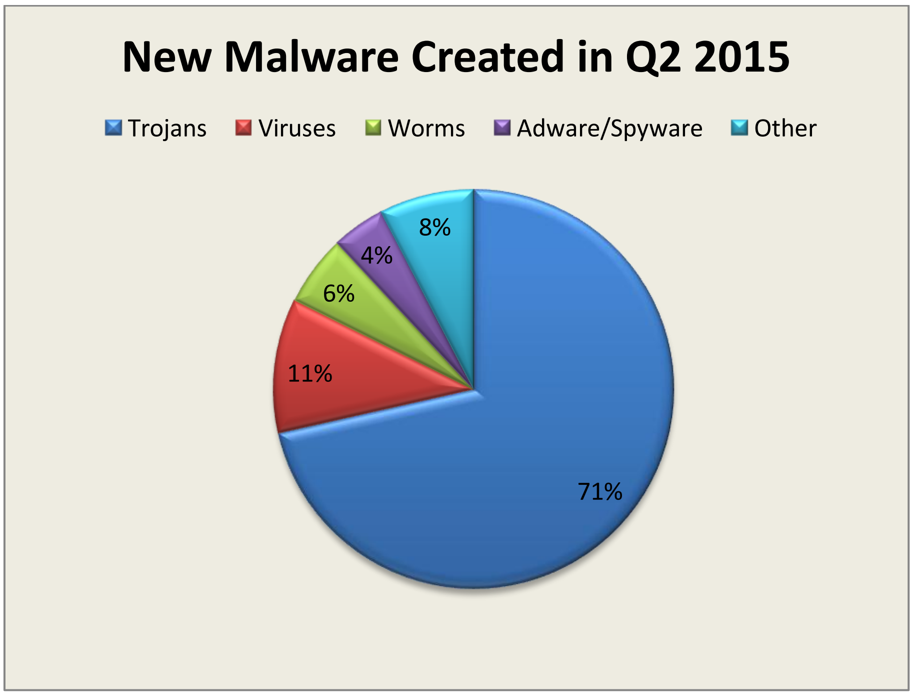


Figure 2.3: The percentages of new malware in 2015

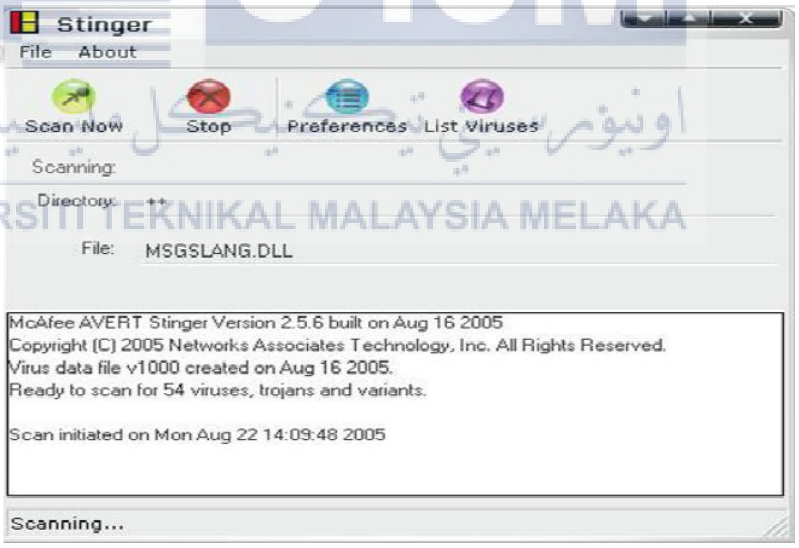
The Figure 2.3 shows the percentages of new malware created in Q2 for year 2015. Based on this figure, Trojans is the most common malware detected which are 71.16% of all samples witnessed during Q2 and the second higher followed by viruses which are 10.83% of all samples. Another new detected malware is worms which just get 6% of all samples. Adware or spyware get the lowest of the new detected malware which just 4% of all samples. Lastly, the other new detected malware only get 8% of all samples.(Santillan, 2015)

3. Infection

The system will get infection by Windows Vulnerabilities, network connections or shared files, Pop-Ups, E-mail, and free application. This causes slow system performance, internet homepage changed, computer crashes, no indication, new toolbars appeared, and many more.

4. Extraction

For extraction, user can use any virus specific removal tools. There have two examples virus specific removal tools which are Stinger and Symantec.



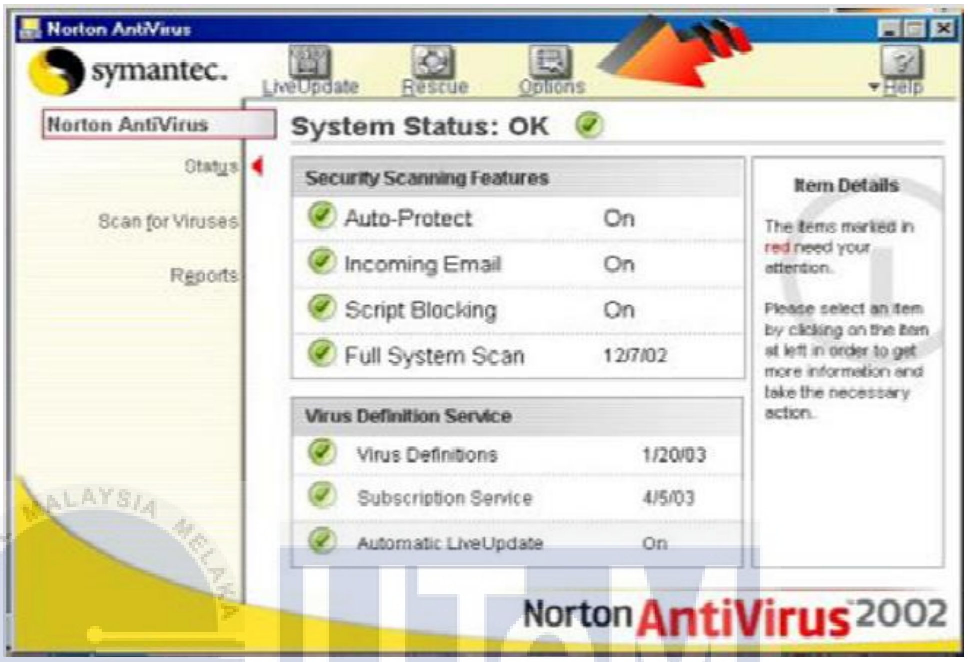
MSGSLANG.DLL

```
McAfee AVERT Stinger Version 2.5.6 built on Aug 16 2005
Copyright (C) 2005 Networks Associates Technology, Inc. All Rights Reserved.
Virus data file v1000 created on Aug 16 2005.
Ready to scan for 54 viruses, trojans and variants.

Scan initiated on Mon Aug 22 14:09:48 2005
```

Scanning...

Figure 2.4: The extraction of the Stinger in MCafee

The Figure 2.4 above shows the example extraction of the Stinger in MCafee.



System Status: OK ✓

Security Scanning Features	
✓ Auto-Protect	On
✓ Incoming Email	On
✓ Script Blocking	On
✓ Full System Scan	12/7/02

Virus Definition Service	
✓ Virus Definitions	1/20/03
✓ Subscription Service	4/5/03
✓ Automatic LiveUpdate	On

Item Details

The items marked in red need your attention.

Please select an item by clicking on the item at left in order to get more information and take the necessary action.

Figure 2.5: Symantec in Norton AntiVirus

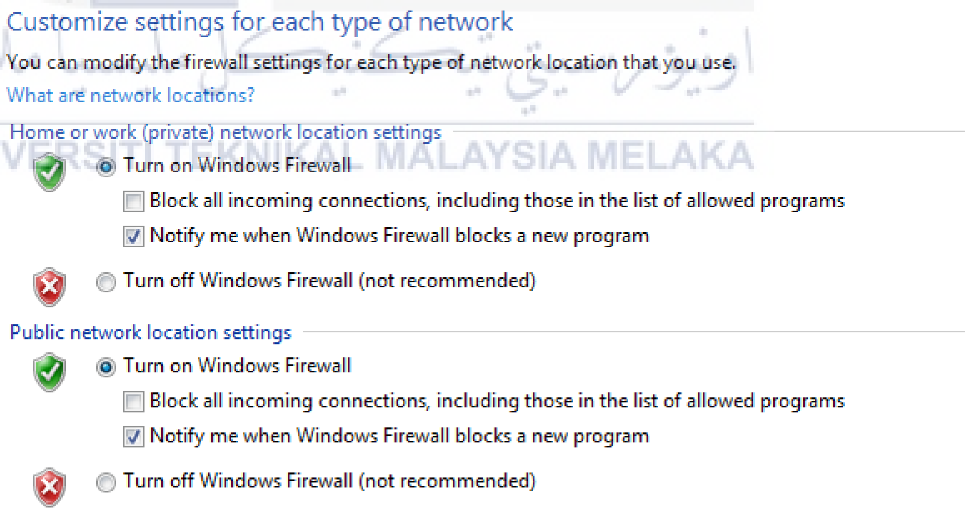
The Figure 2.5 above shows the example extraction of the Symantec in Norton AntiVirus.

5. Protection

For protection, the tools that can be used to protect from Malware are like Aluria Anti-Spyware, CounterSpy, SpySweeper, and Spyware Doctor. Other than that, user can use the tools that can be continuously protected like AVG Anti-Virus, Mcafee Anti-Virus, and Norton or Symantic Anti-Virus.

6. Prevention

For prevention from infection by malware, user needs to update the protection tools and review software being installed. Other than that, make sure the careful of web browsing and e-mail. Lastly, user needs to monitor user's child computer usage.


Customize settings for each type of network

You can modify the firewall settings for each type of network location that you use.

What are network locations?

Home or work (private) network location settings

[✓ Turn on Windows Firewall]

[ ] Block all incoming connections, including those in the list of allowed programs

[✓] Notify me when Windows Firewall blocks a new program

Turn off Windows Firewall (not recommended)

Public network location settings

[✓ Turn on Windows Firewall]

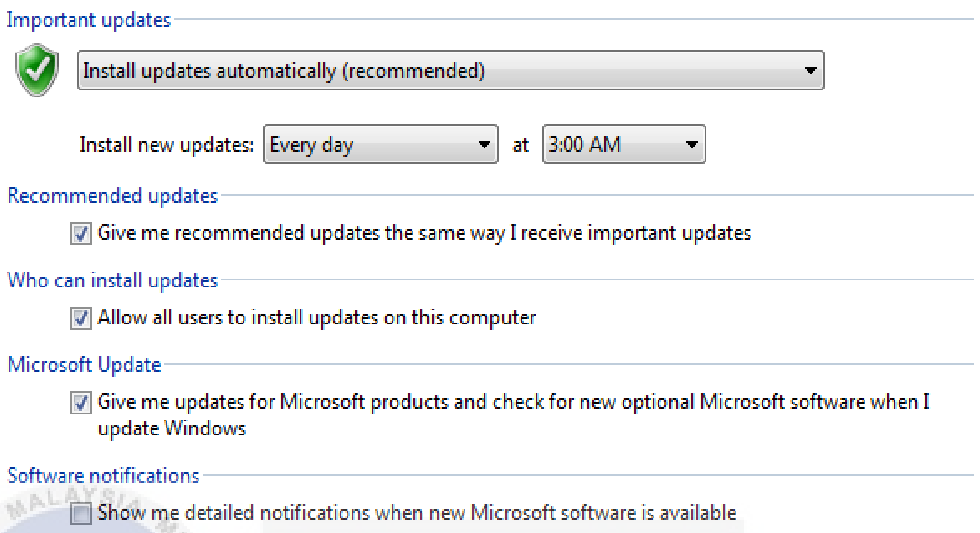
[ ] Block all incoming connections, including those in the list of allowed programs

[✓] Notify me when Windows Firewall blocks a new program

Turn off Windows Firewall (not recommended)

Figure 2.6: Firewall is on

The Figure 2.6 above shows the customize setting for each type of network.


Important updates

Install updates automatically (recommended)

Install new updates: Every day at 3:00 AM

Recommended updates

[✓] Give me recommended updates the same way I receive important updates

Who can install updates

[✓] Allow all users to install updates on this computer

Microsoft Update

[✓] Give me updates for Microsoft products and check for new optional Microsoft software when I update Windows

Software notifications

[ ] Show me detailed notifications when new Microsoft software is available

Figure 2.7: Windows Update is on

The Figure 2.7 above shows the setting of important updates of Windows.

2.2.2 Virus

The virus is the one type of malware which has been explained on the previous sub-chapter. As mentioned on previous sub-chapter, this project will be focus on the virus.

1. Types of virus

There have many types of virus which are:

a) **Boot Viruses:** This virus can attack the boot record, the master boot record, the File Allocation Table (FAT), and the partition table of a computer hard drive. The examples of boot virus are Joshi and Michelangelo.

b) **Stealth Viruses:** This virus can disguise their actions and can be a passive virus which can increase the file size, or can be an active virus which can attack the antivirus software rendering them useless. The example of stealth viruses is Tequila.

c) **Encrypted Virus:** This virus has inbuilt encryption software code which masks the viral code making it difficult to identify and detect the virus. The example of encrypted virus is Cascade.

d) **Polymorphic Virus:** This virus has an inbuilt mechanism that can alter the virus signature. During the process of infection, it creates slightly modified and fully functional copies of itself. The example of polymorphic virus is SMEG.

e) **Macro Virus:** This virus attacks applications that run macros such as Microsoft word. The virus is activated when a document or a template file in which it is embedded, is opened by an application. The example of macro virus is Melissa.

2.2.3 The Sandbox

The sandbox is one of requirement that would be used in this project. This is the software requirement which used on the virtual machine.

1. Overview

The Sandbox is an isolated environment initially used by software developers to test new programming code, (H. Mourad 2015). An Online Sandbox creates isolated environment to general users. So these tools allow various computer programs to run in an isolated environment and prevent them from making any permanent change to other programs and data in computer.

2. Function of the Sandbox

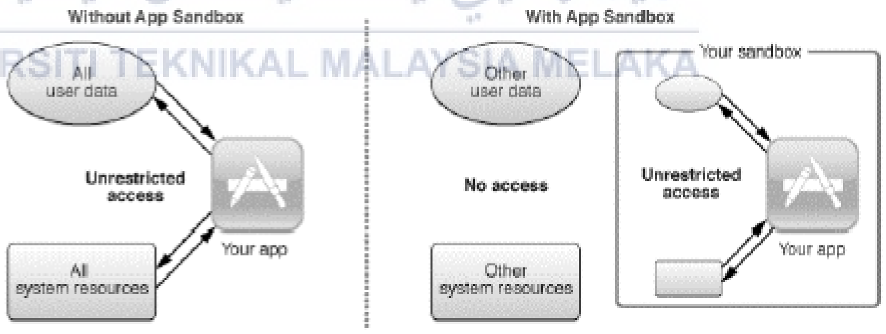


Figure 2.8: Function of the Sandbox

The Figure 2.8 provided above explains the workings of the Sandbox. Without the use of the Sandbox, the infected application would gain all user data and system resources, which h would later be used by someone with a malevolent intent to their benefit. With the deployment of the Sandbox, the infected application will run inside an isolated environment.

3. Types of Sandbox

There have two types of sandbox that can be classified which are online sandbox and standalone sandbox.

a) **Online sandbox**: It is used for enabling the public to submit any malicious file through their webpage, analyze the file, and give a result of the file behaviour for the public. An analyst cannot to customize an online sandbox because it is governed by another organization. An example of online sandbox is Malwr.

b) **Standalone sandbox**: It is used for analyzing a malicious behaviour at a local environment. A dedicated machine are needed which are able to run a virtual machine within it for implementation. The examples of standalone sandbox solutions are Cuckoo, and Sandboxie

4. Techniques of Sandbox

For malware detection, sandbox will conduct dynamic and static malware analysis technique to determine whether a program is considered malicious or not. These two techniques have their advantages and disadvantages.

a) **Static Analysis:** It is a technique for analyzing malware which involves various kinds of tools and techniques to determine whether a file is malicious or not. This technique including decompilation, decryption, pattern matching and static system call analysis. A common approach is filtering binaries by malicious patterns, called signatures. These are the properties that will be examined in a static analysis; file type, file name, size of file, MD5 checksum or hashes and recognition of antivirus detection tools. (Gandotra, Bansal, & Sofat, 2014)

b) **Dynamic Analysis:** It is a technique for analyzing malware which running an application in a controlled environment and monitoring its behaviour. A common approach to dynamic software analysis is sandboxing. With dynamic analysis the domain names that malware is interacting with, IP addresses of the malware Command and Control server, file path locations, registry keys and many more can be detected. (Gandotra et al., 2014)

5. Pros and Cons of Sandbox

By looking to the Figure 2.9 below, the Windows is running with three applications, files and settings, and drivers.

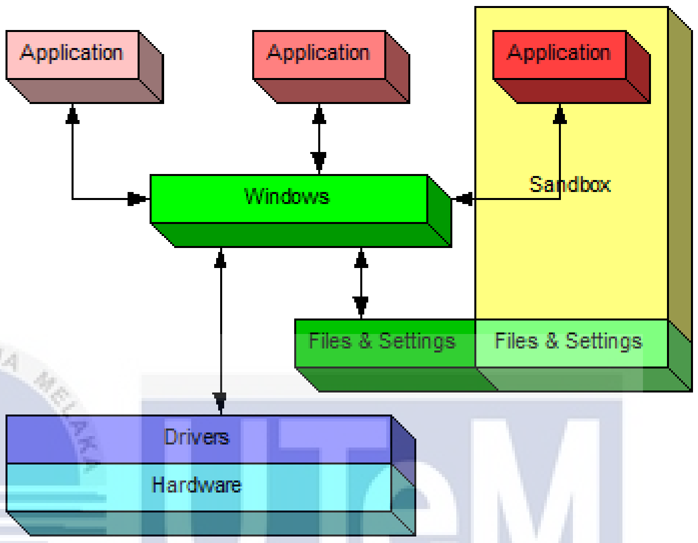


Figure 2.9: The Sandbox under Windows

One of the files and setting is placed around the whole of sandbox. Other than that, one of three applications also placed around the whole of sandbox.

a) **The pros of sandbox**: when the application exits, any malware that may have been downloaded and installed by the sandboxed application is discarded

b) **The cons of sandbox**: When an application is starting to run, it continues to have access to everything that it would were it not sandboxed. Furthermore, it is not visible outside of the Sandbox which means other Windows applications cannot see it. Other than that, it is not saved when the sandboxed application exits.

2.2.4 Anti-Virus

1. Overview

Anti-virus is a security program which to protect the storage or device from getting infected by malware. If computer has become infected by malware, a cyber attacker can use computer to attack another zombie army. With grow up of new malware, there are so many new versions of anti-virus which can detect and protect against all of them. (Williams, 2014)

2. History of Anti-Virus

- **1984**: Fred Cohen published one of the first academic papers on computer viruses.
- **1987**: the first publicly documented removal of a computer virus in the wild was performed by Bernd Fix. At the same year, two antivirus applications for the Atari ST platform were developed by G Data and UVK 2000.
- **1987**: Fred Cohen published a demonstration that there is no algorithm that can perfectly detect all possible viruses. At the same year, the first two heuristic antivirus utilities were released which are Flushot Plus by Ross Greenberg, and Anti4us by Erwin Lanting.
- **1988**: Fred Cohen began to develop strategies for antivirus software that were picked up and continued by later antivirus software developers. At the same year, a mailing list named VIRUS-L was started on the BITNET/EARN network where new viruses and the possibilities of detecting and eliminating viruses were discussed.
- **1990 – 2000**: many antivirus industries were developed to make some research and create antivirus software. For example like Norton Anti-Virus by Symantec, Anti-Virus Guard (AVG) by Jan Gritzbach and Tomas Hofer, and Anti-Virus Guard (AVG) by Jan Gritzbach and Tomas Hofer.
- **2005**: AV-TEST reported that there were 333,425 unique malware samples (based on MD5) in their database. Then, in 2007, it reported a number of 5,490,960 new unique malware samples (based on MD5) only for that year.

- **2012 – 2013**: Antivirus firms reported a new malware samples range from 300.000 to over 500.000 per day.

3. How Anti-Virus works

The virus can be detected by using any anti-virus software. The virus detection method can be classified onto two types which are signature-based detection and behavioural -based detection. (Landage & Wankhade, 2013)

a) **Signature-based detection:** It is a technique which detects malware by comparing pattern against the signature of database. The disassembled code of malware binary is examined to create these signatures. Then, that disassembled code is analyzed and features are extracted. These features are used in constructing the signature of particular malware family.

b) **Behavioural-based detection:** It is a technique which analyzes the behaviour of known or unknown malwares. This technique can be divided in two phases; training phase and detection phase. Training phase is a phase which observing the behaviour of system without any attack. Detection phase is a phase which comparing a profile against the current behaviour and the differences are marked as suitable attack.

2.3 Critical review of current problem and justification

For the critical review of current problem, several virus detection techniques are collected. Some existing antiviruses also gathered. All of these will be compared to make some ideas for develop new anti-virus. Finally, these collected data will be implemented to create a new antivirus for this project.

2.3.1 The Comparison of virus detection method

Table 2.1: The comparison of virus detection method

Methods/ Parameter	**Signature-Based Detection**	**Anomaly-Based Detection**	**Code Emulation**
Strength	Efficient	New malware	Encrypted viruses
Limitation	New malware	Unproven	Complex
Cost	Low	Costly to implement	Costly to implement
Accuracy	More if database is updated	Less	More
Advantages	The result is more accurate and this method is simple to implement	New threat can be detected without need to update the database	Can be applied to metamorphic viruses that use single or multiple encryptions.
Disadvantages	This method just detect simple virus only and Signature database must be updated regularly	If malicious activity looks like normal to the system, it will never send an alarm.	Become too slow if the decryption loop is very long.

The Table 2.1 shows the comparison between three virus detection methods. Three of these also have their advantages and disadvantages. Based this table, Signature-based detection have a simple method which the result of this method is more accurate compare to anomaly-based detection and code emulation. Other than that, it has a lowest cost on implementation compare to anomaly-based detection and code emulation. So that's why most of anti-viruses are using signature-based detection.

2.3.2 Comparison the three Anti-Virus on Windows

Table 2.2: The comparison of three anti-viruses on Windows

Software	On-demand scan	On-access scan	CloudAV	Heuristics	AntiSpam
Avast Free Anti-virus	✓	✓	✓	✓	x
Avira Free Anti-virus	✓	✓	x	✓	✓
AVG Free Anti-virus	✓	✓	x	✓	x

The Table 2.2 shows the comparison between three anti-viruses on Windows which are Avast Free Anti-virus, Avira Free Anti-virus, and AVG Free Anti-virus. On-demand scan defines the manually scanning to the devices by user while on-access scan defines the automatically scanning every file on the devices. CloudAV is similar like on-access scan which is automatically scanning but it scans on the cloud. Heuristic defines a faster technique or quick solution on solving the problems. Lastly, AntiSpam defines the security which prevents from scam and spam on email.

Based on these three anti-virus, all of these are using on demand-scan, on- access scan, and heuristics while CloudAV just used by Avast free anti-virus and AntiSpam just used by Avira free anti-virus.

2.3.3 The Pros and Cons of three Anti-Virus

Table 2.3: The pros and cons of three anti-viruses on Windows

Software	The Pros	The Cons
Avast Free Anti-virus	It has good capability on scanning and light on the system	It has a lot of pop-up’s and needs improvement on cloud reputation
Avira Free Anti-virus	It has high quality signatures and very light on the system	It does not need firewall and also weak protection on heuristic.
AVG Free Anti-virus	It has a faster full scan and great scores in PCMag's hands-on tests and also in independent lab tests.	The PC tune-up component shuts down after one-day free trial.

The Table 2.3 shows the pros and cons of three anti-viruses on Windows which are Avast Free Anti-virus, Avira Free Anti-virus, and AVG Free Anti-virus.

2.4 Proposal Solution

2.4.1 Flowchart of Proposed Solution

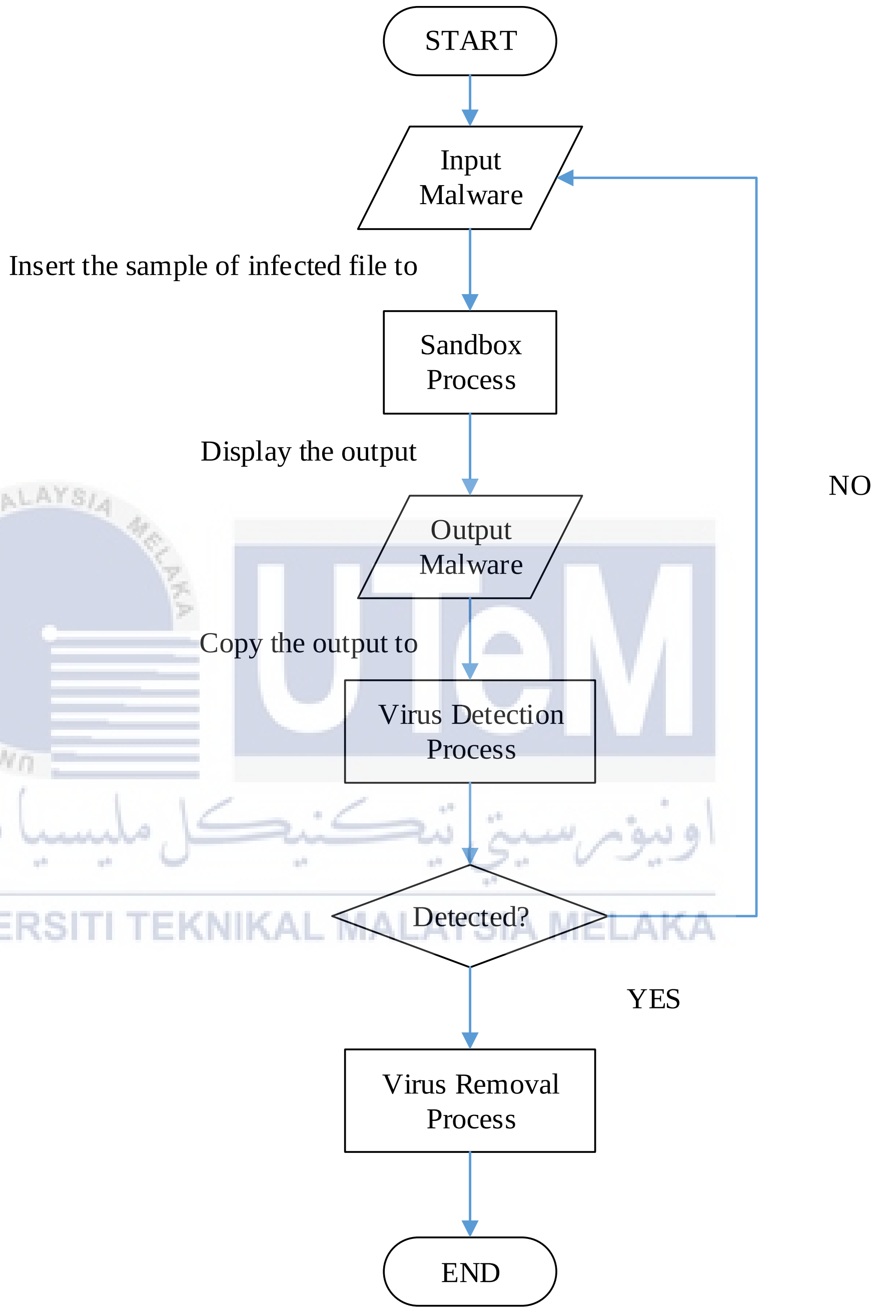


Figure 2.10: The flowchart of Proposed Solution

The Figure 2.10 shows the flowchart of proposed solution for this project. The details of each process will be clarified on the next sub-chapter.

2.4.2 Techniques

In this project, the Auto Virus Removal version 1 (AVRv1) is more concentrate on the virus only. There have many several techniques that will be used.

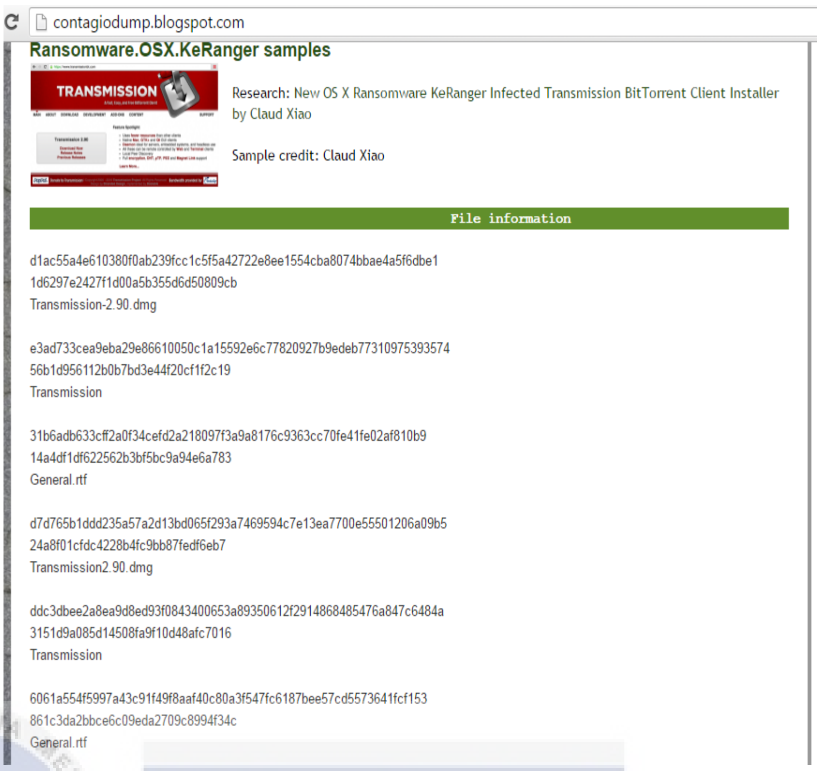
1. Heuristic Detection

The main essence of each method is to analyze the suspicious file's characteristics and behaviour to determine if it is indeed malware. One of the heuristic techniques that will be used is File Emulation. This technique is known as sandbox testing or dynamic scanning.

a) File Emulation

The first step is by getting any sample of file that has been infected by malware. This sample can be found on the website.

The Figure 2.11 below shows the one example of samples of files that have been infected by malware on the website: http://contagiodump.blogspot.com/


contagiodump.blogspot.com

Ransomware.OSX.KeRanger samples

Research: New OS X Ransomware KeRanger Infected Transmission BitTorrent Client Installer by Claud Xiao

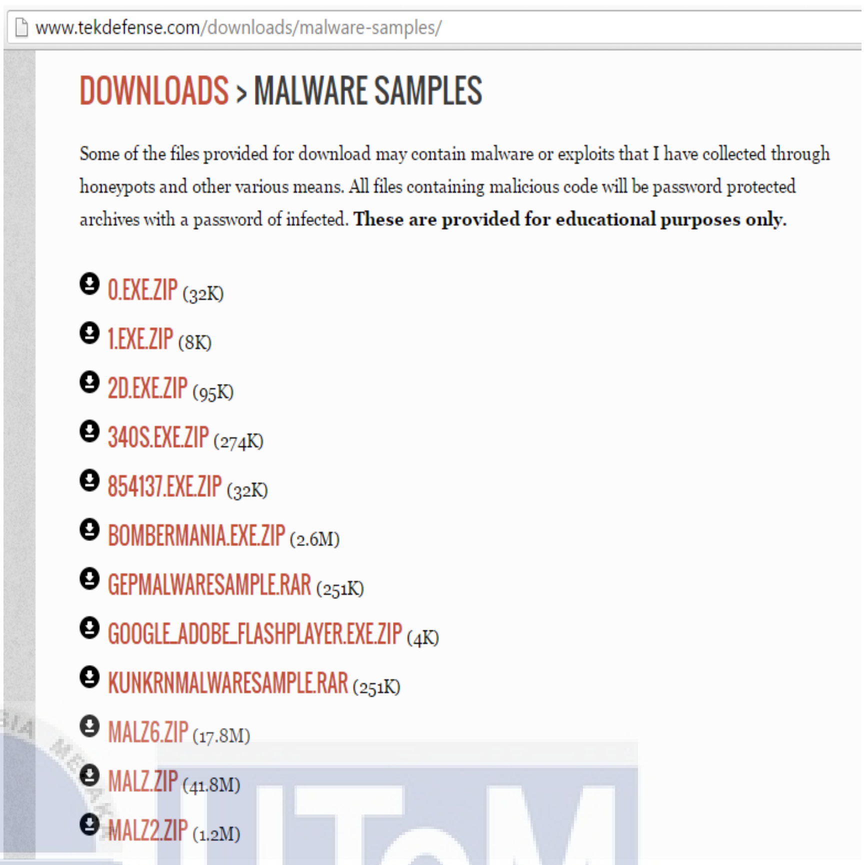
Sample credit: Claud Xiao

File information

```
d1ac55a4e610380f0ab239fcc1c5f5a42722e8ee1554cba8074bbae4a5f6dbe1
1d6297e2427f1d00a5b355d6d50809cb
Transmission-2.90.dmg

e3ad733cea9eba29e86610050c1a15592e6c77820927b9edeb77310975393574
56b1d956112b0b7bd3e44f20cf1f2c19
Transmission

31b6adb633cff2a0f34cefd2a218097f3a9a8176c9363cc70fe41fe02af810b9
14a4df1df622562b3bf5bc9a94e6a783
General.rtf

d7d765b1ddd235a57a2d13bd065f293a7469594c7e13ea7700e55501206a09b5
24a8f01cfdc4228b4fc9bb87fedf6eb7
Transmission2.90.dmg

ddc3dbee2a8ea9d8ed93f0843400653a89350612f2914868485476a847c6484a
3151d9a085d14508fa9f10d48afc7016
Transmission

6061a554f5997a43c91f49f8aaf40c80a3f547fc6187bee57cd5573641fcf153
861c3da2bbce6c09eda2709c8994f34c
General.rtf
```

Figure 2.11: The samples of files infected by malware

The Figure 2.12 below shows the other example of samples of files that have been infected by malware on the website: http://www.tekdefense.com/downloads/malware-samples/


www.tekdefense.com/downloads/malware-samples/

DOWNLOADS > MALWARE SAMPLES

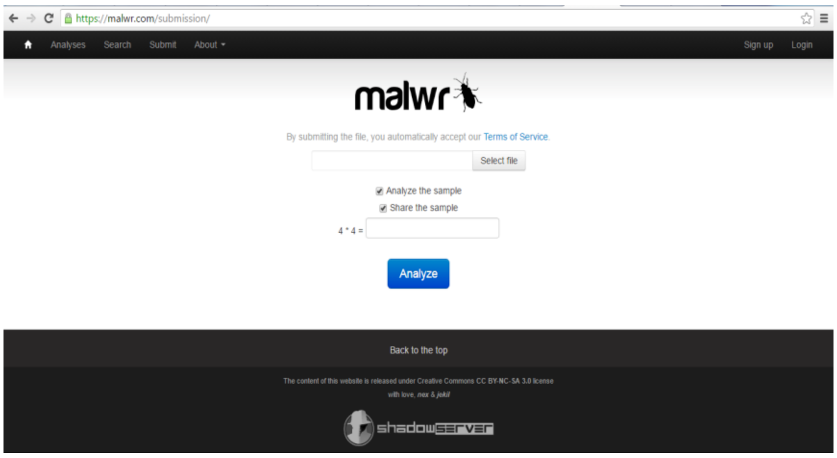
Some of the files provided for download may contain malware or exploits that I have collected through honeypots and other various means. All files containing malicious code will be password protected archives with a password of infected. **These are provided for educational purposes only.**

- 0.EXE.ZIP (32K)
- 1.EXE.ZIP (8K)
- 2D.EXE.ZIP (95K)
- 340S.EXE.ZIP (274K)
- 854137.EXE.ZIP (32K)
- BOMBERMANIA.EXE.ZIP (2.6M)
- GEPMALWARESAMPLE.RAR (251K)
- GOOGLE_ADOBE_FLASHPLAYER.EXE.ZIP (4K)
- KUNKRNMALWARESAMPLE.RAR (251K)
- MALZ6.ZIP (17.8M)
- MALZ.ZIP (41.8M)
- MALZ2.ZIP (1.2M)

Figure 2.12: The other samples of files infected by malware

Just downloading any provided files and make sure that opening and running that's file on the platform of VMWare to avoid damage on real PC.

After insert the sample of infected file in the Online Sandbox, it will analyze the behaviour or malware. This step can be conducted by going to the Online Sandbox's website: https://malwr.com/submission/ .



By submitting the file, you automatically accept our Terms of Service.

☑ Analyze the sample

☑ Share the sample

4 * 4 =

The content of this website is released under Creative Commons CC BY-NC-SA 3.0 license

with love, *nex* & *jekil*

Figure 2.13: The Infected file submission

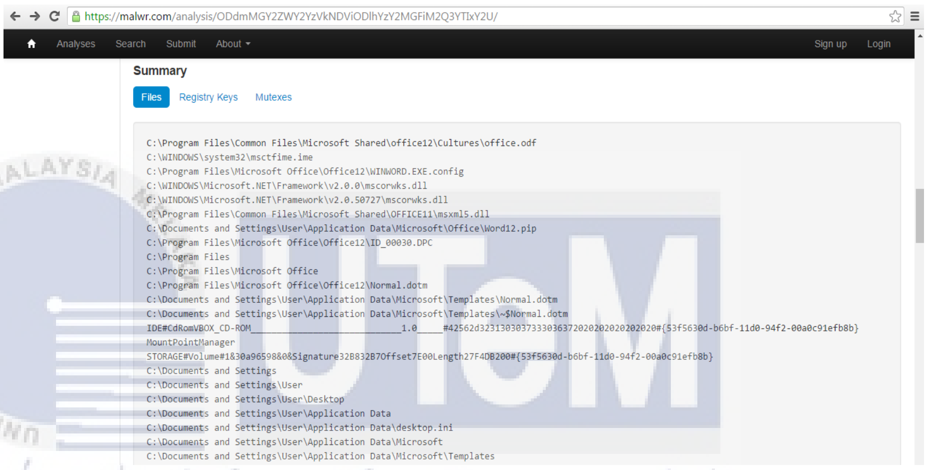
The Figure 2.13 above shows the one example of website of the Online Sandbox. Just selecting the infected file that has been downloaded and clicking the button analyze. Based on the flowchart on Figure 2.11, this is the data which is the input malware.

After clicking the button analyze, the Online Sandbox will keep on running and analyzing the behaviour of malware. Based on the flowchart on Figure 2.11, this process is on the Sandbox process.

Finally, the result from the Online Sandbox will be displayed. This result will be inserted to the source code of the Auto Virus Removal version 1 (AVRv1). Based on the flowchart on Figure 2.11, this is the data which is the output malware.

2. Signature-Based Detection

Then, copy all the samples of files that infected by malware in Windows to source code of the Auto Virus Removal version 1 (AVRv1). All of these samples can be found on the website: https://malwr.com/analysis/ODdmMGY2ZWY2YzVkNDViODlhYzY2MGFiM2Q3YTIxY2U/.



https://malwr.com/analysis/ODdmMGY2ZWY2YzVkNDViODlhYzY2MGFiM2Q3YTIxY2U/
Analyses
Search
Submit
About
Sign up
Login
Summary
Files
Registry Keys
Mutexes

```
C:\Program Files\Common Files\Microsoft Shared\office12\Cultures\office.odf
C:\WINDOWS\system32\msctfime.ime
C:\Program Files\Microsoft Office\Office12\WINWORD.EXE.config
C:\WINDOWS\Microsoft.NET\Framework\v2.0.0\mscorwks.dll
C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\mscorwks.dll
C:\Program Files\Common Files\Microsoft Shared\OFFICE11\msxml5.dll
C:\Documents and Settings\User\Application Data\Microsoft\Office\Word12.pip
C:\Program Files\Microsoft Office\Office12\ID_00030.DPC
C:\Program Files
C:\Program Files\Microsoft Office
C:\Program Files\Microsoft Office\Office12\Normal.dotm
C:\Documents and Settings\User\Application Data\Microsoft\Templates\Normal.dotm
C:\Documents and Settings\User\Application Data\Microsoft\Templates\~$Normal.dotm
IDE#CdRomVBOX_CD-ROM_____________________________1.0_____#42562d3231303037333036372020202020202020#{53f5630d-b6bf-11d0-94f2-00a0c91efb8b}
MountPointManager
STORAGE#Volume#1&30a96598&0&Signature32B832B7Offset7E00Length27F4DB200#{53f5630d-b6bf-11d0-94f2-00a0c91efb8b}
C:\Documents and Settings
C:\Documents and Settings\User
C:\Documents and Settings\User\Desktop
C:\Documents and Settings\User\Application Data
C:\Documents and Settings\User\Application Data\desktop.ini
C:\Documents and Settings\User\Application Data\Microsoft
C:\Documents and Settings\User\Application Data\Microsoft\Templates
```

Figure 2.14: The samples of infected files in Windows

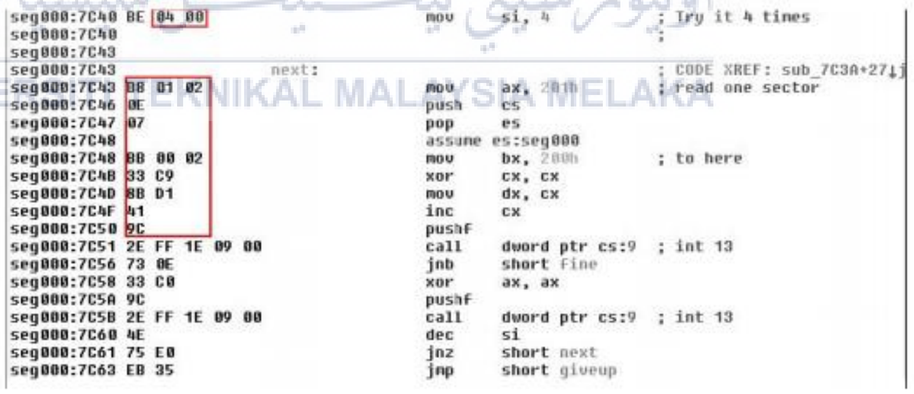
The Figure 2.14 above shows the list of files that infected by malware in Windows. Then, signatures the virus to registry in the source code of the Auto Virus Removal version 1 (AVRv1).

After completing the process of the Sandbox, copy the all files from the result of the Online Sandbox to the source code of the Auto Virus Removal version 1 (AVRv1). Based on the flowchart on Figure 2.11, this situation is on the virus detection process.

By running the Auto Virus Removal version 1 (AVRv1), it will try to detect the virus. In this situation, if the file signature from the source code is match with the file signature found in the Online Sandbox, so it will remove the detected virus automatically. Based on the flowchart on Figure 2.11, this situation is on the virus removal process.

If did not match, repeat the process by inserting the new sample of infected files to the Online Sandbox**.** Based on the flowchart on Figure 2.11, this situation is on decision process whether it is detected or not.

For an example, 83EB 0274 EB0E 740A 81EB 0301 0000 is the signature of the input file. This input will be searched in the signature database. Then, the file will be categorized as W32/Beast virus since 83EB 0274 EB0E 740A 81EB 0301 0000 is the signature of the W32/Beast virus. A same pattern used to detect Stoned virus is shown in Figure 2.15.



```
seg000:7C40 BE 04 00                     mov     si, 4             ; Try it 4 times
seg000:7C40                                                        ;
seg000:7C43
seg000:7C43                next:                                   ; CODE XREF: sub_7C3A+27↓j
seg000:7C43 B8 01 02                     mov     ax, 201h          ; read one sector
seg000:7C46 0E                           push    cs
seg000:7C47 07                           pop     es
seg000:7C48                              assume es:seg000
seg000:7C48 BB 00 02                     mov     bx, 200h          ; to here
seg000:7C4B 33 C9                        xor     cx, cx
seg000:7C4D 8B D1                        mov     dx, cx
seg000:7C4F 41                           inc     cx
seg000:7C50 9C                           pushf
seg000:7C51 2E FF 1E 09 00               call    dword ptr cs:9    ; int 13
seg000:7C56 73 0E                        jnb     short fine
seg000:7C58 33 C0                        xor     ax, ax
seg000:7C5A 9C                           pushf
seg000:7C5B 2E FF 1E 09 00               call    dword ptr cs:9    ; int 13
seg000:7C60 4E                           dec     si
seg000:7C61 75 E0                        jnz     short next
seg000:7C63 EB 35                        jmp     short giveup
```

Figure 2.15: Search pattern for Stone virus

3. On-Demand Scanning

For this project, the Auto Virus Removal version 1 (AVRv1) also using on-Demand Scanning technique in which scanning the files manually.

2.5 Conclusion

In conclusion, it can be understood on what the pattern of infected file from the output of the Online Sandbox. The techniques to detect the samples of infected file also have been considered on this project. The next chapter is about project methodology. It will cover about organization and flow of the project.

CHAPTER III

METHODOLOGY

3.1. Introduction

Project methodology is next process after doing the literature review in the project. It determine how best to plan, develop, control, and deliver a project throughout the continuous implementation process until successful completion and termination. The purpose of project methodology is to allow for controlling the entire management process through effective decision making and problem solving while ensuring the success of specific process, approaches, techniques, methods, and technologies.

3.2. Project Methodology

For this project, Waterfall Methodology has been chosen as the best method for the development process.

3.2.1. Waterfall Model

Waterfall Model was introduced by Winston W. Royce in 1970 to software development. It contains the phase of Requirement Gathering and Analysis, System Design, Implementation, Testing and Maintenance.. (Bassil, 2012)

Requirement Gathering and Analysis Phase is known as Software Requirement Analysis (SRS). It is included in specification process which consists of functional requirements and non-functional requirements. The functional requirements more focus on how the system should provide, how the system should react to particular inputs, and how the system should behave in particular situations. The non-functional requirements more constraints on the services or functions offered by the system such as timing constraints, constraints on the development process, and standards. All of these will be captured and documented in a requirement specification document.

Design Phase is the process of planning and problem solving for a software solution. This phase included in development process. This phase more focuses on how to interface with component, which component to use, how the system will behave, and how will the system respond on certain invocation. It consists of algorithm design, software architecture design, concept design, graphical user interface design, database conceptual schema and logical design, and data structure definition. The output from this phase is Software Detail Design (SDD) and Unit Test Plan (UTP).

Implementation Phase is the process of converting the whole requirements and blueprint into a production environment. This phase included in development process which consists of write program, debug program, code review, integrate modules, and Unit Testing. By getting the input from Design Phase, the system is developed in units. Each unit is developed and tested for its functionality. The output from this phase is Result of Unit Test Plan.

Testing Phase is a process for checking that a software solution meets the original requirements and specifications and that it accomplishes its intended purpose. This phase included in validation process which consists of integration testing, code review, debugging, and user acceptance test. All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any errors. The output from this phase is Software Test Result (STR).

Maintenance Phase is the process of modifying a software solution after delivery and deployment to refine output, correct errors, and improve performance and quality. This phase included in evolution process which more focuses on system in operation. Another activity in this phase is managing files and data. The maintenance activities can be more performed by changing request or adding features.

The Figure 3.1 below shows the Waterfall model which seen as following steadily downwards like a waterfall.

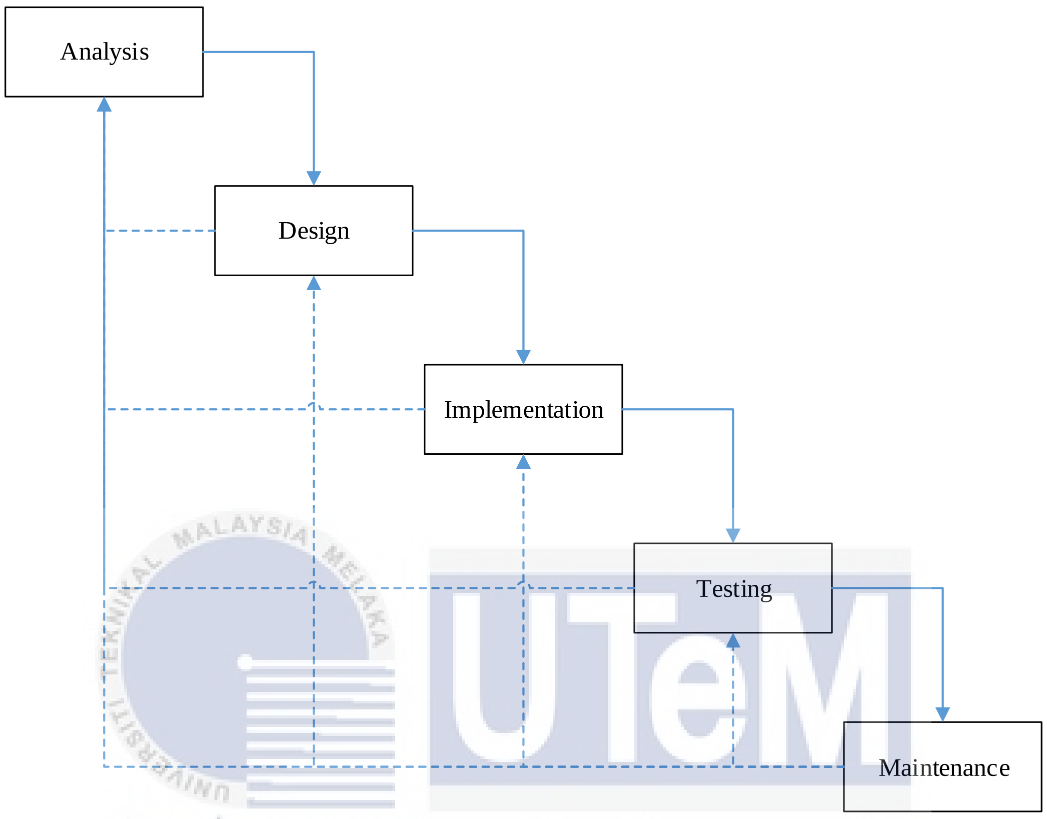


Figure 3.1: The Waterfall Model for this project methodology

In this model, one phase has to finish before move to the next phase Those five main phases will be scheduled in project schedulling.

1. **Analysis:** It will cover the activities needed for the chapter of introduction, chapter of literature review, and chapter of analysis. This phase has been made by collecting resources from journal, internet, book, and making some discussion with supervisor too. All the related resources

will be implemented to get the best method for developing this project. This phase also will be clearly plan the activities involved in creating the project through Gantt chart.

2. **Design:** It will cover all the design of the system. This phase also focus on the object placement in user interface in order to make it more user friendly. Any interface will be explained about the input and output design.

3. **Implementation:** It will cover the installation of software needed for the project. It will continue with creating the designs planned in Design phase and enhance with some codes and algorithms to make it functions properly. Any error found during the implementation will be fixed as well.

4. **Testing:** It will cover many testing on this project which is starting from the plan testing, strategy testing, design testing, and finally with result and analysis testing. Those testing will be scheduled and any error will be solved as well.

5. **Maintenance:** System will observe for certain period of time to make sure it fully operational and free of errors. Maintenance phase then continue with final presentation to supervisors which determine either that’s project will be passed or failed. Finally, final report must be done after doling some corrections and submit to supervisor.

3.2.2. Research Framework

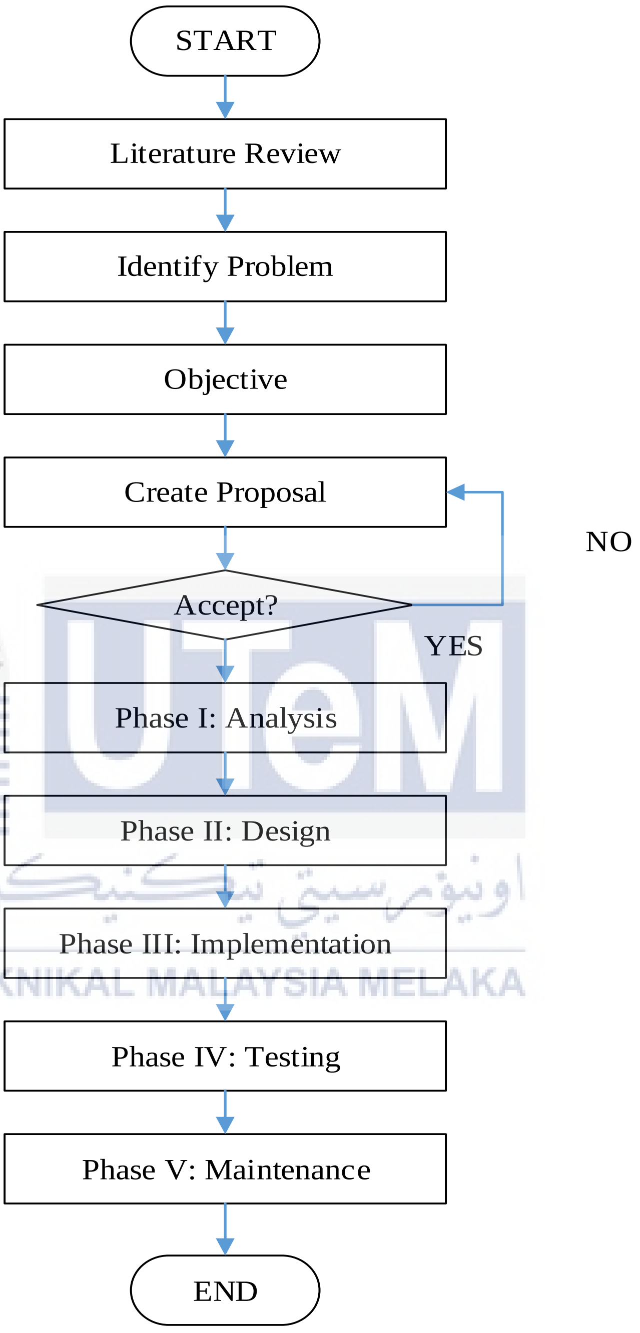


Figure 3.2: The Research Framework of project development

The Figure 3.2 above shows the research framework of project development. It is starting with finding any material and resources on literature review and identifies their problem. After the problem have considered, create the objectives based on the problem solutions. Then, a proposal will be created as beginning for developing a system. When a proposal has been accepted, the waterfall model is started to use. This model consist of five phases which are analysis phase, design phase, implementation phase, testing phase, and lastly maintenance phase.

3.3. Project Milestones

Project milestones or project scheduling is one of the important things while developing a project. There are some activities that planned through Gantt chart base on the main activities and days required to implement for each of the activities planned.

Table 3.1: Project Schedule and Milestones

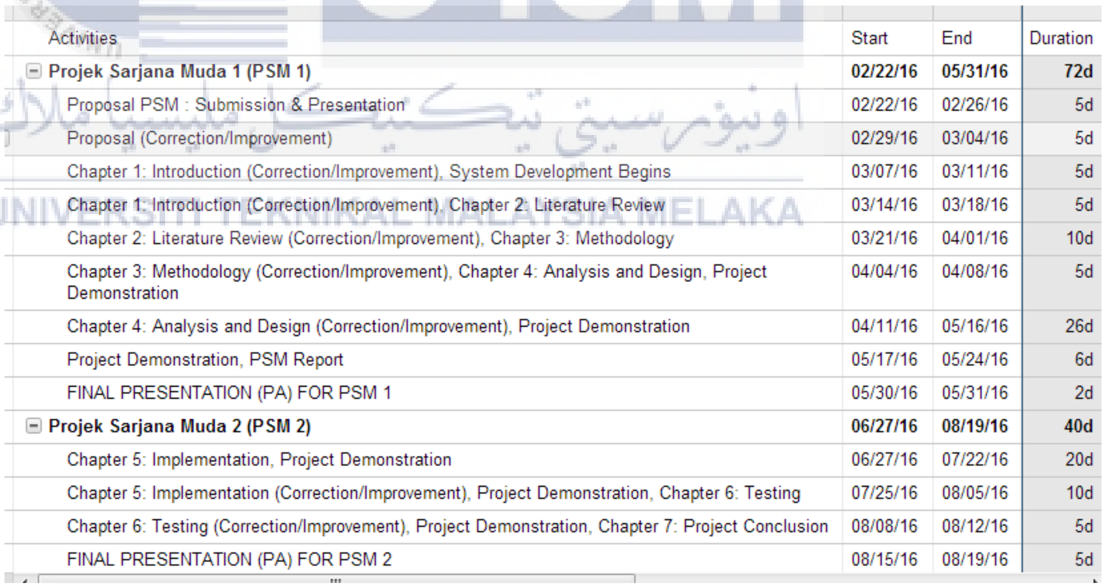
Week	Date	Activity
1	22 Feb 2016 – 26 Feb 2016	- Proposal PSM : Submission & Presentation - Proposal assessment and verification
2	29 Feb 2016 – 4 Mar 2016	- Proposal (Correction/Improvement) - Chapter 1: Introduction - List of supervisor/title
3	7 Mar 2016 – 11 Mar 2016	- Chapter 1: Introduction (Correction/Improvement) - System Development Begins
4	14 Mar 2016 – 18 Mar 2016	- Chapter 1: Introduction (Correction/Improvement) - Chapter 2: Literature Review
5	21 Mar 2016 – 1 April 2016	- Chapter 2: Literature Review (Correction/Improvement) - Chapter 3: Methodology
7	4 April 2016 – 8 April 2016	- Chapter 3: Methodology (Correction/Improvement) - Chapter 4: Analysis and Design - Project Demonstration

8	11 April 2016 – 16 April 2016	- Chapter 4: Analysis and Design (Correction/Improvement) - Project Demonstration
9	11 April 2016 – 16 May 2016	- Chapter 4: Analysis and Design (Correction/Improvement) - Project Demonstration
10	17 May 2016 – 24 May 2016	- Project Demonstration - PSM Report
11	30 May 2016 – 31 May 2016	- FINAL PRESENTATION (PA) FOR PSM 1
12	27 Jun 2016 - 22 July 2016	- Chapter 5: Implementation - Project Demonstration
13	25 July 2016 - 5 August 2016	- Chapter 5: Implementation (Correction/Improvement) - Project Demonstration - Chapter 6: Testing
14	8 August 2016 - 12 August 2016	- Chapter 6: Testing (Correction/Improvement) - Project Demonstration

		- Chapter 7: Project Conclusion
15	15 August 2016 - 19 August 2016	- FINAL PRESENTATION (PA) FOR PSM 2

Table 3.1 show the project schedule for PSM. Those activities are included from the early project of PSM 1 until the end of PSM.

3.4. Gantt Chart



Activities	Start	End	Duration
Projek Sarjana Muda 1 (PSM 1)	**02/22/16**	**05/31/16**	**72d**
Proposal PSM : Submission & Presentation	02/22/16	02/26/16	5d
Proposal (Correction/Improvement)	02/29/16	03/04/16	5d
Chapter 1: Introduction (Correction/Improvement), System Development Begins	03/07/16	03/11/16	5d
Chapter 1: Introduction (Correction/Improvement), Chapter 2: Literature Review	03/14/16	03/18/16	5d
Chapter 2: Literature Review (Correction/Improvement), Chapter 3: Methodology	03/21/16	04/01/16	10d
Chapter 3: Methodology (Correction/Improvement), Chapter 4: Analysis and Design, Project Demonstration	04/04/16	04/08/16	5d
Chapter 4: Analysis and Design (Correction/Improvement), Project Demonstration	04/11/16	05/16/16	26d
Project Demonstration, PSM Report	05/17/16	05/24/16	6d
FINAL PRESENTATION (PA) FOR PSM 1	05/30/16	05/31/16	2d
Projek Sarjana Muda 2 (PSM 2)	**06/27/16**	**08/19/16**	**40d**
Chapter 5: Implementation, Project Demonstration	06/27/16	07/22/16	20d
Chapter 5: Implementation (Correction/Improvement), Project Demonstration, Chapter 6: Testing	07/25/16	08/05/16	10d
Chapter 6: Testing (Correction/Improvement), Project Demonstration, Chapter 7: Project Conclusion	08/08/16	08/12/16	5d
FINAL PRESENTATION (PA) FOR PSM 2	08/15/16	08/19/16	5d

Figure 3.2: The time table of Gantt chart

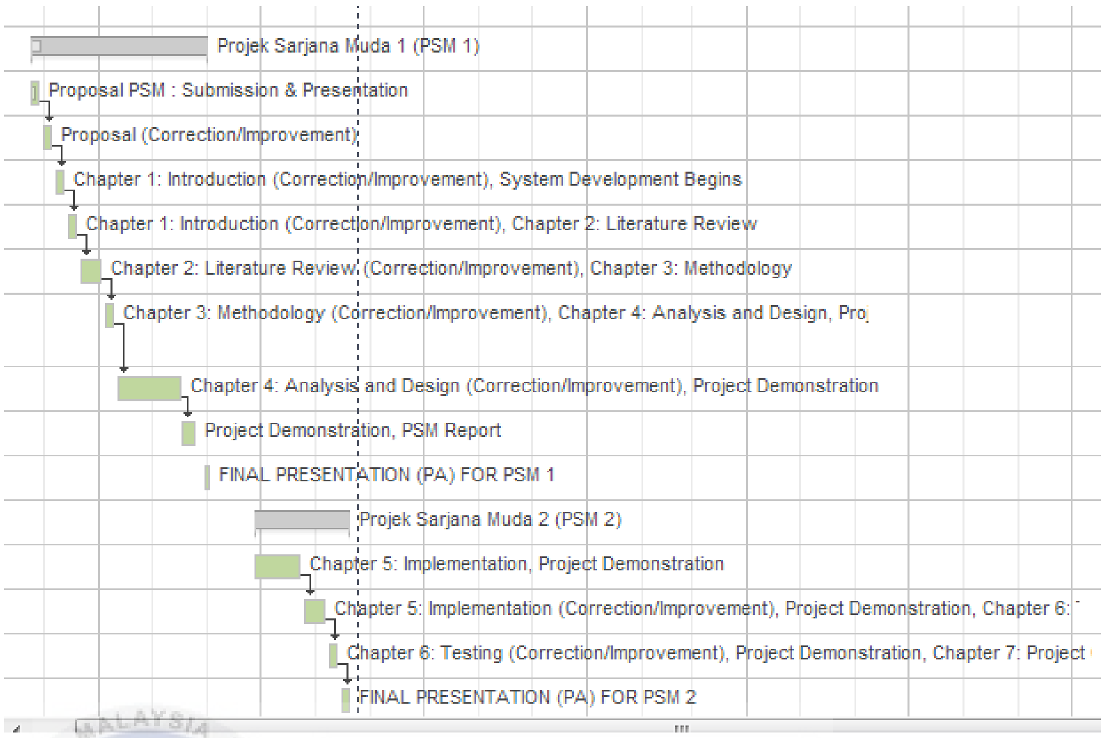


Figure 3.3: The timeline of Gantt chart

The Figure 3.2 and Figure 3.3 shows the Gantt chart of project based on project milestone given. By looking to this Gantt chart, it takes 72 days to complete the tasks on PSM 1 while it takes 40 days to finish the task on PSM 2. The total days to complete these tasks are 112 days.

3.5. Conclusion

For this chapter, methodology provides developer more understand on how to manage the project development. By referring to the implementation of resources on literature review, it helps developer more understand about the problem and opportunities of this project on methodology. Other than that, the software and hardware requirement may also be determined during project development. The next chapter is about analysis and design. It will cover the problem analysis and make the design from problem solutions.

CHAPTER IV

ANALYSIS AND DESIGN

4.1. Introduction

In this chapter, it will be focusing on the analysis and design of the system. The analysis that will be discussed which are problem analysis and requirement analysis. The design that will be reviewed consists of high-level design and detailed design.

The problem analysis will investigate and describe the current system. The result of investigation will be visualized using appropriate diagram. It will explain in detail about the problem statement that has been mentioned in the chapter 1. The requirement analysis include data requirement, functional requirement, non-functional requirement, and others requirement. Other requirement will describe about software requirement and hardware requirement that will be used while developing this system.

Software design is activity that acts as a bridge between requirements and the implementation of the software. It provides structure to any artefacts which decomposes system into parts, assigns responsibilities, and ensures that parts fit together to achieve a global goal. By *Thayer, R. H. (2004), Software Design Part 1, IEEE Software 110*, software design is the process of defining the software architecture, components, modules, interfaces, and data for a software system to satisfy specified requirements. The design is needed in this chapter whereby in this part the analysis designs to show the overall process of this project.

4.2. Flow Chart of the proposed system

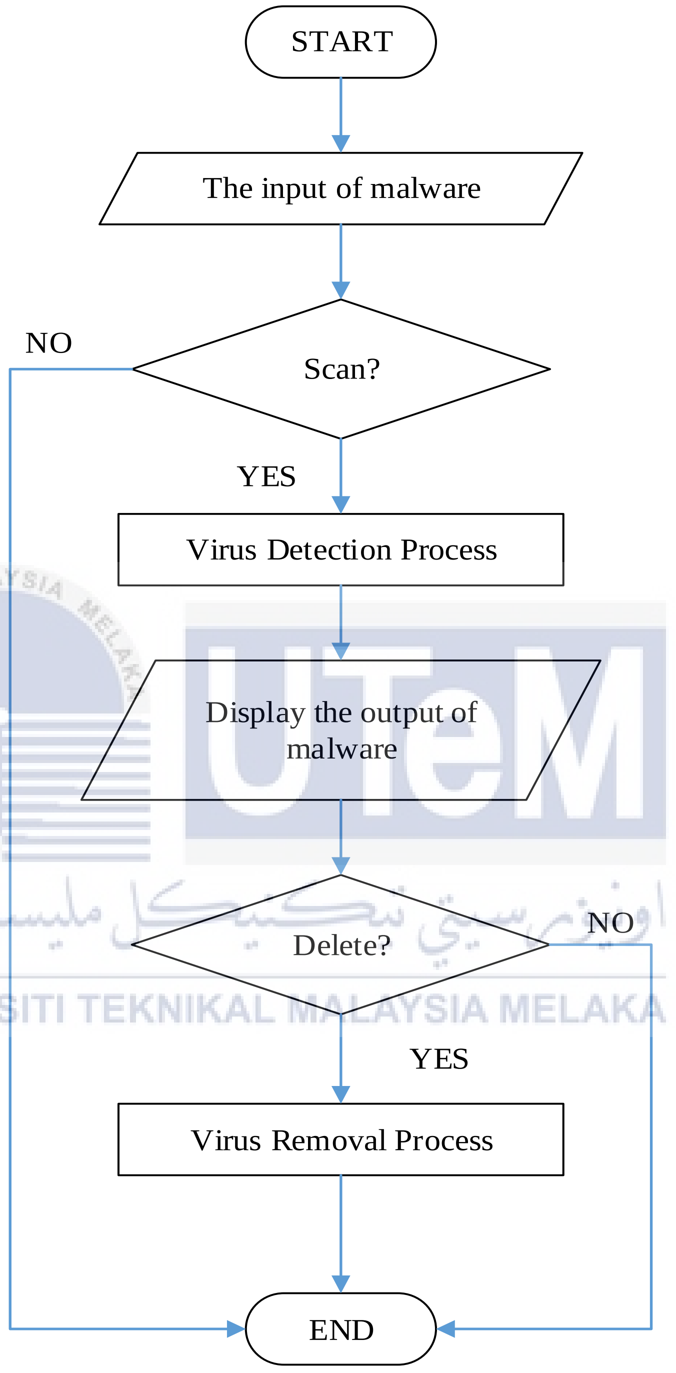


Figure 4.1: The flowchart of the proposed system

When scanning the virus infected storage with anti-virus software, it will inform to user that no threat detected which was really confusing because not all anti-virus can detect this virus. That is the main problem of the current system. Based on the flowchart on Figure 4.1, this problem generally will happen on the display of the output of malware.

In addition, somebody who didn't know on how to remove this virus is just formatting the infected storage as last choice. So, the virus with all the important or non-important data will be lost together. Other than that, the details about the techniques on how to use the virus remover tool at default web page are not more user-friendly and uninformative to user.

4.3. Requirement Analysis

For the requirement analysis, it is based on what system can do, what system must do, and what the user can do with system. There have several requirements which are data requirement, functional requirement, non-functional requirement, software requirement, and hardware requirement.

4.3.1. Data Requirement

As mentioned the justification technique on the literature review before, after the Online Sandbox analyzes the sample of infected file, the result from the Online Sandbox will be displayed. These result will be inserted to the source code of the Auto Virus Removal version 1 (AVRv1) for doing the virus detection process. These results are the input data for the Auto Virus Removal version 1 (AVRv1).

Then, the Auto Virus Removal version 1 (AVRv1) will detect whether the file signature from the source code is match with the file signature of this input data. If it match, the Auto Virus Removal version 1 (AVRv1) will remove it automatically which called virus removal process. These matched files are the output data for the Auto Virus Removal version 1 (AVRv1).

4.3.2. Functional Requirement

For the functional requirement, it is more focus on what the system is supposed to do. The Figure 4.2 shows the Data Flow Diagram (DFD) for a functional requirement in the system. There have two parts of devices which are local disk drive and removal drive. Both of these will be scanned and the detected files will be deleted.

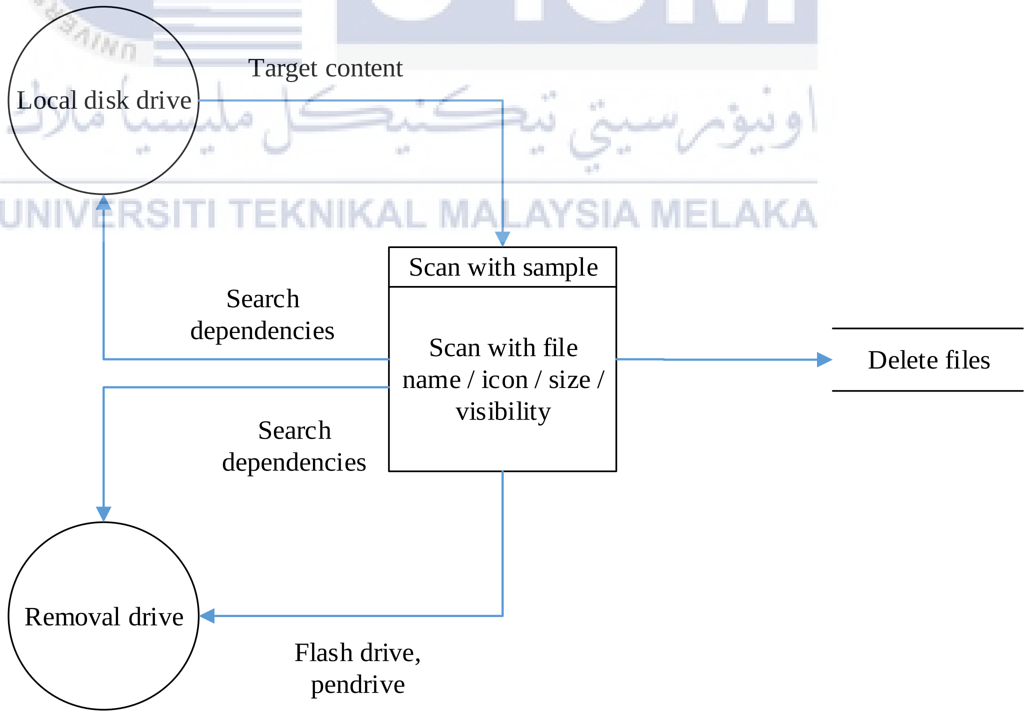


Figure 4.2: The Data Flow Diagram (DFD) for level 0 of the system

4.3.3. Non-functional Requirement

The non-functional requirement is more based on how the system is supposed to be. As using the signature-based detection technique, the virus will be removed if the file signature found in the output of the Online Sandbox is match with the file signature in the source code of the Auto Virus Removal version 1 (AVRv1). The Auto Virus Removal version 1 (AVRv1) will detect and remove the whole of virus only. These viruses may a shortcut virus, boot virus, program virus, and so on.

4.3.4. Other Requirement

For the other requirement, there are including software requirement and hardware requirement.

1. Software Requirement

For the software requirement, it contains six requirements that will be used for developing this project. These are:

a) **The Online Sandbox:** This software is used to obtain the behaviours of malware on a file when it runs. It is important to know which files are infected by malware to be listed and moved in the source code of project.

b) **VMware Workstation:** This software is virtual machine that used to test the functionality of the Auto Virus Removal Version 1 (AVRv1) by detecting and removing the files that infected by the malware from the output of the Online Sandbox. The kind of this solution is to avoid damage on real machine especially the registries on windows.

c) **Java Development Kit:** This software is used in order to let the operating system to be able to run java program.

d) **Eclipse Java Neon:** This software is used as platform for design system interface. This software provides drag-and-drop function and so the interface can be developed easily and quickly.

e) **Microsoft Project:** This software is used for producing Gantt chart and other related with project management.

f) **Notepad++:** This software is used to create a code on HTML, CSS, and JavaScript. After finish create the code, just running and launching in Chrome, Internet Explorer, or Mozilla Firefox to see the interface of webpage.

2. Hardware Requirement

For the hardware requirement, it contains four requirements that will be used for developing this project. These are:

Table 4.1: The hardware requirements

Hardware	Specifications
RAM	4 GB
Hard Disk	500 GB
Processor	Intel Premium i5
Operating System	Windows 7

The Table 4.1 above shows the hardware requirements which are RAM, hard disk, processor, and operating system. This specification also is considered for each hardware requirement.

4.4. High-level Design

For the High-level design, it will discuss about the architecture that would be used for developing this system. It consist of system architecture and user interface design.

4.4.1. System Architecture

System architecture provide fundamental framework for structuring the system and guides the development of the design. It also provides a way of analyzing systems at high-level of abstraction. The design process for identifying the sub-systems making up a system and the framework for sub-system control and communication is architectural design. In short, it is the output of the design process.

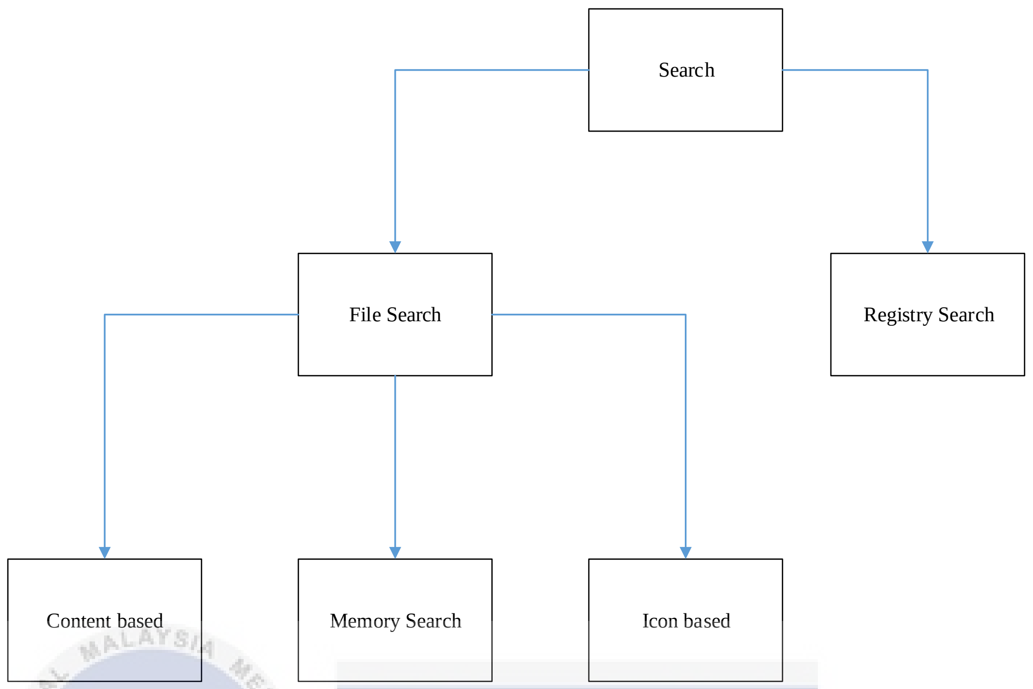


Figure 4.3: The system architecture

The Figure 4.3 shows the system architecture in high-level design. By looking to this figure, it contains three types of search which are memory search, file search, and registry search while this memory search is included in the file search. Other items that contains in file search are content based and icon based.

Based on this system architecture, the project will concentrate on file search and content based only.

4.4.2. User Interface Design

For user interface design, there have two types of devices to choose for scanning which are the computer and the pendrive. User need to run the administrator of the system first.

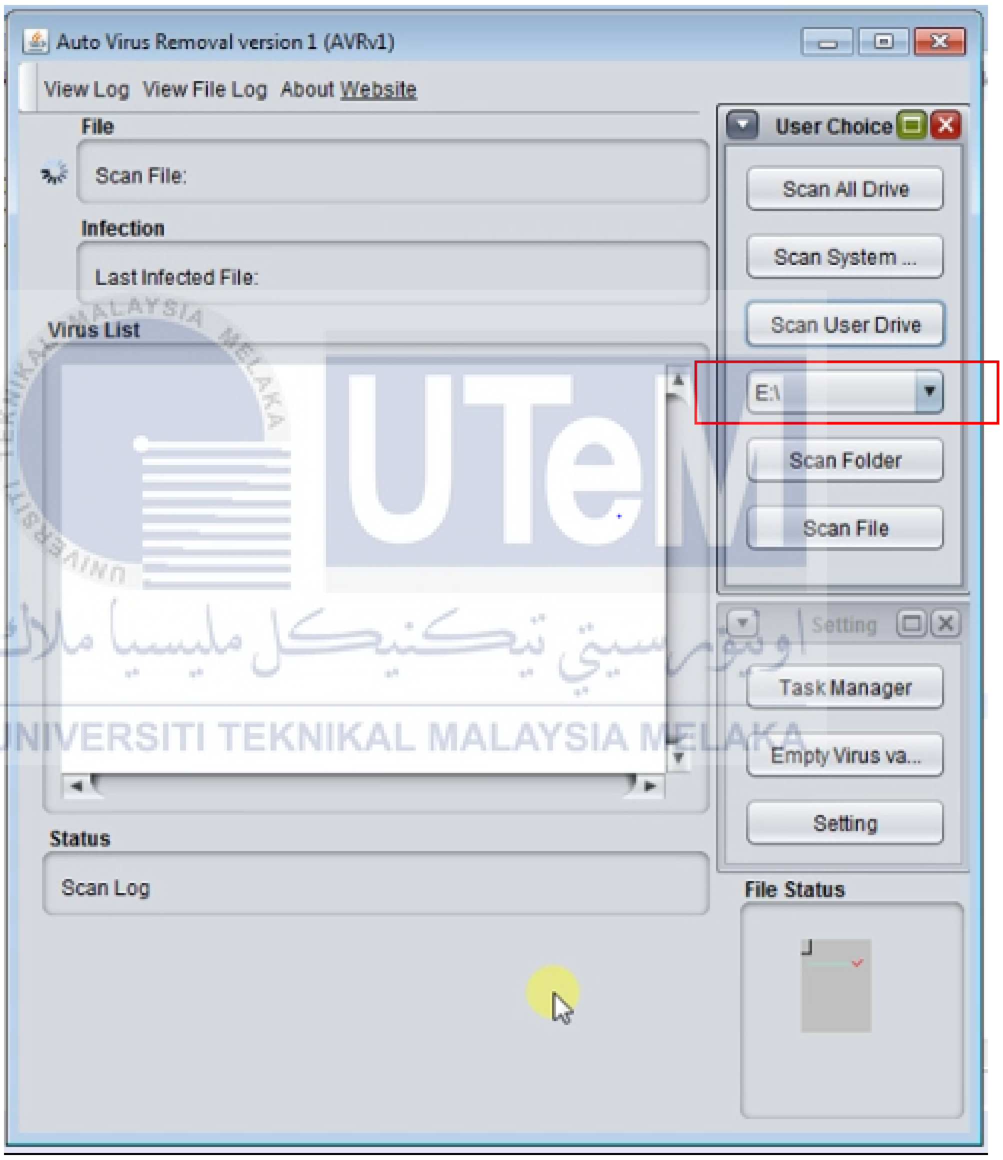
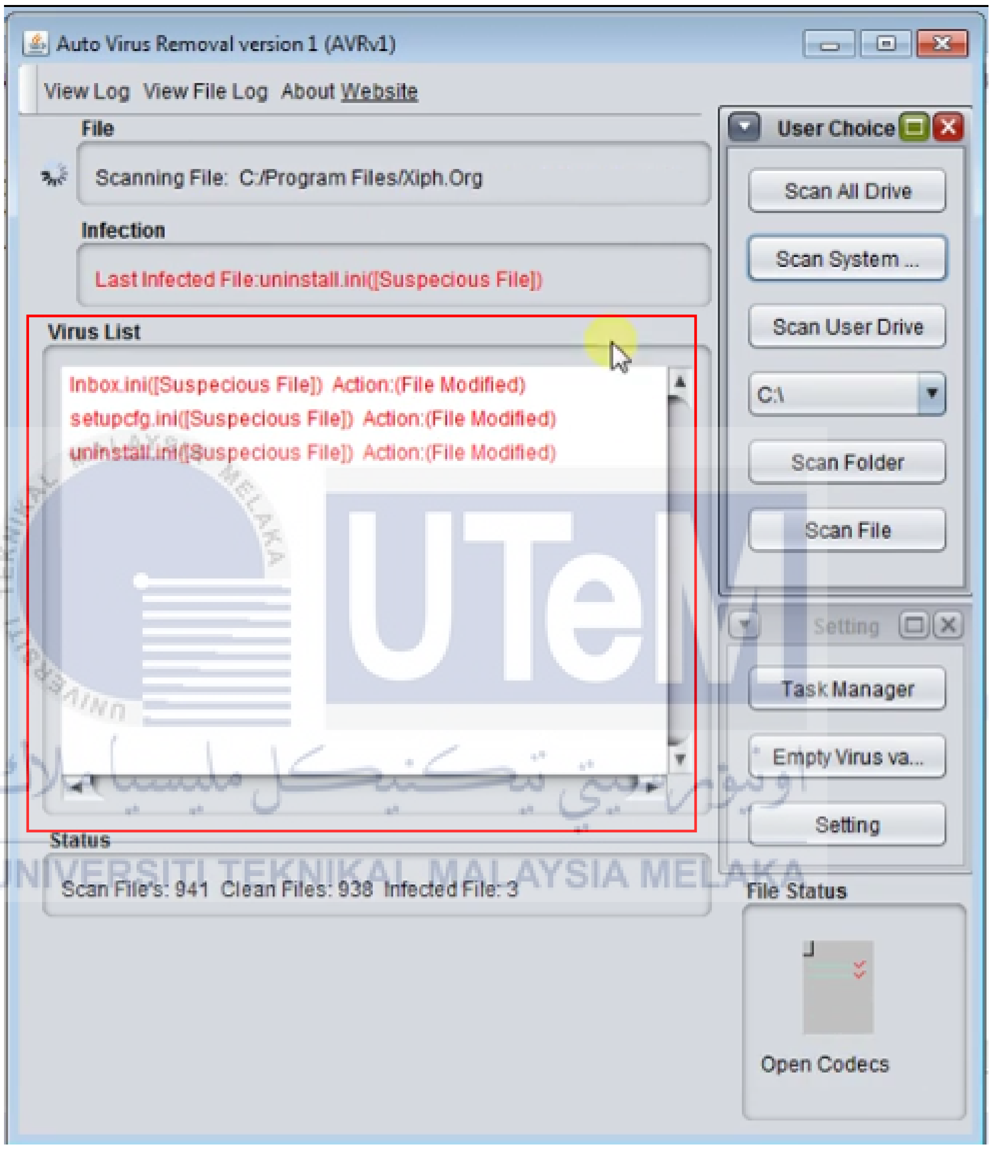


Figure 4.4: The home page of the system

The Figure 4.4 shows the home page of the system. User will be given the option either to choose on the computer or external device for scanning the device as shown on the mark given in this figure.



Scanning File: C:/Program Files/Xiph.Org

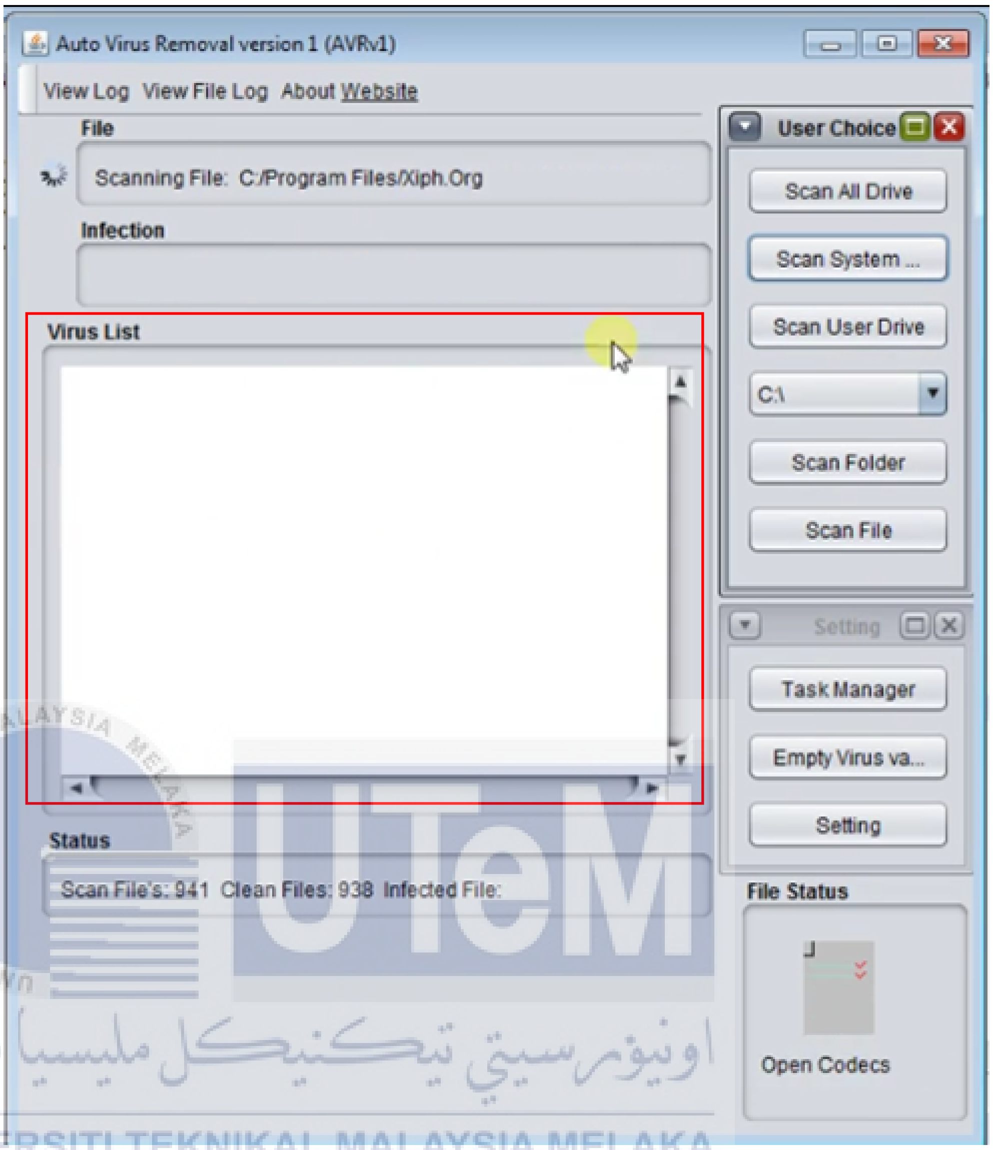
Last Infected File:uninstall.ini([Suspecious File])

Inbox.ini([Suspecious File]) Action:(File Modified)
setupcfg.ini([Suspecious File]) Action:(File Modified)
uninstall.ini([Suspecious File]) Action:(File Modified)

Scan File's: 941 Clean Files: 938 Infected File: 3

Figure 4.5: The interface in the part of computer scanning

The Figure 4.5 shows the scanning process and the list of infected file will be displayed on the mark given in this figure. . Just click any scan button and it will start running.



Scanning File: C:/Program Files/Xiph.Org

Scan File's: 941 Clean Files: 938 Infected File:

Figure 4.6: The result after scanning process

The result of the scanning process will be displayed after the computer scanned. All infected files will automatically deleted after the infected files detected. Just scanning again on the same device and the list of infected files that detected before will not appear for the second time as shown in Figure 4.6.

4.5. Detailed Design

For the detailed design, it will explain more detail about the logic of the design and the approach to satisfying the requirements. It consists of software design.

4.5.1. Software Design

For software design, the specific function of the system will be explained based on the Data Flow Diagram (DFD) on the Figure 4.2.

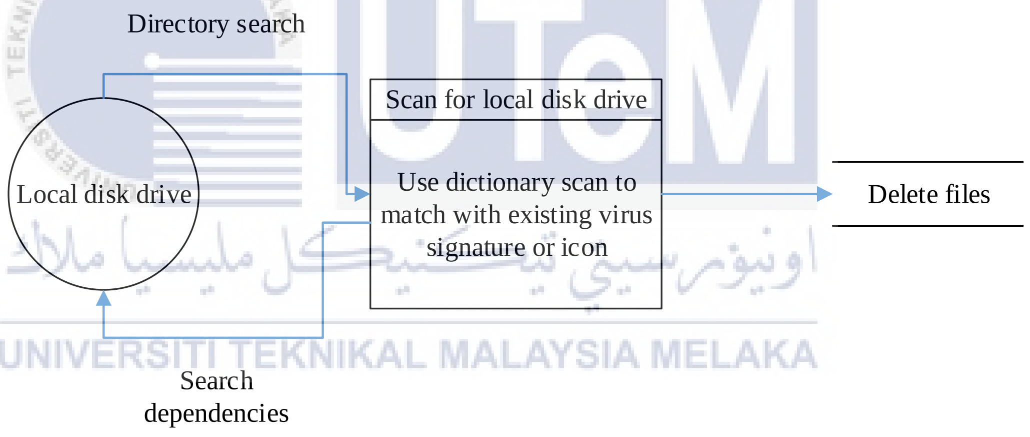

Directory search
Scan for local disk drive
Local disk drive
Use dictionary scan to match with existing virus signature or icon
Delete files
Search dependencies

Figure 4.7: The Data Flow Diagram (DFD) for level 1 on the computer scanning

The Figure 4.7 shows the Data Flow Diagram (DFD) for level 1 on the computer scanning. For scanning the local disk drive, it will use the signature-based detection and delete the detected files.

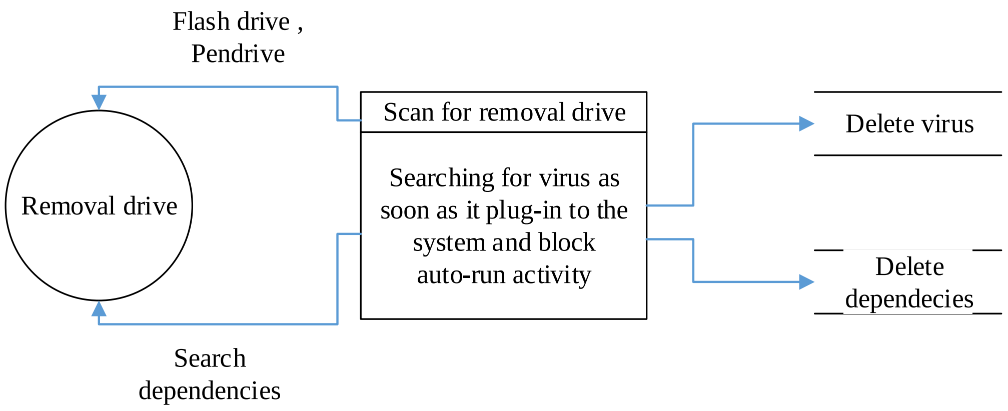


Figure 4.8: The Data Flow Diagram (DFD) for level 1 on the pendrive scanning

The Figure 4.8 shows the Data Flow Diagram (DFD) for level 1 on the pendrive scanning. For scanning the removal drive, it will search for virus as soon as it plug-in to the system and block auto-run activity. Finally, delete the virus and their dependencies.

4.6. Conclusion

This chapter has reviewed a design that covered all the process of this project which is system design for the whole project and explanation the details of this project. In the next chapter, the implementation of this system will be discussed.

CHAPTER V

IMPLEMENTATION

5.1. Introduction

In this chapter, it will be focusing on the software development environment setup, software configuration management, and implementation status of this system. The activities involved in this chapter are define the development environment setup, configuration management setup, and design setup. Other than that, it will explain about the software tools that used for configuration tool, and describe the progress of the development status of each module. As the result, the software tools can be configured correctly and so the system can be implemented successfully.

5.2. Software Development Environment setup

The implementation of this project will be identified in this section. This project is standalone information-based system which is developed using Java Development Kit. The extra plug-in that used for this system is Online Sandbox. The Figure 5.1 shows the software development environment setup architecture of the Auto Virus Removal version 1(AVRv1).

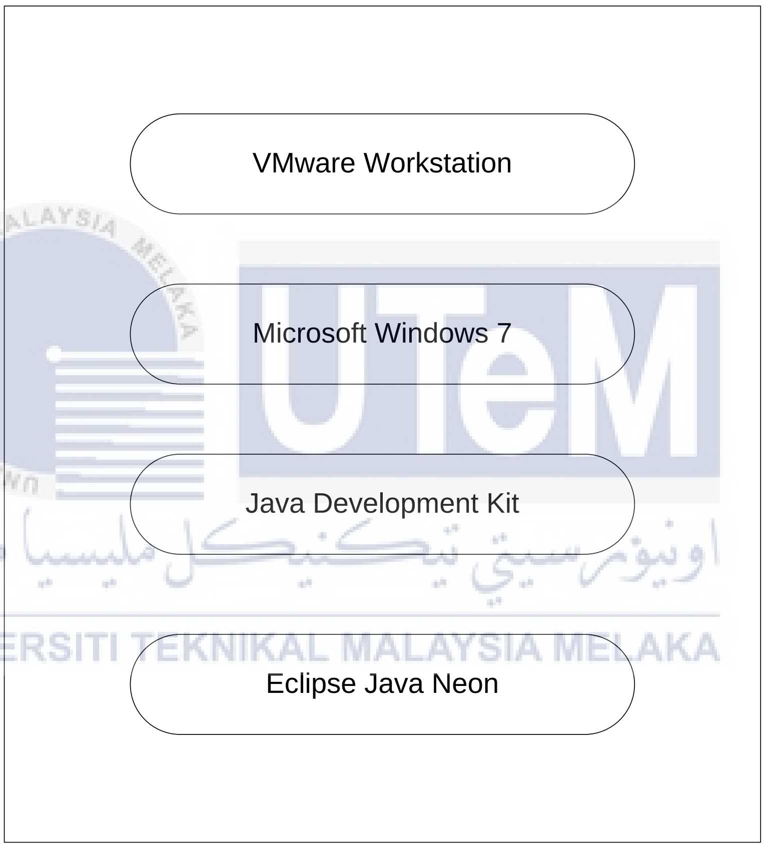


Figure 5.1: The Software Development Environment Setup Architecture

5.2.1. Installation of required software

VMware Workstation was installed and used as the platform to run this system on the personal computer. This system should be running on the virtual machine in order to protect physical machine from virus infection. Other than that, Microsoft Windows 7 was installed on the platform of VMware Workstation. This operating system was selected because nowadays many users are using Microsoft window 7 compared to another operating system. Java Development Kit (JDK) was installed in order to let the operating system to be able to run java program. Lastly, Eclipse Java Neon was installed which provides Graphical User Interface (GUI) for develop the interfaces for the system. In addition, Graphical User Interface (GUI) provides drag-and-drop function which can develop the interfaces easily and quickly.

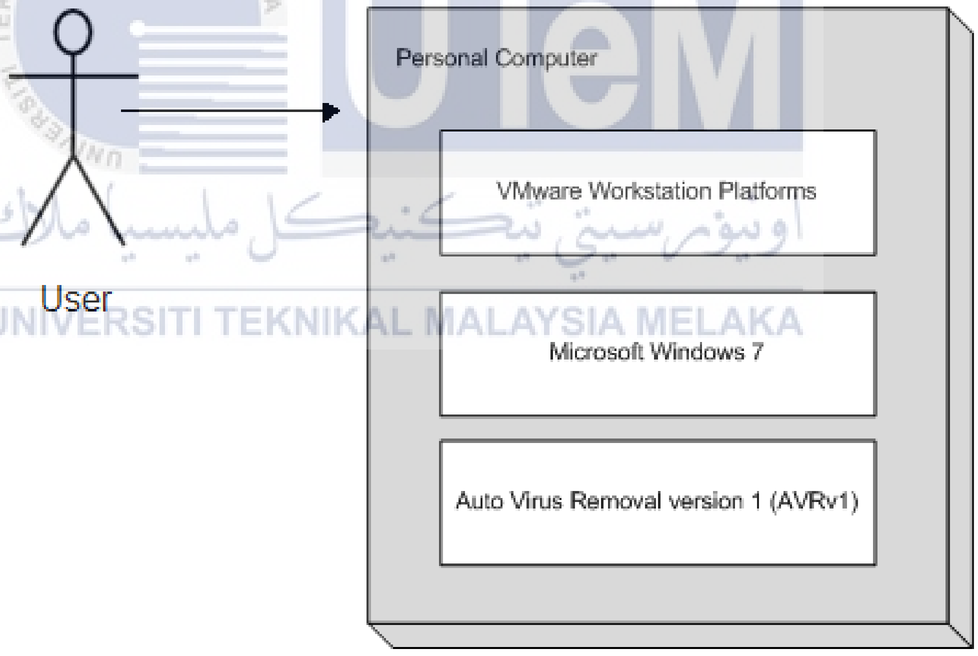

User
Personal Computer
VMware Workstation Platforms
Microsoft Windows 7
Auto Virus Removal version 1 (AVRv1)

Figure 5.2: Deployment Diagram

The Figure 5.2 shows the arrangement of software and hardware to run the Auto Virus Removal version 1 (AVRv1). The personal computer is a user and will be conducted as a required hardware.

5.2.2. Indicator of compromise

After all components implemented, some of the virus program (.exe) should be running on the platform of VMware in order to know their characteristics. After the virus program (.exe) ran, some of the files in the windows was infected. What can be concluded about this situation is the viruses can infect any file at any time. The Figure 5.3 shows other symptom of virus infection.

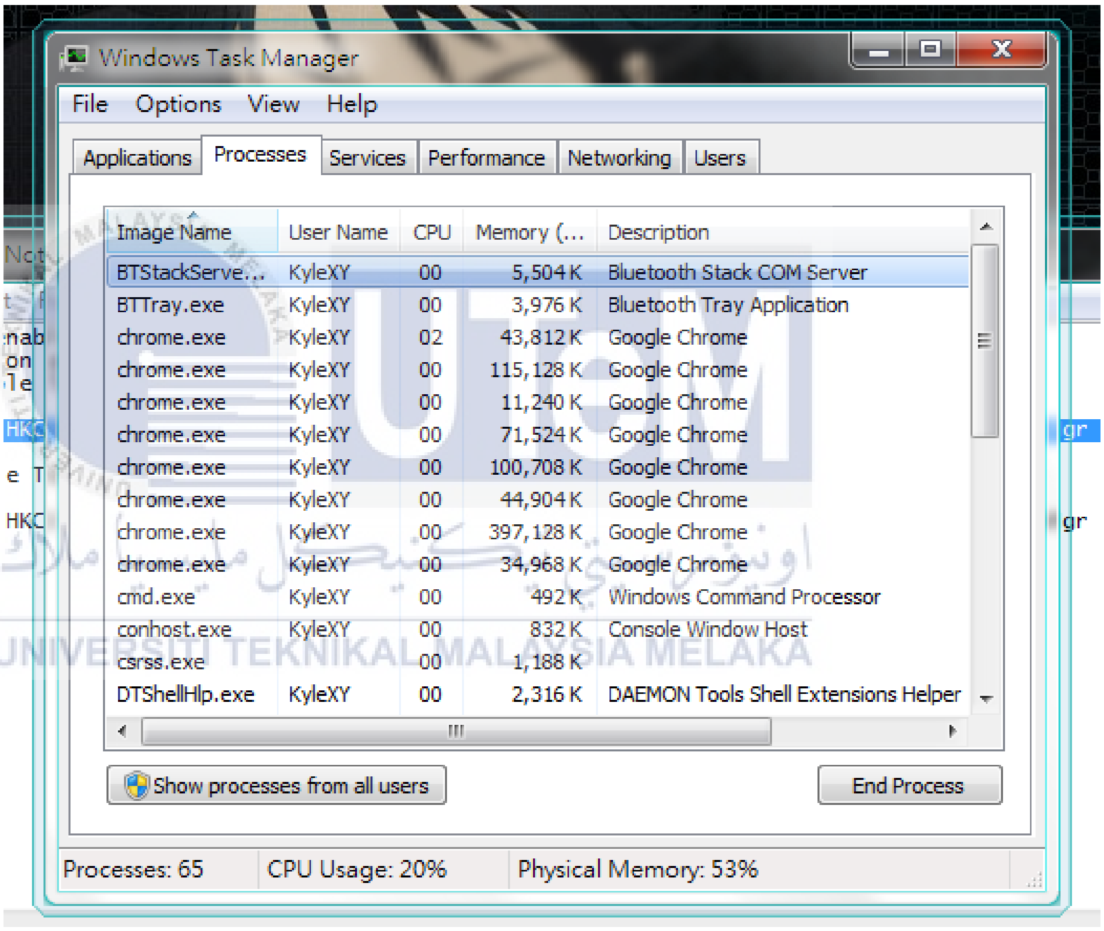
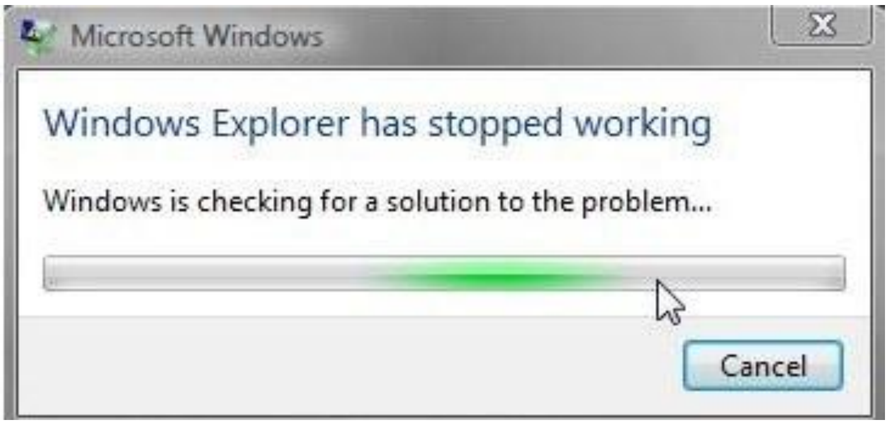


Image Name	User Name	CPU	Memory (...	Description
BTStackServe...	KyleXY	00	5,504 K	Bluetooth Stack COM Server
BTTray.exe	KyleXY	00	3,976 K	Bluetooth Tray Application
chrome.exe	KyleXY	02	43,812 K	Google Chrome
chrome.exe	KyleXY	00	115,128 K	Google Chrome
chrome.exe	KyleXY	00	11,240 K	Google Chrome
chrome.exe	KyleXY	00	71,524 K	Google Chrome
chrome.exe	KyleXY	00	100,708 K	Google Chrome
chrome.exe	KyleXY	00	44,904 K	Google Chrome
chrome.exe	KyleXY	00	397,128 K	Google Chrome
chrome.exe	KyleXY	00	34,968 K	Google Chrome
cmd.exe	KyleXY	00	492 K	Windows Command Processor
conhost.exe	KyleXY	00	832 K	Console Window Host
csrss.exe		00	1,188 K	
DTShellHlp.exe	KyleXY	00	2,316 K	DAEMON Tools Shell Extensions Helper

Processes: 65 CPU Usage: 20% Physical Memory: 53%

Figure 5.3: Virus infected task manager

Another symptom of virus infection such as the performance of program become slow and sometime the program will be noticed like 'Not responding' or 'Windows Explorer has stopped working' such as the Figure 5.4 below.


Microsoft Windows

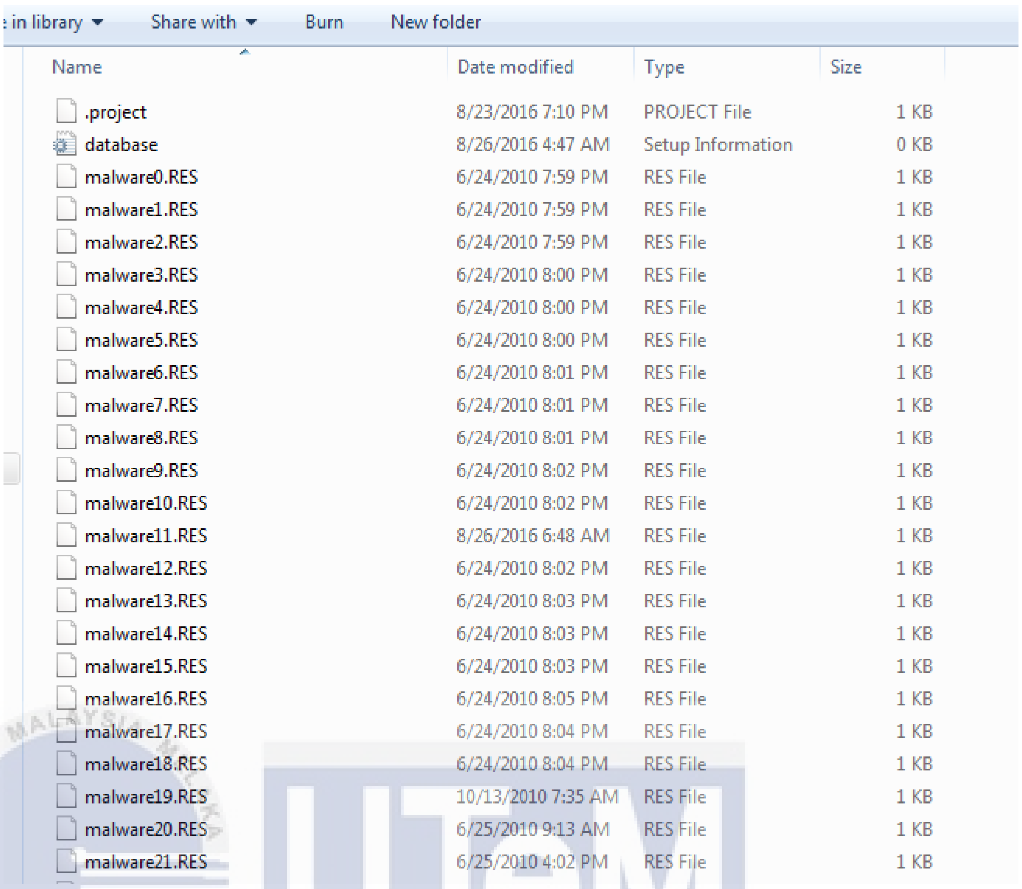
Windows Explorer has stopped working

Windows is checking for a solution to the problem...

Figure 5.4: The problem of Microsoft Windows

5.2.3. Update the file signature in database

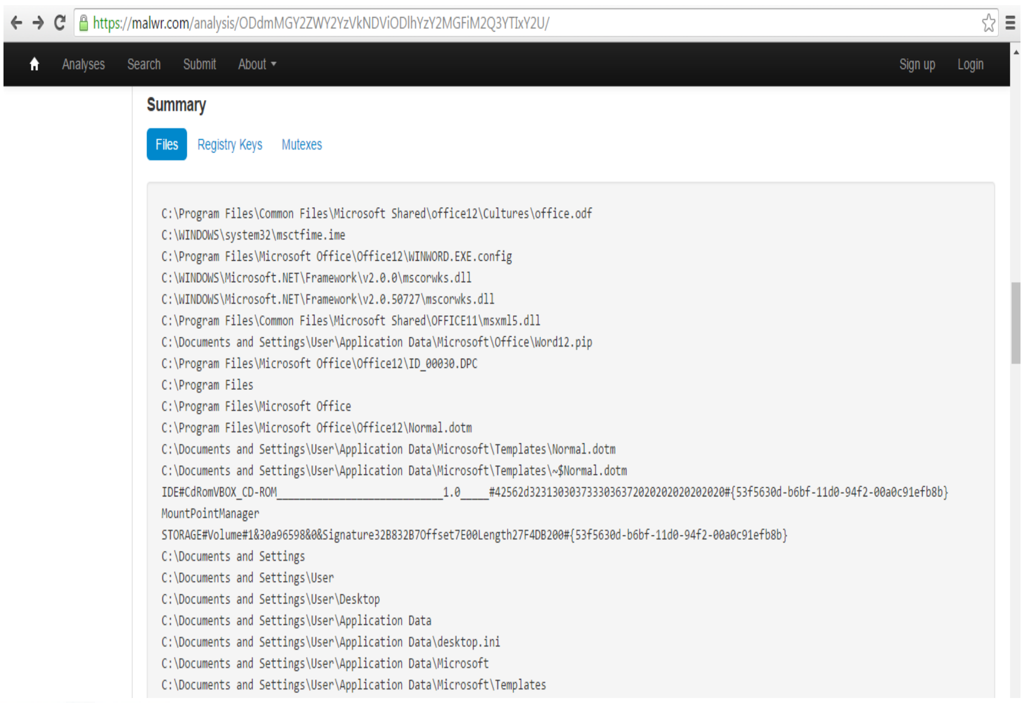
The file signature in the database is the input of the Auto Virus Removal version 1 (AVRv1) which is used to match with the pattern of infected file. If the pattern of infected file is matched with the pattern file in database of the system, it means that infected file is detected and it will be automatically removed.



Name	Date modified	Type	Size
.project	8/23/2016 7:10 PM	PROJECT File	1 KB
database	8/26/2016 4:47 AM	Setup Information	0 KB
malware0.RES	6/24/2010 7:59 PM	RES File	1 KB
malware1.RES	6/24/2010 7:59 PM	RES File	1 KB
malware2.RES	6/24/2010 7:59 PM	RES File	1 KB
malware3.RES	6/24/2010 8:00 PM	RES File	1 KB
malware4.RES	6/24/2010 8:00 PM	RES File	1 KB
malware5.RES	6/24/2010 8:00 PM	RES File	1 KB
malware6.RES	6/24/2010 8:01 PM	RES File	1 KB
malware7.RES	6/24/2010 8:01 PM	RES File	1 KB
malware8.RES	6/24/2010 8:01 PM	RES File	1 KB
malware9.RES	6/24/2010 8:02 PM	RES File	1 KB
malware10.RES	6/24/2010 8:02 PM	RES File	1 KB
malware11.RES	8/26/2016 6:48 AM	RES File	1 KB
malware12.RES	6/24/2010 8:02 PM	RES File	1 KB
malware13.RES	6/24/2010 8:03 PM	RES File	1 KB
malware14.RES	6/24/2010 8:03 PM	RES File	1 KB
malware15.RES	6/24/2010 8:03 PM	RES File	1 KB
malware16.RES	6/24/2010 8:05 PM	RES File	1 KB
malware17.RES	6/24/2010 8:04 PM	RES File	1 KB
malware18.RES	6/24/2010 8:04 PM	RES File	1 KB
malware19.RES	10/13/2010 7:35 AM	RES File	1 KB
malware20.RES	6/25/2010 9:13 AM	RES File	1 KB
malware21.RES	6/25/2010 4:02 PM	RES File	1 KB

Figure 5.5: The list of file signature in the database of the system

The Figure 5.5 shows the list of file signature in the database of the system. Each of file has one pattern that has been collected from the output of Online Sandbox. There have many pattern of file such as desktop.ini, Inbox.ini, malware1.exe, netmsg.dll, and many more. If user want to update the file signature, just open the malware file by notepad, then replace the new pattern of infected file on that malware file and save it manually. The new pattern of infected file can get from the output of Online Sandbox as shown in the Figure 5.6 below and just copy the pattern of infected file only.


https://malwr.com/analysis/ODdmMGY2ZWY2YzVkNDViODlhYzY2MGFiM2Q3YTIxY2U/
Analyses
Search
Submit
About
Sign up
Login
Summary
Files
Registry Keys
Mutexes

```
C:\Program Files\Common Files\Microsoft Shared\office12\Cultures\office.odf
C:\WINDOWS\system32\msctfime.ime
C:\Program Files\Microsoft Office\Office12\WINWORD.EXE.config
C:\WINDOWS\Microsoft.NET\Framework\v2.0.0\mscorwks.dll
C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\mscorwks.dll
C:\Program Files\Common Files\Microsoft Shared\OFFICE11\msxml5.dll
C:\Documents and Settings\User\Application Data\Microsoft\Office\Word12.pip
C:\Program Files\Microsoft Office\Office12\ID_00030.DPC
C:\Program Files
C:\Program Files\Microsoft Office
C:\Program Files\Microsoft Office\Office12\Normal.dotm
C:\Documents and Settings\User\Application Data\Microsoft\Templates\Normal.dotm
C:\Documents and Settings\User\Application Data\Microsoft\Templates\~$Normal.dotm
IDE#CdRomVBOX_CD-ROM_____________________________1.0_____#42562d3231303037333036372020202020202020#{53f5630d-b6bf-11d0-94f2-00a0c91efb8b}
MountPointManager
STORAGE#Volume#1&30a96598&0&Signature32B832B7Offset7E00Length27F4DB200#{53f5630d-b6bf-11d0-94f2-00a0c91efb8b}
C:\Documents and Settings
C:\Documents and Settings\User
C:\Documents and Settings\User\Desktop
C:\Documents and Settings\User\Application Data
C:\Documents and Settings\User\Application Data\desktop.ini
C:\Documents and Settings\User\Application Data\Microsoft
C:\Documents and Settings\User\Application Data\Microsoft\Templates
```

Figure 5.6: The output of Online Sandbox

5.3. Software Configuration Management

This section has a sub-section which is Configuration Environment Setup. It is more focus on the design setup and also on the configuration management setup.

5.3.1. Configuration environment setup

After Eclipse Java Neon was installed, a project with name "Auto Virus Removal version 1" has been created. The figures show the pseudo code and their function of each interface of the system:

a. Scan the system, file, and folder

i. Description: This function is to scan the system, file and folder in windows

ii. Input:

virusname.text: string - String value contained in *virusname*

type.text: string - String value contained in *type*

wild.text: string - String value contained in *wild*

threatlevel.text: string - String value contained in *threatlevel*

affectedplatform.value: number - Integer value contained in *affectedplatform*

Output:

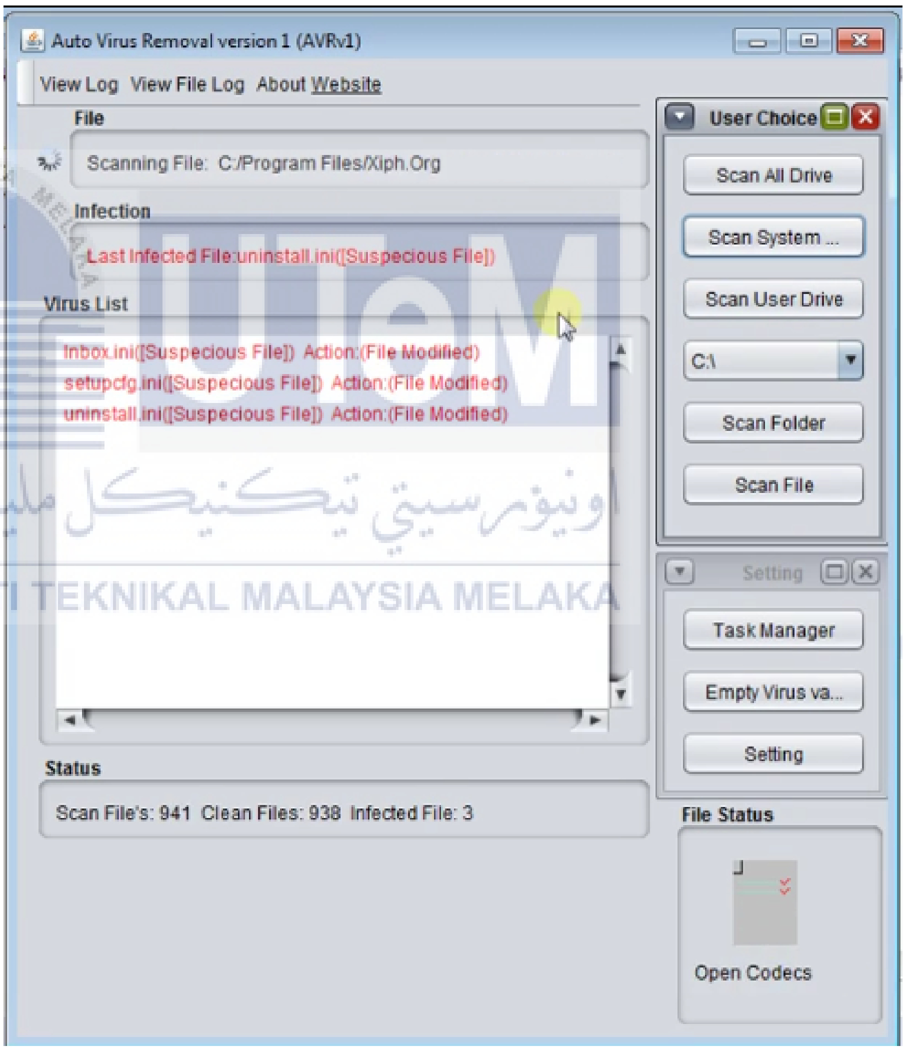
N/A

iii. Pseudo code:

```
for(int io=0;io<point;io++)
                {
                        if(  database[io].equals(
str221[i1] ) )
                        {

jl2.setForeground(Color.red);
                                                jl2.setText("Last
Infected File:"+str221[i1]+"([trojen-gender])");




System.out.println("\n\n************* $Virus Found$
***********");
                                                virus v=new
virus(strRoot[iq]+str22[i]+"/"+str221[i1]);


        System.out.println("2."+strRoot[iq]+str22[i]+"/"+str22
1[i1]);
                                                vir++;
```

```
                                        infected=infected+1;

                        mu.addElement(str221[i1]+"([trojen-gender]) Action:(File
                        Modified)");
                                        }

                                                }
```

iv. Interface:



Scanning File: C:/Program Files/Xiph.Org

Last Infected File:uninstall.ini([Suspecious File])

inbox.ini([Suspecious File]) Action:(File Modified)
setupcfg.ini([Suspecious File]) Action:(File Modified)
uninstall.ini([Suspecious File]) Action:(File Modified)

Scan File's: 941 Clean Files: 938 Infected File: 3

Figure 5.7: The interface of scanning process

b. Delete the infected file

i. Description:

This is the function to delete the infected file that matched with in database. The infected file that detected before will not detected for the second time when rescan on the same device.

ii. Input:

virusname.text: string - String value contained in virusname

Output:

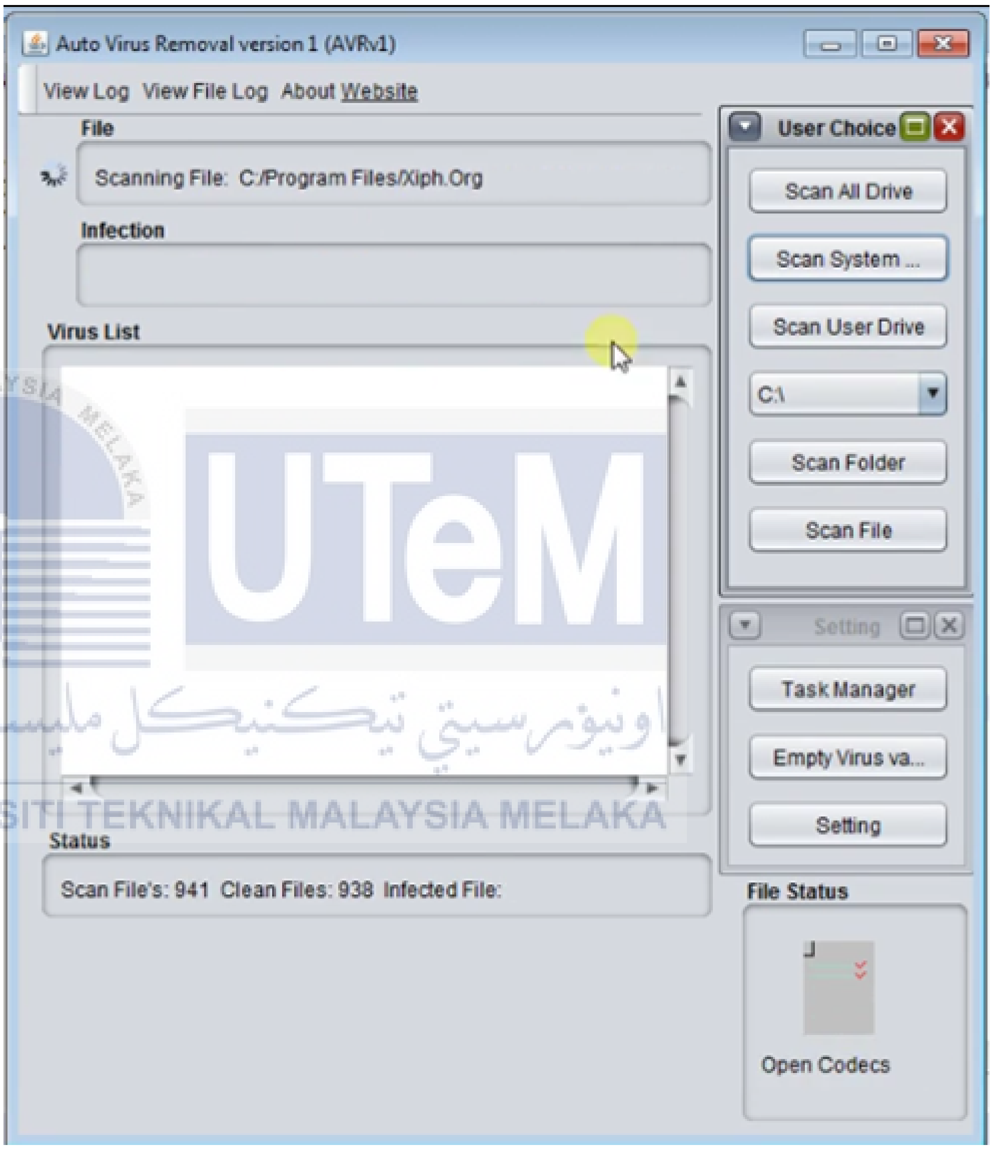
N/A

iii. Pseudo code:

```
class virus{
        BufferedReader br=null;
        public String fname="";
        int count=0;
        int ans=0;
        int flag=0;
        JButton b1,b2,b3,b4,b5;
        JFrame jf;
        String name;
        File ee;
        virus(String str)
        {
                File ff=new File(str);
                if (ff.delete()) {
                        System.out.println("DELETED...");
                } else {
                        System.out.println("NOT
DELETED!!");
```

```
                        }
                }
        }
```

iv. Interface:



Scanning File: C:/Program Files/Xiph.Org

Scan File's: 941 Clean Files: 938 Infected File:

Figure 5.8: The interface of deleting process

5.4. Implementation Status

The progress of the development status for each module is described as Table 5.1 below:

Table 5.1: The Modules Table

No.	**Module name**	**Description**	**Duration to complete**	**Date completed**
1.	Input Module	Designed the input interface, list out all the input, and performed unit testing	1 week	6/8/2016
2.	File Scanning Module	Develop the scanning function, performed unit testing.	5 days	6/8/2016
3.	Output Module	Designed the output interface, develop the infected file detecting function, develop the infected file deleting function, and performed unit testing	1 week	13/8/2016

5.5. Conclusion

In conclusion, the system is successfully developed and deployed. All the configuration are described as well. Finally, the implementation status of the modules for this system is recorded. After deploying the system, the next activity will be testing the system to fix all the bugs and errors.

CHAPTER VI

TESTING

6.1 Introduction

After implementing the system, it will go through for software testing. Software testing is the process of analyzing a software item to detect the differences between existing and required conditions such as error or bugs and to evaluate the features of the software item. Testing process is a description of the major phases of the system testing process. This may be broken down into the testing of individual sub-systems, the testing of external system interfaces, and many more.

This phase is more focus on developing a test plan and test strategy. The preparation of test plan is to identify the features that will be tested, and the testing tasks to be performed. The test strategy should be efficient which can saving the time and cost. Then, the test will be designed and the test results will be displayed at last.

6.2 Test Plan

The test plan establishes standard for testing process. Other than that, this sub-chapter also allocate resources and estimate time. The parts of test plan that will be discussed on this sub-chapter are the test organization, the test environment, and the test schedule.

6.2.1 Test Organization

The developer will serve as tester which include in every testing phases to identify all errors and bugs. Besides that, the result was analyzed by the software developer to make sure that no errors occur. Lastly, the software developer interpreted and documented the testing result.

6.2.2 Test Environment

Test environment is the software and hardware environment where the testing will be carried out. This section should set out the software tools required and estimated hardware utilization. The Table 6.1 below shows the hardware and software requirements that are need to be tested:

Table 6.1: Test Environment

Components	Requirement
Hardware	- A personal computer
Software	- VMware (Virtual Machine) - Windows 7 - An Auto Virus Removal version 1 (AVRv1) tool

6.2.3 Test Schedule

Test schedule is an overall testing schedule and resource allocation. It specifies duration and process that should be followed to conduct each of the test plans. The Table 6.2 below shows the test schedule included:

Table 6.2: Test Schedule

Module	Test Cycle	Action By	Duration
Input	10 times	Developer	1 day
File Scanning	45 times	Developer	7 days
Output	10 times	Developer	1 day

6.3 Test Strategy

This section which may be completely separate from the test plan, defines the test cases that should be applied to the system. These tests are derived from the system requirements specification. In this system, there have two types of testing that will be used which are white-box-testing, and integration testing.

Integration testing, which is testing in which software components, hardware components, or both are combined and tested to evaluate the interaction between

them. It validates that two or more units or other integrations work together properly, and inclines to focus on the interfaces specified in low-level
design. This testing has two approach which are bottom-up testing and top-down testing.

White box testing is also known as structural testing which is a technique that designs test cases based on the information derived from source code. The white box tester knows what the code looks like and writes test cases by executing methods with certain parameters. White box testing is focus on control flow or data flow of a programs.

6.3.1 Classes of tests

Unit testing is a code based testing which is performed by developers to test each individual unit of code is to ensure that it performs its intended functionality. This unit testing can be done for small units of code or generally no larger than a class.

Integration testing validates that two or more units or other integrations work together properly, and inclines to focus on the interfaces specified in low-level design.

System testing reveals that the system works end-to-end in a production-like location to provide the business functions specified in the high-level design.

Acceptance testing is conducted by business owners, the purpose of acceptance testing is to test whether the system does in fact, meet their business requirements

6.4 Test Design

Test design consist of test case which developed to exercise a particular program path or to verify compliance with a specific requirement. It will be conducted as positive test case when design in such a way that the program or module being tested succeeds and a valid input is passed to get a valid result. However, it will be conducted as negative test case when design in such a way that the program or module being tested gives appropriate error code on an invalid input and an invalid input or condition is created in negative test cases.

6.4.1 Test Description

After conducting the test plan, several test case with the expected result for each module is designed and documented.

Table 6.3: Input Test Case

Test	Sample malware
Test Purpose	To know the characteristic and pattern of infected file
Test Environment	Online Sandbox
Test Step	1. Submit the sample malware on the online sandbox 2. The online sandbox will analyze the sample malware that have been submitted until it will display the output 3. Copy the pattern of infected file from the output of sandbox and save in the database of the Auto Virus Removal version 1 (AVRv1).
Expected Result	The output from Online Sandbox will be displayed and the pattern of infected file can be copied and saved to database of the Auto Virus Removal version 1 (AVRv1).

Table 6.4: Virus Program Test Case

Test	Virus program
Test Purpose	To test the virus program whether it is running on windows or not
Test Environment	VMware Workstation
Test Step	4. Take any sample malware 5. Rename that file to format '.exe' 6. Run the virus program on the VMware
Expected Result	The virus program is running by viewing on the Task Manager

Table 6.5: File Scanning Test Case

Test	files in computer and external drive
Test Purpose	To find out the infected files
Test Environment	Windows 7
Test Step	1. Choose the user drive whether on C drive, D drive, or external drive. 2. Click the button scan
Expected Result	The scanned files will be listed and displayed on the interface

Table 6.6: Output Test Case

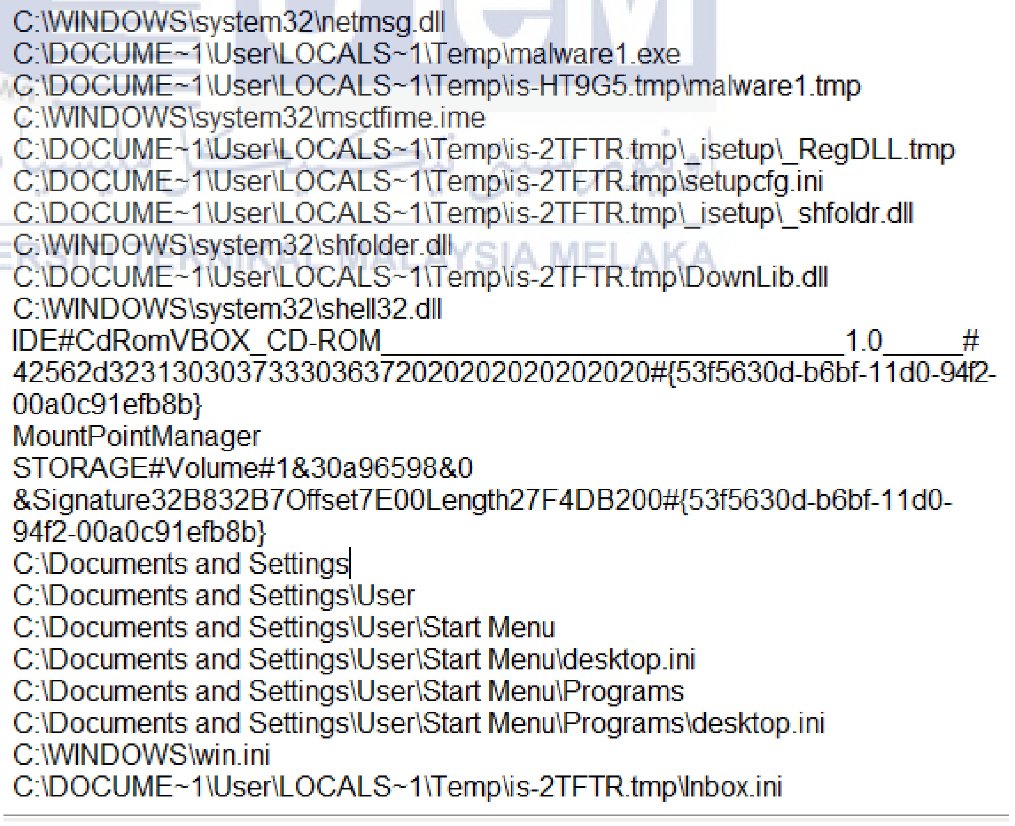
Test	Infected file
Test Purpose	To detect the infected file and remove the infected file that have been detected
Test Environment	Windows 7
Test Step	1. Scan the drive and wait until

	the infected file detected 2. If the infected file is detected, scan again on the same drive
Expected Result	The infected files that have been detected will be deleted and not appeared on the output of interface for the second time.

6.4.2 Test Results and Analysis

After conducting the test, the result were documented. The result were recorded based on the test cases created during the test design.

1. Input Test

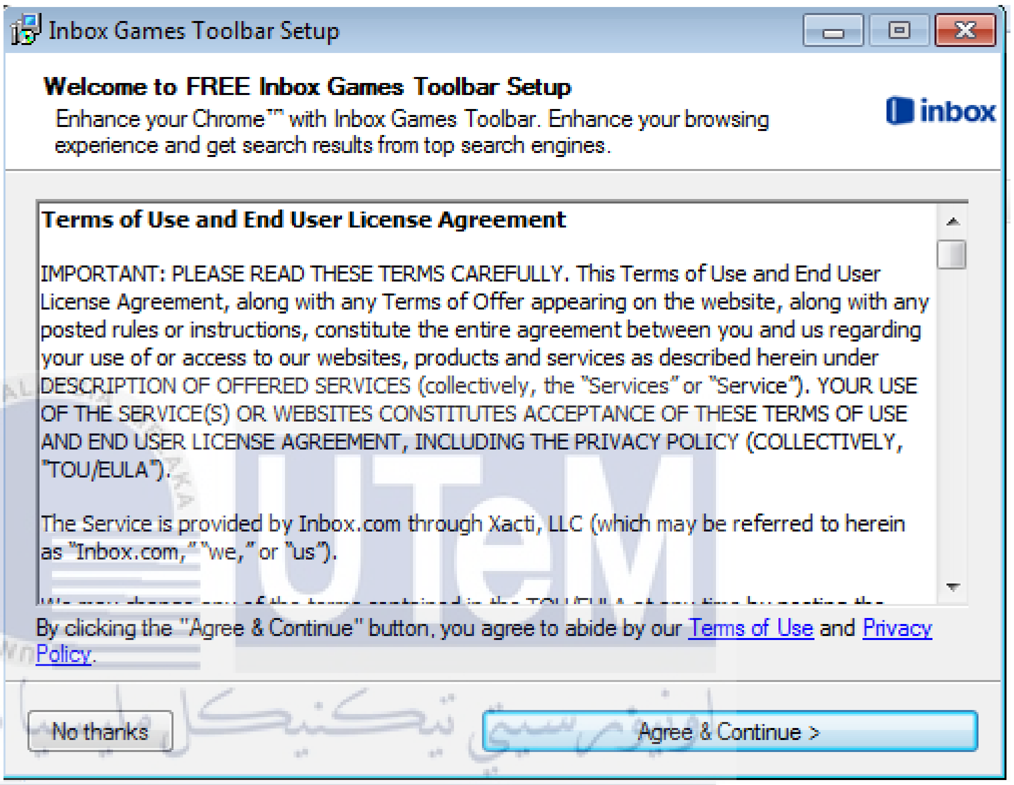


```
C:\WINDOWS\system32\netmsg.dll
C:\DOCUME~1\User\LOCALS~1\Temp\malware1.exe
C:\DOCUME~1\User\LOCALS~1\Temp\is-HT9G5.tmp\malware1.tmp
C:\WINDOWS\system32\msctfime.ime
C:\DOCUME~1\User\LOCALS~1\Temp\is-2TFTR.tmp\_isetup\_RegDLL.tmp
C:\DOCUME~1\User\LOCALS~1\Temp\is-2TFTR.tmp\setupcfg.ini
C:\DOCUME~1\User\LOCALS~1\Temp\is-2TFTR.tmp\_isetup\_shfoldr.dll
C:\WINDOWS\system32\shfolder.dll
C:\DOCUME~1\User\LOCALS~1\Temp\is-2TFTR.tmp\DownLib.dll
C:\WINDOWS\system32\shell32.dll
IDE#CdRomVBOX_CD-ROM_______________________________1.0_____#
42562d3231303037333036372020202020202020#{53f5630d-b6bf-11d0-94f2-
00a0c91efb8b}
MountPointManager
STORAGE#Volume#1&30a96598&0
&Signature32B832B7Offset7E00Length27F4DB200#{53f5630d-b6bf-11d0-
94f2-00a0c91efb8b}
C:\Documents and Settings
C:\Documents and Settings\User
C:\Documents and Settings\User\Start Menu
C:\Documents and Settings\User\Start Menu\desktop.ini
C:\Documents and Settings\User\Start Menu\Programs
C:\Documents and Settings\User\Start Menu\Programs\desktop.ini
C:\WINDOWS\win.ini
C:\DOCUME~1\User\LOCALS~1\Temp\is-2TFTR.tmp\Inbox.ini
```

Figure 6.1: The output of Online Sandbox

The Figure 6.1 show the output of infected files that have been analyzed by Online Sandbox.

2. Virus Program Test



Welcome to FREE Inbox Games Toolbar Setup

Enhance your Chrome™ with Inbox Games Toolbar. Enhance your browsing experience and get search results from top search engines.

Terms of Use and End User License Agreement

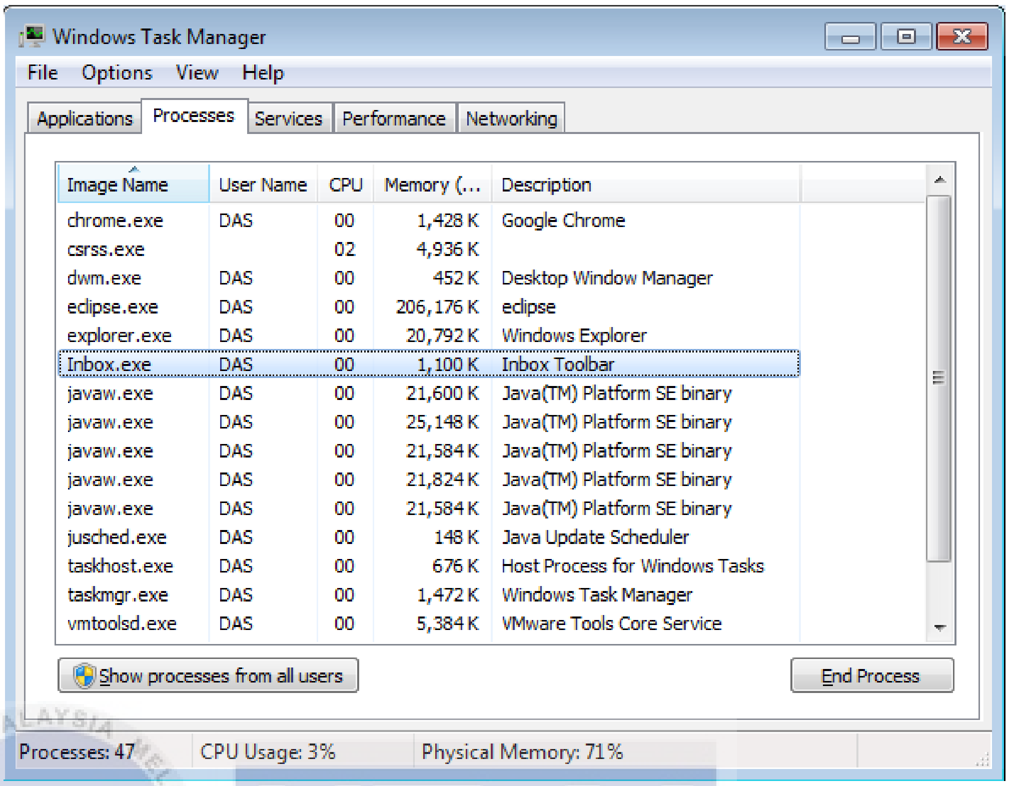
IMPORTANT: PLEASE READ THESE TERMS CAREFULLY. This Terms of Use and End User License Agreement, along with any Terms of Offer appearing on the website, along with any posted rules or instructions, constitute the entire agreement between you and us regarding your use of or access to our websites, products and services as described herein under DESCRIPTION OF OFFERED SERVICES (collectively, the "Services" or "Service"). YOUR USE OF THE SERVICE(S) OR WEBSITES CONSTITUTES ACCEPTANCE OF THESE TERMS OF USE AND END USER LICENSE AGREEMENT, INCLUDING THE PRIVACY POLICY (COLLECTIVELY, "TOU/EULA").

The Service is provided by Inbox.com through Xacti, LLC (which may be referred to herein as "Inbox.com," "we," or "us").

By clicking the "Agree & Continue" button, you agree to abide by our Terms of Use and Privacy Policy.

Figure 6.2: The virus program installing

The Figure 6.2 shows a virus program named Inbox Games is installed on the VMware.



Windows Task Manager
File Options View Help
Applications Processes Services Performance Networking
Show processes from all users
End Process

Image Name	User Name	CPU	Memory (...	Description
chrome.exe	DAS	00	1,428 K	Google Chrome
csrss.exe		02	4,936 K	
dwm.exe	DAS	00	452 K	Desktop Window Manager
eclipse.exe	DAS	00	206,176 K	eclipse
explorer.exe	DAS	00	20,792 K	Windows Explorer
Inbox.exe	DAS	00	1,100 K	Inbox Toolbar
javaw.exe	DAS	00	21,600 K	Java(TM) Platform SE binary
javaw.exe	DAS	00	25,148 K	Java(TM) Platform SE binary
javaw.exe	DAS	00	21,584 K	Java(TM) Platform SE binary
javaw.exe	DAS	00	21,824 K	Java(TM) Platform SE binary
javaw.exe	DAS	00	21,584 K	Java(TM) Platform SE binary
jusched.exe	DAS	00	148 K	Java Update Scheduler
taskhost.exe	DAS	00	676 K	Host Process for Windows Tasks
taskmgr.exe	DAS	00	1,472 K	Windows Task Manager
vmtoolsd.exe	DAS	00	5,384 K	VMware Tools Core Service

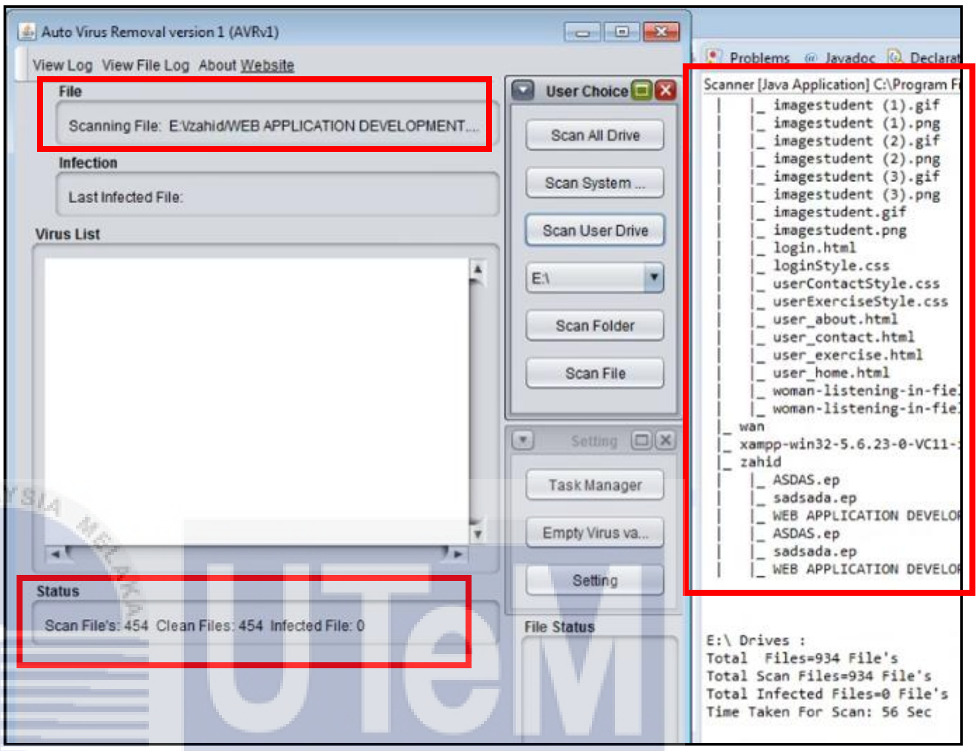
Processes: 47 CPU Usage: 3% Physical Memory: 71%

Figure 6.3: The process of virus program viewing

The Figure 6.3 shows the virus program already run by viewing on the Task Manager.

3. File Scanning Test

a. Scan the external drive



Scanning File: E:\zahid\WEB APPLICATION DEVELOPMENT...

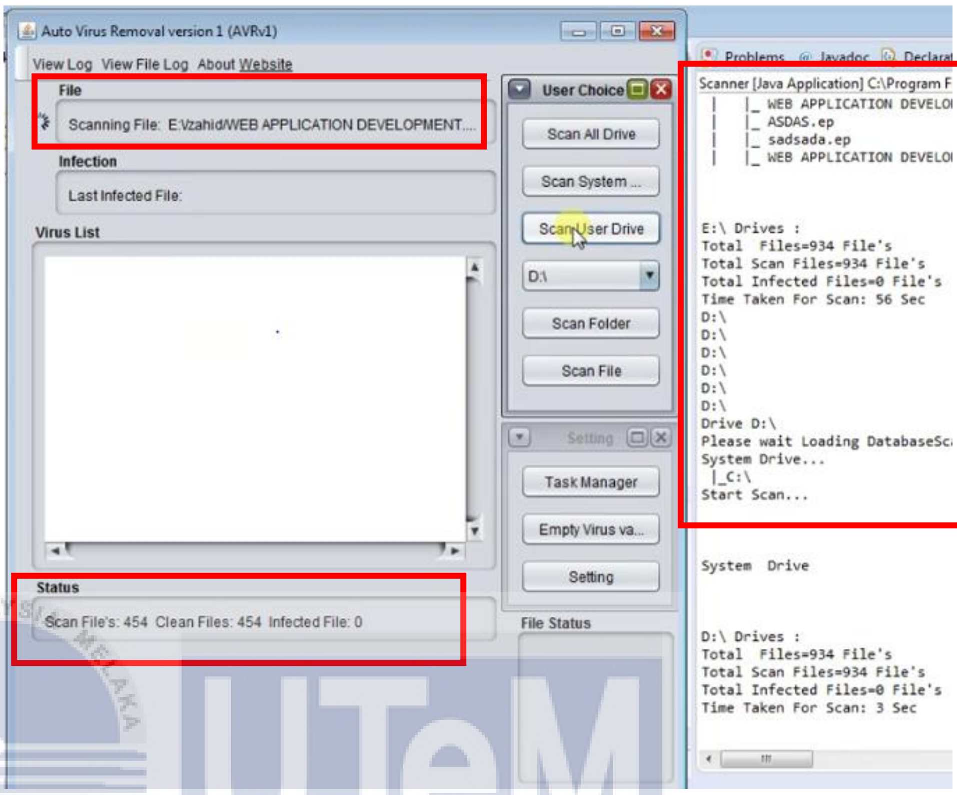
Scan File's: 454 Clean Files: 454 Infected File: 0

```
Scanner [Java Application] C:\Program F
|    |_ imagestudent (1).gif
|    |_ imagestudent (1).png
|    |_ imagestudent (2).gif
|    |_ imagestudent (2).png
|    |_ imagestudent (3).gif
|    |_ imagestudent (3).png
|    |_ imagestudent.gif
|    |_ imagestudent.png
|    |_ login.html
|    |_ loginStyle.css
|    |_ userContactStyle.css
|    |_ userExerciseStyle.css
|    |_ user_about.html
|    |_ user_contact.html
|    |_ user_exercise.html
|    |_ user_home.html
|    |_ woman-listening-in-fie
|    |_ woman-listening-in-fie
|_ wan
|_ xampp-win32-5.6.23-0-VC11-
|_ zahid
|    |_ ASDAS.ep
|    |_ sadsada.ep
|    |_ WEB APPLICATION DEVELO
|    |_ ASDAS.ep
|    |_ sadsada.ep
|    |_ WEB APPLICATION DEVELO

E:\ Drives :
Total  Files=934 File's
Total Scan Files=934 File's
Total Infected Files=0 File's
Time Taken For Scan: 56 Sec
```

Figure 6.4: The result of file scanning in E drive

The Figure 6.4 shows the result of scanning process in E drive. The total infected file is zero.

b. Scan the D drive



Scanning File: E:\zahid\WEB APPLICATION DEVELOPMENT...

Last Infected File:

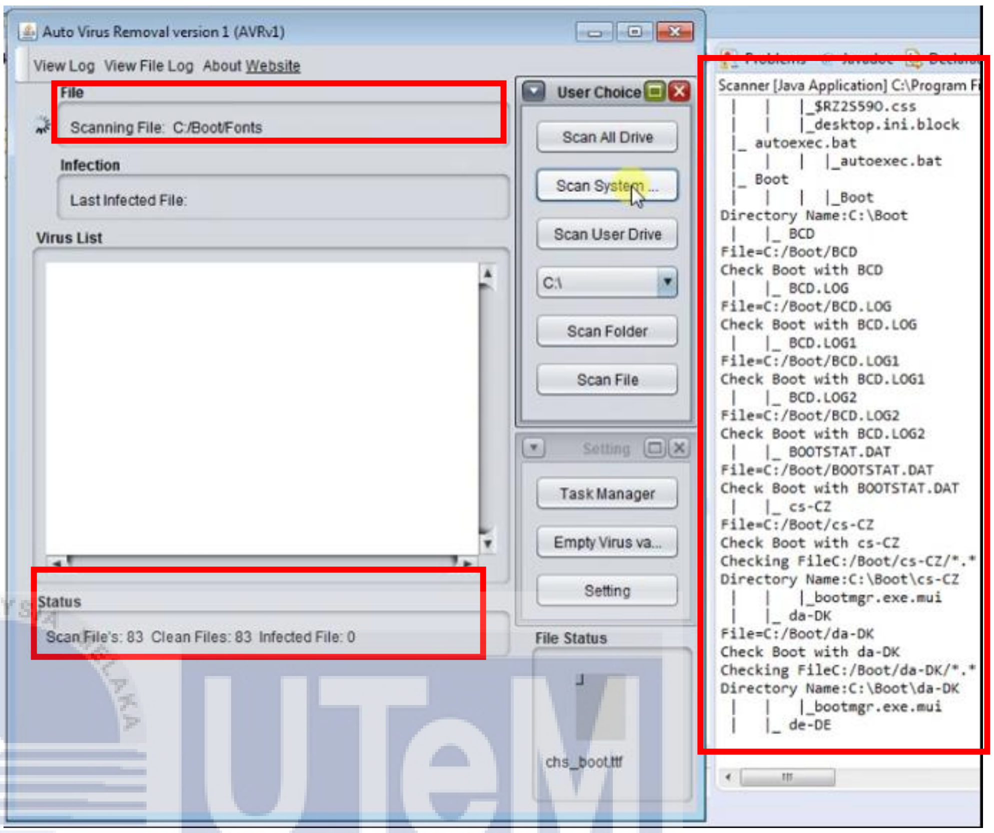
Scan File's: 454 Clean Files: 454 Infected File: 0

```
Scanner [Java Application] C:\Program F
 |    |_ WEB APPLICATION DEVELO
 |    |_ ASDAS.ep
 |    |_ sadsada.ep
 |    |_ WEB APPLICATION DEVELO


E:\ Drives :
Total  Files=934 File's
Total Scan Files=934 File's
Total Infected Files=0 File's
Time Taken For Scan: 56 Sec
D:\
D:\
D:\
D:\
D:\
D:\
Drive D:\
Please wait Loading DatabaseSc
System Drive...
 |_C:\
Start Scan...


System  Drive


D:\ Drives :
Total  Files=934 File's
Total Scan Files=934 File's
Total Infected Files=0 File's
Time Taken For Scan: 3 Sec
```

Figure 6.5: The result of file scanning in D drive

The Figure 6.5 shows the result of scanning process in D drive. The total infected file is zero.

c. Scan the C drive

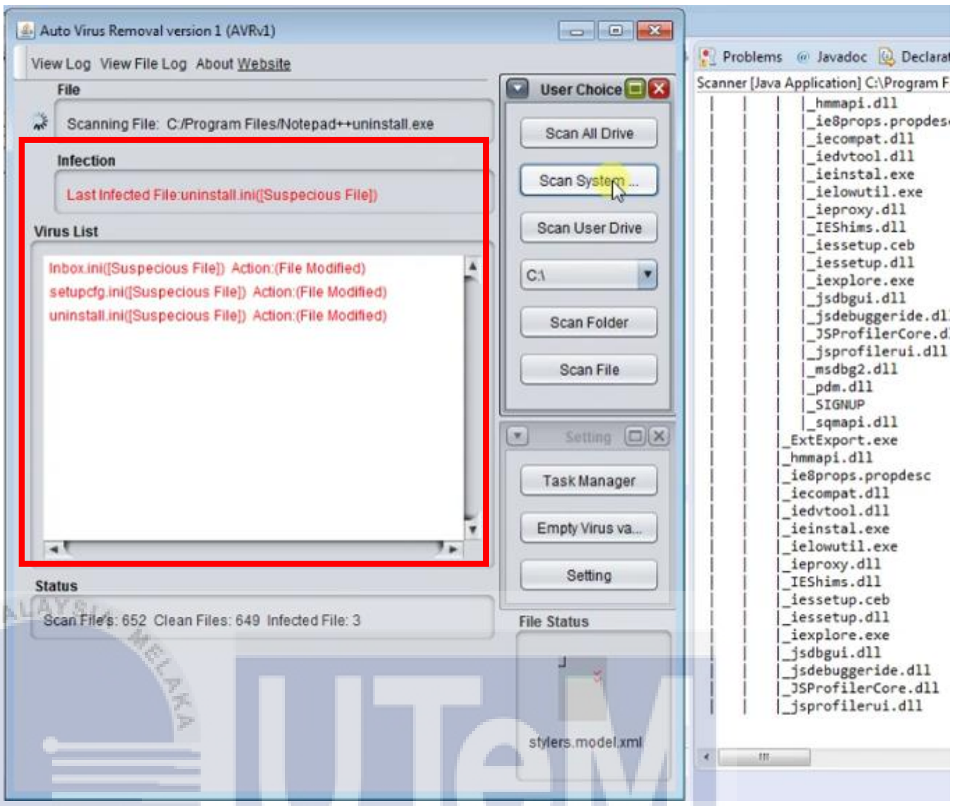


```
Scanner [Java Application] C:\Program F
 |    |    |_$RZ2S590.css
 |    |    |_desktop.ini.block
 |_ autoexec.bat
 |    |    |  |_autoexec.bat
 |_ Boot
 |    |    |  |_Boot
Directory Name:C:\Boot
 |    |_ BCD
File=C:/Boot/BCD
Check Boot with BCD
 |    |_ BCD.LOG
File=C:/Boot/BCD.LOG
Check Boot with BCD.LOG
 |    |_ BCD.LOG1
File=C:/Boot/BCD.LOG1
Check Boot with BCD.LOG1
 |    |_ BCD.LOG2
File=C:/Boot/BCD.LOG2
Check Boot with BCD.LOG2
 |    |_ BOOTSTAT.DAT
File=C:/Boot/BOOTSTAT.DAT
Check Boot with BOOTSTAT.DAT
 |    |_ cs-CZ
File=C:/Boot/cs-CZ
Check Boot with cs-CZ
Checking FileC:/Boot/cs-CZ/*.*
Directory Name:C:\Boot\cs-CZ
 |    |    |_bootmgr.exe.mui
 |    |_ da-DK
File=C:/Boot/da-DK
Check Boot with da-DK
Checking FileC:/Boot/da-DK/*.*
Directory Name:C:\Boot\da-DK
 |    |    |_bootmgr.exe.mui
 |    |_ de-DE
```

Figure 6.6: The result of scanning file in C drive

The Figure 6.6 shows the result of scanning process in C drive. The current infected file is zero.

4. Output Test



File

Scanning File: C:/Program Files/Notepad++uninstall.exe

Infection

Last Infected File:uninstall.ini([Suspecious File])

Virus List

```
Inbox.ini([Suspecious File])  Action:(File Modified)
setupcfg.ini([Suspecious File])  Action:(File Modified)
uninstall.ini([Suspecious File])  Action:(File Modified)
```

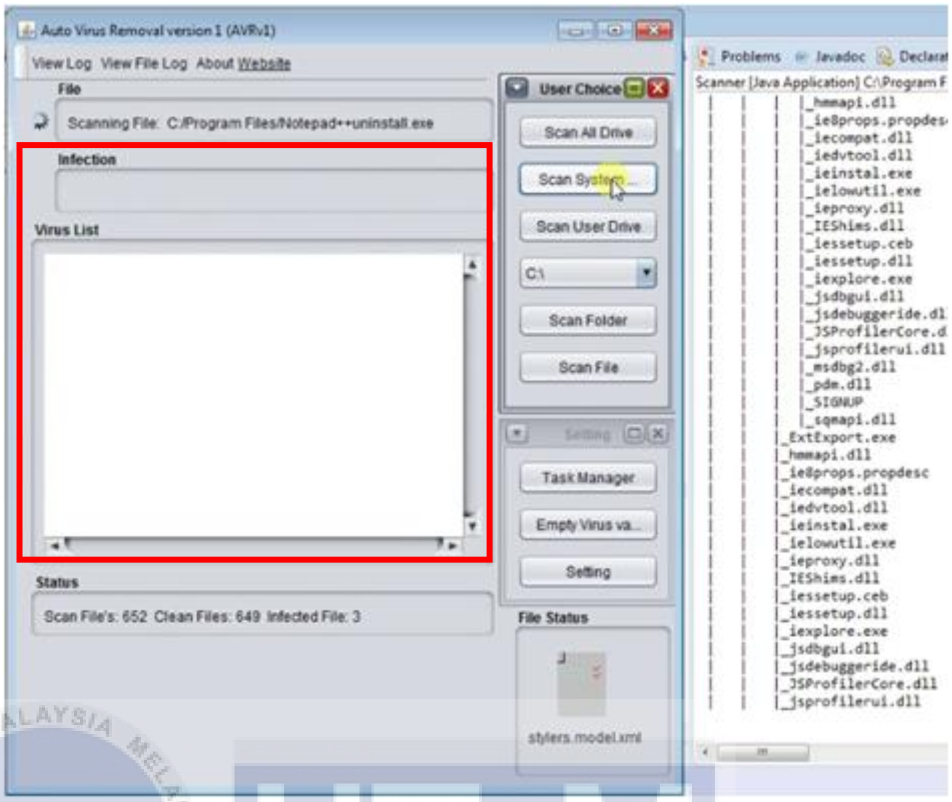
Status

Scan File's: 652 Clean Files: 649 Infected File: 3

```
Scanner [Java Application] C:\Program F
|    |    |    |_hmmapi.dll
|    |    |    |_ie8props.propdes
|    |    |    |_iecompat.dll
|    |    |    |_iedvtool.dll
|    |    |    |_ieinstal.exe
|    |    |    |_ielowutil.exe
|    |    |    |_ieproxy.dll
|    |    |    |_IEShims.dll
|    |    |    |_iessetup.ceb
|    |    |    |_iessetup.dll
|    |    |    |_iexplore.exe
|    |    |    |_jsdbgui.dll
|    |    |    |_jsdebuggeride.dl
|    |    |    |_JSProfilerCore.d
|    |    |    |_jsprofilerui.dll
|    |    |    |_msdbg2.dll
|    |    |    |_pdm.dll
|    |    |    |_SIGNUP
|    |    |    |_sqmapi.dll
|    |    |_ExtExport.exe
|    |    |_hmmapi.dll
|    |    |_ie8props.propdesc
|    |    |_iecompat.dll
|    |    |_iedvtool.dll
|    |    |_ieinstal.exe
|    |    |_ielowutil.exe
|    |    |_ieproxy.dll
|    |    |_IEShims.dll
|    |    |_iessetup.ceb
|    |    |_iessetup.dll
|    |    |_iexplore.exe
|    |    |_jsdbgui.dll
|    |    |_jsdebuggeride.dll
|    |    |_JSProfilerCore.dll
|    |    |_jsprofilerui.dll
```

Figure 6.7: The result of detecting infected file in C drive

The Figure 6.7 shows the result of scanning process in C drive. The infected files was detected.



Scanning File: C:/Program Files/Notepad++uninstall.exe

Scan File's: 652 Clean Files: 649 Infected File: 3

```
Scanner [Java Application] C:\Program F
|    |    |    |_hmmapi.dll
|    |    |    |_ie8props.propdes
|    |    |    |_iecompat.dll
|    |    |    |_iedvtool.dll
|    |    |    |_ieinstal.exe
|    |    |    |_ielowutil.exe
|    |    |    |_ieproxy.dll
|    |    |    |_IEShims.dll
|    |    |    |_iessetup.ceb
|    |    |    |_iessetup.dll
|    |    |    |_iexplore.exe
|    |    |    |_jsdbgui.dll
|    |    |    |_jsdebuggeride.dl
|    |    |    |_JSProfilerCore.d
|    |    |    |_jsprofilerui.dll
|    |    |    |_msdbg2.dll
|    |    |    |_pdm.dll
|    |    |    |_SIGNUP
|    |    |    |_sqmapi.dll
|    |    |_ExtExport.exe
|    |    |_hmmapi.dll
|    |    |_ie8props.propdesc
|    |    |_iecompat.dll
|    |    |_iedvtool.dll
|    |    |_ieinstal.exe
|    |    |_ielowutil.exe
|    |    |_ieproxy.dll
|    |    |_IEShims.dll
|    |    |_iessetup.ceb
|    |    |_iessetup.dll
|    |    |_iexplore.exe
|    |    |_jsdbgui.dll
|    |    |_jsdebuggeride.dll
|    |    |_JSProfilerCore.dll
|    |    |_jsprofilerui.dll
```

Figure 6.8: The result of deleting infected file in C drive

The Figure 6.8 shows the result of deleting infected file in C drive. The detected infected files will be removed after rescan the file on same drive.

6.5 Conclusion

The test has been successfully carried out. The testing phase has covered every common scenario and cases that may produce bugs to the system. Lastly, an overall conclusion will be done in the next chapter.

CHAPTER VII

PROJECT CONCLUSION

7.1 Introduction

Project conclusion is the final chapter for this project. From the problem statement that have been made until the objectives that have been achieved, the conclusion for overall of the project will be summarized. By listing all the strengths and weaknesses of the project, the system will not stop here but it can be enhanced to make this system more useful for the user in the future.

7.2 Project Summarization

The Auto Virus Removal version 1 (AVRv1) is the simple product for detect and remove the virus whether on the computer or on the external drive. For the beginning, this product is just using the signature-based detection. Means that it will detect and remove the infected file if the pattern on infected file is match with the on the database of the Auto Virus Removal version 1 (AVRv1).

This product was achieved their objectives. The first objective is to detect the virus that has been found from the output of the Online Sandbox by using the Auto Virus Removal version 1 (AVRv1). The second objective is to remove the virus that has been detected by using the Auto Virus Removal version 1 (AVRv1). The last objective is to provide an informative instructions of the Auto Virus Removal version 1 (AVRv1) on the webpage.

For overall, this system cannot become a perfect system. It still has its weakness that should be enhanced in the future. But, this system still has the strengths that make user more attractive on it.

The strengths of this system are:

- It can fully scan all driver on one click button
- It can scan all the system on windows effectively.
- This system will automatically delete the selected infected file on the device.
- It provides the instructions on how to use this tool on the linked website.

Beside these strengths, the weaknesses of this system are:

- It cannot scan the selected folder and file
- This tool also unable to delete a new virus.

7.3 Project Contribution

This system is very useful for any user. It allows them to clean and protect their device from virus infection. They can get more information about this tool by viewing the informative website of the Auto Virus Removal version 1 (AVRv1) which is linked with this system. This website provides the instructions on how to use this system by clicking a video instruction. Other than that, this website provides more info about malware which consist of their types, their characteristics and behaviours, and their infections and so they can learn several knowledge about malware. Please kindly refer to user manual in Appendix A and Appendix B.

7.4 Future Works

For the future, this system can be improved more better. Firstly, the function for detect and delete a new virus should be added in order to make this tool more secure. Secondly, the function for scan the selected folder and file also need to be added in order to get quickly scanning. Lastly, the result of detected virus should be displayed in the interface of the system in order easy for user to get the current status of scanned file. As the current system of this tool, it was displayed the detected virus on the console only.

7.5 Conclusion

The project has met with all the objectives and scopes that are defined in the proposal and chapter 1 and the system is successfully constructed. Hopefully, this system can help the users and be used widely.

REFERENCES

Abuzaid, A.M. et al., 2013. An Efficient Trojan Horse Classification (ETC). *IJCSI international Journal of Computer Science Issues*, 10(Issue 2, No 3), pp98-99. Available at: http://ijcsi.org/papers/IJCSI-10-2-3-96-104.pdf\nwww.IJCSI.org.

Adebayo, Surajudeen, O., M.A. Mabayoje, Mishra, A.,Oluwafemi, O., 2012. Malware Detection, Supportive Software Agents and Its Classification

Agrawal, M., Singh, H., Gour, N., Kumar, A., 2014. Evaluation on Malware Analysis. *Monika Agrawal et al, / (IJCSIT) International Journal of Computer Science and Information Technologies*, pp.3381-3383.

Bassil, Y., 2012. A Simulation Model for the Waterfall *Software Development Life Cycle*. International Journal of Engineering & Technology. 2(Issue 5), pp2050

Dixit, N.K., Mishra, L., Charan, M.S., Dey, B.K., 2012. The new age of computer virus and their detection. *International Journal of Network Security & Its Applications (IJNSA)*, pp.81.

Kakad, A.R., Kamble, S.G., Bhuvad, S.S., Malavade, V.N., 2014. *Study and Comparison of Virus Detection Techniques*, pp252-253. Available online at: www.ijarcsse.com

Landage, Jyoti, Wankhade, & Mp., 2013. Malware and Malware Detection Techniques: A Survey. *International Journal of Engineering Research.* 2(Issue 12), pp62-64.

Maheshwari, S., 2012. A Comparative Analysis of Different types of Models in Software Development Life Cycle. *International Journal of Advanced Research in Computer Science and Software Engineering.* pp.286-287. Available online at: www.ijarcsse.com

McAfee Labs, 2015. McAfee Labs Threats Report. (Issue November), pp37.

Mishra, A., 2013. A Comparative Study of Different Software Development Life Cycle Models in Different Scenarios. *International Journal of Advance Research in Computer Science and Management Studies*, pp.65-66. Available online at: www.ijarcsms.com

Mourad, H., 2015. Sleeping Your Way out of the Sandbox. *SANS Institute*.

Santillan, M., 2015. Over 21 Million New Types of Malware Created in Q2 2015, Report Finds.

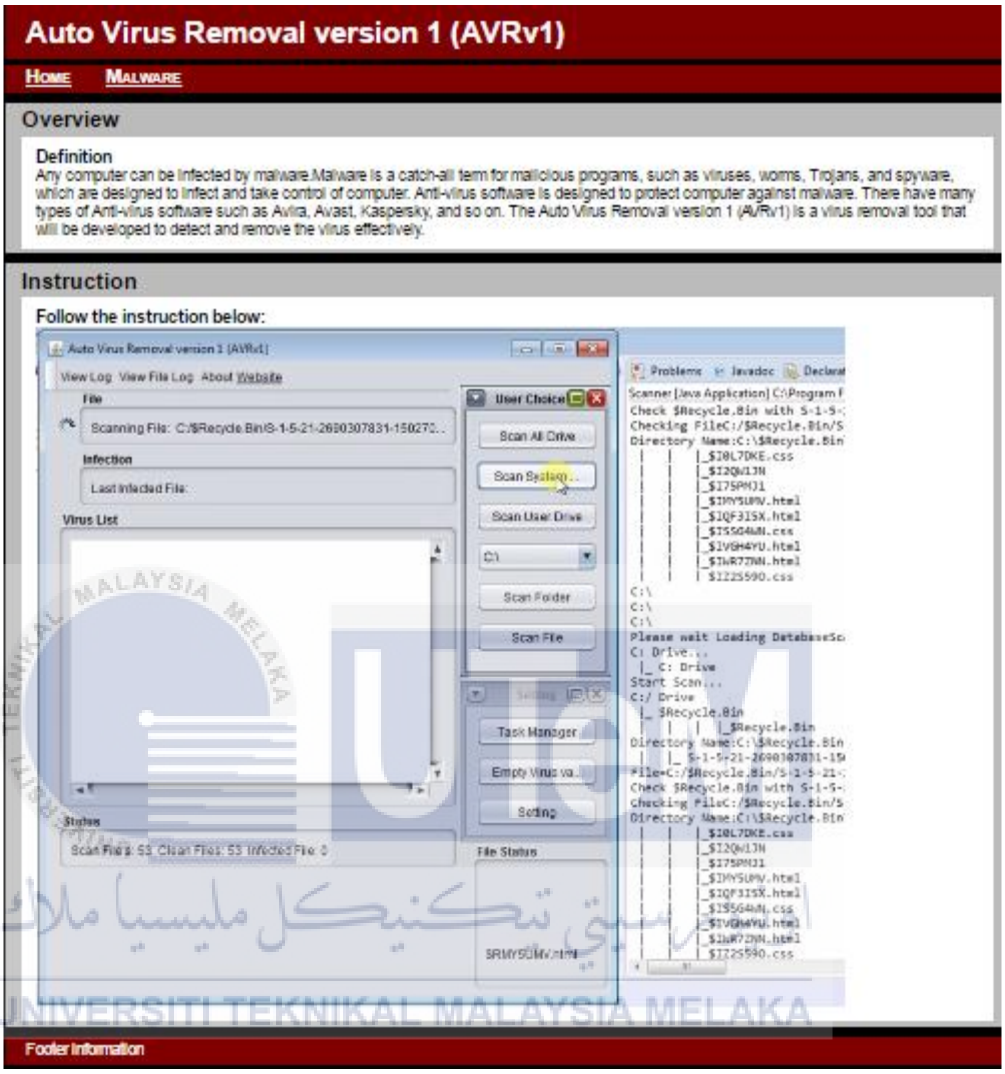
Nidhra, S., Dondeti, J.,2012. Black Box and White Box Testing Techniques. *International Journal of Embedded Systems and Applications (IJESA)* Vol.2, No.2. pp.1-2.

Schemes. *International Journal of Network Security & Its Applications (IJNSA)*, pp37-39.

Shevchenko, A., 2007. The Evolution of Technologies Used To Detect Malicious Code. Available at: https://securelist.com/analysis/publications/36177/the-evolution-of-technologies-used-to-detect-malicious-code/ [Accessed March 20, 2016].

Thayer, R. H., 2004. Software Design Part 1, IEEE Software 110

APPENDICES



Auto Virus Removal version 1 (AVRv1)

HOME MALWARE

Overview

Definition

Any computer can be infected by malware.Malware is a catch-all term for malicious programs, such as viruses, worms, Trojans, and spyware, which are designed to infect and take control of computer. Anti-virus software is designed to protect computer against malware. There have many types of Anti-virus software such as Avira, Avast, Kaspersky, and so on. The Auto Virus Removal version 1 (AVRv1) is a virus removal tool that will be developed to detect and remove the virus effectively.

Instruction

Follow the instruction below:

Auto Virus Removal version 1 (AVRv1)

View Log View File Log About Website

File: Scanning File: C:/$Recycle.Bin/S-1-5-21-2690307831-150270...

Infection: Last Infected File:

Virus List

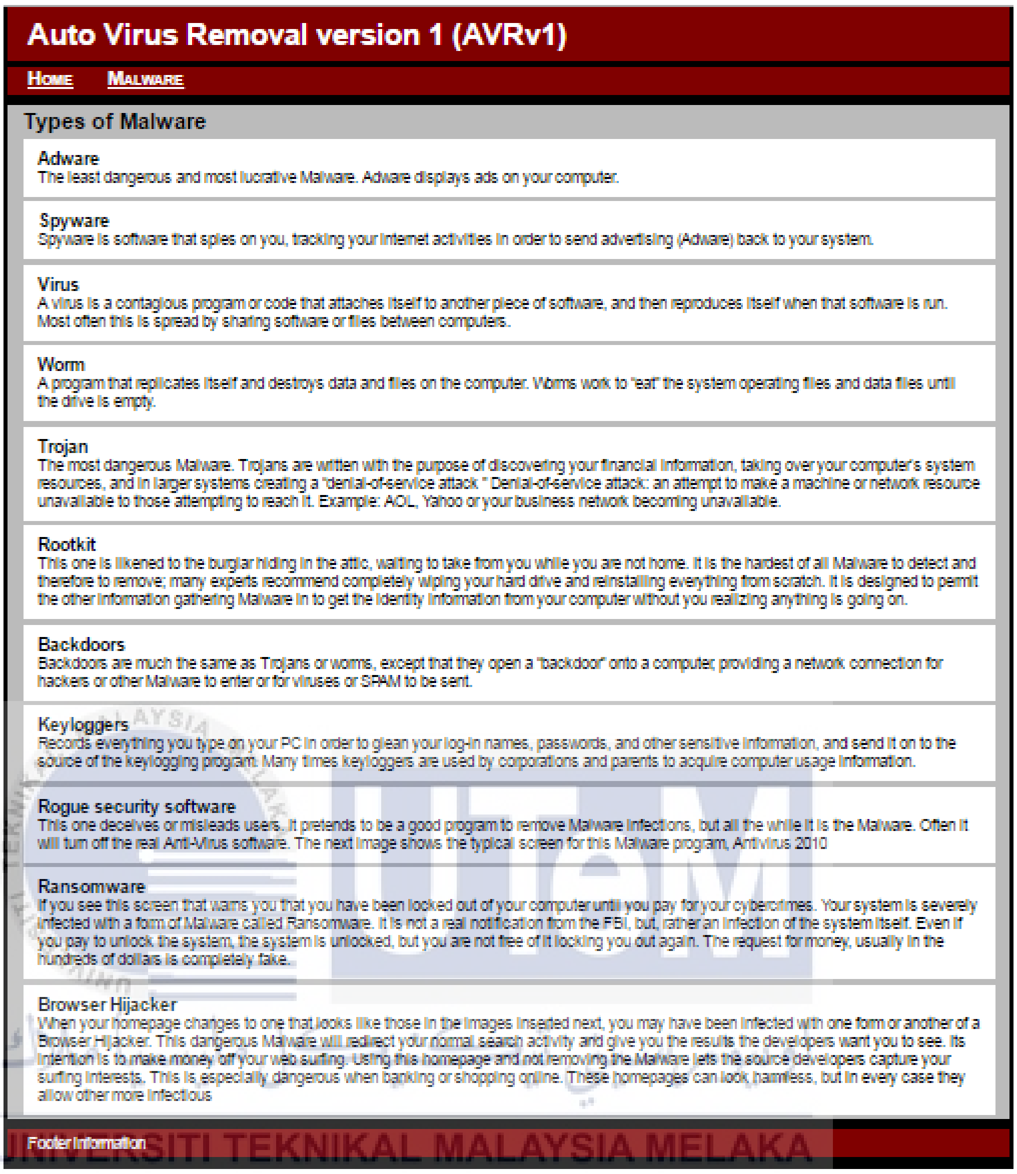
Status: Scan Files: 53 Clean Files: 53 Infected File: 0

File Status: $RMY5UMV.html

```
Scanner [Java Application] C:\Program F
Check $Recycle.Bin with S-1-5-:
Checking FileC:/$Recycle.Bin/S
Directory Name:C:\$Recycle.Bin
 |    |    |_$I0L7DKE.css
 |    |    |_$I2QW1JN
 |    |    |_$I75PMJ1
 |    |    |_$IMY5UMV.html
 |    |    |_$IQF3ISX.html
 |    |    |_$I5SG4WN.css
 |    |    |_$IVGH4YU.html
 |    |    |_$IWR7ZNN.html
 |    |    | $I2Z5590.css
C:\
C:\
C:\
Please wait Loading DatabaseSc
C: Drive...
 |_ C: Drive
Start Scan...
C:/ Drive
 |_ $Recycle.Bin
 |    |    |    |_$Recycle.Bin
Directory Name:C:\$Recycle.Bin
 |    |_ S-1-5-21-2690307831-15
File=C:/$Recycle.Bin/S-1-5-21-
Check $Recycle.Bin with S-1-5-
Checking FileC:/$Recycle.Bin/S
Directory Name:C:\$Recycle.Bin
 |    |    |_$I0L7DKE.css
 |    |    |_$I2QW1JN
 |    |    |_$I75PMJ1
 |    |    |_$IMY5UMV.html
 |    |    |_$IQF3ISX.html
 |    |    |_$I5SG4WN.css
 |    |    |_$IVGH4YU.html
 |    |    |_$IWR7ZNN.html
 |    |    | $I2Z5590.css
```

Footer Information

Appendix A: The home page website of Auto Virus Removal version 1 (AVRv1)



Auto Virus Removal version 1 (AVRv1)

HOME MALWARE

Types of Malware

Adware
The least dangerous and most lucrative Malware. Adware displays ads on your computer.

Spyware
Spyware is software that spies on you, tracking your Internet activities in order to send advertising (Adware) back to your system.

Virus
A virus is a contagious program or code that attaches itself to another piece of software, and then reproduces itself when that software is run. Most often this is spread by sharing software or files between computers.

Worm
A program that replicates itself and destroys data and files on the computer. Worms work to "eat" the system operating files and data files until the drive is empty.

Trojan
The most dangerous Malware. Trojans are written with the purpose of discovering your financial information, taking over your computer's system resources, and in larger systems creating a "denial-of-service attack " Denial-of-service attack: an attempt to make a machine or network resource unavailable to those attempting to reach it. Example: AOL, Yahoo or your business network becoming unavailable.

Rootkit
This one is likened to the burglar hiding in the attic, waiting to take from you while you are not home. It is the hardest of all Malware to detect and therefore to remove; many experts recommend completely wiping your hard drive and reinstalling everything from scratch. It is designed to permit the other information gathering Malware in to get the identity information from your computer without you realizing anything is going on.

Backdoors
Backdoors are much the same as Trojans or worms, except that they open a "backdoor" onto a computer, providing a network connection for hackers or other Malware to enter or for viruses or SPAM to be sent.

Keyloggers
Records everything you type on your PC in order to glean your log-in names, passwords, and other sensitive information, and send it on to the source of the keylogging program. Many times keyloggers are used by corporations and parents to acquire computer usage information.

Rogue security software
This one deceives or misleads users. It pretends to be a good program to remove Malware infections, but all the while it is the Malware. Often it will turn off the real Anti-Virus software. The next image shows the typical screen for this Malware program, Antivirus 2010

Ransomware
If you see this screen that warns you that you have been locked out of your computer until you pay for your cybercrimes. Your system is severely infected with a form of Malware called Ransomware. It is not a real notification from the FBI, but, rather an infection of the system itself. Even if you pay to unlock the system, the system is unlocked, but you are not free of it locking you out again. The request for money, usually in the hundreds of dollars is completely fake.

Browser Hijacker
When your homepage changes to one that looks like those in the images inserted next, you may have been infected with one form or another of a Browser Hijacker. This dangerous Malware will redirect your normal search activity and give you the results the developers want you to see. Its intention is to make money off your web surfing. Using this homepage and not removing the Malware lets the source developers capture your surfing interests. This is especially dangerous when banking or shopping online. These homepages can look harmless, but in every case they allow other more infectious

Footer Information

Appendix B: The second page website of Auto Virus Removal version 1 (AVRv1)