

**BACKDOOR ATTACK DETECTION BASED ON STEPPING STONE
DETECTION APPROACH**

KHALID ABDULRAZZAQ ABDULNABI AL-MINSHID



UNIVERSITI UTARA MALAYSIA

2014

Backdoor Attack Detection Based on Stepping Stone Detection Approach

A dissertation submitted to Dean of Research and Postgraduate Studies
Office

in partial Fulfillment of the requirement for the degree

Master of Science (Information Technology)

Universiti Utara Malaysia



By

Khalid Abdulrazzaq Abdalnabi Al-Minshid

Permission to Use

In presenting this dissertation in fulfilment of the requirements for a Master of Science in Information Technology (MSc. IT) from Universiti Utara Malaysia, I agree that the Universiti Library may make it freely available for inspection. I further agree that permission for the copying of this dissertation in any manner, in whole or in part, for scholarly purpose may be granted by my supervisor or, in their absence, by the Dean of Awang Had Salleh Graduate School of Arts and Sciences. It is understood that any copying or publication or use of this dissertation or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to Universiti Utara Malaysia for any scholarly use which may be made of any material from my dissertation.

Requests for permission to copy or to make other use of materials in this dissertation, in whole or in part, should be addressed to:



UUM
Universiti Utara Malaysia

Dean of Awang Had Salleh Graduate School of Arts and Sciences

UUM College of Arts and Sciences

Universiti Utara Malaysia

06010 UUM Sintok

Kedah Darul Aman

Abstract

Network intruders usually use a series of hosts (stepping stones) to conceal the tracks of their intrusion in the network. This type of intrusion can be detected through an approach called Stepping Stone Detection (SSD). In the past years, SSD was confined to the detection of only this type of intrusion. In this dissertation, we consider the use of SSD concepts in the field of backdoor attack detection. The application of SSD in this field results in many advantages. First, the use of SSD makes the backdoor attack detection and the scan process time faster. Second, this technique detects all types of backdoor attack, both known and unknown, even if the backdoor attack is encrypted. Third, this technique reduces the large storage resources used by traditional antivirus tools in detecting backdoor attacks. This study contributes to the field by extending the application of SSD-based techniques, which are usually used in SSD-based environments only, into backdoor attack detection environments. Through an experiment, the accuracy of SSD-based backdoor attack detection is shown as very high.

Keywords: Stepping stone, stepping stone detection, backdoor, hacker, intrusion

Acknowledgement

“In the name of Allah the Most Beneficent and Most Merciful”

All praises and thanks to the Almighty, Allah (SWT), who helps me to finish this dissertation. Allah gives me the opportunity, strength and the ability to complete my study for Master degree after a long time of continuous work. No volume of words is enough to express my gratitude towards my supervisor, Dr. Mohd Nizam Omar, who has been very concerned and gave me many interesting, valuable and sincere feedbacks throughout his supervision. Indeed, I found in his experience the main reference of my research, I greatly benefited from his detailed comments and insights that helped me clarify ideas in “Backdoor Attack Detection Based on Stepping Stone Detection Approach”.

I sincerely thanks to my evaluators, Dr. Shahrudin bin Awang Nor and Dr. Ahmad Suki Bin Che Mohamed Arif, and thanks to Dr. Mohd. Hasbullah bin Omar, Prof. Madya Dr. Faudziah Bt Ahmad, Dr. Nooraini Binti Yusoff, and other committee members, for graciously reviewing this work and giving valuable suggestion and comments on my work. I would also like to say a big thanks to all UUM lecturers and staff members at the School of Computing who were kind enough to give me their precious time and assistance, without which I would not have been able to complete this Master’s dissertation. I am indebted and thankful to all Malaysian people who are very friendly and make us feel that we are not strangers in Malaysia. Last but not least, the words cannot express my gratitude to my family, especially my mother, my dear brothers Salam, Hamid and Wissam, my sisters, my faithful wife, my sons Ahmed and Murtdha and my five daughters, Duha, Saja, Nor, Tbark and Baneen. Words cannot describe their constant love, care, concern, patience, throughout the two years of my study abroad. I’m forever thankful, grateful, and indebted to them. I dedicate the accomplishment of this dissertation to my father, may Allah bless him!, my affectionate mother, and to the twin of my spirit, my wife.

“Thank you UUM”

Khalid Al-Minshid

TABLE OF CONTENTS

PERMISSION TO USE	i
ABSTRACT.....	ii
ACKNOWLEDGEMENT	iii
TABLE OF CONTENTS	iv
CHAPTER ONE INTRODUCTION	1
1.1 Introduction	1
1.2 Research Background	2
1.3 Problem Statement	4
1.4 Research Question.....	5
1.5 Research Objectives.....	5
1.6 Scope.....	6
1.7 Significance of the Research	6
1.8 Summary.....	7
CHAPTER TWO LITERATURE REVIEW.....	8
2.1 Introduction	8
2.2 Terminology	9
2.2.1 Network Security Terminology	9
2.2.2 SSD Terminology	13
2.3 Backdoor Attack.....	15
2.3.1 Types of Backdoors.....	15
2.3.2 Authors and Users of Backdoors	17
2.3.3 Backdoor Detectors	18
2.3.4 Recent Backdoor's Detection Approaches and Related Works	22
2.4 Stepping Stone.....	24
2.4.1 Stepping Stone Chain	24
2.4.2 SSD Approach	24
2.4.3 SSD Evolution and Related Work	26
2.4.3.1 The Past of SSD.....	26

2.4.3.2 Current SSD	27
2.4.3.3 Future SSD	30
2.4.3.4 Emerging Fields for Application of SSD	31
2.4.4 SSD Issues.....	31
2.4.4.1 Interactive and Non interactive Connection	31
2.4.4.2 Positive and Negative False	33
2.4.4.3 Passive and Active Detection.....	33
2.4.4.4 SDD Matching Concepts.....	34
2.4.5 SSD Techniques.....	36
2.4.6 SSD Models	40
2.4.6.1 HSSD Model	41
2.4.6.2 NSSD Model:.....	42
2.5 SSD and Backdoor	45
2.6 Summary.....	47
CHAPTER THREE RESEARCH METHODOLOGY	48
3.1 Introduction	48
3.2 Operational Framework	48
3.3 Research Design.....	50
3.4 Subject and Information Sources	51
3.5 Experimental Process and Data Gathering	52
3.6 Data Analysis	52
3.7 Evaluation.....	53
3.8 Tools	54
CHAPTER FOUR SAMPLING AND EXPERIMENTAL SETUP	55
4.1 Sampling.....	55
4.2 Materials and Experiment Setup	62
4.3 Challenges and Solutions.....	65
4.4 Experiment Steps	68
4.5 Summary.....	70

CHAPTER FIVE RESEARCH FINDINGS AND DISCUSSION.....	71
5.1 Introduction	71
5.2 Data Analysis	71
5.3 Findings	83
5.4 Results and Evaluation.....	89
CHAPTER SIX CONCLUSION AND FUTURE WORK.....	94
6.1 Conclusion.....	94
6.2 Research Contributions.....	95
6.3 Future Work.....	95
REFERENCES	96
PUBLICATIONS	101



UUM
Universiti Utara Malaysia

LIST OF TABLES

Table 2.1: Signature-based and Anomaly-based Characteristics	21
Table 2.2: Prior Works for Stepping Stone Detection Approach	29
Table 2.3: Characteristics of SSD Techniques	38
Table 2.4: Characteristics of SSD Models	44
Table 3.1: The relation between attributes and variables	50
Table 5.1: The detection ratio result for the known backdoors	90
Table 5.2: The initial values for the detection result for 10 samples	91
Table 5.3: TPR and FPR for the 10 known backdoors	91
Table 5.4: The detection ratio result for the unique samples	91



LIST OF FIGURES

Figure 1.1: Stepping Stones Chain Intrusion.....	3
Figure 2.1: The Layer in the TCP/IP model and OSI model.....	10
Figure 2.2 : TCP packet structure	10
Figure 2.3: IP header structure	11
Figure 2.4: Stepping Stone Connection Chain	13
Figure 2.5: Organization of backdoor detection.....	20
Figure 2.6: One-to-one relationship	34
Figure 2.7: One-to-many relationship.....	35
Figure 2.8: Many-to-many relationship	35
Figure 2.9: General Classification of SSD	40
Figure 2.10: SSD Host-based model design.....	41
Figure 2.11: SSD Network-based model design.....	42
Figure 2.12: Backdoor Attack Traffic.....	45
Figure 3.1: Operational Framework.....	49
Figure 3.2: The relationship between variables and attributes	51
Figure 4.1: The interface of Spy Net Client's software	56
Figure 4.2 : The interface of Sub7 Gold client's software.....	57
Figure 4.3: The tools that can be used to encrypt and make new samples	58
Figure 4.4: The interface to one of the encryption tools.....	59
Figure 4.5: Test result for the sample UUM_Backdoor before the encryption.....	60
Figure 4.6: Test result for the sample (UUM_Backdoor) after the encryption	60
Figure 4.7: Eset Smart Security 6 test result for the sample after the encryption.	61
Figure 4.8: Network Topology used for Offline Design testbed.....	63
Figure 4.9: Backdoor's client (attacker) software that used offline design	63
Figure 4.10: Network Topology used for Online Design testbed	64
Figure 4.11:UUM_Backdoor in virtual machine software (VMware) environment	65
Figure 4.12: UUM_Backdoor in real environment	66
Figure 4.13: Virtual Machine software environment.....	67
Figure 4.14: System restore method in Virtual Machine software	67

Figure 4.15: Eset Smart Security 6 tool process.	68
Figure 4.16: Using Wireshark tool to capture the network packets	69
Figure 5.1 : Scenario (1), the flow between the backdoor and the attacker	73
Figure 5.2 : Scenario (1), the capture packets in the victim side	73
Figure 5.3: Scenario (1), the capture packets in the attacker side	74
Figure 5.4: Scenario (2), flow between the backdoor and the host of the attacker ...	75
Figure 5.5: Scenario (2), Poison backdoor in the victim side.....	76
Figure 5.6: Scenario (2), Poison backdoor in the attacker side.	76
Figure 5.7: Scenario (3), the victim host is active and the attacker host is offline ...	77
Figure 5.8: Scenario (3), the capture packets in the victim side.	78
Figure 5.9: Scenario (4), the flow between the APT backdoor and the attacker.....	79
Figure 5.10: Scenario (4), the backdoor use outgoing flow only	80
Figure 5.11 : Scenario (5), using the intermediate server	81
Figure 5.12 : Scenario (5), the capture packets in the victim side	82
Figure 5.13: The information of the intermediate online server	82
Figure 5.14 : The activity graph of the backdoor	84
Figure 5.15: The backdoor activity	84
Figure 5.16: Backdoor detection based on the round trip time (RTT) technique	85
Figure 5.17: Backdoor's scenario without round trip time	86
Figure 5.18: Detection Backdoor Technique Based on Stepping Stone Approach ..	88
Figure 5.19: The detection result for the known samples	90
Figure 5.20: Avira Antivirus Scan Process Time.....	92
Figure 5.21: Eset Smart Security 7 Scan Process Time.....	93
Figure 5.22: SSD Detection Time	93

LIST OF APPENDICES

Appendix A The Snapshots to SSD Results	102
Appendix B The Snapshots to Antivirus and IDS Results	109



CHAPTER ONE

INTRODUCTION

1.1 Introduction

Network applications are an important part of our daily lives. We cannot dispense with the use of these networks. At the same time, security attacks have been dramatically increasing. Security attacks come from users who do not have authorization to access the network and use the software. Most of the time, an unauthorized access is run by using a special malicious software called “malware.”

In the last ten years, malware attacks have become a common crime story online. Nowadays, well-known threats, including viruses, worms, trojans, backdoors, exploits, password stealers, and spyware, have reached millions, and among these threats, the backdoor attack has a high rate of intrusion across global networks around the world (Microsoft, 2012).

The backdoor attack is a hidden technique used to gain remote access to a machine or another system without authentication. It was a major threat in recent years and is one of the threats that cause serious concerns because the outbound it generates consists of several types of packages and exerts dangerous control over a range of hosts (B. Choi & Cho, 2012). As such, detecting backdoors has become an urgent demand today.

1.2 Research Background

Several techniques are used to detect backdoor attacks. All these techniques can be classified into two types, namely, signature-based and behavior-based techniques (Sonawane, Prasad, & Pardeshi, 2012). In signature-based detectors, a sequence of features unique to the backdoor attack is used to detect the backdoor attack. Most intrusion detection systems (IDS) are signature-based (Kang, Kim, Kim, Kwon, & Im, 2011). Behavior-based methods focus on analyzing malicious behavior. Such behavior includes addresses of the backdoor destination and source, the types of attachment in which they are embedded, and the statistical anomalies in backdoor-infected systems (Modi et al., 2012).

Signature-based techniques have less scanning time and few false positives. However, unknown backdoors can easily evade detection. In addition, signature-based techniques do not have the ability to deal with obfuscation. In the same way, behavior-based techniques cannot detect a lot of polymorphic backdoors in the present environment (Maarof & Osman, 2012). Moreover, behavior-based techniques suffer from two limitations, namely, the high false alarm rate and the complexity involved in determining which features should be learned in the training phase (Idika & Mathur, 2007).

Given these limitations, several researchers began looking for a new technique that can detect backdoor attacks. One of the successful techniques discovered in recent years and has yielded great outcomes in the field of security is the stepping stone detection (SSD).

In the past years, SSD was confined only in the detection of intrusions run by intruders through a chain of hosts in the network (stepping stone chain intrusion), as shown in Figure 1.1. This method is usually used by intruders to conceal their intrusion track in the network.

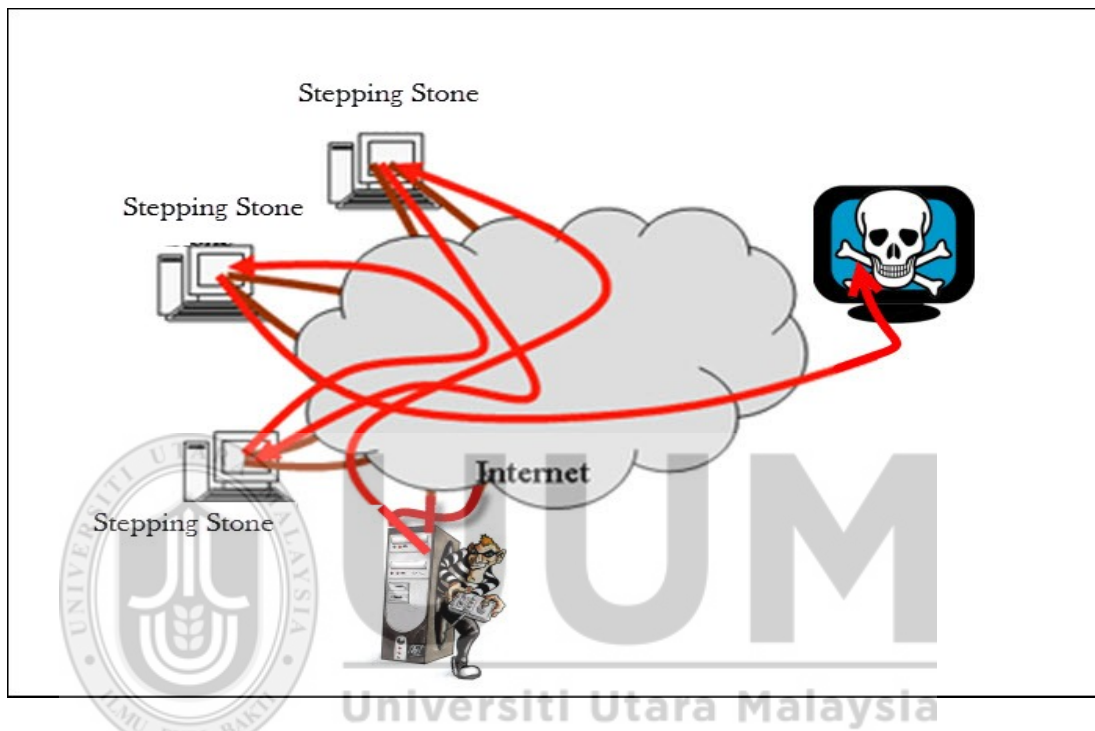


Figure 1.1: Stepping Stones Chain Intrusion

In fact, the SSD approach is quite flexible and dependable in detecting interactive connections (Ping, Wanlei, & Yini, 2010). Thus, given that the connections of backdoor attacks consist of interactive connections, the SSD theory can be extended to the detection of backdoor attacks by using concepts taken from SSD based-research (Omar, Amphawan, & Din, 2012).

1.3 Problem Statement

In the last decade, backdoor attacks have emerged as one of the most serious threats and major intrusion into global networks around the world (Microsoft, 2012). Undoubtedly, all computer viruses are undesirable, but backdoor viruses are especially dangerous because they can bypass normal authentication systems and use a hidden technique that allows a remote attacker to access and forward a user's personal information (B. Choi & Cho, 2012).

Antivirus utilities have an important function in overcoming backdoor problems. However, pattern-based signatures are the most common technique employed for backdoor detection. This technique requires the right signature to be embedded into the antivirus and detection (Prasad, Babu, & Rao, 2013). Moreover, to address a novel attack intrusion or an encrypted attack intrusion, antivirus tools apply more complex techniques that exhaust the resources of the system. In addition, daily zero-attacks and false positives have become the most challenging problems in the backdoor detection field (Maarof & Osman, 2012). While, the advantage of any detector lies in their simplicity, speed and accuracy (Sathyanarayan, Kohli, & Bruhadeshwar, 2008). For all above, the present study proposes the detection of backdoor attacks through the use of the simple concepts of SSD-based research to enhance the speed and accuracy and reduce the storage resources used by traditional antivirus tools.

1.4 Research Question

This research intends to solve detecting backdoor attack by using the SSD approach. The main question is “How the backdoor problem can be detected by using the SSD approach?”. Subsequent questions of the main research question are as follows:

- i) Which SDD technique is the suitable solution to solve detecting the backdoor attack problem?
- ii) How SSD approach can be developed to overcome the backdoor attack problem?
- iii) How the proposed SSD approach will be evaluated?



UUM
Universiti Utara Malaysia

1.5 Research Objectives

The objectives of this research are:

- i) To identify the requirements of the suitable SSD-based technique.
- ii) To develop approach that can detect the backdoor attack problem by using SSD-based approach.
- iii) To evaluate the capability of the proposed approach by conducting a well-planned experiment.

1.6 Scope

The scope of this research is as follows:

- 1- To detect the backdoor attack problem in host-based environment.
- 2- To use a technique that based on SSD techniques.

Due to the fact, the Local Area Network (LAN) architecture easy to control (Omar, 2011), LAN is chosen as suitable network architecture to run the experiment of this research.

1.7 Significance of the Research

Many advantages can be gained from the application of the SSD approach in the detection of backdoor attacks. The significance of this research consists of the following:

- 1- Reduction of the scanning process time: The SSD approach increases the speed of the detection of backdoor attacks. Therefore, it also reduces the time gap between detection and response (Omar, 2005).
- 2- Enhancement of the accuracy of backdoor attack detection: The SSD-based technique employed in this study is based on the use of interactive connections. As such, it can detect all backdoor types that cannot be detected by traditional antivirus tools, including known and unknown types, even if the backdoor is encrypted.

- 3- Reduction of the storage space occupied: This technique can reduce the large storage space used by traditional antivirus tools.
- 4- This research study contributes to the body of knowledge in the domain of research. It is done by extending the application of SSD-based techniques, which are usually used only in SSD-based environments.

1.8 Summary

This chapter described the background of the research. It outlines the problem, questions, and objectives of the research and also points out its scope and significance. The chapter states the problems related to backdoor attack detection that have prompted researchers to look for a new approach to enhance the accuracy and speed of the detection and reduce the large storage resources used by traditional antivirus tools. This study intends to solve these problems by using concepts related to SSD, one of the most successful techniques developed in recent years, which has yielded great outcomes in the field of security. Many advantages can be gained from the application of SSD concepts in the detection of backdoor attacks, such as the enhancement of the accuracy and speed of the detection and the reduction in the storage space used for backdoor attack detection.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

This chapter aims to provide background information required in understanding the subsequent chapters. It provides a review of the existing literatures on the SSD approach and backdoor attacks. Furthermore, it also discusses works related to such issues.

Section 2.1 lists the basic terminology related to backdoor attacks and the SSD approach. Section 2.2 outlines the basic characteristics, typology, history, and new developments of backdoor attacks. It ends with an overview and discussion of the techniques and methods used in detecting backdoor attacks, with a special focus on the current and future developments of these techniques. Section 2.3 introduces the main concepts, evolution, and related works of the SSD approach and discusses the SSD techniques that have been used. Section 2.4 concentrates on the techniques that can be used by the SSD approach to detect backdoor attacks. Section 2.5 illustrates the relationship between backdoor attacks and the SSD approach and justifies the application of SSD in detecting backdoor attacks. Finally, Section 2.6 summarizes the entire chapter.

2.2 Terminology

Firstly, we have to present some definitions that will be used in this research. In this section, there are two parts, the backdoor terms and SSD terms.

2.2.1 Network Security Terminology

In order to understand the literature review on detection of the backdoor attack it is necessary to understand the general network security concepts with a focus on the Transmissions Control Protocol/Internet Protocol (TCP/IP) that is used by most of networks.

Malware: Malware word comes from two words, malicious and software. It is a program that is designed to be harmful.

Backdoor: A hidden technique is used for getting remote access to a machine or other system that without authentication.

Intrusion: An illegal act of entering to a computer, network or any system.

TCP/ IP packet: A simple unit of the network transmission over TCP/ IP protocol, these packets fit into the network layer of TCP/ IP model at the source. The TCP/IP model was created after the Open System Interconnection OSI, model which defines a networking framework to implement protocols in seven layers.

Figure 2.1 illustrates the layer in the TCP/IP model and OSI models. After being modeled at the network layer, packets are encoded into bits, and then pass to the data link layer. From the data link layer, the packets are inserted into frames, and then passed to the physical layer. At the destination, the process is reversed respectively. All the TCP/ IP packets are included two major pieces to transfer over the Internet switching: the packet header and the data.

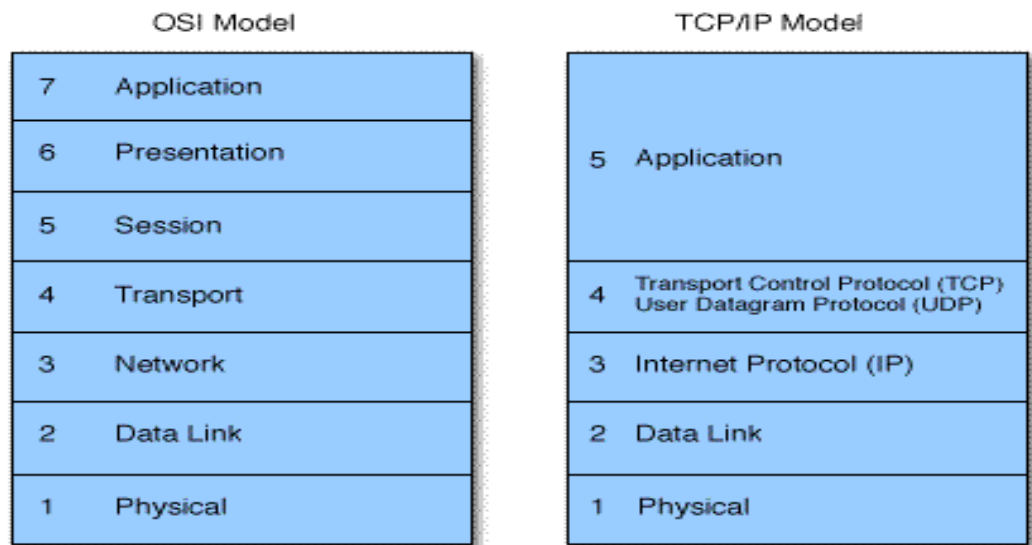


Figure 2.1: The Layer in the TCP/IP model and OSI model

The TCP/ IP packet header includes several pieces of information about the packet itself. For example, the IP addresses for both sender and destination are included in each packet to determine the packet path (Kurose & Ross, 2012). Figure 2.2 shows TCP packet structure

Field	Explanation
Source port	TCP port of sending host
Destination port	TCP port of destination host
Sequence number	Ensures all bytes have been received
Ack number	The sequence number of the next byte
Data length	Length of the TCP segment
Reserved	Reserved
Flags	What content is in the segment
Window	How much space is in the TCP windows
Checksum	Ensures validity of the header
Urgent Pointer	If urgent data is being sent, this specifies the end of that data in the segment

Figure 2.2 : TCP packet structure

Source Port (16)		Destination Port (16)	
Sequence Number (32)			
Acknowledgement Number (32)			
Data offset	Reserved (6)	Flags (6)	Window (16)
Checksum (16)		Urgent (16)	
Options and Padding			
Data (Varies)			

Figure 2.3: IP header structure

The packet header of IPv4 as shown in Figure 2.3 consists of 14 fields. However, the 14th is optional. The most significant bits are considered to come first. For example, the version field in IP address header is actually found in the four most significant bits of the first byte because it's length only 4 bits. The following information is available for IPv4 packet header.

Source IP address: It specifies; the sender of the packet.

Destination IP address: It indicates the receiver of the packet.

Total length (16-31 bits): This 16-bit field defines the entire datagram size, including header and data, in bytes.

Flags: it is used to control or identify fragments.

Versions (0-3 bits): the four-bit version in IP header.

IHL (4-7 bits): it is use to determine the header length.

Type of service (TOS) 8-13 bits: The original definition of the TOS was for a sending host to specify the datagram preference as it made way through an Internet. For instance, one host could set TOS to prefer low delay and high reliable in high throughput. It is not used widely in practice implementation.

Identification: it is used for uniquely identifying fragments of an original IP datagram.

Fragment offset: it specifies the offset of a particular fragment relative to the beginning of the original un-fragmented IP datagram.

Time to live (TTL): it helps to prevent datagram from persistent or going in circling on the Internet.

Protocol: this field is used to define the protocol in datagram.

Header Checksum: this field is used for error-checking of the header. It is used to re-check the transferred bits that reach to destination in a correct sending order. .

Options: an optional field that may be not used within the packet life cycle in the Internet.

2.2.2 SSD Terminology

Also, we have to present definitions which are related to SSD approach.

Assume attacker logs in from host 1 and ultimately connects to host n , which is the goal host to the attacker, through host 2 until host $n-1$, as illustrated in Figure 2.4.

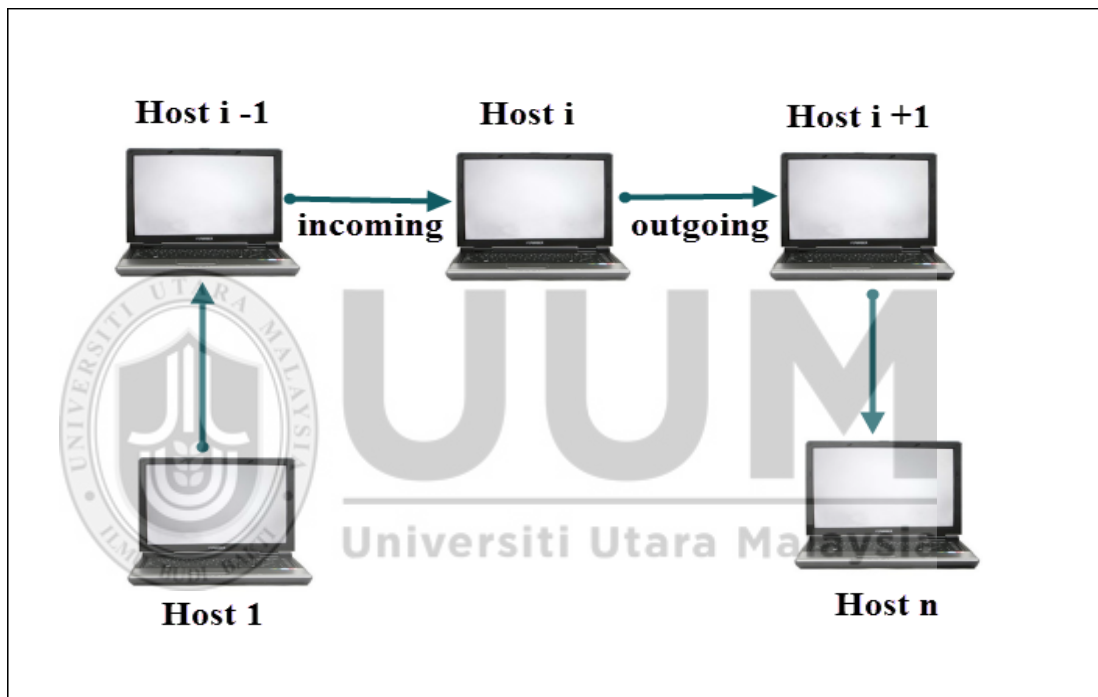


Figure 2.4: Stepping Stone Connection Chain

Connection: The connection of user logging from host to next host is called connection session between the two hosts.

Chain: Given n hosts H_1 to H_n , a series (chain) of connections is defined as a chain $C = \langle C_1, C_2, \dots, C_{n-1} \rangle$ where C_i is a chain of connection between host H_i and host H_{i+1} for a series $i = 1$ to $n-1$.

Stepping-Stone: The intermediary host of a connection chain which is invaded by the attacker.

Downstream and upstream: If a direction is along a user's login direction from attacker to victim as shown in the arrows in Figure 2.3, it is called downstream. Otherwise, it is called upstream.

False positive: False positive means when the detector detects a backdoor in a file does not have backdoor.

False negative: False negative means when the detector does not detect the backdoor in a file has infected by backdoor.



2.3 Backdoor Attack

Sub7 is the first backdoor code was created in 1990 which enabled any user backdoor (hacker) to get access to a victim's computer (Decloedt & Van Heerden, 2010). According to the mentioned study above, the backdoors have been used as tools by programmers for many years to check and debug applications, this is generally preferred when a programmer (developer) is programming or improving a software application that needs using authentication in order to test and run the application. This software (backdoor) becomes a big threat once dishonest software discovers and using them to gain illegal access to the victim's applications.

Backdoor's intrusion could be via instillation code (software), such as 'Back Orifice' or through related backdoors that were done and left by the program's developers. Popular backdoor programs from the 90's that used for mischief were Netbus Back, Orifice and Sub7. Examples to the latest popular backdoors are: Aimot, DsBot, Egg Drop, Hupigon, VanBot and Mo Sucker (Decloedt & Van Heerden, 2010).

2.3.1 Types of Backdoors

According to the study (Idika & Mathur, 2007), there are two types of backdoor attacks. The first type of backdoor in an Operating System (OS) or a complicated application is a technique to pass normal authentication and get access. Over the development period of application or the OS, the hackers add back doors for various purposes. The second type of backdoor can be installed program or could be updating (insert code) to existing software. The setup code in the victim's

computer may allow a hacker log on to the computer without authentication. A backdoor's task usually begins by facilitating the operation of illegal access and using a password and software application or any action to illegal users.

According to (Declodt & Van Heerden, 2010), there are many techniques are used by the backdoor to hunt his victim. The first, by combining the backdoor's code with legal software such as combine the backdoor with Microsoft word application or any operating system. The second method is, when the backdoor uses the harmful software such as worms, viruses or any malware in order to reach his victim. The third method, by exploiting security holes in computer application or OS such as XP operating system holes. The fourth method, by change the compiler's code so as to insert backdoor code inside the compiler's segment code. The last method, when the hacker tricks the user in order to setup the backdoor. However, usually the attackers setup the backdoor by using an automated method.

According to the study (Dittmann, Karpuschewski, Fruth, Petzel, & Munder, 2010), there are two types of communication are used by the backdoor in order to connect with his attacker. First type is direct connection or client server connection, the attacker in this situation is the client, and the victim host is the server. Usually, the client's software has a graphic user interface (GUI) that makes the attacker remotely control the victim. Second type is indirect connection, this type is very similar to the direct connection, but the client generates the backdoor in the intermediate hosts. The data exchange is same as in the direct communication.

2.3.2 Authors and Users of Backdoors

Different names are used for naming the makers (writers) and users of the backdoors. Hackers, black hats and crackers are the most famous names. These names are called on all organizations and persons that using the backdoors or create them; these names could be an internal or external threat such as the threat of spying by a foreign government (Idika & Mathur, 2007).

According to the above mentioned study, the installation of the backdoor in the victim's machine is going through two stages. The first stage is setting phase and the second stage is post release, it is coming when the victim becomes ready to intend backdoor instillation. However, all hackers install the backdoor during post release stage due to the instillation of the backdoor during the first stage is needed to install it manual.

Generally, when the hackers intended to make a new backdoor, usually they use one or two methods, obfuscation and behavior add/modify in order to avoid the antivirus. The obfuscation is used in order to hide the real intentions of the backdoors. Behavior add/modify, effectively creates a new application, although the core of the backdoor may not have changed (Decloedt & Van Heerden, 2010).

2.3.3 Backdoor Detectors

There are many ways that make the computer gets infected by the backdoor attack. Backdoors attack can be bundled with shareware or other download software. It is not difficult for many types of the backdoor to pass the firewall of the system (Mudzingwa & Agrawal, 2012; Salimi & Arastouie, 2011). Therefore, most of these systems are provided with the second defense wall that is the backdoor detector, it is any technique or method that uses to protect the computer. The backdoor detector may or may not combine with the operating system.

According to Decloedt (2010), the backdoor's detectors have two inputs. First, the database (signature) or the knowledge of the backdoor's behavior. Second, the software which is under test. Generally, these detectors compare the backdoor's signature with the known patterns (database). This type called signature technique. However, this technique cannot face a new backdoor's code (Mudzingwa & Agrawal, 2012; Salimi & Arastouie, 2011). Anomaly based detection uses its knowledge to check the normal behavior and detect the backdoor. This type includes a special rules set in order to decide, is it backdoor or not. However, this method cannot detect a lot of polymorphic viruses (Modi et al., 2012; Mudzingwa & Agrawal, 2012). Figure 2.5 shows the organization of backdoor detection, each technique can use one of three methods: dynamic, static, or hybrid.

In general, the backdoor detection techniques classify as follows:

a) Anomaly based Detection :

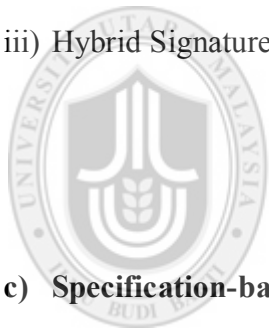
- i) Dynamic Anomaly
- ii) Static Anomaly
- iii) Hybrid Anomaly

b) Signature-based detection

- i) Dynamic Signature
- ii) Static Signature
- iii) Hybrid Signature

c) Specification-based Detection

- i) Dynamic Specification
- ii) Static Specification
- iii) Hybrid Specification



UUM

Universiti Utara Malaysia

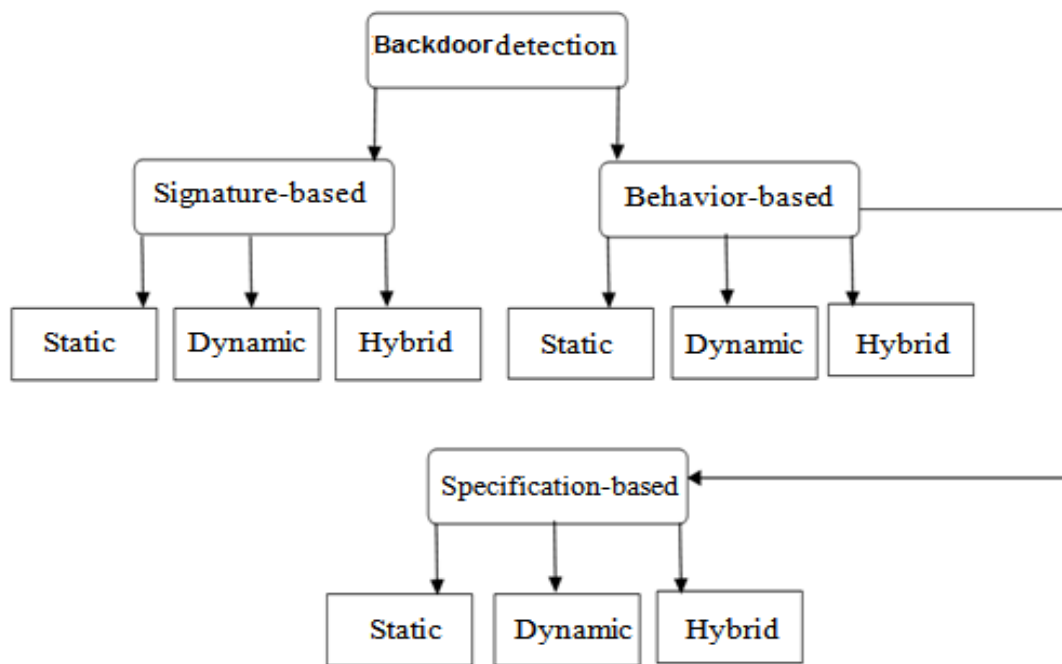


Figure 2.5: Organization of backdoor detection

Mudzingwa and Agrawal, (2012), and Modi, (2012), illustrated the advantages and disadvantages for both two types of the backdoor detectors as follows: For Signature-based technique can easily evade it by unknown backdoor and cannot face any obfuscation. All that make overall rate of the accuracy is low. While, for Anomaly-based technique, the overall false positive (FPR) and overall false negative (FNR) are high. Certainly, the accuracy is a crucial term in intrusion detection system efficiency. Therefore, they are (signature-based approach and anomaly-based approach) not powerful enough when facing the backdoors attack. Table 2.1 illustrates the main characteristics for both types.

Table 2.1: Signature-based and Anomaly-based Characteristics

Advantages	Disadvantages
Signature-based Approach	
<ul style="list-style-type: none"> i) Identifies intrusion by matching captured patterns with preconfigured knowledge base. ii) High detection accuracy for previously known attacks. iii) Low computational cost. 	<ul style="list-style-type: none"> i) Unknown backdoors can easily evade detection and cannot deal with simple obfuscation ii) Accuracy Rate / overall rate is lower iii) Need to be updated
Anomaly-based Technique	
<ul style="list-style-type: none"> i) Uses statistical test on collected behavior to identify intrusion. ii) Can lower the false alarm rate for unknown attacks. 	<ul style="list-style-type: none"> i) Not able to detect a lot of polymorphic viruses present (Packers). ii) Low / Maintenance iii) Scalability: the least scalable methodology due the time it requires to learn and build its baseline profiles iv) Require more time to configure, learn, and tune the environment. v) Requires more resources to manage the high volumes of alerts it produces vi) Overall False Positives / High vii) Overall-False Negatives / High

2.3.4 Recent Backdoor's Detection Approaches and Related Works

Several types of detection technique are supposed to detect the backdoor attack problem, which are however not always possible. One of these techniques based on the backdoor's signature that used in the most of the Antivirus utilities. The process of this technique in the infected machine by matching the features of the backdoor (signature) with a pre-existing database affiliated to Antivirus. However, this technique fails when the signature of the backdoor attack is not existed in the database of the antivirus (Balzarotti et al., 2010; Radmand, 2009). Furthermore, the backdoor can be modified in order to change the backdoor's signature to a new one.

Network communication monitor is one of the methods which have been used to detect the backdoor attacks. For example, running the (nstat-a) code will listen on TCP port 113. This type of connection can be found using the NIDS model (Radmand, 2009). However, the common method to detect the backdoor with the NIDS is to check each packet that is linked to backdoor activities. One way is to look for the string in the network stream which has more effectively than antivirus tools. However, the processing in this method will exhaust the resources and is not feasible.

The data mining techniques have been used also in order to detect the malware in general (Siddiqui, Wang, & Lee, 2008). Most of these techniques appeared a high accuracy, but all these works are complex and exhaust the resources such as the memory and the performance of the host. In the same way, some of the studies have used Artificial Intelligence strategies such as (Salimi & Arastouie, 2011), it is a novel approach for backdoor attack detection. Genetic algorithms (GA)

and Artificial Neural Network (ANN) are used in this approach. ANN is designated for classifying system features and forecasting the percentage of the backdoor existing probability. And Genetic Algorithm (GA) in order to give a deterministic answers to the issue. However, it has the same disadvantages of the complex techniques.

Another way to detect the backdoor proposed by Kabiri and Ghorbani (2005), they proposed using Machine Learning (ML) techniques. However, in this method some of an unknown file can be classified as malicious or benign. Some of these studies are applying machine learning methods on the content-based feature (Menahem, Shabtai, Rokach, & Elovici, 2009).

Some studies have gone over deferent trends such as the study (Waksman & Sethumadhavan, 2011), it discussed the possibility of hardware components that can contain hidden backdoors, which can be enabled with catastrophic effects or for ill-gotten profit. According to the mentioned study above, these backdoors can be inserted by a malicious insider on the design team or a third-party IP provider. They proposed the techniques that allow us to build trustworthy hardware systems from components designed by un-trusted designers or procured from un-trusted third-party IP providers.

In generally speaking, all above studies consider the issue of detecting the backdoor attacks problem. As well, this research considers the same issue, but in a so different way.

2.4 Stepping Stone

There are two main concepts are involved by the name of stepping stone, stepping stones attack (chain), and stepping stone detection SSD approach.

2.4.1 Stepping Stone Chain

In order to hide the track of intrusion in the network, the intruders use a series of hosts on the network which are called (stepping stones chain). This kind of intrusion can be detected through use an approach called, SSD (Kampasi, Zhang, Di Crescenzo, Ghosh, & Talpade, 2007).

2.4.2 SSD Approach

Since a stepping stone is just forwarding attack traffic through all stepping-stone connection chain, the connection's traffic in the same connection chain must have similar characteristics. Therefore, the problem of detecting stepping stones comes down to find correlated connections with the same characteristics. A new method has emerged in order to solve this problem would be make comparisons between the content of the incoming packets and the content of outgoing packets within a network to detect packets which, have the same value (content). This method is called SSD approach.

SSD approach is a system to analyze the traffic of the connection and identify which connections are stepping stone connections or identify which connection pair are correlated connections. Correlated connections are a pair of connections, which are in the same way of connection chain. On the chain connection, the connection which is closest to the attacker is called the upstream

connection. While the connection which is closest to the victim is called the downstream (Shullich, Chu, Ji, & Chen, 2011).

As well as the evolution of computer technology, the researchers have suggested new techniques for detecting a stepping stone intrusion like Content-Based Thumbprint (Staniford and Heberlein, 1995), Time-Based Approach (Zhang and Paxson, 2000), Deviation-Based Approach (Yoda and Etoh, 2000), Round-Trip Time Approach (Yung 2002, Yang and Huang, 2005), and Packet Number Difference-Base Approach (Donoho et al., 2002, Blum, Song, and Venkataraman, 2004). The next section will show all these related approaches in more details.



2.4.3 SSD Evolution and Related Work

To understand on how SSD works in details, we have to start with the historical evolution of this theory. The SSD evolution includes three phases: The first phase is the past of SSD, second phase is present and the third phase is the expectation of what will be SSD in the future. Based on historical reviews that done by Omar, (2012) and Shullich, Chu, Ji, and Chen, (2011). The evolution steps of SSD are reviewed in the next four sections.

2.4.3.1 The Past of SSD

According to (Omar et al., 2012; Shullich et al., 2011), in SSD research, Staniford in 1995 proposed the concept of ‘thumbprint’ that summarized the packet’s content by providing it with a unique identity which differentiated it from other packets. However, the thumbprint method was not suitable for encrypted connections (Shullich et al., 2011). After that, in 2000 (Yoda & Etoh) and (Zhang & Paxson) are proposed on/off and deviation methods respectively. But, these two methods were prone to high false positive and active perturbation problems (Omar et al., 2012).

In 2006 (Yang & Huang), proposed the ‘reply-echo’ technique to minimize the false positive problem and Donoho, Flesia, Shankar, Paxson, Coit, and Staniford (2002), proposed solving the perturbation problem using Active Perturbation Attack (APA), APA is a technique created by the intruder to influence the SSD process. In the same year, Wang, Reeves and Wu (2002), applied the Inter Packet Delay (IPD) technique to overcome the stepping stone problem by a new use proposed of data that is more effective in detecting stepping stones.

After Yang, and Huang (2006), first introduced a new technique, Round Trip Time (RTT), which is for reducing the false positive rate, past SSD researches began conducting experiments related to Yang and Huang (2006)'s research.

In 2004, Jianhua & Huang introduced the "Step-Function" and "Conservative & Heuristic" Jianhua, Hai, Hao and Zong which were methods enhanced from Yang, and Huang (2006) methods. Meanwhile, in 2003 Strayer, Jones, Castineyra, Levin and Hain were focused on the wireless environment in detecting stepping stones.

In conclusion, the past SSD research focused on the right data type to be used in the SSD approach. The differences lie only in different types of data (e.g. Data, time, inter-packet delay) and their concentration on RTT at the end of the past SSD period.

2.4.3.2 Current SSD

According to (Omar et al., 2012; Shullich et al., 2011). Blum, Song, and Benkataraman (2004), Almulhem (2006), Venkateshaiah (2006) and Wu and Huang (2002) have shown that SSD researchers have changed their focus from enhancing the SSD approach to how make SSD more power in face perturbation problem. This can be seen in research by Blum, et al., (2004) that re-directed SSD research towards gaiting less false positive and false negative rates.

In 2007, Venkateshaiah and Wright created a method to influence SSD and Jianqiang (2006) provided a testbed through which the SSD approach can be examined. In 2007, Almulhem and Traore on the other hand, provided SSD taxonomy to expose those outside the field to SSD.

The researches on the present SSD have become more widespread with the introduction of Artificial Intelligence (AI) techniques. Research which applies AI techniques are referred to as RTT-based research. This effort was started by Yang (2009) who proposed the using of data mining technique to mine for TCP/IP packets in the effort of finding RTT. AI application was continued by Wu & Huang who introduced the Neural Network technique that focuses on finding RTT. From the discussion on AI techniques that have been used, it seems that their technique had the potential of solving SSD problems.

The present SSD researches are focused on issues beyond past SSD research, and introduce new discoveries to the SSD researches. The introduction of different AI techniques used to detect RTT and later to detect stepping stones, shows that the present SSD is evolving. Furthermore, the present SSD shows that the extensive buffering method used as perturb to the present SSD approach exists. There are also studies, which focus on confidence bound, false positive and false negative rates. Attached testbed, which is much needed in SSD research, has also been proposed by Jianqiang (2006). Eventually, this study concluded the most of previous related work in Table 2.2.

Table 2.2: Prior Works for Stepping Stone Detection Approach

System	Characteristic	Function	Authors	Year
Thumbprint	Content	Identify correlated connections	Staniford and Herberlein	1995
ON/OFF	Timing	Identify correlated connections	Zhang and. Paxson	2000
Deviation	Timing	Identify correlated connections	Yoda and Etoh	2000
IPD	Timing	Identify correlated connections	Wang, et al.	2002
Multiscale	Character Count	Identify correlated connections	Donoho, et al.	2002
Send-ack/Send-echo	RTT	Identify abnormal connections	Yung	2002
Watermark	Timing	Identify correlated connections	Wang, D. Reeves	2003
State-Space	Packet events based	Correlation algorithm	Strayer, et al.	2003
Detect-Attacks	Packet Count	Identify correlated connections	Blum, et al.	2004
RTT-Thumbprints	RTT	Identify correlated connections	Yang, and Huang	2005
(Delay + Chaff) S-I, S-II, S-III and S-IV	Timing	Identify correlated connections	Zhang, et al.	2006
DMV	Packet Count	Identify correlated connections	He and Tong	2006
Step-Function	RTT	Identify abnormal connections	Yang, and Huang	2006
DM	Timing	Identify correlated connections	He and Tong	2006
Watermark Secrecy	Packet Count	Trace-Back	Peng, et al.	2006
Anomaly	Other	Identify abnormal connections	Kampasi et al.	2007
Request-Response	Packet Count	Identify correlated connections	Huang et al.	2007
Dropped packet	Packet Count	Other (Optimization)	Omar et al.	2008
Sketching	Timing	Identify correlated connections	Coskun and Memon	2009
Step-Function	RTT	Identify abnormal connections	Ping ,Wanlei, and Yin	2010
chaff packets	Timing	quick-response real-time	Kuo et al.	2010
watermark	Other (BACKLIT)	Identify abnormal connections	Luo et al.	2011
Step-Function	RTT	Identify abnormal connections	Li	2011
watermark	Other collaboratively	Trace-Back	Houmansadr and Borisov	2012
Watermark	inter-packet delays	Trace-Back	Gong, Rodrigues, and Kiyavash	2013

2.4.3.3 Future SSD

According to (Omar et al., 2012), future SSD would focus on the development of SSD testbed. The standard testbed is necessary to the SSD-based research to execute the standard experiment or testing. In the testbed, the requirements, the tools and the topology that will be used are well defined. So far, SSD research has only depended on the testbed developed by Jianqiang et al., (2006). Unfortunately, this testbed has still to be made known to the public. Moreover, from the readings it was found that a standard SSD testbed does not exist to date and most researchers use their own testbed. Because of the use of AI techniques in the SSD environment, future SSD should focus on the developing of testbed that will enhance AI SSD.

The concept of hybrid SSD is also another possibility that could become the research focus for future SSD. More often than not, the past and present SSD research have only depended on network-based SSD (NSSD) to Robert et al., (2001), Staniford (1995), Yoda and Etoh (2000). Although these studies did not clearly define their SSD as NSSD, the use of network packets as main source of the SSD process shows that it is NSSD. Studies by Almulhem and Traore (2007) and Wang and Reeves (2003) have divided the SSD approach into network-based and host-based SSD (HSSD).

From the discussion on past, current and future SSD, it is concluded that all of the researchers focus to the main usage of SSD to detect stepping stone either in host or network-based environment. No such a research that realized the other usage of the SSD in other fields of research.

2.4.3.4 Emerging Fields for Application of SSD

As discussed in previous sections, SSD-based research was mostly limited to only field of detection stepping stones without looking to the full capabilities of stepping stone detection in other fields of research. Only the study (Omar et al., 2012), that suggested using the potential applications of SSD in other fields.

2.4.4 SSD Issues

In addition to study the evolution of SSD, certainly, some issues associated SSD and our proposed research should as well be discussed.

2.4.4.1 Interactive and Non interactive Connection

The connection that should be set up before commands can given and the intrusion can interactively occur is called remote login or an interactive connection (Omar, 2011). An interactive connection is used by many applications; such as SSH, talent religion and so forth. The non interactive connections are the connections that do not need to continue connecting. For example e-mail, FTP and so forth.

SSD research was starting with using an interactive connection method (Staniford-Chen and Herberlein, 1995), (Zhang and Paxson, 2000) and (Yoda and Etoth, 2000). Till today the most of the researchers are focusing on interactive connections. The latest studies, such as those done by Wu and Huang (2010), Almualem and Traore (2010), Omar (2011) and Li (2011), have used interactive connections as fields for SSD. There are many reasons behind why the researchers focus on interactive connections, these reasons are listed below.

i) Most frequently used by attackers

Most of SSD studies are using interactive connections as a result to most attackers use interactive connections as their medium to run the attacks or intrusion actions.

ii) Provides active information to be captured

Broadly, interactive connections will depart updated information for each connection that has been created this information makes SSD possible. However, according to Carver, (2010) several times an intruder is left undetected there is a higher likely that the intruder would undetected forever. As such, an intruder should be captured as soon as he is detected.

iii) The possibility of doing On-line Process

Because of the interactive connections provide live network streams, it is potential for researchers to improve a system that can repel attacks and capture the intruder online. With this feature SSD can be used support Intrusion Detection System (IDS) to check whether the connection is used for intrusion or otherwise (Omar, 2011).

2.4.4.2 Positive and Negative False

The accuracy of IDS is also one of the important factors in network security but as network traffic is so complicated it is impossible to meet the perfect level of accuracy. There are two types of error results in IDS, false positive and false negative. False positive occur when IDS erroneously detect the right traffic. While false negative occur when IDS undetected the unwanted traffic. Both are the problems for security administrators. The greater number of false positive may be acceptable but they can create a huge burden for security administrator, as it has to deal with cumbersome amount of data. On the other hand, false negative do not give any opportunity to the administrator because it is undetected.

2.4.4.3 Passive and Active Detection

When SSD-based approach needs to monitor the traffic of the network or process all the time to detect stepping stones, the detection will be passive approach. While for active approach, it monitors only the necessary data in a well defined period of time (Omar, 2011). In the passive SSD approach, all the resources in detecting stepping stone are employed. This involves the using of CPU, memory and network to get the detection. It is different from an active SSD approach where the process for detecting the stepping stones is only executed when stepping stone is detected, this will reduce the using of resources such as CPU, memory and network.

2.4.4.4 SDD Matching Concepts

According to (Ni, Yang, Zhang, & Song, 2008) and (Sobh, 2008), the relationship between the Send and Echo packets, should takes one of the following cases:

- i) One-to-one relationship: In this case, there are one incoming packet and one outgoing packet. The detection happens only when the incoming packet matches the outgoing packet through the intermediate host as shown in Figure 2.6.

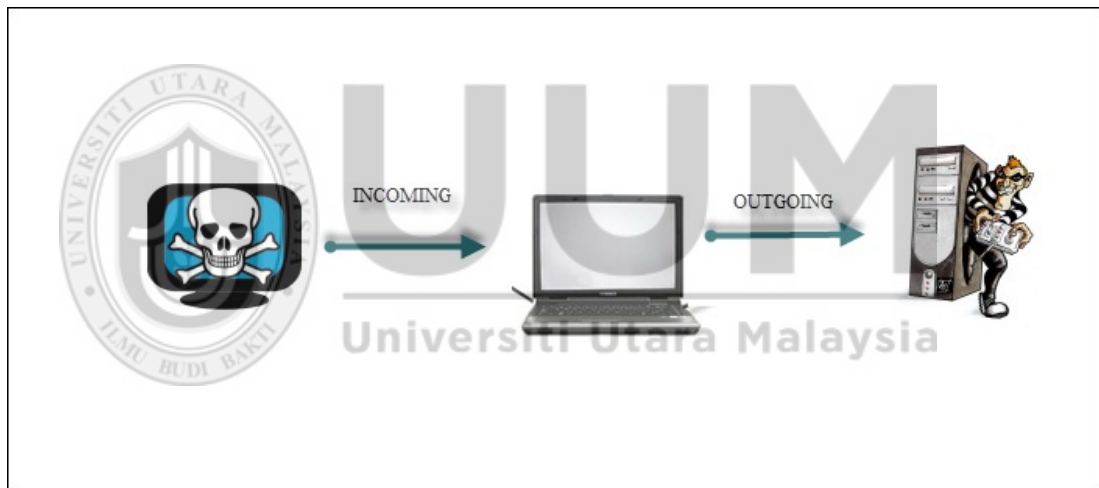


Figure 2.6: One-to-one relationship

- ii) One-to-many relationship: In this case, there is one incoming packet and many outgoing packets. The detection happens only when the incoming packet matches one of the outgoing packets through the intermediate host as shown in Figure 2.7.

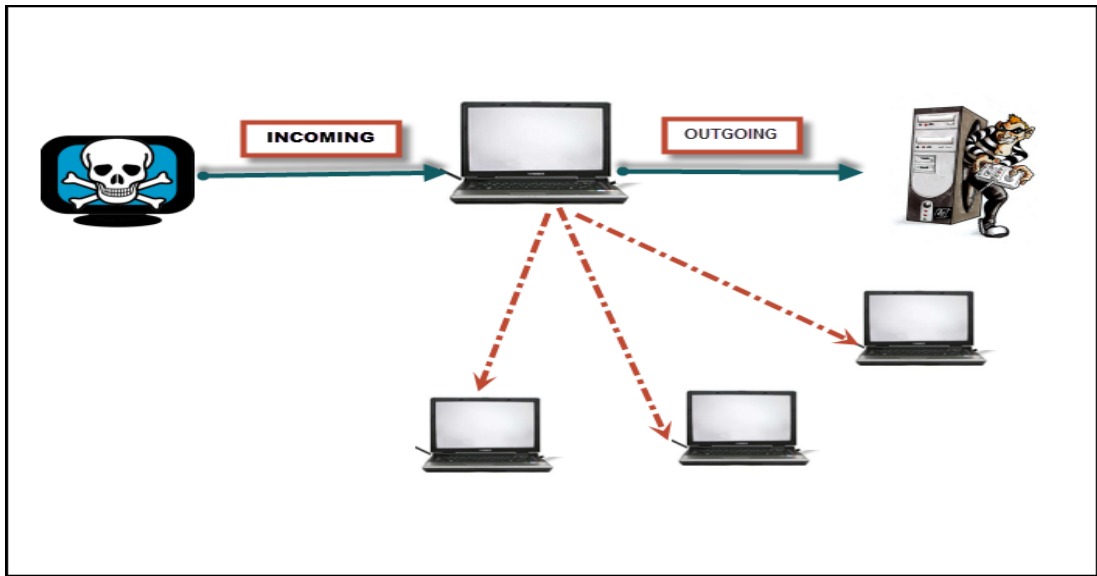


Figure 2.7: One-to-many relationship

iii) Many-to-many relationship: In this case, there are many incoming packets and many outgoing packets. The detection happens only when one of the incoming packets is matched to one of the outgoing packets through the intermediate host as shown in Figure 2.8.

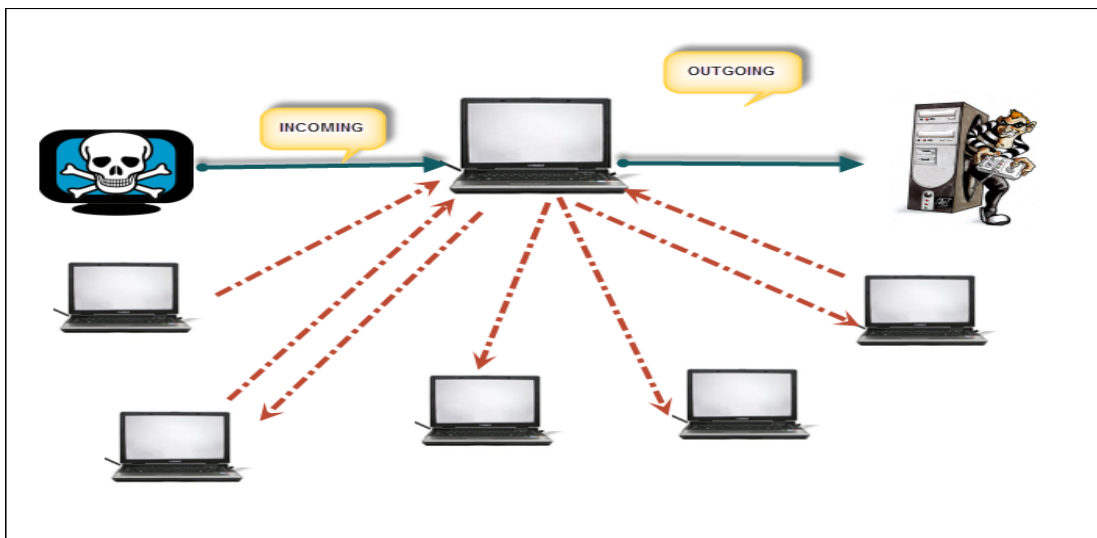


Figure 2.8: Many-to-many relationship

As conclusion, the main concept of SSD approach is the matching between the incoming and outgoing streams. However, in the backdoor situation, the backdoor repeats the connection for many times for the same specific port (Omar, Amphawan, & Din, 2013; Paxson & Zhang, 2000). In other words, the activity of the backdoor is repeated several times and each conversation contains the same amount of data that occurs for the same amount of time, and the outgoing packets for the backdoor should be matched. Therefore, in order to detect the backdoor attack, we need only to compare between two outgoing repeated sessions, if they are matched that means the source of the packets is backdoor attack otherwise, the source is clean.

2.4.5 SSD Techniques

Many techniques have been proposed in order to detect the matched packets, and to solve the stepping stones problem. However, there are four main techniques are used widely, there are: RTT based, Timing based, Deviation based and Packet number based. Table 2.3 illustrates the main characteristics for each technique.

For the Timing-based, there are more than six concern reasons make it not suitable to detect the backdoor attack. For the Packet number based is a weak in terms of resisting intruders when use it alone, therefore, must be use it with another matching metric to avoid the FPR (Yang & Lee, 2008). For Deviation-based technique, in addition to it has all disadvantages of Timing based technique, it is a network-based correlation scheme (Yang & Lee, 2008). Therefore, it is out of this study scope (Host-based).

For RTT technique, it is one of the main techniques that used by SSD and provides accurate of SSD. It is a simple but effective SSD which can be used in a real network environment. (Li, 2011; Ping et al., 2010). The time that taken for a packet to travel between the departure source and the final destination host and return again is called, Round Trip Time (RTT). In this concept, the (exporter) departure source is the host that sending the packet and the destination is the host or the system that receiving the packet and retransmits it again. However, RTT is one of several factors affecting latency in the network. The RTT can range from a few milliseconds of time (thousandths of a second) under perfect conditions between very closely spaced two nodes to several seconds under negative conditions between two nodes separated by a long distance.

In fact, several methods have been used in order to match two packets based on the time stamp or RTT, such as (Kuo & Huang, 2008), (Li, Zhou, & Wang, 2010), and (Ni et al., 2008). However, all these studies explained that, the matching is happened if, there are two kinds of connections are closely. In other words, the machining between two packets does not require to be identical hundred percent.

Table 2.3: Characteristics of SSD Techniques

Timing based Technique Characteristics

- i) Large latency and its variation (Zhang & Paxson, 2000)
- ii) Sensitive to the use of countermeasures by the attacker (Wang & Reeves, 2011)
- iii) It can be easily manipulated by intruders (Yang & Lee, 2008).
- iv) In many cases, the timing-based algorithm missed a stepping stone simply because the connections were exceedingly short (Zhang & Paxson, 2000)
- v) It requires that the packets of connections have precise and synchronized timestamps in order to correlate them properly. This makes it difficult or impractical to correlate the measurements taken at different points in the network (Yang & Lee, 2008)
- vi) Only use Send or Echo packets (Ping et al., 2010)
- vii) It is observed that a large number of legitimate stepping-stone users routinely traverse a network for a variety of purposes (Yang & Lee, 2008).

Packet number based Technique Characteristics

- i) Use Send or Echo packets only (Li, 2011)
 - ii) Due to the fact that the upper bound of the number of packets required to monitor is large, while the lower bound of the amount of chaff needed to evade this detection is small (Yang & Lee, 2008).
-

Round Trip Time Technique Characteristics

- i) Use Send and Echo packets together in order to detect stepping stones (Ping et al., 2010).
- ii) Can filter unsymmetrical Internet packets and chaff packets (Ping et al., 2010).
- iii) Can also be more resistant to network imperfections and intruder evasion than any other type of approach (Ping et al., 2010).

Deviation-Based Technique Characteristics

- i) It has all the problems of the Time-Based Approach (Ping et al., 2010).
 - ii) A network-based correlation scheme (Yang & Lee, 2008)
 - iii) Computing deviation is not efficient (Yang & Lee, 2008)
 - iv) It is not applicable for a compressed session because it depends on the size of a packet (Yang & Lee, 2008)
 - v) It cannot correlate connections where padding is added to the payload because it can correlate only the TCP connections that have one-to-one correspondences in their TCP sequence numbers (Yang & Lee, 2008)
-

2.4.6 SSD Models

Based on the location where the analysis takes place, SSD is divided into two types, which are:

- 1- Network based SSD, (NSSD)
- 2- Host based SSD, (HSSD).

Figure 2.9 shows the general classification of SSD. Each type has a special technique for securing information and monitoring, and each type involved special pros and cons.

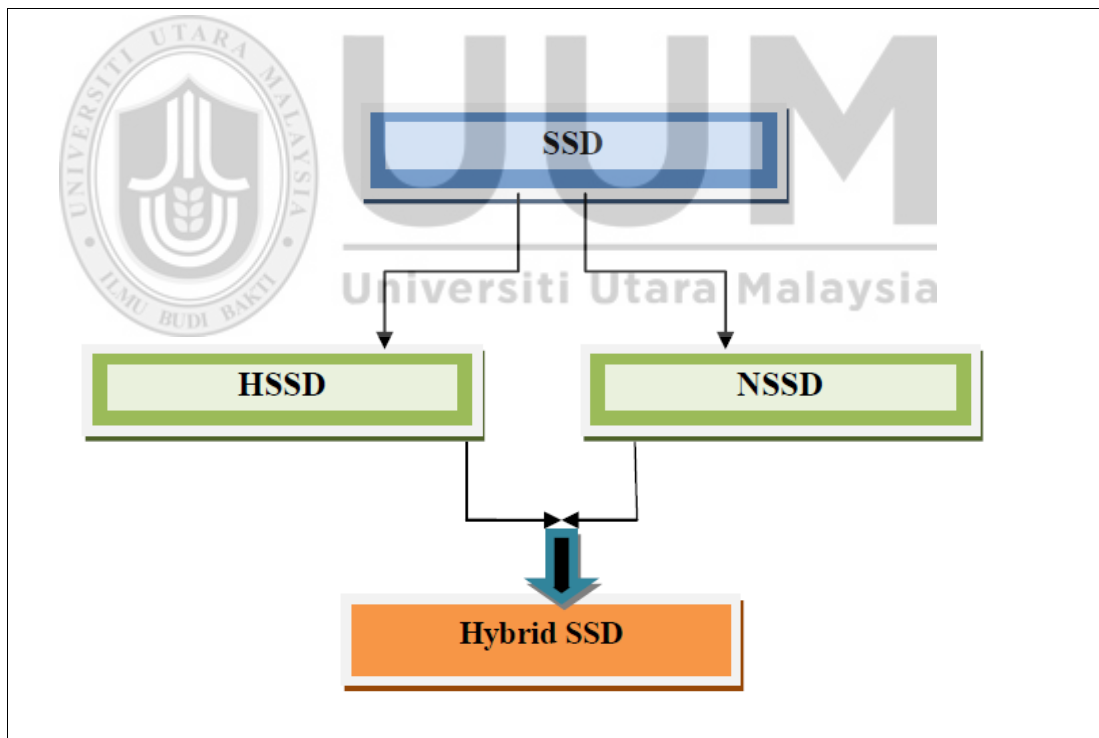


Figure 2.9: General Classification of SSD

2.4.6.1 HSSD Model

HSSD: is called on SSD when the management unit on each host. HSSD monitors the traffic on its workstation by utilizing the host resources to find any intrusion attack. (Sonawane et al., 2012). Figure 2.10 shows the design of HSSD.

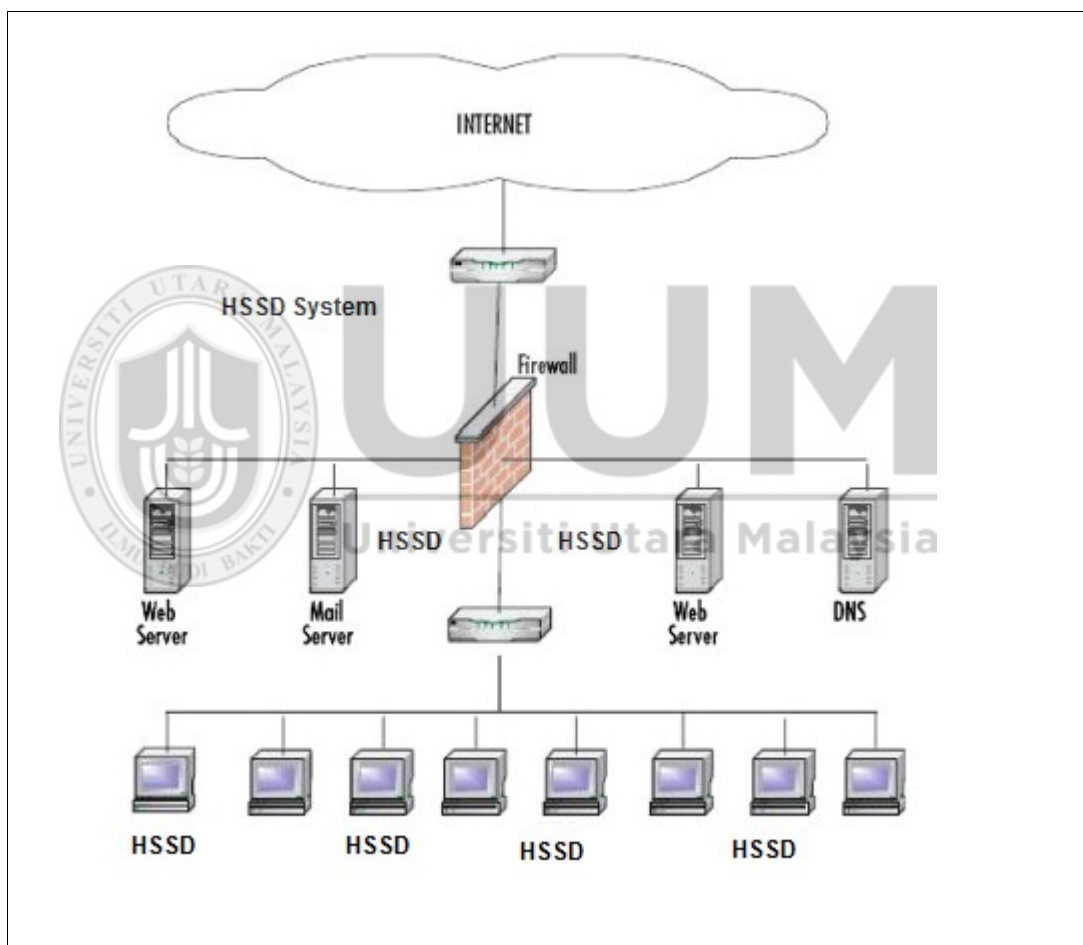


Figure 2.10: SSD Host-based model design

Properties of HSSD:

- i) In HSSD, so long as the backdoor sends a log message, backdoor traffic that is directed at a system will not be missed.
- ii) The HSSD can detect if the backdoor has been successful by testing log messages or other indications on the system
- iii) The HSSD can be used to identify unauthorized

2.4.6.2 NSSD Model:

In this model the management unit as stand-alone devices on all components of network. NSSD checks the traffic on the network to detect intrusion attacks (Sonawane et al., 2012). Figure 2.11 shows the architecture of this model.

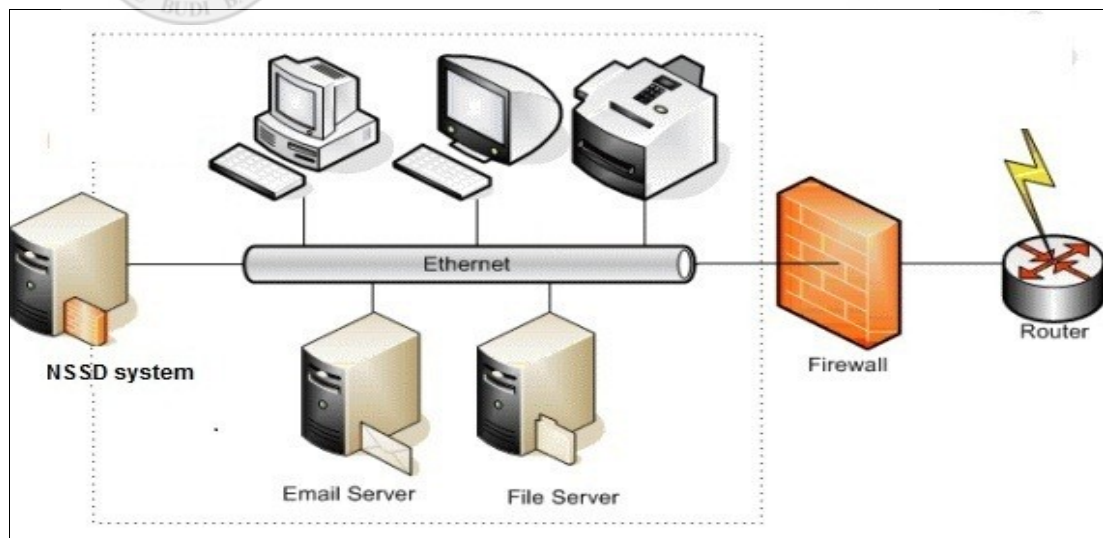


Figure 2.11: NSSD Network-based model design

Properties of NSSD:

- i) We can hide NSSD unit among the network components, so we can monitor the attacker without knowing that.
- ii) NSSD is less costly; by using a single unit in a network we can monitor all traffic.
- iii) The NSSD can detect the contents of all packets that travel to a target system

Sometime, combine both NSSD and HSSD to identify the attacks. In this type both kinds of SSD can be used simultaneously. Hybrid SSD based is called to this type. Table 2.4, illustrates the main reasons for why the host-based have been chosen, and why we did not choose the network-based or Hybrid SSD based.

For the network-based, it helps only to detect the external intrusions; furthermore, it is not easy to detect the intrusion from the encrypted traffic in addition to that, it is not easy to detect the network intrusion in a virtual network. So, what is the result if the backdoor is external has encrypted traffic in the virtual network? In short, it is not suitable for detecting the backdoor attack. While for Hybrid SSD based, in addition to, it has the disadvantages of the NSSD design; it is difficult to understand, in other words; it is complex. Therefore, it is not suitable method for this research also.

Table 2.4: Characteristics of SSD Models

Type	Limitations/Challenges	Characteristics/strengths	Position
HSSD	<ul style="list-style-type: none"> i) Need to install on each host ii) It active (monitor intrusion) only on the host. 	<ul style="list-style-type: none"> i) Detect intrusion by monitoring host's file system, network events or system calls. ii) It does not need to extra hardware 	On each host
NSSD	<ul style="list-style-type: none"> i) Not easy to detect intrusions from encrypted traffic. ii) Not easy to detect network intrusion in a virtual networks. iii) It uses only to detect the external intrusion. 	<ul style="list-style-type: none"> i) Detect intrusions by monitoring network traffic. ii) Use to monitor multiple systems at same time. iii) Need to place only on underlying network. 	In external network
Hybrid SSD based	<ul style="list-style-type: none"> i) New and very difficult to understand. 	<ul style="list-style-type: none"> i) The user able to monitor and analyze communications 	In hypervisor.

2.5 SSD and Backdoor

Nowadays, backdoors are one of the most challenges of network growth, because they involve opening a security hole (or backdoor) on the computer, and then use it by the attacker to gain remote access and to download more viruses (Shabtai, Kanonov, Elovici, Glezer, & Weiss, 2012). Figure 2.12, shows a backdoor working method.

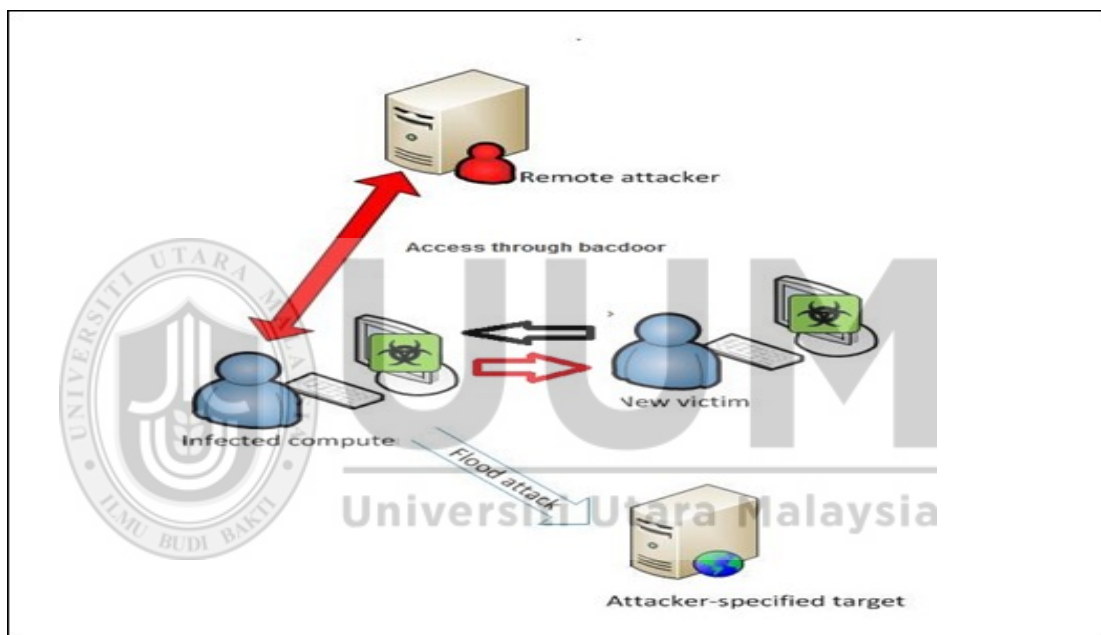


Figure 2.12: Backdoor Attack Traffic

The problem of backdoor attack detection has been appeared when the types of viruses, (including the backdoor) increased dramatically. The problem was on, to find the connection of the backdoors among a stream of authorized network traffic initially appears very difficult. However, the interactive traffic has specific features quite different from most machines driven traffics (longer idle periods, smaller packet sizes). Therefore, it is possible to find efficiently for such traffic.

In 2000, a general algorithm was proposed by Zhang and Paxson, for detecting the interactive of the backdoor. The main concept of this theory is developed a group of algorithms to detect several kinds of interactive connections.

According to Zhang and Paxson (2000), these algorithms are possible to implement to a connection flows and whenever they are detecting interactive connections that using a port that is not a standard. The main idea was firstly, by looking for small packets (because most backdoors commands are short, the size packet, they used “20 bytes” to define “small” packets) which have large interval time (fall between 10 mSec and 2 Sec). Then by looking for frequent for small size packets. For directionality key, they only consider the traffic that sent by the starter of a connection. In other words, this algorithm reflects; the risk of attack is coming with any flows consists of less than 8 packets or less than 2 seconds where a flow has one direction. This tool of the detection algorithms is clear and convincing (Paxson & Zhang, 2000).

As conclusion, the study of Paxson and Zhang used the signature of the interactive traffic. Our research considers the same issue, using the interactive traffic in order to detect the backdoor. But, by using deferent technique based on the simple concepts that taken from SSD based-research, and does not use the traffic signature method that used by Paxson and Zhang.

2.6 Summary

This chapter illustrated a background and other related works to this research (Backdoor Attack Detection Based on Stepping Stone Detection Approach). The chapter identified the requirements of the SSD approach techniques that can be used in detecting the backdoor attack, and justified why choosing Round Trip Time based and packets number based as suitable techniques.

This chapter described the concepts of SSD detecting method, and explained how SSD can be used for detecting the backdoor attack. Moreover, this chapter also illustrated why choosing a host-based architecture remains the best option. Furthermore, it justified why a new approach has to be proposed when various other approaches exist, such as eht signature-based and anomaly-based detection approaches.



CHAPTER THREE

RESEARCH METHODOLOGY

3.1 Introduction

Section 3.2 begins with a discussion of the operational framework. Section 3.3 presents the research design. Section 3.4 shows the sources of the data for this research. Section 3.5 concentrates on the techniques that can be used to gather data. Section 3.6 explains the method of data analysis. Section 3.7 explains the method of evaluation. Finally, Section 3.8 describes the tools used in this study. The purpose of this research is to solve the problem of backdoor attacks through the use of concepts related to the SSD approach.

3.2 Operational Framework

The operational framework of this research is illustrated in Figure 3.1. In general, the framework is divided into three sections. The first section (preliminary framework) includes the search for a suitable technique that can be used in the SSD approach to detect the backdoor attack based on a review of the literature and related works.

The second section of the framework illustrates how we can use SSD techniques in the new approach in real environments. In the third section, an actual test is conducted to produce the required results. Afterward, the analysis and results are evaluated. This study adopts methodologies similar to those in previous studies, such as those of (Prasad et al., 2013) and (Omar, 2011).

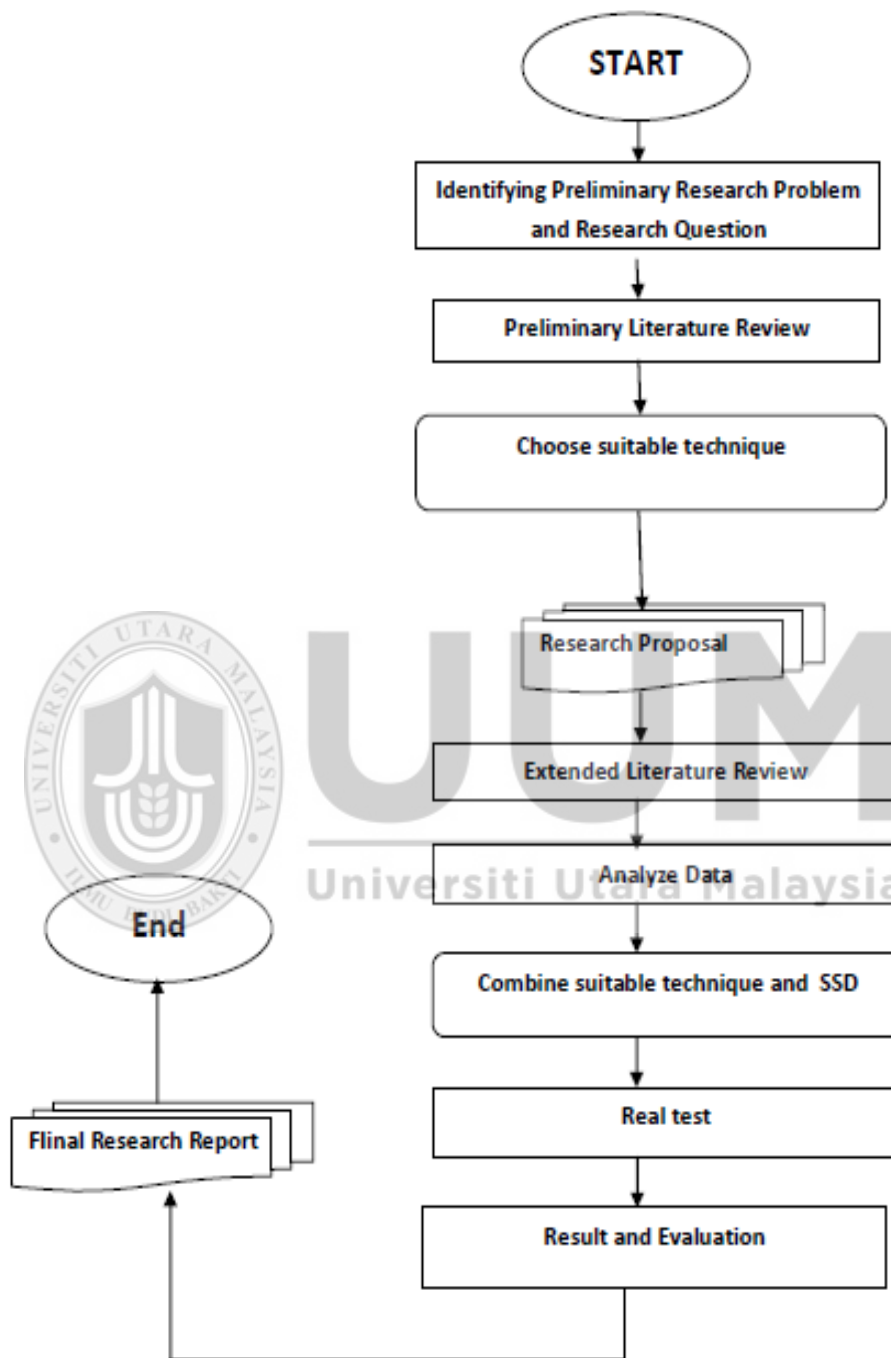


Figure 3.1: Operational Framework

3.3 Research Design

As mentioned in the second chapter, SSD is a flexible system that is not difficult to develop. Figure 3.2 shows all the independent and dependent variables and other attributes of the proposed research. The evaluation of the proposed approach depends on three variables: True Positive Rates (TPR), False Positive Rate (FPR) and Scan Process Time.

All the attributes and variables in this proposed research are illustrated in Table 3.1.

Table 3.1: The relation between attributes and variables

Attributes	Variables
Inp	Input
TPR	True Positive Rates
FPR	False Positive Rate
O	Output
Scan Process Time	Scan Process Time

Adopted from (Omar, 2011)

Figure 3.2 also shows how the proposed approach is measured based on the variables (FPR, TFR and Scan Process Time) and the same variables to other related approaches (antivirus and IDS). The result of O (Output) will be analyzed in percentages and compared with those other related approaches.

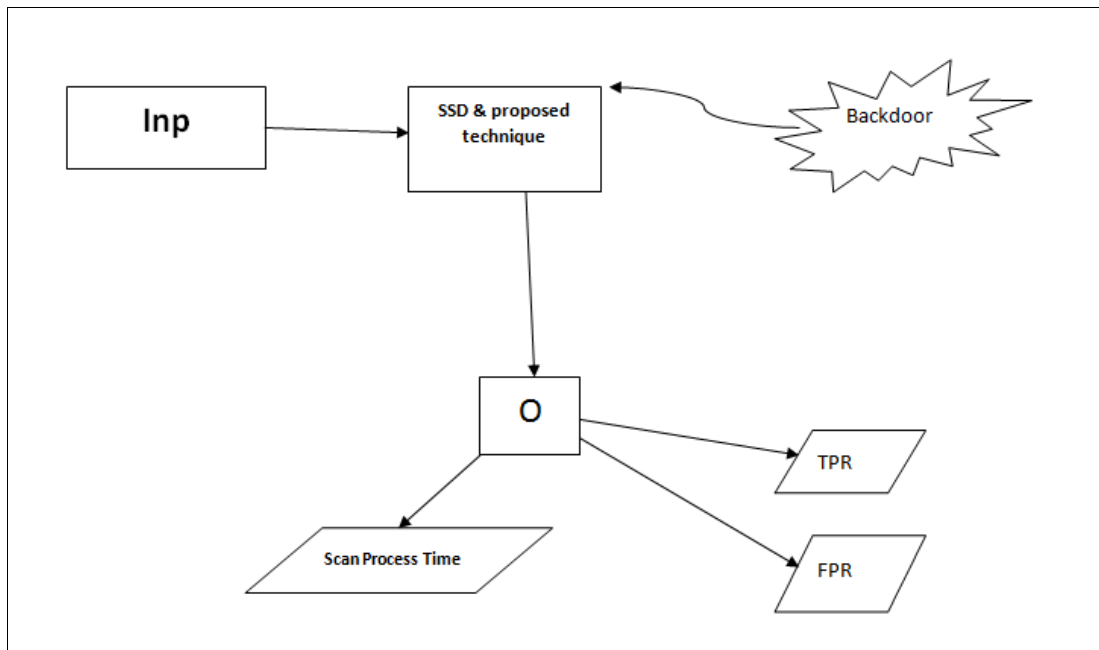


Figure 3.2: The relationship between variables and attributes

3.4 Subject and Information Sources

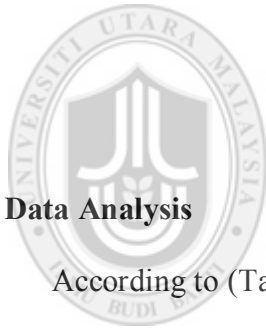
This study involves research on both SSD and backdoor detection. It uses documented sources, such as journals, books, and conferences. A crucial part of this research depends on an experiment designed to prove the results. The sources of the data include books, magazines, and so on. The Internet is also used to obtain actual information.

3.5 Experimental Process and Data Gathering

Most of the research results are based on the experiment. Therefore, in gathering the data, this study employs the following steps:

- 1- A real backdoor attack is used by manually injecting ten different types of backdoor attack into clean machines installed individually.
- 2- The SSD approach is used to detect all ten backdoor attacks one by one.
- 3- The detection rates (TPR) and miss-detection rate (FPR) are captured.
- 4- Steps (1) and (2) are repeated using the antivirus and IDS

Notably, this method is similar to that of Borders, Zhao, & Prakash (2006).



UUM

3.6 Data Analysis

According to (Tahan, Rokach, & Shahar, 2012), TRP, FPR, and scan process time are the main parameters in analyzing the data. TPR is the detection rate and FPR is the miss-detection rate. A high TPR and low scan process time (speed) prove the success of the proposed approach. Therefore, we compared the TRP, FRP, and scan process time of the proposed approach with those of the antivirus and IDS for evaluation.

3.7 Evaluation

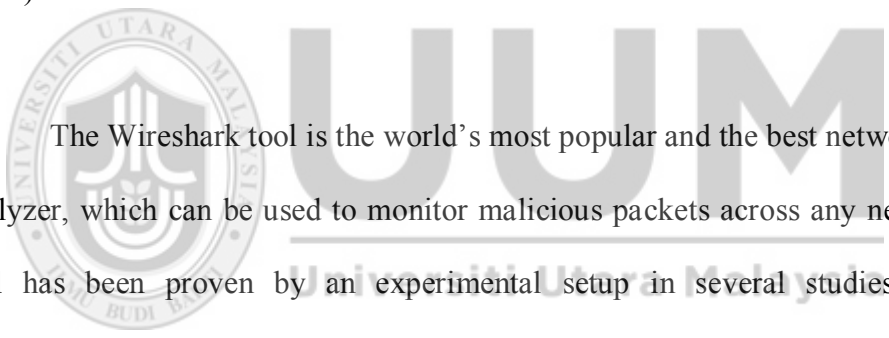
To evaluate the results of SSD-based backdoor attack detection, the output of this technique and the output of the antivirus and IDS were compared. Therefore, the comparison hinges on the three variables of TRP, FPR, and scan process time for both the proposed approach and the antivirus and IDS. The study of (Tahan et al., 2012) lists the performance measurements as follows:

- 1) TP : True Positive
- 2) FP : False Positive
- 3) TN : True Negative
- 4) FN : False Negative
- 5) $N = (FP + TN)$
- 6) $P = (TP + FN)$
- 7) $FPR = FP / (FP + TN)$: (False Positive Rate)
- 8) $TPR = TP / (TP + FN)$: (True Positive Rate)
- 9) $ACC = (TP + TN) / (P + N)$: (Accuracy)
- 10) $PPV = TP / (TP + FP)$: (Positive Predictive Value)
- 11) $NPV = TN / (TN + FN)$: (Negative Predictive Value)
- 12) $BER = 0.5 (FN/P + FP/N)$: (Balanced Error Rate)

3.8 Tools

The tools for this proposed research are:

- i) Hardware: LAN network includes the hosts (computers), cables and hub switches.
- ii) Software: Windows operating system as the operating system and Wireshark tool (W. Foundation, 2013), which can be used to detect intruder packets across any network (Agrawal et al., 2010).
- iii) Ten different types of backdoor files. Backdoor files will be gathered from the Internet source.
- iv) Antivirus and IDS.



The Wireshark tool is the world's most popular and the best network protocol analyzer, which can be used to monitor malicious packets across any network. This tool has been proven by an experimental setup in several studies (Banerjee, Vashishtha, & Saxena, 2010). Furthermore, this tool has been used by many studies similar to ours, such as those by (W. S. Choi & Choi, 2013), (Soni, 2013) and (Mohan, 2013).

CHAPTER FOUR

SAMPLING AND EXPERIMENTAL SETUP

This chapter describes how the backdoor samples were gathered and how the experiment process was conducted in relation to the research objectives. Section 4.1 illustrates how the samples were gathered. Section 4.2 describes the materials, method, and network architectures used in the test bed. Section 4.3 outlines the steps of experimental method. Section 4.4 concludes the chapter.

4.1 Sampling

Gathering different types of backdoor files (servers) from Internet sources, such as (G. T. I. S. Center, n.d.), (NETRESEC, 2010) and (Mila, 2013). However, obtaining the client's software tools used by the attacker in obtaining remote access is not easy because all the hackers are distributed across the servers throughout the Internet. But, they keep the client's software tools to themselves. In fact, the client's software tool is a software written by the hacker to obtain remote access and edit the backdoor server(s) or generate new server(s). Therefore, the client's software is a basic tool needed to determine the source host, the destination host, the port number, and the period of time between the sent packets. In the same way, to evade the antivirus, hackers use different tools to make the backdoor server unique or to encrypt it. This research uses two types of backdoor attack samples.

1. Known backdoor samples

First, we started with a study of how hackers use client's software tools to generate a new backdoor attack and how they obtain remote access. The backdoor attacks generated by these tools are used as the first type of sample, most of which can be detected by one or more antivirus tools. Table 4.1 illustrates the client's software tools that can be used to create the backdoor servers and Figures 4.1 and 4.2 show examples to the interfaces of the client's software.

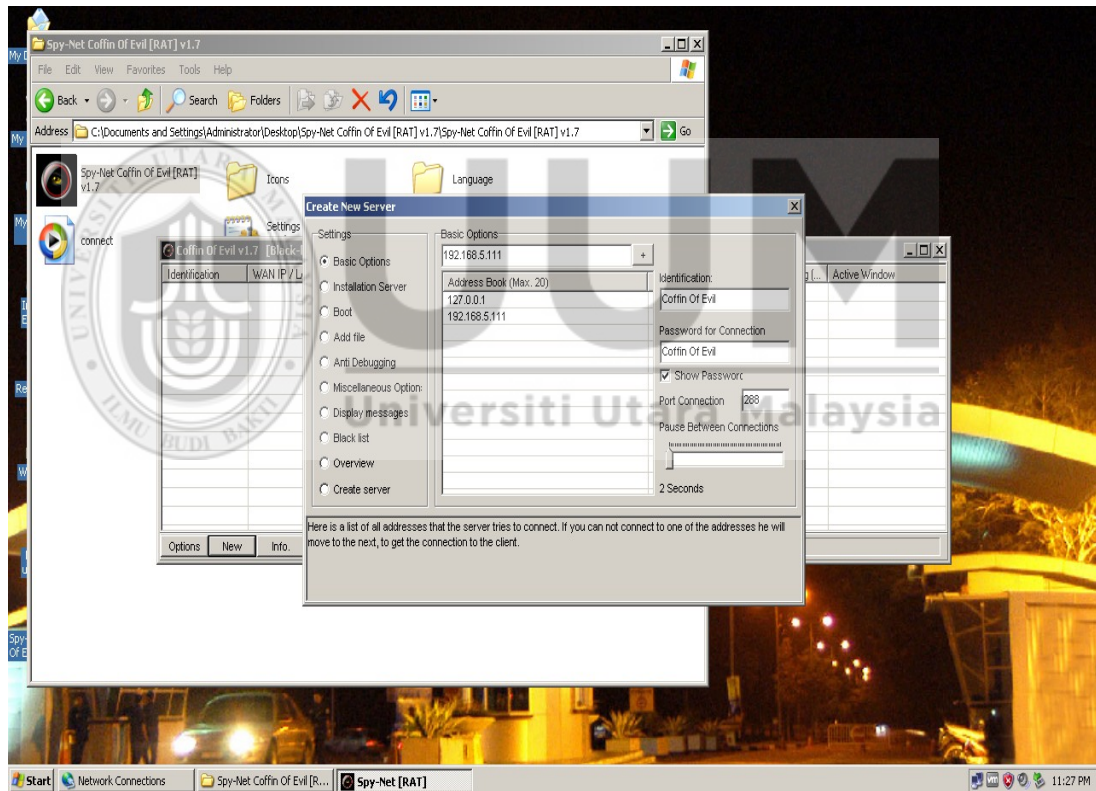


Figure 4.1: The interface of Spy Net Client's software

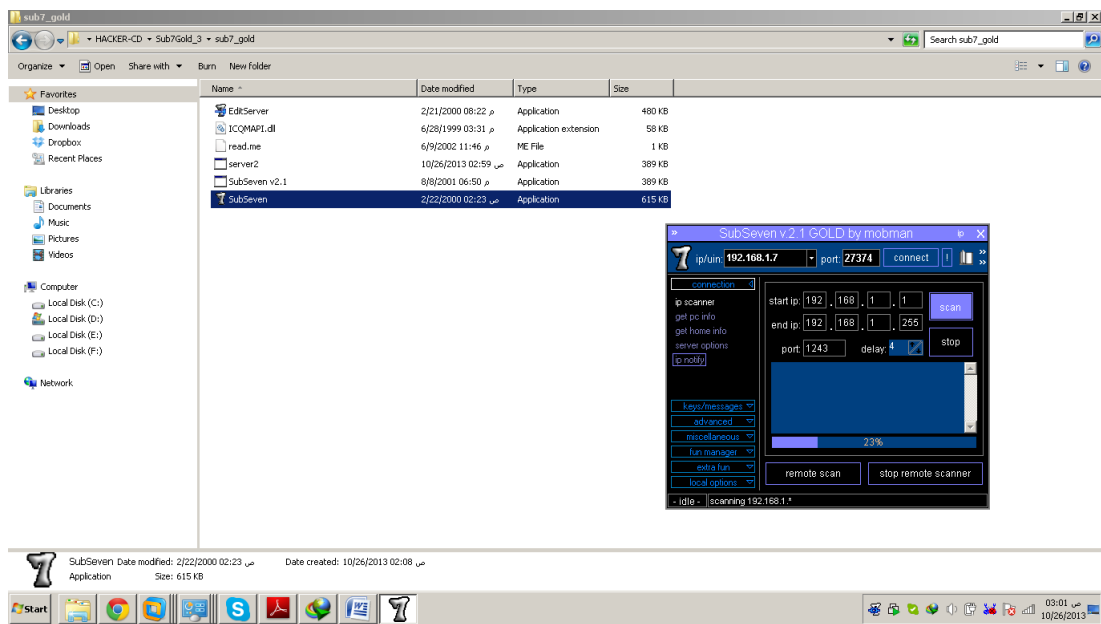


Figure 4.2 : The interface of Sub7 Gold client's software

Table 4.1: Client's software tools that were used to create backdoor server

Client's software	Backdoor type	Source of tool
Bifrost Coffin Of Evil 1.2.1d	Executable file	www.dev-point.com
Bifrost un_pack	Executable file	www.dev-point.com
SpY-NeT v2.6 ArA	Executable file	www.dev-point.com
Njrat041afixed	Executable file	www.dev-point.com
XtremeRATv2.9_2	Executable file	www.dev-point.com
Sub7Gold_3	Executable file	http://www.law-uni.net
DarkCometRAT33FWB	Executable file	www.dev-point.com
Bozok 1.1	Executable file	http://ss-rat.blogspot.com
PI2.3.2	Executable file	http://www.poisonyivy-rat.com
Poison Ivy 2.3.2.AR	Executable file	www.dev-point.com
njRAT-v0.6.4	Executable file	http://garabezy.blogspot.com

2. Unique and an encrypted backdoor samples

In addition to the client's software, we used extra tools to encrypt or to make the server a unique backdoor attack to test the SSD techniques with the encrypted and the unique backdoor attack. Figure 4.3 shows the tools that can be used to encrypt and make new samples and Figure 4.4 shows the interface to one of these software tools.

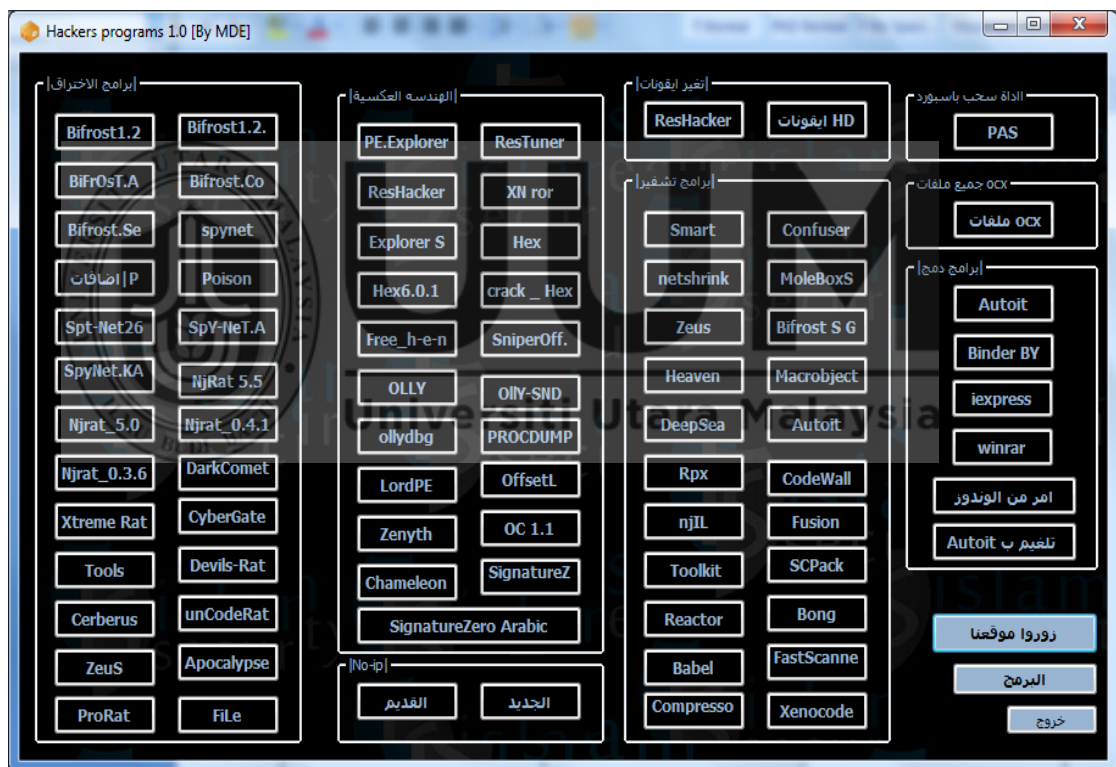


Figure 4.3: The tools that can be used to encrypt and make new samples

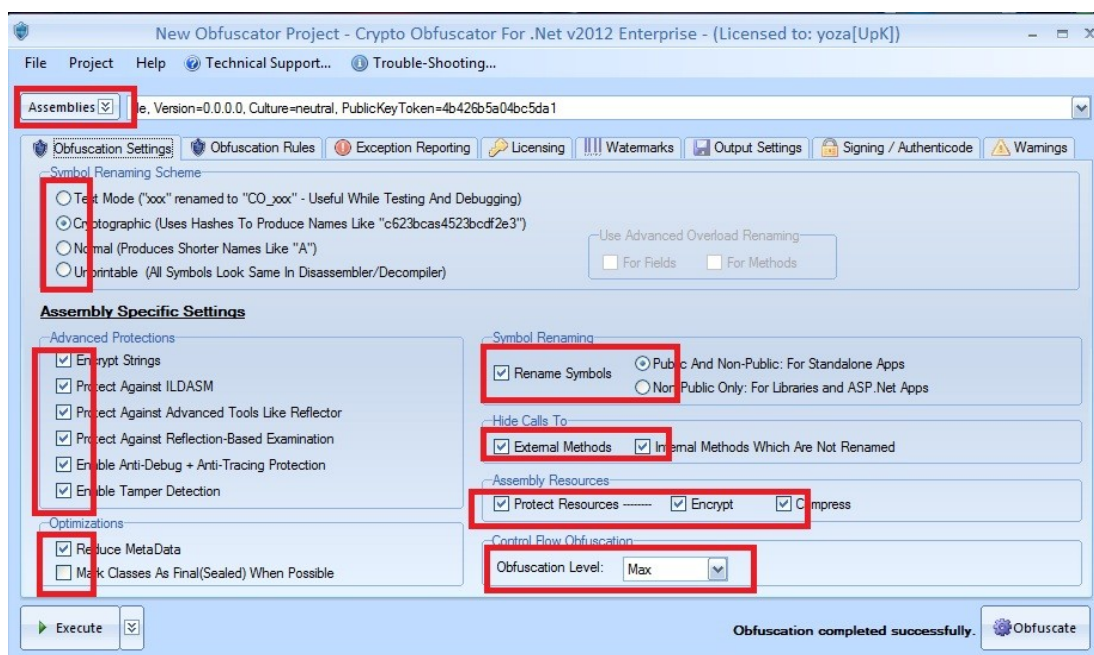


Figure 4.4: The interface to one of the encryption tools

To prove that the samples are backdoors, we tested them through the Virustotal website service (Virustotal, 2013), which uses more than 45 antivirus programs in the scanning process. Furthermore, we ran an extra test by using Avira, Avast, and Eset1 antivirus tools, which were individually installed in the victim host. Figure 4.5 shows the test results for the sample UUM_Backdor before the encryption. The resultant detection ratio is 11/48. Figure 4.6 shows the test result for the same sample after the encryption by the same tool, the resultant detection ratio is 0/48. Figure 4.7 shows the test result for the same sample by Eset Smart Security 6 antivirus, the result of detection ratio is 0/1. The sample (UUM_Backdor) is generated by Njrat v.05 tool

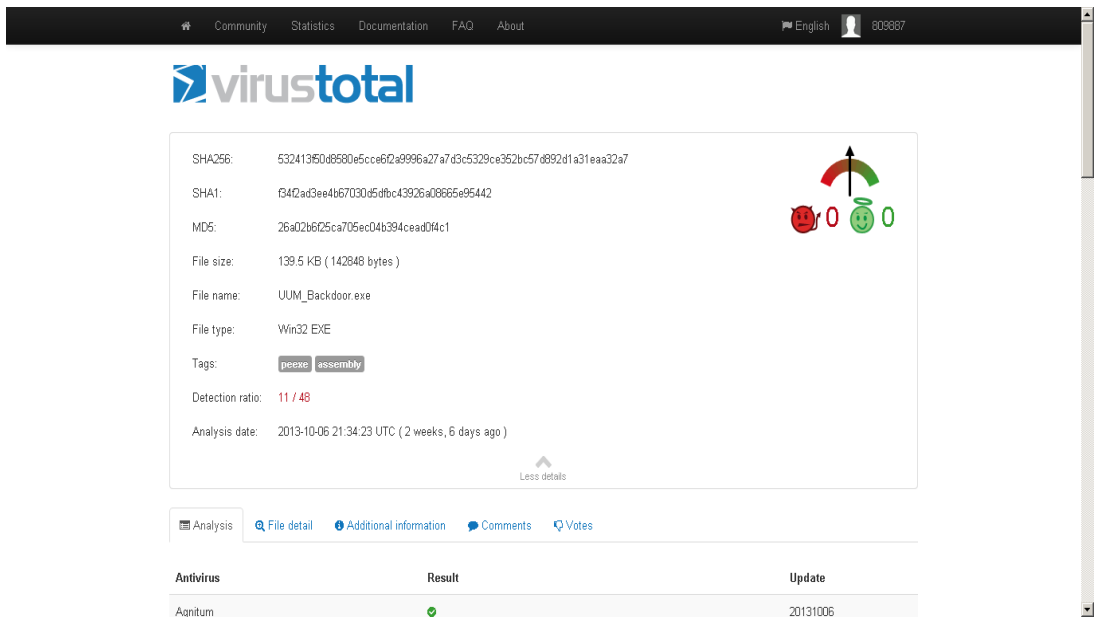


Figure 4.5: Test result for the sample UUM_Backdoor before the encryption



Figure 4.6: Test result for the sample (UUM_Backdoor) after the encryption

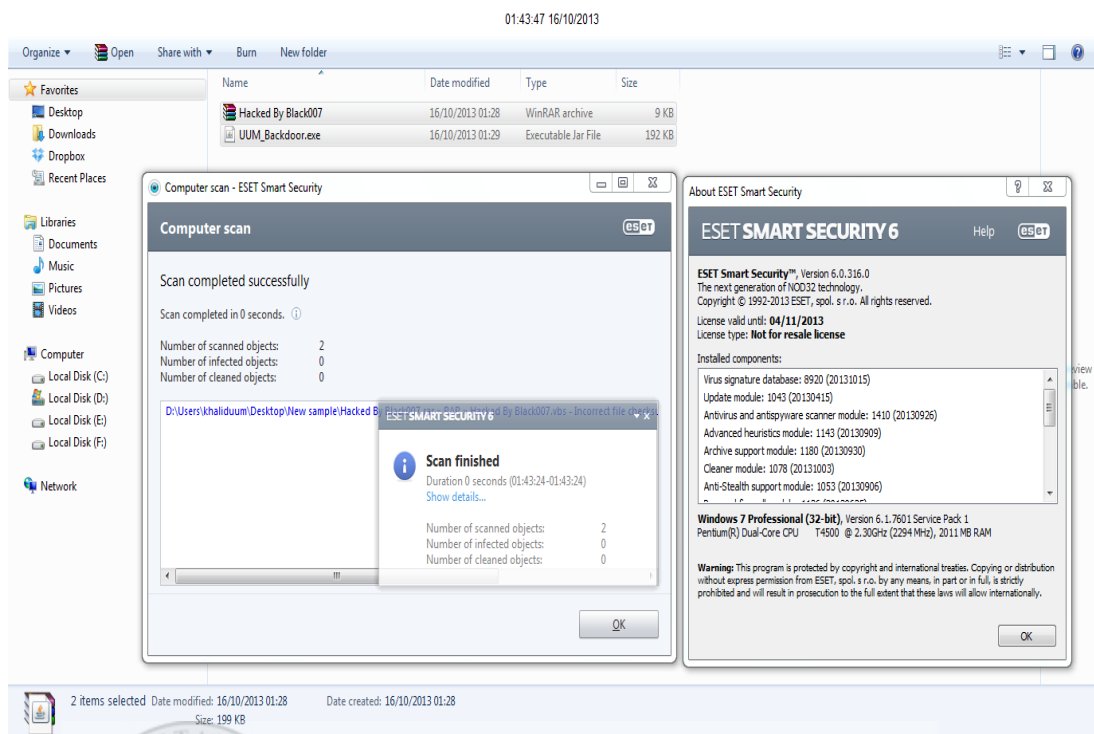


Figure 4.7: Eset Smart Security 6 test result for the sample after the encryption.

The three steps above prove in a practical manner our research problem, “All antivirus tools are not enough to face the backdoor attack.” Therefore, the zero-attacks daily and false positives are the most challenging problems in the field of backdoor detection. Furthermore, the three previous steps illustrate how we can obtain new known or unique backdoor attack samples. In addition to these samples, we gathered some dataset (Wiresharke Pcap files) from Internet sources, such as (Mila, 2013). Table 4.2 shows the samples and dataset (Wiresharke Pcap files), that used in this research.

Table 4.2: Some of samples and dataset that used in this research

Sample name		Sample name	
1	Njrat v.604 server	13	BIN_8202 /dataset
2	Coffin Of Evil-Server	14	PlugX/dataset
3	Cyper-GAT server	15	Taidoor/dataset
4	Bozok1.1 server	16	Enfal_Lurid/dataset
5	Backdoor.Win32.Agent.rb	17	LURK/dataset
6	SpY-NeT v2.6 server	18	DNSWatch_protux/dataset
7	Bifrost un_pack server	19	Mediana/dataset
8	XtremeRAT server	20	TrojaPage/dataset
9	Njrat v.401 server	21	LetsGo_yahoosb/dataset
10	Poison server	22	RTF_Mongall_Dropper_Cve/dataset
11	SubSeven v2.1R server	23	TrojanCookies/dataset
12	IXESHE/dataset	24	Gh0st-gif/dataset

4.2 Materials and Experiment Setup

As described in the Chapter Three (Methodology), the experiment has been run with four steps in (LAN) as a controlled environment. For example, the first testbed was done by using LAN without Internet (Offline Design) as shown in Figure 4.8, by setup the backdoor client's software (attacker) in the host (B), (192.168.5.45) and the backdoor server (UUM_Bacdoor.exe) on the host (A), (192.168.5.46). The Source Port: (1177), Destination Port: (52361).

Figure 4.9, shows the backdoor client's software (attacker) that used in this architecture. To capture the packets between the attacker and the victim host, we run Wireshark tool in both sides.

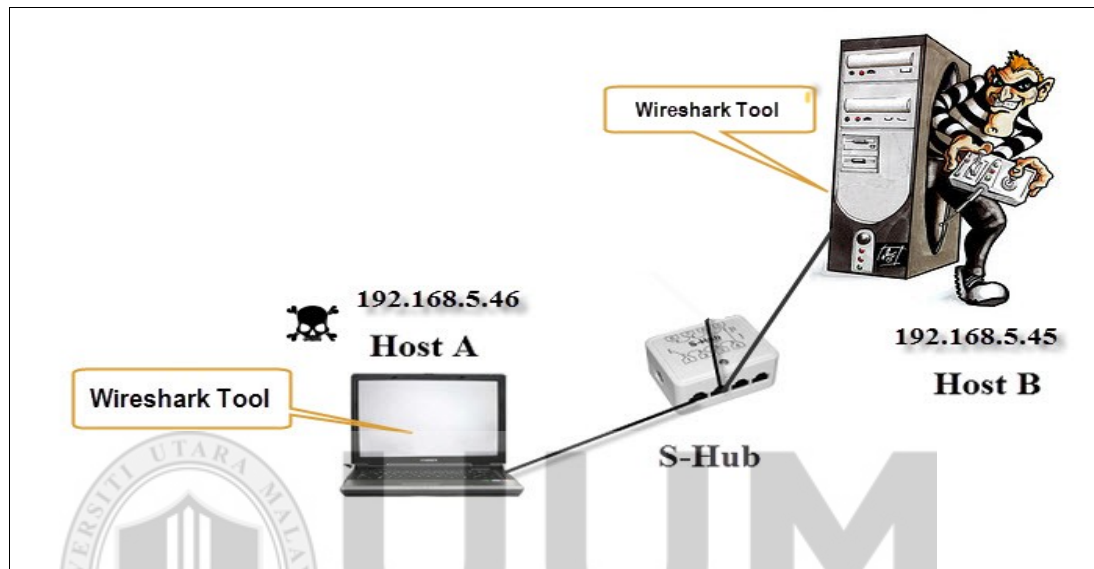


Figure 4.8: Network Topology used for Offline Design testbed

Name	IP	PC	User	Install Date	Country	OS	Cam	Ver.	Ping	Active Window
HacKed_E2230F4	192.168.5.46	BOOGY-PC	boogy	2013-07-20	IRQ	Win 7 Ultimate SP1 x86	Yes	0.5.0E	2ms	http://www1.de

Name	HacKed_E2230F4
Host	192.168.5.45:1177
Dir	%TEMP%
Exe	backdoor_uum.exe

Figure 4.9: Backdoor's client (attacker) software that used offline design

The second design is used because some of the backdoors are not connecting directly to the attacker, and they use an intermediate online server between them, therefore we used the online design testbed as shown in Figure 4.10 for this specific case. For example, the experiment, that run with (Xtreem) backdoor sample. To study this case from both sides, we run the Wireshark tool in the both sides. However, there is no activity between the attacker and the intermediate online server, because the information was sent to the email that related to the online server. In fact, this design is quite similar to the environment of the backdoor activity over the Internet environment. Usually in the Internet network situation is not possible to capture the packets on the attacker side.

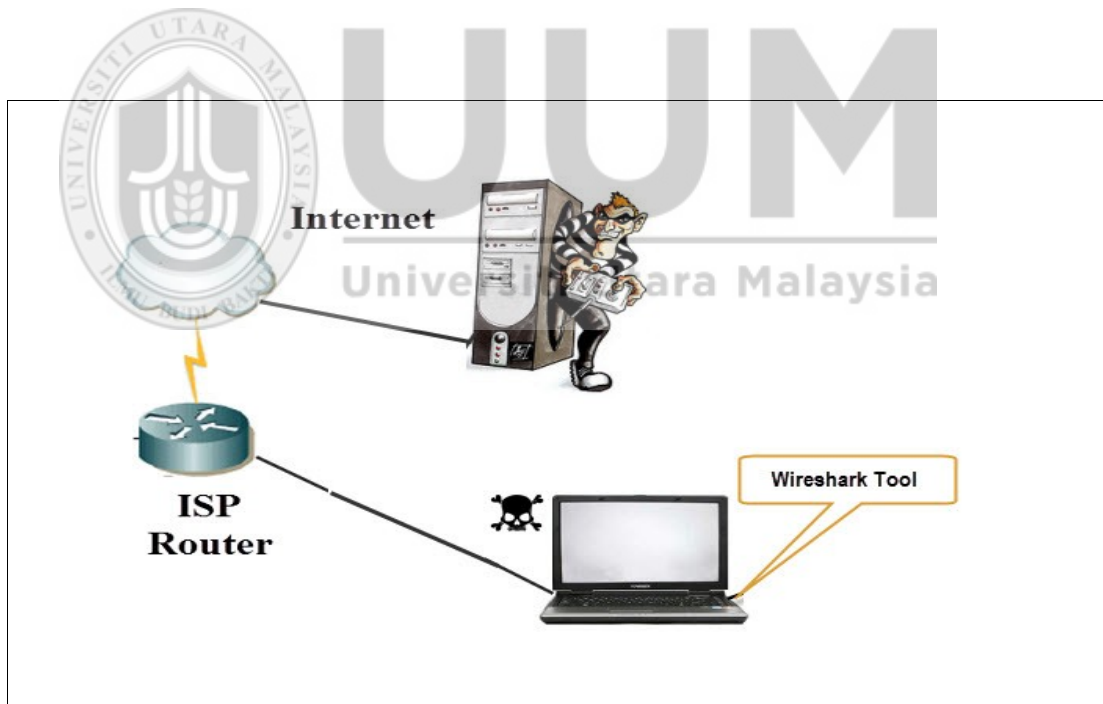


Figure 4.10: Network Topology used for Online Design testbed

4.3 Challenges and Solutions

A real environment was used in the experiment: two computers, a hub switch, and cables to make a connection between the hosts. However, many challenges beset this method. To remove the backdoor files and to ensure the host is cleaned, we had to reformat the victim host after each experiment. As a result, the method took a long time for each experiment. Therefore, we used the virtual machine software (VMware. Inc, 2013), as the environment in which we ran our experiments. To verify this tool, we started by repeating the first experiment conducted in a real environment for the same sample with the same setting of the network service. The results are similar for both types as shown in Figures 4.11 and 4.12.

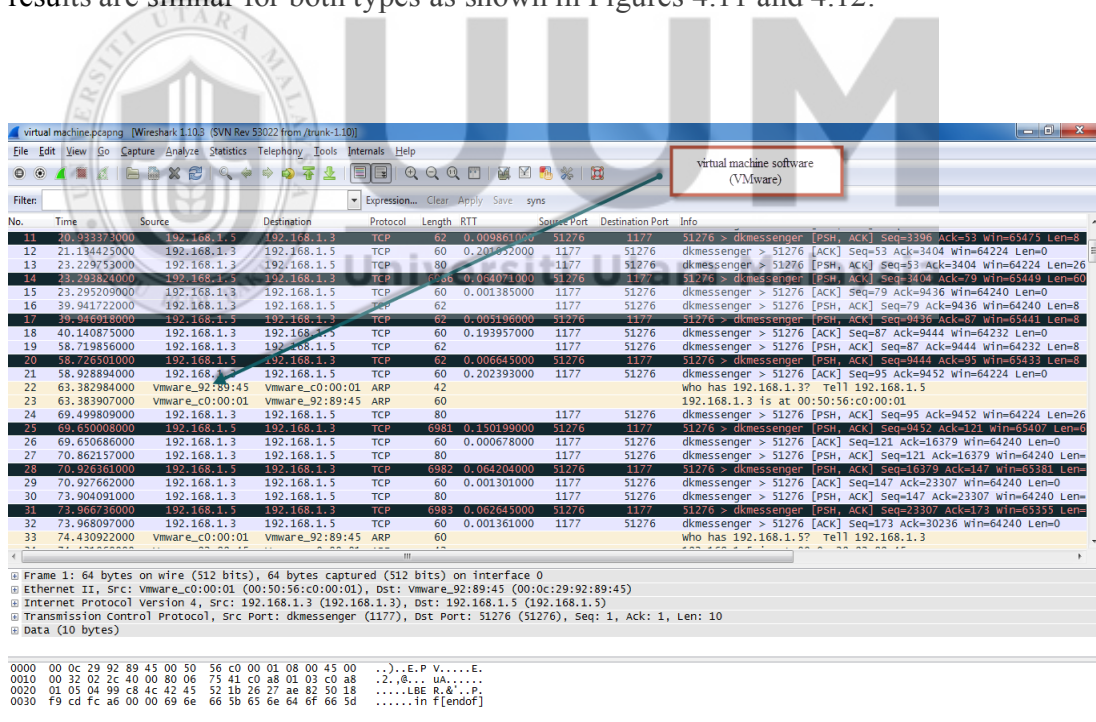


Figure 4.11: UUM Backdoor sample in virtual machine software environment

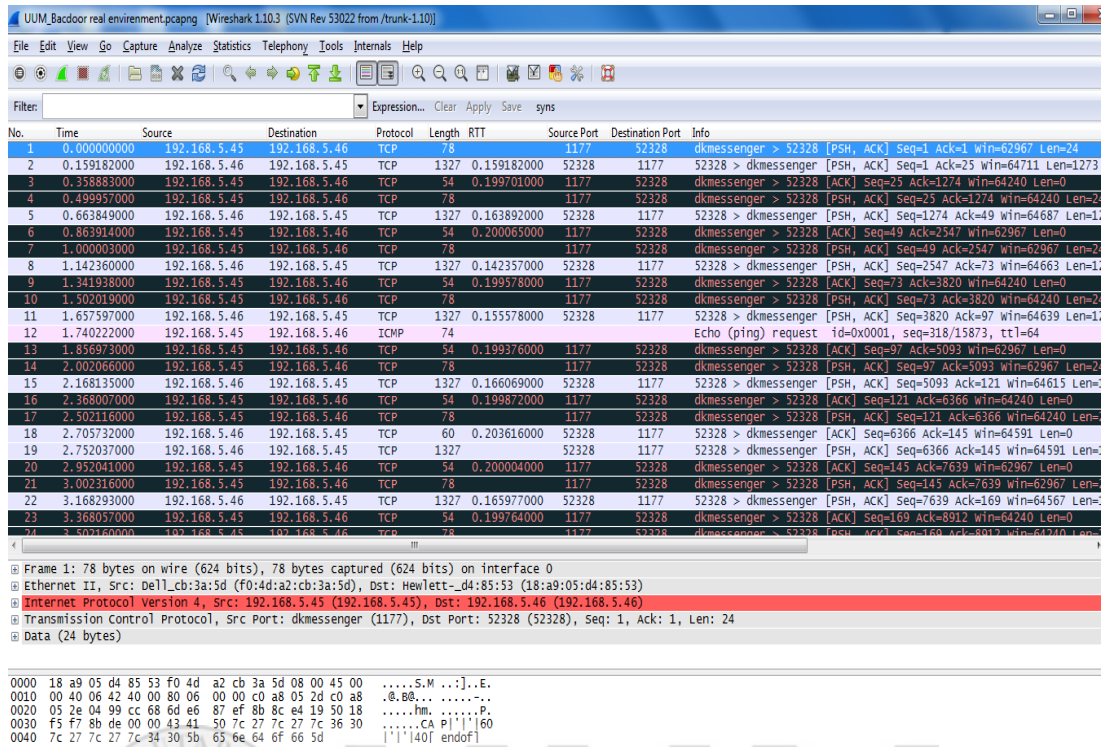


Figure 4.12: UUM_Backdoor sample in real environment

Furthermore, the virtual machine software environment (VMware) was also used in similar works, such as those by (Borders et al., 2006), (Crawford & Peterson, 2013) and (Gribble, Levy, Moshchuk, & Bragin, 2013). Figure 4.13 shows the Virtual Machine software environment.

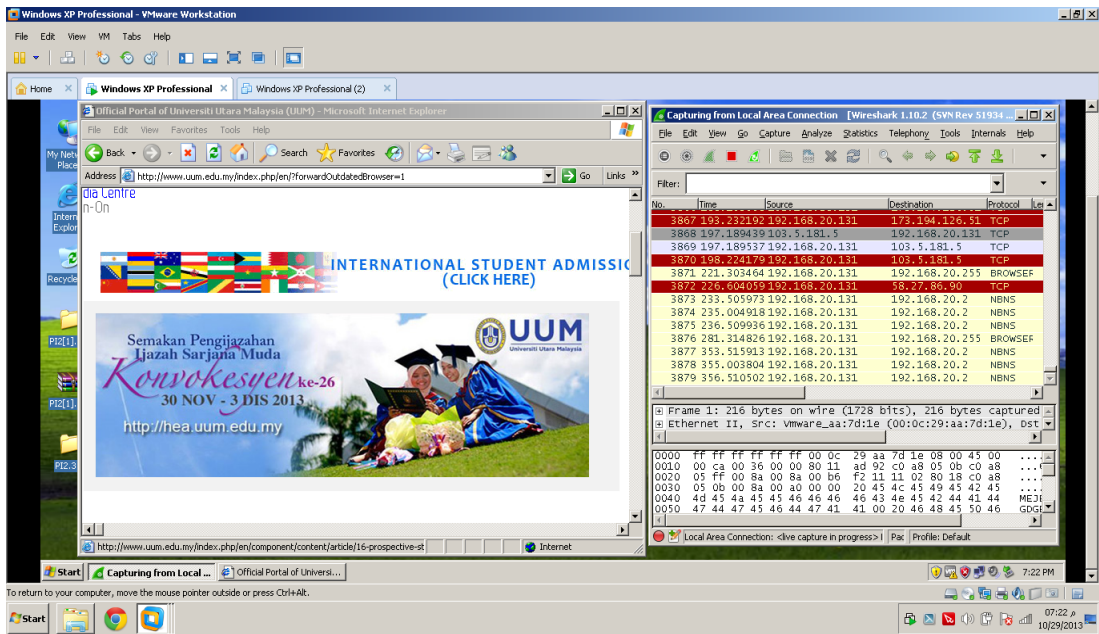


Figure 4.13: Virtual Machine software environment

After each experiment in the virtual machine software, we used the snapshot restoring option to make sure that the host victim reverts to the first statement of the operating system (clean). This task is very easy and very fast, as shown in Figure 4.14.

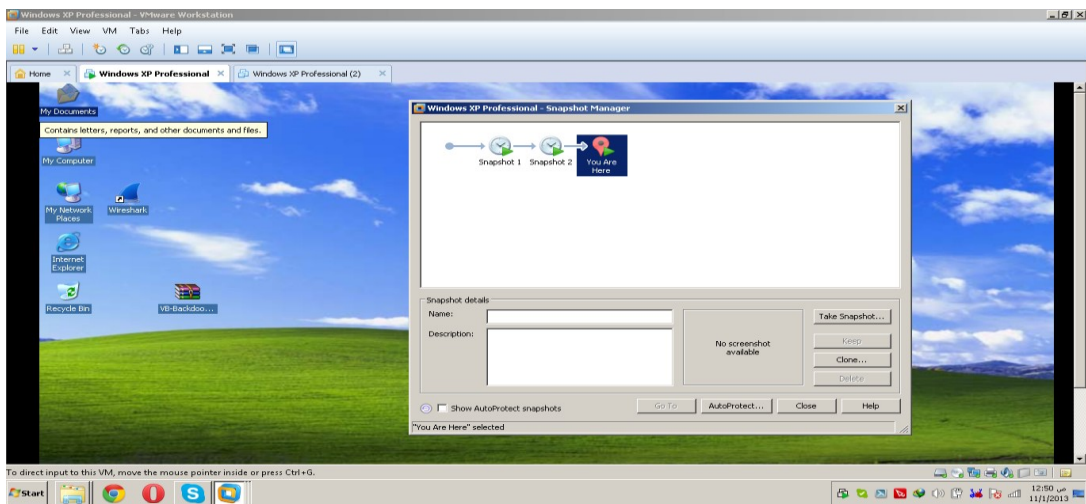


Figure 4.14: System restore method in Virtual Machine software

4.4 Experiment Steps

In details, the experiment is involved the following four steps:

- 1- The backdoor samples (server) generated by the client's software tools or gathered from the Internet source.
- 2- The pre test was done for all samples one by one by using Virustotal service. For some of these samples, the same sample was encrypted or changed by the encrypted software tool to make it as a unique or encrypted sample. Then, the second test was run with the new sample by using the same Virustotal service.
- 3- Extra testing was done by antivirus tools (Avg, Avira and Eset smart security 6 tools) that installed individually on the host as shown in Figure 4.15.

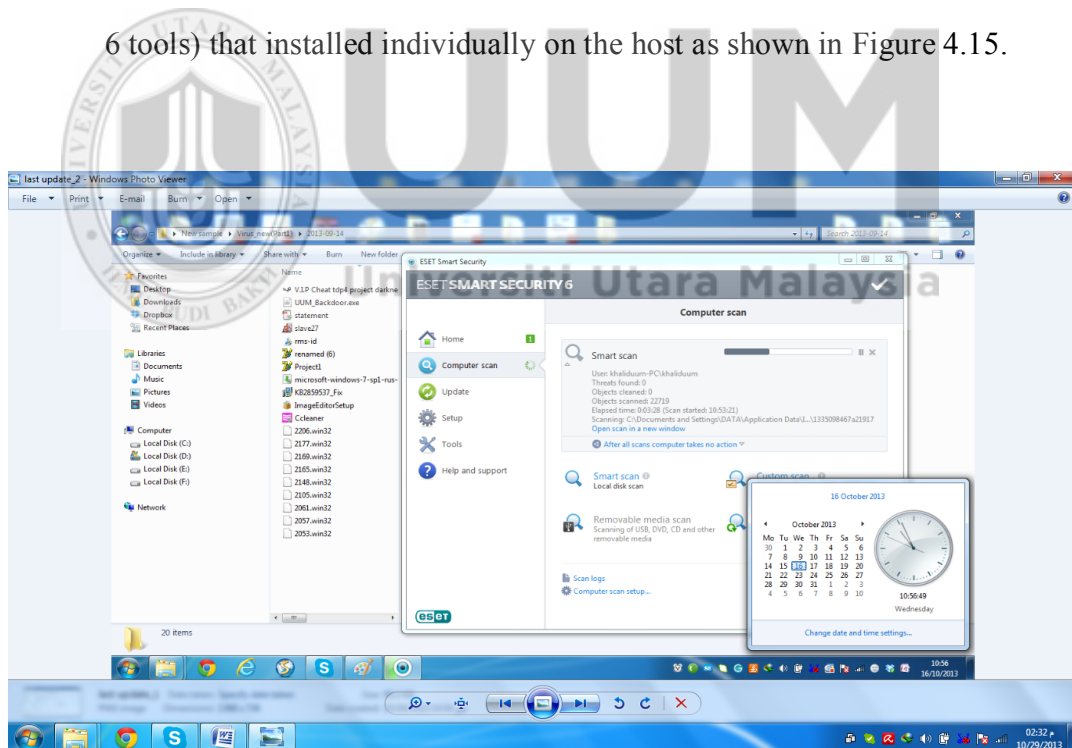


Figure 4.15: Eset Smart Security 6 tool process.

- 4- The next step was, injecting the backdoor sample individually in the host, and run Wireshark tool to capture the incoming and outgoing network packets on the monitored hosts for TCP sessions as shown in Figure 4.16. The Wireshark tool captured all the information that will be used as to provide next processes.

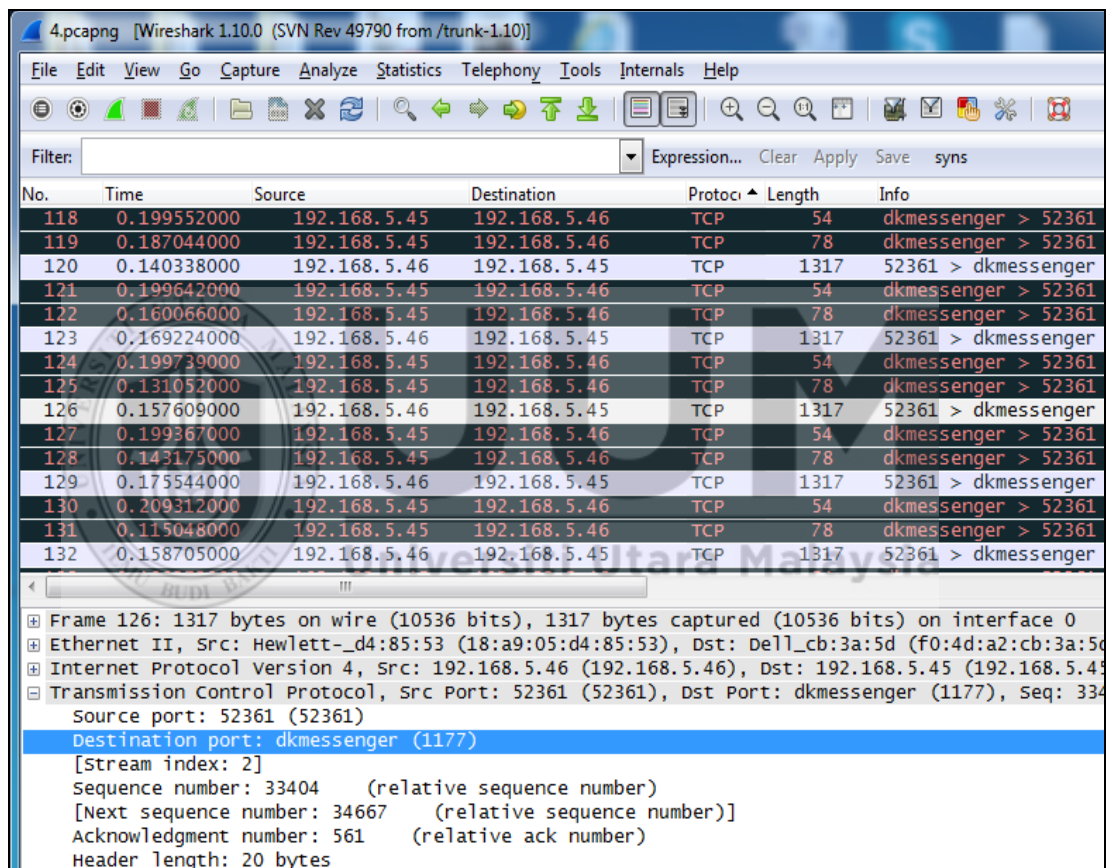


Figure 4.16: Using Wireshark tool to capture the network packets

4.5 Summary

This chapter describes how the samples were gathered and how the experiment process was conducted in relation to the research objectives. The network architectures used in the test bed and the steps of experimental method were also described. All the tools that were used in the sampling and experiment method were stated. Both two types, real environment and virtual environment have been used in the experiments implementation. The verification of virtual environment has done before use it. The Wireshark tool was used to capture the traffic as Pcap files that contain all the information, which will be used in the succeeding analysis. Most of the Pcap captures used in this research as samples are attached in appendix A.



CHAPTER FIVE

RESEARCH FINDINGS AND DISCUSSION

5.1 Introduction

As described at the beginning of this dissertation, false positive and false negative rates are the most challenging problems confronting the field of backdoor attack detection. Furthermore, all the techniques of the signature-based and anomaly-based approaches, which are used to address the backdoor attack problem, are not powerful enough when faced with unknown backdoor, obfuscation, or polymorphic viruses. As a result, the overall rate of accuracy of such techniques is low. This study intends to overcome the backdoor attack problem by using stepping stone detection (SSD) concepts. The previous chapters described how the backdoor attack problem can be detected using the concepts related to the SSD approach. The purpose of this chapter is to evaluate the capability of the proposed approach.

5.2 Data Analysis

The second chapter pointed out that many techniques have been proposed to solve the stepping stone problem. The four main techniques that are widely used in detecting the stepping stone problem include the round trip time (RTT)-based techniques, the timing-based, the deviation-based, and the packet number-based techniques. All the above four techniques are involved in the SDD approach. However, as we explained in Chapter Two, for the timing-based technique, more

than six concern reasons make it unsuitable in detecting backdoor attacks. Moreover, the deviation-based technique has all the disadvantages of the timing-based technique and is a network-based correlation scheme (Yang & Lee, 2008). Therefore, we also cannot use it because it lies beyond the scope of this study (host-based). As a result, we are left with the RTT-based and packet number-based techniques. The data analysis will evaluate if the RTT- based and Packet number-based techniques can detect the backdoor attack based on what has already been conducted in the experiment by using more than ten backdoor samples.

From the experiment, five scenarios were used by backdoor attacks to transmit the data from the victim host to the attacker host. We analyzed all five scenarios one by one to discuss the effectiveness of the two SDD techniques above in dealing with the scenarios and to determine which technique(s) can be used for all five scenarios.

1- Scenario (1):

When the victim host and the attacker host are active and there are incoming and outgoing flows between them. Figure 5.1 shows the traffic between the backdoor and the attacker in two sessions. Figure 5.2 shows the capture packets by Wireshark tool in the victim side and Figure 5.3 shows the capture packets by Wireshark tool in the attacker side. This scenario was shown in the experiment that run by using the backdoors attack samples NjRatv.0401, RTF_Mongall_Dropper and Bifrost.

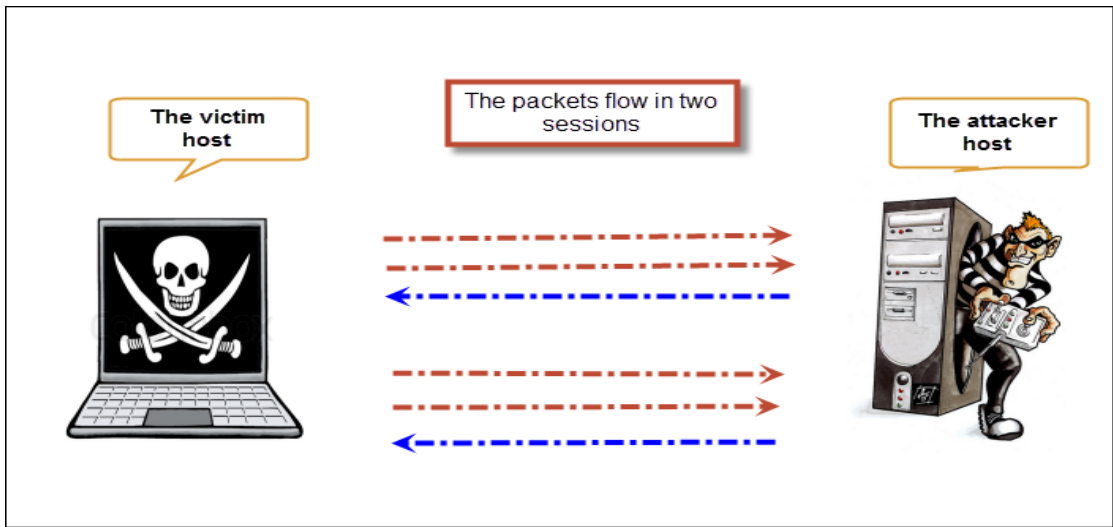


Figure 5.1 : Scenario (1), the flow between the backdoor and the attacker

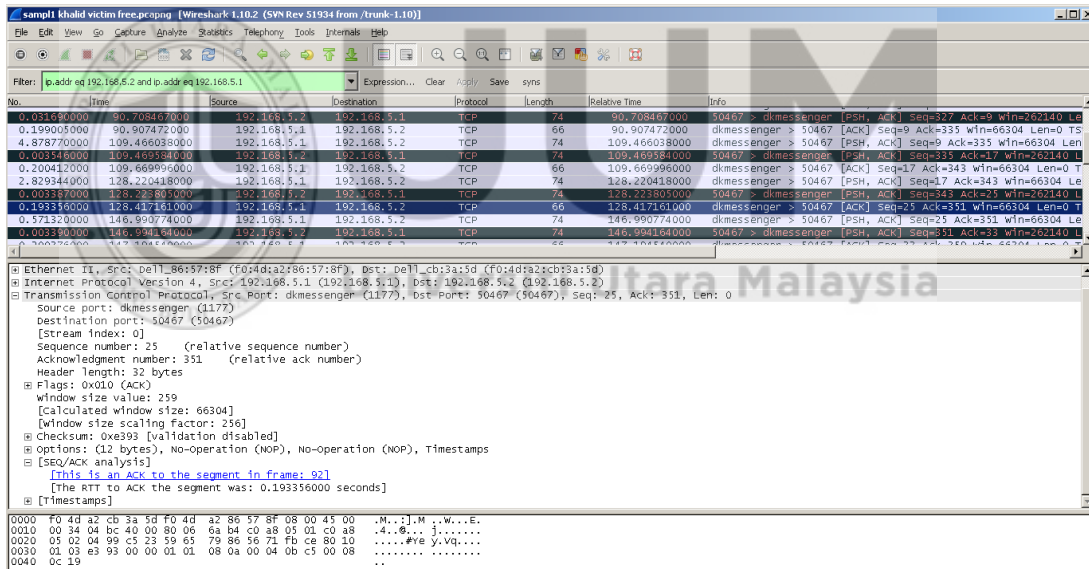


Figure 5.2 : Scenario (1), the capture packets in the victim side

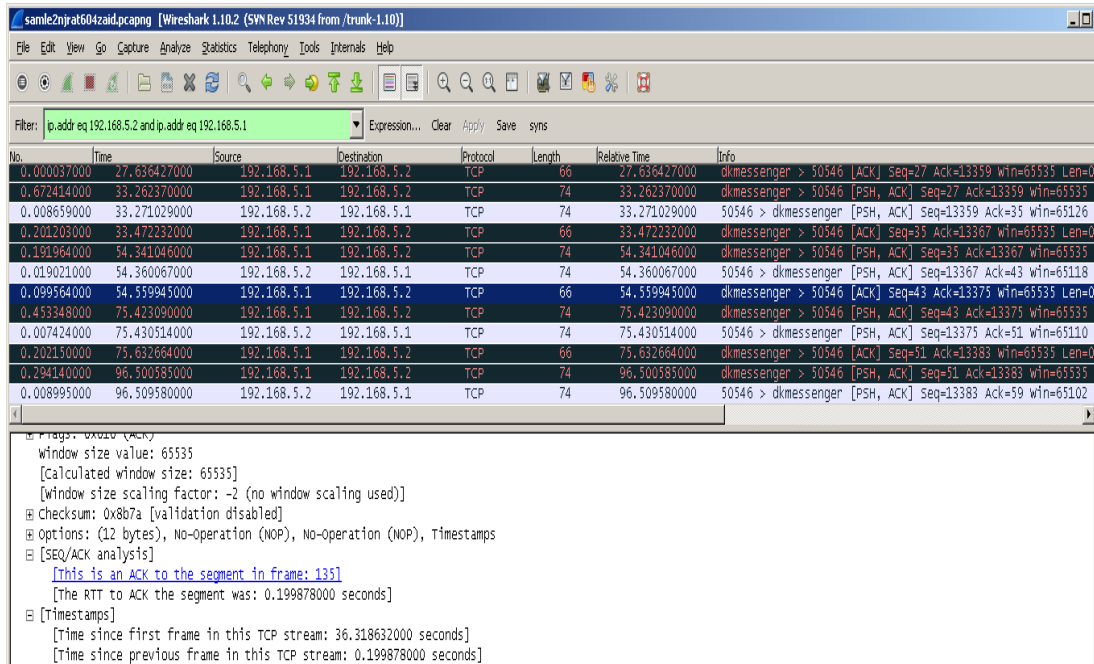


Figure 5.3: Scenario (1), the capture packets in the attacker side

As shown in the three figures above, it can obtain all the metrics that used in all SDD techniques such as the time between the segments, the number of packets and the round trip time (RTT). The important notes from this experiment are:

- a) Each session contains the same number of packets.
- b) All sessions are matched. For example, they have the same time period between the segment, the same length of packets and the same RTT.

2- Scenario (2):

When the victim host is active, and the attacker (client's software) is not active, but his host is online. This scenario was shown in the experiment that run by using the backdoor attack sample (Poison's server).

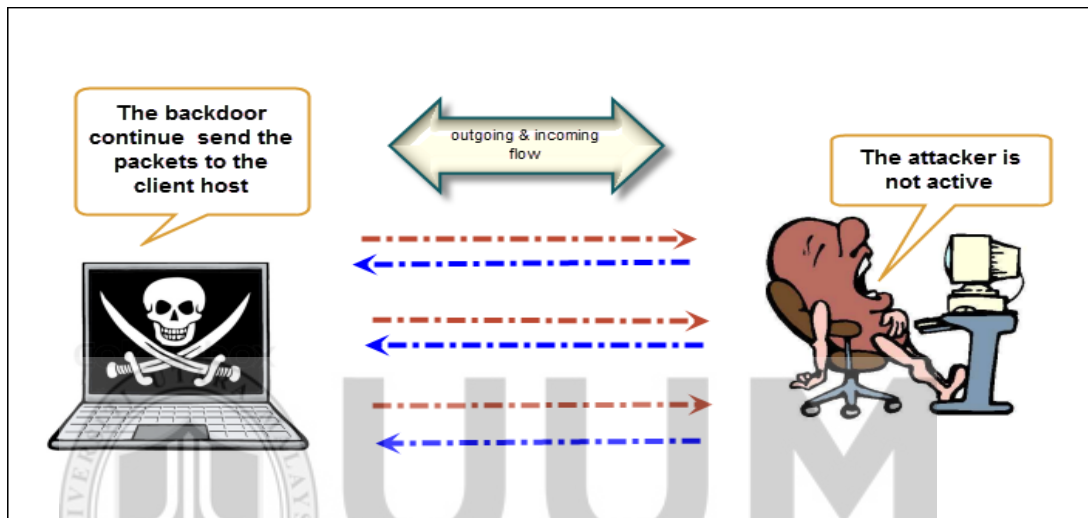


Figure 5.4: Scenario (2), the flow between the backdoor and the host of the attacker

In this scenario, the backdoor tries connect with the client's software which is not active inside online host. Therefore, the session is canceled every time with RST flag. Figure 5.4 shows the flow between the backdoor and the host of the attacker in three sessions. Figure 5.5 shows the capture packets by Wireshark tool in the victim side and figure 5.6 shows the capture packets by Wireshark tool in the attacker side.

poison when the client sleep_victim side.pcapng [Wireshark 1.10.2 (SVN Rev 51934 from /trunk-1.10)]

Filter: tcp

No.	Time	Source	Destination	Protocol	Length	Relative Time	Info
0.000030000	196.224256000	192.168.5.22	192.168.5.11	TCP	62	196.224256000	cma > edm-manager [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SA
0.001878000	196.226134000	192.168.5.11	192.168.5.22	TCP	60	196.226134000	edm-manager > cma [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
0.416336000	196.642470000	192.168.5.22	192.168.5.11	TCP	62	196.642470000	[TCP Retransmission] cma > edm-manager [SYN] Seq=0 Win=64
0.000401000	196.642871000	192.168.5.11	192.168.5.22	TCP	60	196.642871000	edm-manager > cma [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
0.544779000	197.187650000	192.168.5.22	192.168.5.11	TCP	62	197.187650000	[TCP Retransmission] cma > edm-manager [SYN] Seq=0 Win=64
0.008102000	197.195752000	192.168.5.11	192.168.5.22	TCP	60	197.195752000	edm-manager > cma [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
4.311714000	207.271659000	192.168.5.22	192.168.5.11	TCP	62	207.271659000	optima-vnet > edm-manager [SYN] Seq=0 Win=64240 Len=0 MSS
0.000654000	207.272313000	192.168.5.11	192.168.5.22	TCP	60	207.272313000	edm-manager > optima-vnet [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
0.523116000	207.795429000	192.168.5.22	192.168.5.11	TCP	62	207.795429000	[TCP Retransmission] optima-vnet > edm-manager [SYN] Seq=0
0.000392000	207.795821000	192.168.5.11	192.168.5.22	TCP	60	207.795821000	edm-manager > optima-vnet [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
0.436608000	208.232429000	192.168.5.22	192.168.5.11	TCP	62	208.232429000	[TCP Retransmission] optima-vnet > edm-manager [SYN] Seq=0
0.000346000	208.232775000	192.168.5.11	192.168.5.22	TCP	60	208.232775000	edm-manager > optima-vnet [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

Internet Protocol Version 4, Src: 192.168.5.11 (192.168.5.11), Dst: 192.168.5.22 (192.168.5.22)

Transmission Control Protocol, Src Port: edm-manager (3460), Dst Port: cma (1050), Seq: 1, Ack: 1, Len: 0

Source port: edm-manager (3460)

Destination port: cma (1050)

[Stream index: 1]

Sequence number: 1 (relative sequence number)

Acknowledgment number: 1 (relative ack number)

Header length: 20 bytes

Flags: 0x014 (RST, ACK)

Window size value: 0

[Calculated window size: 0]

[Window size scaling factor: -2 (no window scaling used)]

Checksum: 0xdd0c (validation disabled)

[SEQ/ACK analysis]

This is an ACK to the segment in frame: 47

[The RTT to ACK the segment was: 0.008102000 seconds]

[Timestamps]

```

0000 00 0c 29 44 8a 04 00 0c 29 aa 7d 1e 08 00 45 00  ..)D....).E.
0010 00 28 06 c6 00 00 80 06 a8 98 c0 a8 05 0b c0 a8  ..(.....
0020 05 16 0d 84 04 1b 00 00 00 00 12 a6 22 ae 50 14  ..X...P.
0030 00 00 0d 0c 00 00 00 00 00 00 00 00 00 00 00  ..

```

Figure 5.5: Scenario (2), Poison backdoor in the victim side

poison the clien sleep -attacker side.pcapng [Wireshark 1.10.2 (SVN Rev 51934 from /trunk-1.10)]

Filter: tcp

No.	Time	Source	Destination	Protocol	Length	Relative Time	Info
0.000596000	105.079594000	192.168.5.22	192.168.5.11	TCP	62	105.079594000	cma > edm-manager [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SA
0.001047000	105.080641000	192.168.5.11	192.168.5.22	TCP	54	105.080641000	edm-manager > cma [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
0.417112000	105.497753000	192.168.5.22	192.168.5.11	TCP	62	105.497753000	[TCP Retransmission] cma > edm-manager [SYN] Seq=0 Win=642
0.000066000	105.497819000	192.168.5.11	192.168.5.22	TCP	54	105.497819000	edm-manager > cma [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
0.552632000	106.050451000	192.168.5.22	192.168.5.11	TCP	62	106.050451000	[TCP Retransmission] cma > edm-manager [SYN] Seq=0 Win=642
0.000065000	106.050516000	192.168.5.11	192.168.5.22	TCP	54	106.050516000	edm-manager > cma [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
4.311905000	116.127052000	192.168.5.22	192.168.5.11	TCP	62	116.127052000	optima-vnet > edm-manager [SYN] Seq=0 Win=64240 Len=0 MSS=
0.000106000	116.127158000	192.168.5.11	192.168.5.22	TCP	54	116.127158000	edm-manager > optima-vnet [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
0.523343000	116.650692000	192.168.5.22	192.168.5.11	TCP	62	116.650692000	[TCP Retransmission] optima-vnet > edm-manager [SYN] Seq=0
0.000058000	116.650730000	192.168.5.11	192.168.5.22	TCP	54	116.650730000	edm-manager > optima-vnet [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
0.436899000	117.087629000	192.168.5.22	192.168.5.11	TCP	62	117.087629000	[TCP Retransmission] optima-vnet > edm-manager [SYN] Seq=0
0.000048000	117.087677000	192.168.5.11	192.168.5.22	TCP	54	117.087677000	edm-manager > optima-vnet [RST, ACK] Seq=1 Ack=1 Win=0 Len=0

Internet Protocol Version 4, Src: 192.168.5.11 (192.168.5.11), Dst: 192.168.5.22 (192.168.5.22)

Transmission Control Protocol, Src Port: edm-manager (3460), Dst Port: optima-vnet (1051), Seq: 1, Ack: 1, Len: 0

Source port: edm-manager (3460)

Destination port: optima-vnet (1051)

[Stream index: 1]

Sequence number: 1 (relative sequence number)

Acknowledgment number: 1 (relative ack number)

Header length: 20 bytes

Flags: 0x014 (RST, ACK)

Window size value: 0

[Calculated window size: 0]

[Window size scaling factor: -2 (no window scaling used)]

Checksum: 0x9d05 (validation disabled)

[SEQ/ACK analysis]

[Timestamps]

[Time since first frame in this TCP stream: 0.523678000 seconds]

[Time since previous frame in this TCP stream: 0.000058000 seconds]

```

0000 00 0c 29 44 8a 04 00 0c 29 aa 7d 1e 08 00 45 00  ..)D....).E.
0010 00 28 06 c8 00 00 80 06 a8 96 c0 a8 05 0b c0 a8  ..(.....
0020 05 16 0d 84 04 1b 00 00 00 00 58 00 1d ba 50 14  ..X...P.
0030 00 00 0d 0c 00 00 00 00 00 00 00 00 00 00 00  ..

```

Figure 5.6: Scenario (2), Poison backdoor in the attacker side.

The important notes from this experiment are:

- a) Each session contains the same number of packets.
- b) All sessions are matched. For example, they have the same time period between the segment, the same length of packets and the same RTT.

3- Scenario (3):

When the victim host active, and the attacker host are offline.

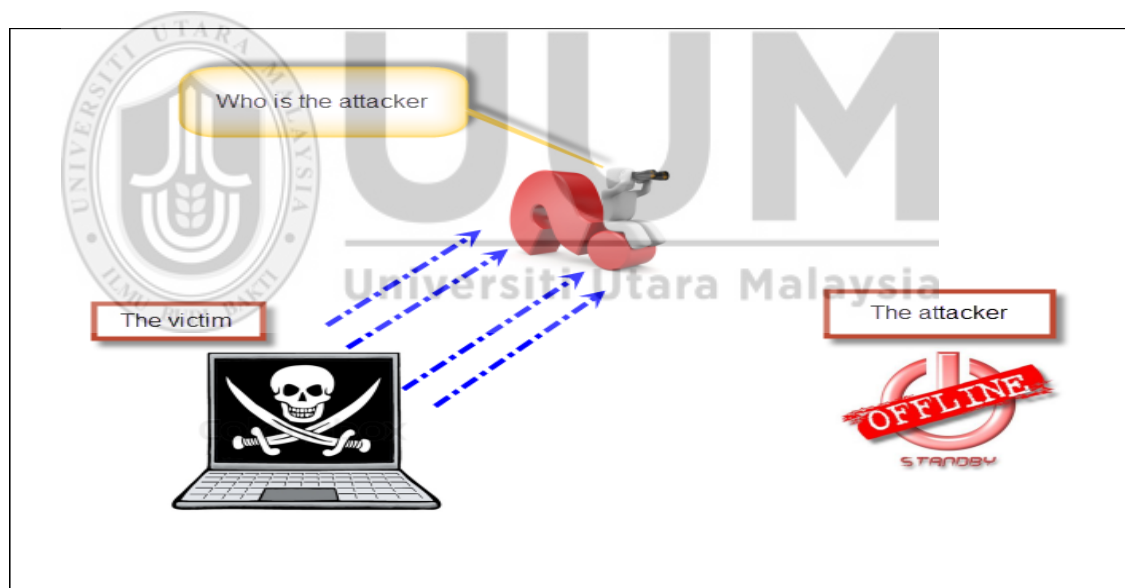


Figure 5.7: Scenario (3), the victim host active, and the attacker host are offline.

This scenario was shown in the experiments that run by using the backdoor attack samples Njrat 604 and Bifrost. Figure 5.7 shows the backdoor tries to find the distention host (the attacker), but the attacker is offline. There is no information or

flow comes from the attacker. Figure 5.8 shows the capture packets by Wireshark tool in the victim in this scenario.

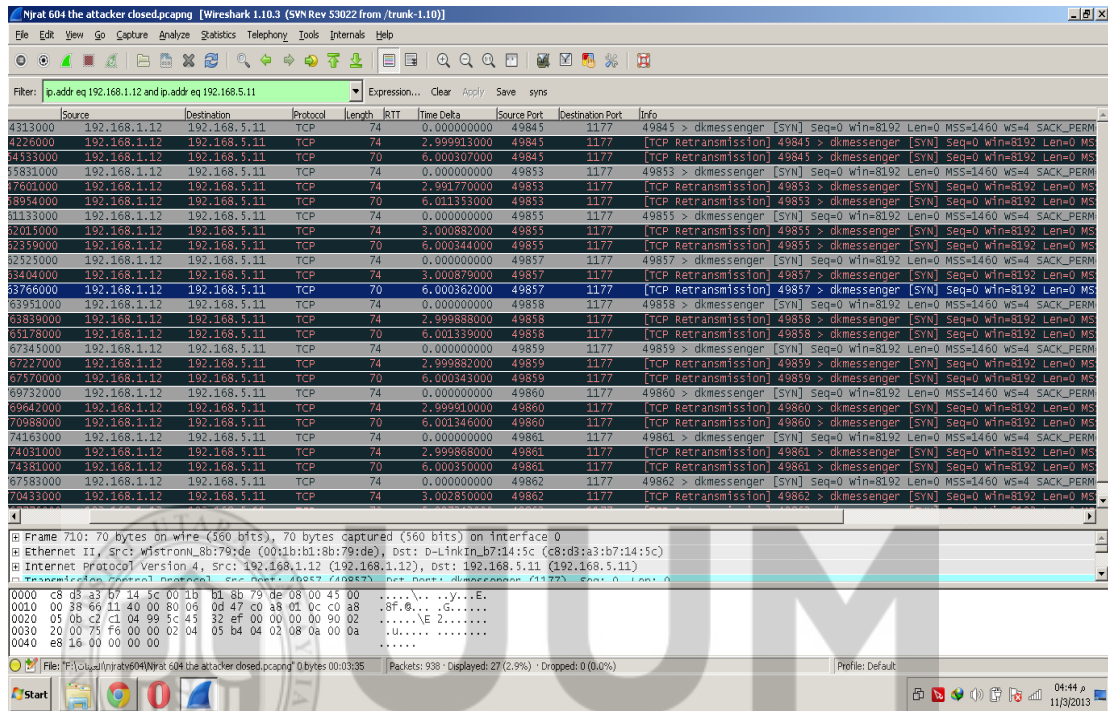


Figure 5.8: Scenario (3), the capture packets in the victim side.

The important notes from this experiment are:

- There is only outgoing packets without ACKs. Therefore, in this case we cannot use the RTT metric.
- Each session contains the same number of packets.

4- Scenario (4):

When the type of the backdoor attack use outgoing flow only such as the Advanced Persistent Threat (APT)s backdoors, that make only outbound connections (Welch, Pearson, Tierney, & Williams, 2012), there is no incoming flow in this scenario. In fact, this type is very dangerous due to, the default setting for most firewalls including Windows system firewall do not monitor the outbound traffic, they only monitor inbound traffic (Sukwong, Kim, & Hoe, 2011).

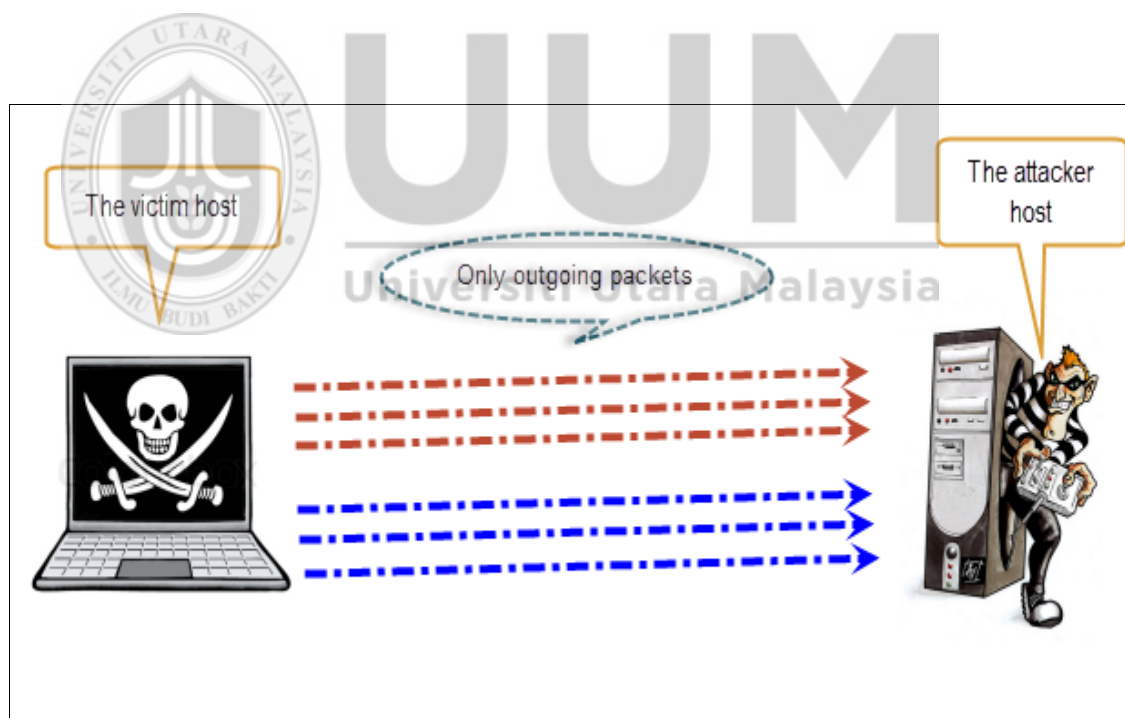


Figure 5.9: Scenario (4), the flow between the APT backdoor and the attacker

Figure 5.9 shows the flow between the backdoor and the attacker host. Figure 5.10 shows the capture packets by Wireshark tool in the victim side.

No.	Time	Source	Destination	Protocol	Length	Relative Time	Info
0.000000	223.912402	172.16.253.131	197.163.56.70	TCP	1514	223.912402	dst > 336 [ACK] Seq=150933 Ack=239 Wln=64002 Len=160
0.000083	223.912485	172.16.253.131	197.163.56.70	TCP	1068	223.912485	dst > 336 [PSH, ACK] Seq=151913 Ack=239 Wln=64002 Len=101
0.009830	223.012315	172.16.253.131	197.163.56.70	TCP	70	223.012315	dst > 336 [PSH, ACK] Seq=152927 Ack=239 Wln=64002 Len=16
0.000376	223.012691	172.16.253.131	197.163.56.70	TCP	1514	223.012691	dst > 336 [ACK] Seq=152943 Ack=239 Wln=64002 Len=1460
0.000051	223.012742	172.16.253.131	197.163.56.70	TCP	1514	223.012742	dst > 336 [ACK] Seq=154403 Ack=239 Wln=64002 Len=1460
0.000114	223.012856	172.16.253.131	197.163.56.70	TCP	1514	223.012856	dst > 336 [ACK] Seq=155863 Ack=239 Wln=64002 Len=1460
0.000054	223.012910	172.16.253.131	197.163.56.70	TCP	1514	223.012910	dst > 336 [ACK] Seq=157323 Ack=239 Wln=64002 Len=1460
0.000098	223.013008	172.16.253.131	197.163.56.70	TCP	1514	223.013008	dst > 336 [ACK] Seq=158783 Ack=239 Wln=64002 Len=1460
0.000045	223.013053	172.16.253.131	197.163.56.70	TCP	1514	223.013053	dst > 336 [ACK] Seq=160243 Ack=239 Wln=64002 Len=1460
0.000097	223.013150	172.16.253.131	197.163.56.70	TCP	1514	223.013150	dst > 336 [ACK] Seq=161703 Ack=239 Wln=64002 Len=1460
0.000043	223.013193	172.16.253.131	197.163.56.70	TCP	1514	223.013193	dst > 336 [ACK] Seq=163163 Ack=239 Wln=64002 Len=1460
0.000101	223.013294	172.16.253.131	197.163.56.70	TCP	1514	223.013294	dst > 336 [ACK] Seq=164623 Ack=239 Wln=64002 Len=1460
0.000080	223.013374	172.16.253.131	197.163.56.70	TCP	1514	223.013374	dst > 336 [ACK] Seq=166083 Ack=239 Wln=64002 Len=1460
0.000084	223.013458	172.16.253.131	197.163.56.70	TCP	1514	223.013458	dst > 336 [ACK] Seq=167543 Ack=239 Wln=64002 Len=1460
0.000081	223.013539	172.16.253.131	197.163.56.70	TCP	1514	223.013539	dst > 336 [ACK] Seq=169003 Ack=239 Wln=64002 Len=1460
0.000080	223.013619	172.16.253.131	197.163.56.70	TCP	1514	223.013619	dst > 336 [ACK] Seq=170463 Ack=239 Wln=64002 Len=1460
0.000083	223.013702	172.16.253.131	197.163.56.70	TCP	1514	223.013702	dst > 336 [ACK] Seq=171923 Ack=239 Wln=64002 Len=1460
0.000090	223.013792	172.16.253.131	197.163.56.70	TCP	1514	223.013792	dst > 336 [ACK] Seq=173383 Ack=239 Wln=64002 Len=1460
0.000086	223.013878	172.16.253.131	197.163.56.70	TCP	1514	223.013878	dst > 336 [ACK] Seq=174843 Ack=239 Wln=64002 Len=1460
0.000102	223.013980	172.16.253.131	197.163.56.70	TCP	1514	223.013980	dst > 336 [ACK] Seq=176303 Ack=239 Wln=64002 Len=1460
0.000045	223.014025	172.16.253.131	197.163.56.70	TCP	1514	223.014025	dst > 336 [ACK] Seq=177763 Ack=239 Wln=64002 Len=1460
0.000102	223.014127	172.16.253.131	197.163.56.70	TCP	1514	223.014127	dst > 336 [ACK] Seq=179223 Ack=239 Wln=64002 Len=1460
0.000045	223.014172	172.16.253.131	197.163.56.70	TCP	1514	223.014172	dst > 336 [ACK] Seq=180683 Ack=239 Wln=64002 Len=1460
0.000085	223.014257	172.16.253.131	197.163.56.70	TCP	1514	223.014257	dst > 336 [ACK] Seq=182143 Ack=239 Wln=64002 Len=1460
0.000083	223.014340	172.16.253.131	197.163.56.70	TCP	1514	223.014340	dst > 336 [ACK] Seq=183603 Ack=239 Wln=64002 Len=1460
0.000083	223.014423	172.16.253.131	197.163.56.70	TCP	702	223.014423	dst > 336 [PSH, ACK] Seq=185063 Ack=239 Wln=64002 Len=648
0.000177	223.014600	172.16.253.131	197.163.56.70	TCP	1514	223.014600	dst > 336 [ACK] Seq=187111 Ack=239 Wln=64002 Len=1460
0.000046	223.014646	172.16.253.131	197.163.56.70	TCP	1514	223.014646	dst > 336 [ACK] Seq=188171 Ack=239 Wln=64002 Len=1460
0.000084	223.014730	172.16.253.131	197.163.56.70	TCP	1514	223.014730	dst > 336 [ACK] Seq=189631 Ack=239 Wln=64002 Len=1460
0.000085	223.014815	172.16.253.131	197.163.56.70	TCP	1068	223.014815	dst > 336 [PSH, ACK] Seq=190091 Ack=239 Wln=64002 Len=101
0.003009	223.107824	172.16.253.131	197.163.56.70	TCP	70	223.107824	dst > 336 [PSH, ACK] Seq=191105 Ack=239 Wln=64002 Len=16
0.000438	223.108262	172.16.253.131	197.163.56.70	TCP	1514	223.108262	dst > 336 [ACK] Seq=191121 Ack=239 Wln=64002 Len=1460
0.000080	223.108320	172.16.253.131	197.163.56.70	TCP	1514	223.108320	dst > 336 [ACK] Seq=192581 Ack=239 Wln=64002 Len=1460

Figure 5.10: Scenario (4), the backdoor use outgoing flow only

The important notes from this experiment are:

- Each session contains the same number of packets, (for this sample the outgoing segments = 30 and the incoming segments = 0).
- All sessions are matched. For example, they have the same number of packets, the same time period between the segment and the same length of packets.
- Each session contains only outgoing packets without backs (ACKs).

5- Scenario (5):

When, the victim and the attacker use one or more than an intermediate server, they use indirect connection.

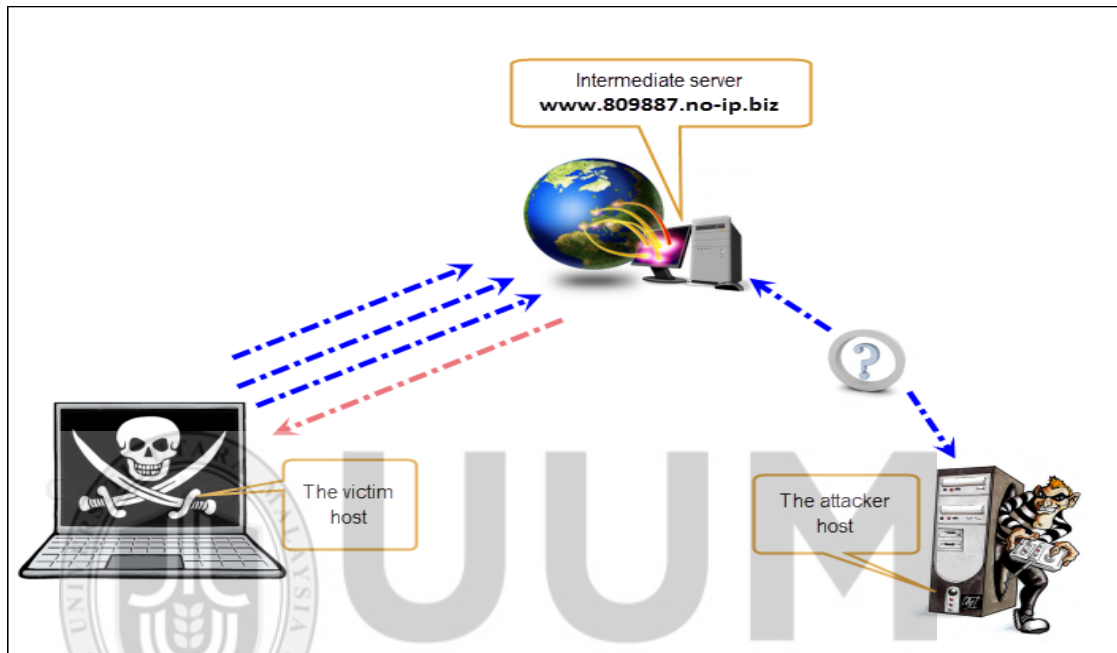


Figure 5.11 : Scenario (5), using the intermediate server

This scenario was shown in the experiment that run with the backdoor attack sample (Xtreem) backdoor. Figure 5.11 shows the flow between the backdoor and the intermediate server. We used 809887.no-ip.biz with real ip-address (175.144.93.33) as an intermediate online server as shown in Figure 5.13. Figure 5.12 shows the capture packets by Wireshark tool in the victim side.

No.	Time	Source	Destination	Protocol	Length	Relative Time
6.015391000	148.747219000	192.168.20.132	175.144.93.33	TCP	62	148.747219000
1.744989000	160.704165000	175.144.93.33	192.168.20.132	TCP	60	160.704165000
0.998985000	161.703150000	192.168.20.132	175.144.93.33	TCP	62	161.703150000
3.013141000	164.716291000	192.168.20.132	175.144.93.33	TCP	62	164.716291000
6.015292000	170.731583000	192.168.20.132	175.144.93.33	TCP	62	170.731583000
11.973819000	182.705402000	175.144.93.33	192.168.20.132	TCP	60	182.705402000
0.996537000	183.701939000	192.168.20.132	175.144.93.33	TCP	62	183.701939000
2.889412000	186.591351000	192.168.20.132	175.144.93.33	TCP	62	186.591351000
0.341852000	192.606774000	192.168.20.132	175.144.93.33	TCP	62	192.606774000
1.437219000	204.706548000	175.144.93.33	192.168.20.132	TCP	60	204.706548000
0.934485000	205.641033000	192.168.20.132	175.144.93.33	TCP	62	205.641033000
2.934607000	208.575640000	192.168.20.132	175.144.93.33	TCP	62	208.575640000
6.015383000	214.591023000	192.168.20.132	175.144.93.33	TCP	62	214.591023000
0.019373000	220.186416000	192.168.20.132	175.144.93.33	TCP	62	220.186416000
3.045133000	223.231549000	192.168.20.132	175.144.93.33	TCP	62	223.231549000
3.413337000	226.644886000	175.144.93.33	192.168.20.132	TCP	60	226.644886000
0.980046000	227.624932000	192.168.20.132	175.144.93.33	TCP	62	227.624932000
1.621992000	229.246924000	192.168.20.132	175.144.93.33	TCP	62	229.246924000
1.312919000	230.559843000	192.168.20.132	175.144.93.33	TCP	62	230.559843000
6.015482000	236.575325000	192.168.20.132	175.144.93.33	TCP	62	236.575325000
4.615474000	241.190799000	175.144.93.33	192.168.20.132	TCP	60	241.190799000
1.000405000	242.191204000	192.168.20.132	175.144.93.33	TCP	62	242.191204000
3.024537000	245.215741000	192.168.20.132	175.144.93.33	TCP	62	245.215741000
1.155996000	248.622198000	175.144.93.33	192.168.20.132	TCP	60	248.622198000
0.890539000	249.657601000	192.168.20.132	175.144.93.33	TCP	62	249.657601000
1.621992000	251.232104000	192.168.20.132	175.144.93.33	TCP	62	251.232104000
1.422163000	252.654267000	192.168.20.132	175.144.93.33	TCP	62	252.654267000
1.700813000	258.669536000	192.168.20.132	175.144.93.33	TCP	62	258.669536000
4.525421000	263.194957000	175.144.93.33	192.168.20.132	TCP	60	263.194957000
0.996059000	264.191016000	192.168.20.132	175.144.93.33	TCP	62	264.191016000
2.899970000	267.090986000	192.168.20.132	175.144.93.33	TCP	62	267.090986000
3.570430000	270.661416000	175.144.93.33	192.168.20.132	TCP	60	270.661416000

Figure 5.12 : Scenario (5), the capture packets in the victim side

The screenshot shows the DNS2IP website interface. At the top, there are navigation links for HOME, DNS2IP, IP2DNS, and DEFINITIONS. A search bar is present with the text "Enter a DNS address:" and the input "809887.no-ip.biz". Below the search bar, the result is displayed: "Result for '809887.no-ip.biz': IP : 175.144.93.33 DNS : 175.144.93.33". The website also features a sidebar with a "Win tickets-Super Show 5" advertisement and a footer with copyright information for YooDev.

Figure 5.13: The information of the intermediate online server

The important notes from this experiment are:

- a) Each session contains the same number of packets.
- b) All sessions are matched. For example, they have the same time period between the segments and the same length of packets.
- c) There is no information about the real attacker.

5.3 Findings

From the analysis of the experimental results and the literature review, we arrived at the following findings:

- 1- As mentioned in Chapter Two, specifically in Sections 2.3.5 and 2.3.6, the main concept of the SSD approach is the match between incoming and outgoing streams. However, in the backdoor situation, the backdoor repeats the connection many times for the same specific port (Omar et al., 2013). The activity of the backdoor is repeated several times, and each conversation contains the same amount of data that occurs within the same period of time. Therefore, the outgoing packets for the backdoor should be matched as shown in Figure 5.14 and 5.15.

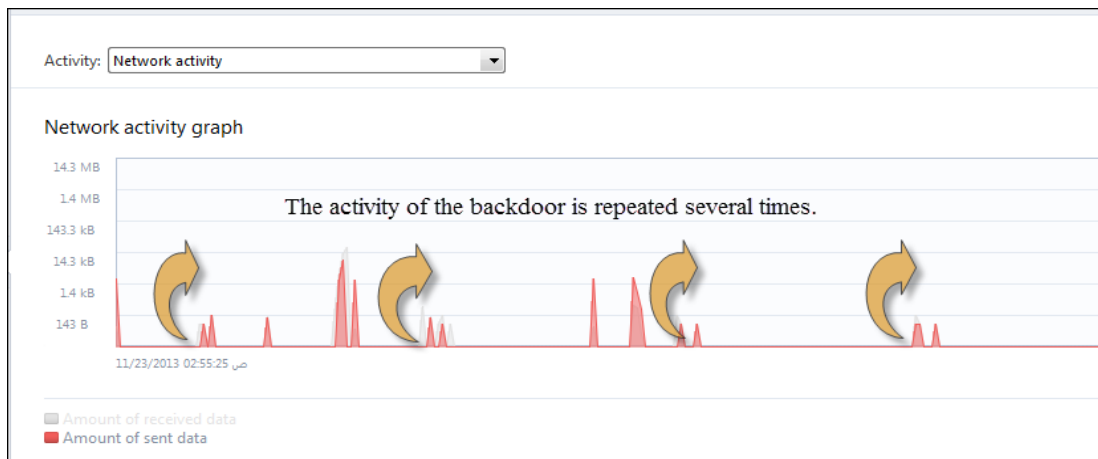


Figure 5.14 : The activity graph of the backdoor

2- Regarding (1), to detect the backdoor attack in all scenarios, we need only to compare the two outgoing repeated packets, which use the same port and are headed to the same destination. If they match, then their source is a backdoor attack. Otherwise, the source is clean. In other words, we have only to match the outgoing packets to detect the backdoor attack.

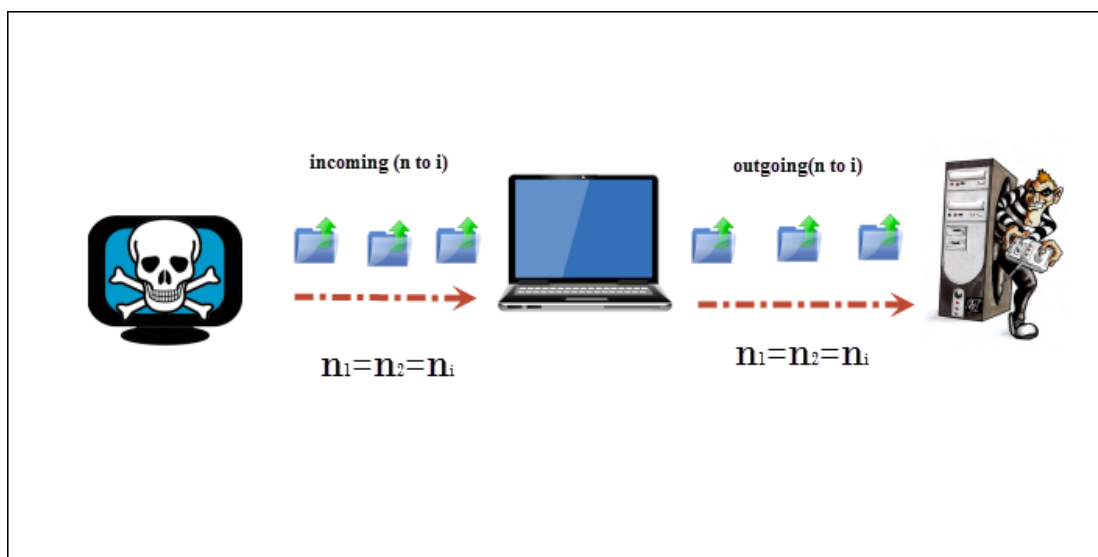


Figure 5.15: The backdoor activity

First, we used the RTT technique for the outgoing packets to detect the matched packets as shown in Figure 5.16. The detection result ratio for the first five samples was 100%. However, after more samples and more experiments, the detection ratio result decreased to 80% for the ten samples.

No.	Time	Source	Destination	Protocol	Length	RTT	Time Delta	Source Port	Destination Port	Info
39.746652	1051.450285	172.16.253.129	125.77.199.30	TCP	62	0.000000000	0.000000000	1075	8000	rdmshc > irdmi [SYN, ACK] Seq=0 Win=64240 Len=0
0.000116	1051.707177	125.77.199.30	172.16.253.129	TCP	60	0.256892000	0.256892000	8000	1075	irdmi > rdmshc [SYN, ACK] Seq=0 Ack=1 Win=6
0.000622	1092.138518	172.16.253.129	125.77.199.30	TCP	62	0.000000000	0.000000000	1076	8000	dab-st1-c > irdmi [SYN, ACK] Seq=0 Win=64240 Len=0
0.199425	1092.390448	125.77.199.30	172.16.253.129	TCP	60	0.251530000	0.251530000	8000	1076	irdmi > dab-st1-c [SYN, ACK] Seq=0 Ack=1 Win=6
39.747143	1132.313626	172.16.253.129	125.77.199.30	TCP	62	0.000000000	0.000000000	1077	8000	ingames > irdmi [SYN, ACK] Seq=0 Win=64240 Len=0
0.252539	1132.786165	125.77.199.30	172.16.253.129	TCP	60	0.252539000	0.252539000	8000	1077	irdmi > ingames [SYN, ACK] Seq=0 Ack=1 Win=6
39.762185	1173.904592	172.16.253.129	125.77.199.30	TCP	62	0.000000000	0.000000000	1078	8000	avocent-proxy > irdmi [SYN, ACK] Seq=0 Win=64240 Len=0
0.254291	1173.158883	125.77.199.30	172.16.253.129	TCP	60	0.254291000	0.254291000	8000	1078	irdmi > avocent-proxy [SYN, ACK] Seq=0 Ack=1 Win=6
39.673413	1213.293865	172.16.253.129	125.77.199.30	TCP	62	0.000000000	0.000000000	1079	8000	asprovataik > irdmi [SYN, ACK] Seq=0 Win=64240 Len=0
0.258757	1213.552622	125.77.199.30	172.16.253.129	TCP	60	0.258757000	0.258757000	8000	1079	irdmi > asprovataik [SYN, ACK] Seq=0 Ack=1 Win=6
0.000631	1253.923214	172.16.253.129	125.77.199.30	TCP	62	0.000000000	0.000000000	1080	8000	socks > irdmi [SYN, ACK] Seq=0 Win=64240 Len=0 MS
0.202969	1254.205571	125.77.199.30	172.16.253.129	TCP	60	0.253257000	0.253257000	8000	1080	irdmi > socks [SYN, ACK] Seq=0 Ack=1 Win=642
39.748003	1294.325890	172.16.253.129	125.77.199.30	TCP	62	0.000000000	0.000000000	1081	8000	pvunwien > irdmi [SYN, ACK] Seq=0 Win=64240 Len=0
0.252029	1294.577919	125.77.199.30	172.16.253.129	TCP	60	0.252029000	0.252029000	8000	1081	irdmi > pvunwien [SYN, ACK] Seq=0 Ack=1 Win=6
39.746650	1334.700660	172.16.253.129	125.77.199.30	TCP	62	0.000000000	0.000000000	1082	8000	amt-esd-prot > irdmi [SYN, ACK] Seq=0 Win=64240 Len=0
0.264592	1334.965252	125.77.199.30	172.16.253.129	TCP	60	0.264592000	0.264592000	8000	1082	irdmi > amt-esd-prot [SYN, ACK] Seq=0 Ack=1 Win=6
9.871814	1375.075979	172.16.253.129	125.77.199.30	TCP	62	0.000000000	0.000000000	1083	8000	ansoft-lm-1 > irdmi [SYN, ACK] Seq=0 Win=64240 Len=0
0.000225	1375.331940	125.77.199.30	172.16.253.129	TCP	60	0.255961000	0.255961000	8000	1083	irdmi > ansoft-lm-1 [SYN, ACK] Seq=0 Ack=1 Win=6
0.000566	1418.287866	172.16.253.129	125.77.199.30	TCP	62	0.000000000	0.000000000	1084	8000	ansoft-lm-2 > irdmi [SYN, ACK] Seq=0 Win=64240 Len=0
0.121942	1418.039124	125.77.199.30	172.16.253.129	TCP	60	0.251438000	0.251438000	8000	1084	irdmi > ansoft-lm-2 [SYN, ACK] Seq=0 Ack=1 Win=6
7.155028	1456.193786	172.16.253.129	125.77.199.30	TCP	62	0.000000000	0.000000000	1085	8000	webobjects > irdmi [SYN, ACK] Seq=0 Win=64240 Len=0
0.254057	1456.407843	125.77.199.30	172.16.253.129	TCP	60	0.254057000	0.254057000	8000	1085	irdmi > webobjects [SYN, ACK] Seq=0 Ack=1 Win=6

Figure 5.16: Backdoor detection based on the round trip time (RTT) technique

3- For scenario (3), when the host of the attacker is offline and for the scenario (4), which uses only the outgoing packets, we cannot use the RTT to detect the backdoor attack. This phenomenon is a result of the lack of acknowledgments (ACKs) for the outgoing packets and the absence of an echo for the send packet. In other words, no RTT exists as shown in Figure 5.17. These reasons also explain why we have two undetected samples when we used the RTT technique.

No.	Time	Source	Destination	Protocol	Length	RTT	Time Delta	Source Port	Destination Port	Info
0.000115	123.719381	172.16.253.131	197.163.56.70	TCP	1514		0.000115000	1047	336	neod1 > 336 [ACK] Seq=103077 Ack=1077 Win=63
0.000095	123.719476	172.16.253.131	197.163.56.70	TCP	1334		0.000095000	1047	336	neod1 > 336 [PSH, ACK] Seq=104537 Ack=1077 W
19.535305	143.254781	172.16.253.131	197.163.56.70	TCP	70		19.535305000	1047	336	neod1 > 336 [PSH, ACK] Seq=105817 Ack=1083 W
0.000226	143.255007	172.16.253.131	197.163.56.70	TCP	86		0.000226000	1047	336	neod1 > 336 [PSH, ACK] Seq=105833 Ack=1083 W
0.459414	143.714421	172.16.253.131	197.163.56.70	TCP	70		0.459414000	1047	336	neod1 > 336 [PSH, ACK] Seq=105865 Ack=1219 W
0.000257	143.714678	172.16.253.131	197.163.56.70	TCP	1514		0.000257000	1047	336	neod1 > 336 [ACK] Seq=105881 Ack=1219 Win=63
0.000051	143.714729	172.16.253.131	197.163.56.70	TCP	1514		0.000051000	1047	336	neod1 > 336 [ACK] Seq=107341 Ack=1219 Win=63
0.000043	143.714772	172.16.253.131	197.163.56.70	TCP	1514		0.000043000	1047	336	neod1 > 336 [ACK] Seq=108801 Ack=1219 Win=63
0.000050	143.714822	172.16.253.131	197.163.56.70	TCP	1514		0.000050000	1047	336	neod1 > 336 [ACK] Seq=110261 Ack=1219 Win=63
0.000063	143.714885	172.16.253.131	197.163.56.70	TCP	1514		0.000063000	1047	336	neod1 > 336 [ACK] Seq=111721 Ack=1219 Win=63
0.000060	143.714945	172.16.253.131	197.163.56.70	TCP	1514		0.000060000	1047	336	neod1 > 336 [ACK] Seq=113181 Ack=1219 Win=63
0.000061	143.715006	172.16.253.131	197.163.56.70	TCP	1514		0.000061000	1047	336	neod1 > 336 [ACK] Seq=114641 Ack=1219 Win=63
0.000137	143.715143	172.16.253.131	197.163.56.70	TCP	1514		0.000137000	1047	336	neod1 > 336 [ACK] Seq=116101 Ack=1219 Win=63
0.000093	143.715236	172.16.253.131	197.163.56.70	TCP	250		0.000093000	1047	336	neod1 > 336 [PSH, ACK] Seq=117561 Ack=1219 W
0.058244	143.773480	172.16.253.131	197.163.56.70	TCP	70		0.058244000	1047	336	neod1 > 336 [PSH, ACK] Seq=117757 Ack=1219 W
0.000268	143.773748	172.16.253.131	197.163.56.70	TCP	1514		0.000268000	1047	336	neod1 > 336 [ACK] Seq=117773 Ack=1219 Win=63
0.000048	143.773796	172.16.253.131	197.163.56.70	TCP	1514		0.000048000	1047	336	neod1 > 336 [ACK] Seq=119233 Ack=1219 Win=63
0.000042	143.773838	172.16.253.131	197.163.56.70	TCP	1514		0.000042000	1047	336	neod1 > 336 [ACK] Seq=120693 Ack=1219 Win=63
0.000098	143.773936	172.16.253.131	197.163.56.70	TCP	1408		0.000098000	1047	336	neod1 > 336 [PSH, ACK] Seq=122153 Ack=1219 W
19.550053	163.323989	172.16.253.131	197.163.56.70	TCP	54		19.550053000	1047	336	neod1 > 336 [ACK] Seq=123507 Ack=1225 Win=63
0.310387	163.634376	172.16.253.131	197.163.56.70	TCP	70		0.310387000	1047	336	neod1 > 336 [PSH, ACK] Seq=123507 Ack=1361 W
0.000289	163.634665	172.16.253.131	197.163.56.70	TCP	88		0.000289000	1047	336	neod1 > 336 [PSH, ACK] Seq=123523 Ack=1361 W
0.096907	163.731572	172.16.253.131	197.163.56.70	TCP	70		0.096907000	1047	336	neod1 > 336 [PSH, ACK] Seq=123557 Ack=1361 W
0.000410	163.731982	172.16.253.131	197.163.56.70	TCP	1514		0.000410000	1047	336	neod1 > 336 [ACK] Seq=123573 Ack=1361 Win=62
0.000068	163.732050	172.16.253.131	197.163.56.70	TCP	1514		0.000068000	1047	336	neod1 > 336 [ACK] Seq=125033 Ack=1361 Win=62
0.000052	163.732102	172.16.253.131	197.163.56.70	TCP	1514		0.000052000	1047	336	neod1 > 336 [ACK] Seq=126103 Ack=1361 Win=63

Figure 5.17: Backdoor's scenario without round trip time

- 4- Usually, the metrics that can be used for the matching include the packet number, the length of the packets, and RTT. However, we cannot use the RTT metric in all scenarios as described above. We have to use the packet number and the length of the packets. However, the study of (Yang & Lee, 2008) found , the lower bound of the amount needed to evade this detection is small. Therefore, we have to evade the session which has small number of packets (less than three packets) if it does not continue repeat itself several times to evade the FPR (Wu & Huang, 2007).
- 5- The difference in the time between the time stamp of the first segment of the first matched session and the time stamp of the last segment of the second matched session determines the time needed by this technique to detect the backdoor attack. .

From the findings above, the final design that can be used to detect the backdoor attack based on stepping stone approach concepts, where the matching between two sessions occurs, involves the following concepts:

1. Both sessions use the same port.
2. Both sessions are headed to the same destination.
3. Both sessions have the same number of packets.
4. Each packet in the first session has the same length of the corresponding packet (the same sequence) in the second session.
5. The session should be repeated several times if the number of packets is less than three packets to evade the FPR that may be caused by Scenario 2.

For the matching process, the metrics are the following: port number, destination address, number of packets, and the length for each packet, in addition to “who sent these packets,” which may be backdoor software.

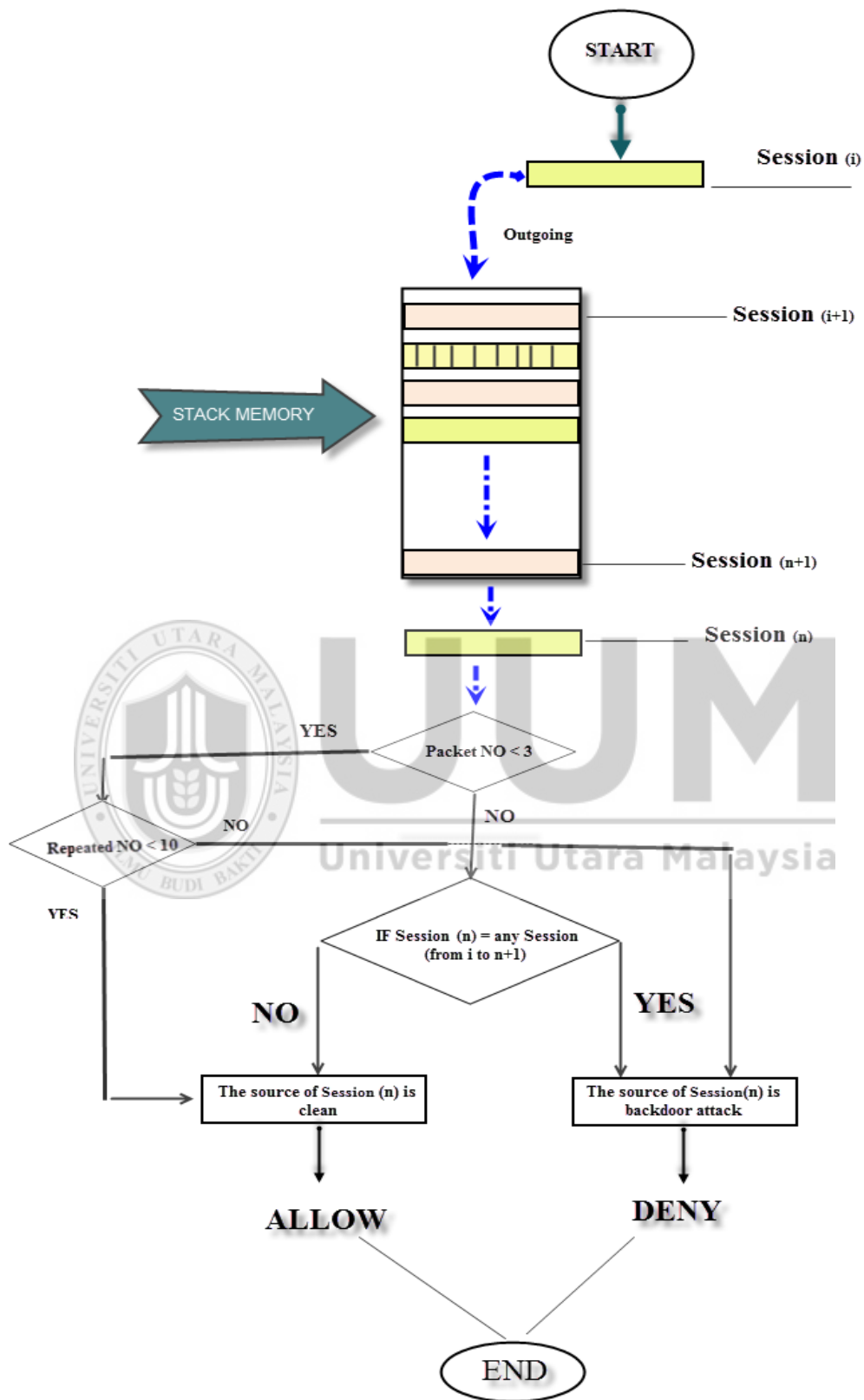


Figure 5.18: Detection Backdoor Attack Technique Based on Stepping Stone Approach

To determine the FNR, we captured the packets many times when the connection is clean, such as the use of trusted software like Internet Explorer, Yahoo Messenger, Whitesmoke, Skype, and Smart VoIP software. Furthermore, to evaluate the capability of the proposed approach, the results of the SSD technique and those of more than 45 IDS and antivirus programs were compared using Virustotal service

5.4 Results and Evaluation

The proposed method is very fast compared with other antivirus systems and IDS as shown in Figures 5.20, 5.21 and 5.22. Furthermore, most of the backdoor attacks detected by the proposed technique were not detected by the said antivirus systems or IDS as shown in Figures 4.5, 4.6 and 5.19.

In general, this technique has a high accuracy. In this study, the technique yielded a ratio of 100% in detecting the backdoor samples used. Table 5.1 shows the detection ratio results of the ten samples while the Table 5.4 shows the detection ratio results for four unique backdoors and Table 5.3 shows the TPR and FPR rates for ten known backdoors.

Date (UTC)	File / URL	Detection ratio
2013-11-20 10:15:27	Bifrost_un_pack.exe b36e4bc67349a1da1f01960202baf8bba92c2f3aa9159422e86326430bfdee0	10 / 47
2013-11-20 10:14:47	Bozok1.1.exe a5e3faab6b3a4fab9eb266ff74d278bf681c10b7a361f9e145429cbef5eb562	11 / 47
2013-11-20 10:13:52	Coffin Of Evil-Server 1.exe fe2e37672d47ea07baefaf11cbef2cf8f9c2c357993e940fb4481d0d86bf13e	9 / 47
2013-11-20 10:13:04	CyperGAT (2).exe 6c2e39dc3342b80603ad3f9992e98821786519b1dcb20a8520c4d8c5921286cb	11 / 47
2013-11-20 10:11:55	njrat041.exe ce595dbdef4df8bd252cb4c175b86a14489da97c3d4b195b8e211243e47da26b	9 / 47
2013-11-20 10:10:05	Poison.exe faff5cfe511d14aeffcc6bafbcf289fec666d4f0cd6d7803ab0d6de0c61e8	6 / 47
2013-11-20 10:09:22	SpY-NeT v2.6.exe 82ea1069810138913e47a9a8fe6882def9f2ac9efe3c7a2c84edf1d2e7f3036	7 / 46
2013-11-20 08:35:44	SubSeven v2.1R.exe	8 / 47

Figure 5.19: Virustotal detection result for the known samples

Table 5.1: The detection ratio result for the known backdoors

Backdoor samples	Virustotal Detection ratio	SSD Detection ratio
Bifrost un_pack	10 / 47	1/1
Bozok1.1	11 / 47	1/1
Coffin Of Evil-Server	9 / 47	1/1
Cyper-GAT	11 / 47	1/1
njrat041	9 / 47	1/1
Poison	6 / 47	1/1
SpY-NeT v2.6	7 / 46	1/1
SubSeven v2.1R	8 / 47	1/1
Sub7old.exe	12 / 47	1/1
XtremeRAT server	9 / 47	1/1
Total Ratio	(average = 9.2/47 = 19.57%)	100%

Table 5.2: The initial values for the detection result for 10 samples

METHOD	FP	FN	TP	TN	P	N
Antivirus and IDS (average)	0	8.042	1.95	5	10	5
SSD	0	0	10	5	10	5

According to the equations in Chapter Three, Section 3.7

Table 5.3: TPR and FPR for the 10 known backdoors

METHOD	TPR	FPR	ACC	PPV	NPV	BER
Antivirus and IDS	%19.57	0%	%46.38	100%	%38.33	%40.21
SSD	100%	0%	100%	100%	100%	0%

According to the equations in Chapter Three, Section 3.7

Table 5.4: The detection ratio result for the unique samples

Backdoor <i>unique</i> samples NO	Virustotal Detection ratio	SSD Detection ratio
4	0%	100%

For the speed (scan process time), the time used by Avira, Avast, and Eset Smart Security antivirus tools and that by the SSD technique are compared. For example, Avira antivirus needs 06:14:07 hours to complete the scan process as shown in Figure 5.20, and Eset Smart Security needs 05:56:42 hours to complete the scan process as shown in figure 5.21. By contrast, 20.29 Seconds is required by the SSD technique in detecting the samples, as described in Section 5.3 No as shown in Figure 5.22.

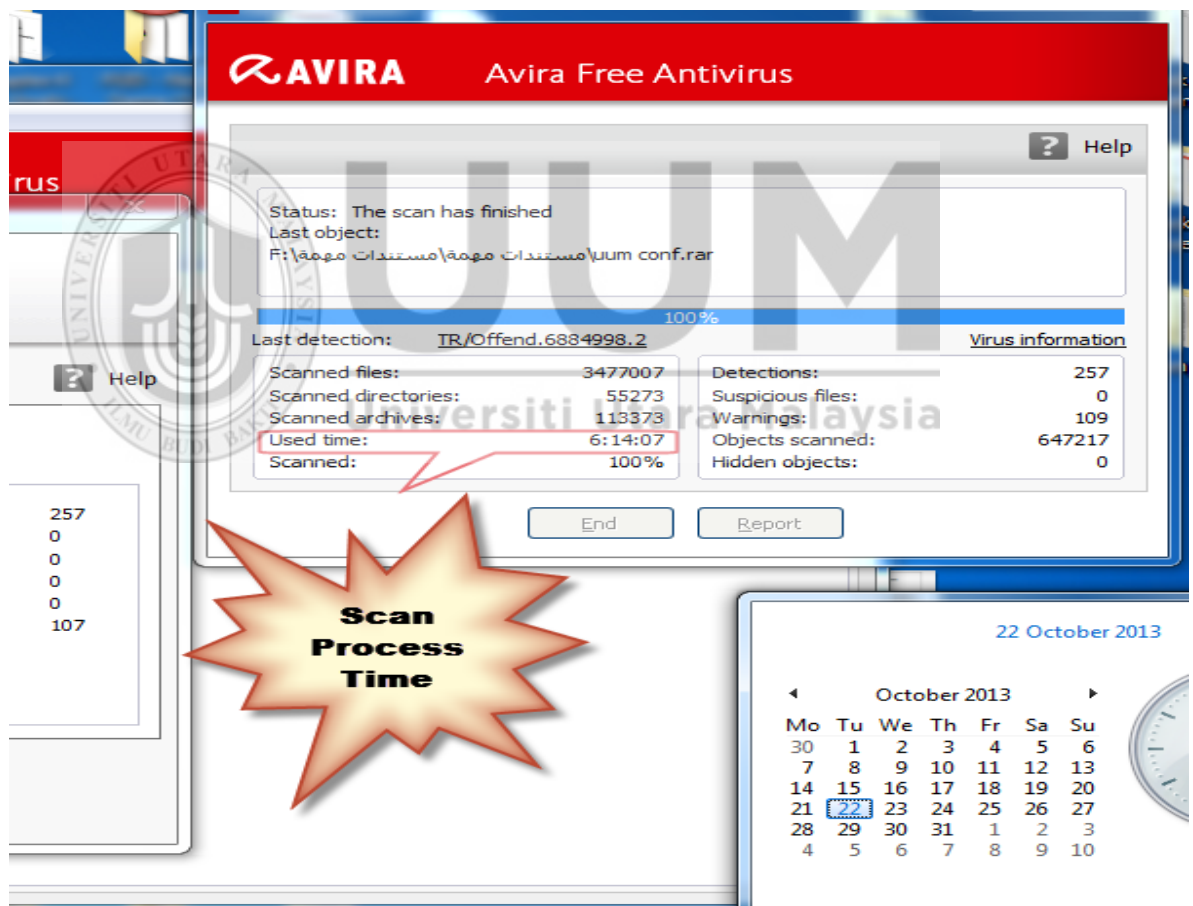


Figure 5.20: Avira Antivirus Scan Process Time

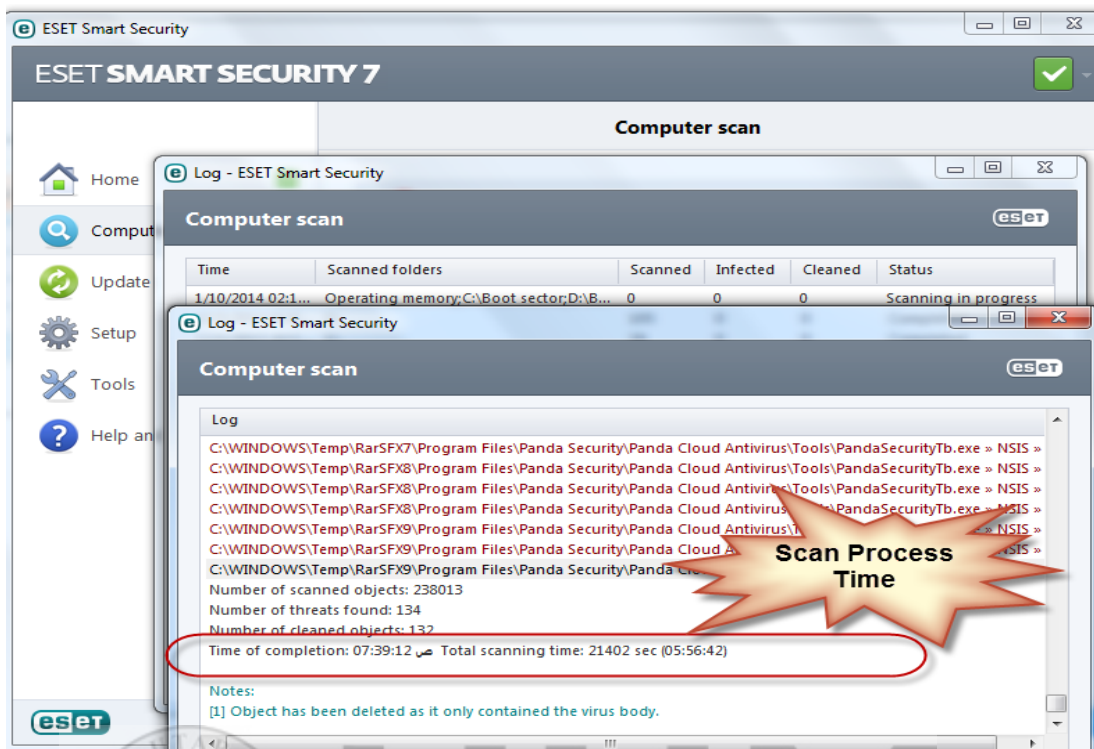


Figure 5.21: Eset Smart Security 7 Scan Process Time

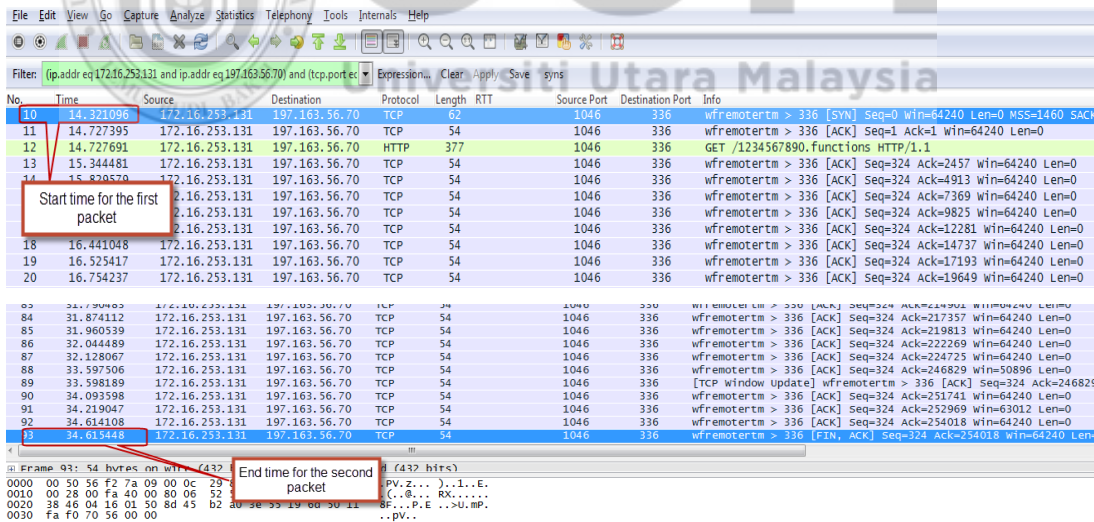


Figure 5.22: SSD Detection Time

CHAPTER SIX

CONCLUSION AND FUTURE WORK

6.1 Conclusion

This research considered the using of Stepping Stone Detection (SSD) approach in detecting backdoor attack to enhance the speed and accuracy of the detection and to reduce the huge storage resources used by traditional antivirus tools. The study clearly showed how the backdoor problem can be detected using the SSD approach and justified which SDD technique is the suitable solution to the backdoor attack problem. Furthermore, this research illustrated how the SSD approach has been developed to overcome the backdoor attack problem and displayed how the capability of the proposed SSD approach can be evaluated by a well-planned experiment.

Moreover, this study also illustrated why choosing a host-based architecture remains the best option. Furthermore, it justified why a new approach has to be proposed when various other approaches exist, such as the signature-based and anomaly-based detection approaches. As a result, this research proved that when the SSD approach is used, one gains a very high true positive with very low false negative rates.

6.2 Research Contributions

The contributions for this research include the following: first, the technique developed reduces the scan process time. The fast detection of the SSD approach makes backdoor detection faster and, at the same time, reduces the detection time. Therefore, the time gap between detection and response is reduced. Second, the technique enhances the accuracy of backdoor attack detection. This research uses an SSD-based technique that is in turn based on the use of interactive connections. As such, the technique is capable of detecting all backdoor types, which cannot be detected by traditional antivirus tools, either known or unknown, even when the backdoor is encrypted. Third, this technique reduces the huge storage resources used by traditional antivirus tools. Fourth, this research expands the field by extending the use of SSD-based techniques, which formerly have been used only in SSD-based environments, in backdoor environments. Last, this research provides new samples and collection of dataset that will be available online for researchers in the field.

6.3 Future Work

In future work, we can enhance the performance of this technique by using more samples. Moreover, the SSD theory remains to be developed alongside an integrated system to detect most of the threats to network security, such as spam and proxy.

REFERENCES

- Agrawal, H., Alberi, J., Bahler, L., Conner, W., Micallef, J., Virodov, A., & Snyder, S. R. (2010). *Preventing insider malware threats using program analysis techniques*. Paper presented at the MILITARY COMMUNICATIONS CONFERENCE, 2010-MILCOM 2010.
- Balzarotti, D., Cova, M., Karlberger, C., Kruegel, C., Kirda, E., & Vigna, G. (2010). *Efficient detection of split personalities in malware*. Paper presented at the Network and Distributed System Security Symposium (NDSS).
- Banerjee, U., Vashishtha, A., & Saxena, M. (2010). Evaluation of the Capabilities of WireShark as a Tool for Intrusion Detection. *International Journal of Computer Applications*, 6(7).
- Borders, K., Zhao, X., & Prakash, A. (2006). *Siren: Catching evasive malware*. Paper presented at the Security and Privacy, 2006 IEEE Symposium on.
- Choi, B., & Cho, K. (2012). Detection of Insider Attacks to the Web Server. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, 3(4), 35-45.
- Choi, W. S., & Choi, S. G. (2013). *An enhanced method for mitigation of network traffic using TCP signalling control*. Paper presented at the Advanced Communication Technology (ICACT), 2013 15th International Conference on.
- Crawford, M., & Peterson, G. (2013). *Insider Threat Detection using Virtual Machine Introspection*. Paper presented at the System Sciences (HICSS), 2013 46th Hawaii International Conference on.
- Decloedt, H. E., & Van Heerden, R. (2010). Rootkits, Trojans, backdoors and new developments.
- Dittmann, J., Karpuschewski, B., Fruth, J., Petzel, M., & Munder, R. (2010). *An exemplary attack scenario: threats to production engineering inspired by the Conficker worm*. Paper presented at the Proceedings of the First International Workshop on Digital Engineering.

- G. T. I. S. Center. (n.d.). Open Malware Retrieved July 13 2013, from <http://oc.gtisc.gatech.edu:8080>
- Gribble, S., Levy, H., Moshchuk, A., & Bragin, T. (2013). Detection of spyware threats withn virtual machine. : US Patent 20,130,014,259.
- Idika, N., & Mathur, A. P. (2007). A survey of malware detection techniques. *Purdue University*, 48.
- Kampasi, A., Zhang, Y., Di Crescenzo, G., Ghosh, A., & Talpade, R. (2007). *Improving stepping stone detection algorithms using anomaly detection techniques*.
- Kang, B., Kim, H. S., Kim, T., Kwon, H., & Im, E. G. (2011). *Fast malware family detection method using control flow graphs*. Paper presented at the Proceedings of the 2011 ACM Symposium on Research in Applied Computation.
- Kuo, Y.-W., & Huang, S.-H. (2008). *An Algorithm to Detect Stepping-Stones in the Presence of Chaff Packets*. Paper presented at the Parallel and Distributed Systems, 2008. ICPADS'08. 14th IEEE International Conference on.
- Kurose, J. F., & Ross, K. W. (2012). *Computer networking*: Pearson Education.
- Li, P. (2011). *Detecting stepping stones in internet environments*. Victoria: Deakin University.
- Li, P., Zhou, W., & Wang, Y. (2010). *Getting the real-time precise round-trip time for stepping stone detection*. Paper presented at the Network and System Security (NSS), 2010 4th International Conference on.
- Maarof, M. A., & Osman, A. H. (2012). Malware Detection Based on Hybrid Signature Behaviour Application Programming Interface Call Graph. *American Journal of Applied Sciences*, 9.
- Menahem, E., Shabtai, A., Rokach, L., & Elovici, Y. (2009). Improving malware detection by applying multi-inducer ensemble. *Computational Statistics & Data Analysis*, 53(4), 1483-1494.
- Microsoft. (2012). Microsoft Security Intelligence Report "*WORLDWIDE THREAT ASSESSMENT*" (Vol. 13): Technical Report.

- Mila. (2013). Contagio Malware Dump Retrieved Sep 30, 2013, from <http://contagiodump.blogspot.com/2013/04/collection-of-pcap-files-from-malware.html#more>
- Modi, C., Patel, D., Borisaniya, B., Patel, H., Patel, A., & Rajarajan, M. (2012). A survey of intrusion detection techniques in cloud. *Journal of Network and Computer Applications*.
- Mohan, R. (2013). Network Analysis and Application Control Software based on Client-Server Architecture. *arXiv preprint arXiv:1304.5015*.
- Mudzingwa, D., & Agrawal, R. (2012). *A study of methodologies used in intrusion detection and prevention systems (IDPS)*. Paper presented at the Southeastcon, 2012 Proceedings of IEEE.
- NETRESEC. (2010, 2013). NETRESEC Retrieved November, 01, 2013, from <http://www.netresec.com>
- Ni, L., Yang, J., Zhang, R., & Song, D. (2008). *Matching TCP/IP Packets to Resist Stepping-Stone Intruders' Evasion*. Paper presented at the System Theory, 2008. SSST 2008. 40th Southeastern Symposium on.
- Omar, M. N. (2005). *The Optimization of Stepping Stone Detection Algorithm in Intrusion Detection System* Master Universiti Teknologi Malaysia, Skudai, Johor,.
- Omar, M. N. (2011). *Approach for Solving Active Perturbation Attack problem in Stepping Stone Detection*. PHD, Universiti Sains Malaysia, Malaysia (USM) Penang.
- Omar, M. N., Amphawan, A., & Din, R. (2012). Evolution of Stepping Stone Detection and Emerging Applications. *11 WSEAS International Conference on Information Security and Privacy (ISP'12)*.
- Omar, M. N., Amphawan, A., & Din, R. (2013). A Stepping Stone Perspective to Detection of Network Threats.
- Paxson, V., & Zhang, Y. (2000). *Detecting backdoors*. Paper presented at the Proc. of 9th USENIX Security Symposium.

- Ping, L., Wanlei, Z., & Yini, W. (2010, 1-3 Sept. 2010). *Getting the Real-Time Precise Round-Trip Time for Stepping Stone Detection*. Paper presented at the Network and System Security (NSS), 2010 4th International Conference on.
- Prasad, M. S., Babu, A. V., & Rao, M. K. B. (2013). An Intrusion Detection System Architecture Based on Neural Networks and Genetic Algorithms. [International Journal of Computer Science and Management Research]. *International Journal of Computer Science and Management Research*, 2.
- Radmand, A. (2009). A ghost in software Retrieved sep, 21, 2013, from <http://cs.columbusstate.edu/cae-ia/StudentPapers/radmand.azadeh.pdf>
- Salimi, E., & Arastouie, N. (2011). *Backdoor Detection System Using Artificial Neural Network and Genetic Algorithm*. Paper presented at the Computational and Information Sciences (ICCIS), 2011 International Conference on.
- Sathyanarayan, V., Kohli, P., & Bruhadeshwar, B. (2008). *Signature generation and detection of malware families*. Paper presented at the Information Security and Privacy.
- Shabtai, A., Kanonov, U., Elovici, Y., Glezer, C., & Weiss, Y. (2012). "Andromaly": a behavioral malware detection framework for android devices. *Journal of Intelligent Information Systems*, 1-30.
- Shullich, R., Chu, J., Ji, P., & Chen, W. (2011). A Survey of Research in Stepping-Stone Detection. *International Journal of Electronic Commerce*, 2(2).
- Siddiqui, M., Wang, M. C., & Lee, J. (2008). *A survey of data mining techniques for malware detection using file features*. Paper presented at the Proceedings of the 46th Annual Southeast Regional Conference on XX.
- Sobh, T. (2008). *Novel algorithms and techniques in telecommunications, automation and industrial electronics*: Springer.
- Sonawane, S., Prasad, G., & Pardeshi, S. (2012). A survey on intrusion detection techniques. *World Journal of Science and Technology*, 2(3).

- Soni, C. (2013). Capturing of HTTP protocol packets in a wireless network. *International Journal of Wired and Wireless Communications*, 1(2), 5-10.
- Sukwong, O., Kim, H. S., & Hoe, J. C. (2011). Commercial antivirus software effectiveness: an empirical study. *Computer*, 63-70.
- Tahan, G., Rokach, L., & Shahar, Y. (2012). Mal-ID: Automatic Malware Detection Using Common Segment Analysis and Meta-Features. *The Journal of Machine Learning Research*, 98888, 949-979.
- Virustotal. (2013). VirusTotal Retrieved July 13, 2013, from <https://www.virustotal.com/>
- VMware. Inc. (2013). VMware software Retrieved Oct 19, 2013, from <https://www.vmware.com/ap>
- W. Foundation. (2013). Wireshark Retrieved July 13, 2013, from <http://www.wireshark.org/>
- Waksman, A., & Sethumadhavan, S. (2011). *Silencing hardware backdoors*. Paper presented at the Security and Privacy (SP), 2011 IEEE Symposium on.
- Wang, X., & Reeves, D. (2011). Robust correlation of encrypted attack traffic through stepping stones by flow watermarking. *Dependable and Secure Computing, IEEE Transactions on*, 8(3), 434-449.
- Welch, V., Pearson, D., Tierney, B., & Williams, J. (2012). Security at the Cyber Border: Exploring Cybersecurity for International Research Network Connections.
- Wu, H.-C., & Huang, S.-H. (2007). *Detecting stepping-stone with Chaff perturbations*. Paper presented at the Advanced Information Networking and Applications Workshops, 2007, AINAW'07. 21st International Conference on.
- Yang, J., & Lee, B. (2008). Detecting Stepping-Stone Intrusion and Resisting Evasion through TCP/IP Packets Cross-Matching *Autonomic and Trusted Computing* (pp. 2-12): Springer.
- Zhang, Y., & Paxson, V. (2000). *Detecting stepping stones*. Paper presented at the Proceedings of the 9th USENIX Security Symposium.

PUBLICATIONS

- 1) Alminshid, K., & Omar, M. N. (2013, 23-25 Sept). *Detecting backdoor using stepping stone detection approach*. Paper presented at the Informatics and Applications (ICIA), 2013 Second International Conference on, Lodz, Poland, published by the IEEE Xplore, index by the (ICIA),2013 Second International Conference Proceeding.
- 2) Basha, A. D., Mnaath, S. H., Alminshid, K., & Umar, I. N. (2013). Importance Applications Mobile Agent technology for Virtual E-learning Environment: Proposed Model. *International Journal of Enhanced Research in Science Technology & Engineering*, 2(5), (24-28), 2319-7463.
- 3) Mnaath, S. H., Basha, A. D., Alminshid, K., & Jamaludin, R. (2013). The Opportunities and Difficulties for M-learning to Enhancing students learning results. *International Journal of Enhanced Research in Science Technology & Engineering*, 2(5), (24-28), 2319-7463.