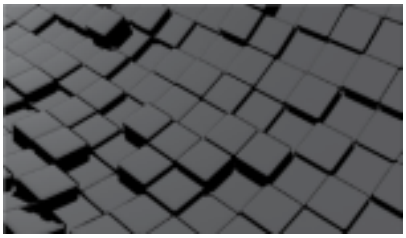
Kali Linux Penetration Testing BIBLE

Gus Khawaja

WILEY



Kali Linux Penetration Testing Bible

Gus Khawaja

Published simultaneously in Canada

ISBN: 978-1-119-71908-3
ISBN: 978-1-119-71964-9 (ebk)
ISBN: 978-1-119-71907-6 (ebk)

Manufactured in the United States of America

Library of Congress Control Number: 2021904285

About the Author

Gus Khawaja is an expert in application security and penetration testing. He is a cybersecurity consultant in Montreal, Canada, and has a depth of experience working with organizations to protect their assets from cyberattacks. He is a published author and an online instructor in the field of cybersecurity.

About the Technical Editor

Corey Ball is a cybersecurity expert with more than 10 years of IT and cybersecurity leadership experience. He specializes in penetration testing APIs, web applications, and networks. He currently has more than 10 cybersecurity certifications including the OSCP, CISSP, CISM, and CCISO. Corey is a cyberse
curity career mentor for Cybrary and the author of the upcoming book *Hacking APIs*. He has a bachelor of arts in English and philosophy from CSU Sacramento.

Acknowledgments

I have been fortunate to share my knowledge and expertise with my audience

through Wiley. I hope this knowledge will make you the best expert in your career as a penetration tester.

I am especially grateful to my family who supported me to deliver 18 full chapters of this book. A full year of nonstop writing takes a lot of guts, but it's here for you, so take advantage of it.

I am blessed that I have background experience in programming that helped me a lot in my career as a penetration tester and as an application security expert. You will realize that these days, having skills in web application architecture will allow you to master this career.

Finally, I would like to thank Wiley's team members who supported me dur ing the journey of writing this amazing book. Without this support, this book would never have seen the light of day!

Contents at a Glance

Contents

Introduction

Kali is a popular Linux distribution used by security professionals and is becoming an important tool for daily use and for certifications. Penetration testers need to master Kali's hundreds of tools for pentesting, digital forensics, and reverse engineering. *Kali Linux Penetration Testing Bible* is a hands-on guide for getting the most from Kali Linux for pentesting. This book is for working cybersecu
rity professionals in offensive, hands-on roles, including red teamers, white hat hackers, and ethical hackers. Defensive specialists will also find this book valuable, as they need to be familiar with the tools used by attackers.

This comprehensive pentesting book covers every aspect of the art and science of penetration testing. It covers topics like building a modern Docker ized environment, the basics of bash language in Linux, finding vulnerabilities in different ways, identifying false positives, and practical penetration testing workflows. You'll also learn to automate penetration testing with Python and dive into advanced subjects like buffer overflow, privilege escalation, and beyond. By reading this book, you will:

- Gain a thorough understanding of the hundreds of penetration testing tools available in Kali Linux.
- Master the entire range of techniques for ethical hacking so you can be more effective in your job and gain coveted certifications.
- Learn how penetration testing works in practice and fill the gaps in your knowledge to become a pentesting expert.
- Discover the tools and techniques that hackers use so you can boost your network's defenses.

What Does This Book Cover?

This book goes deep into the subject of penetration testing. For established penetration testers, this book fills all the practical gaps, so you have one complete resource that will help you as your career progresses. For newcomers to the field, *Kali Linux Penetration Testing Bible* is your best guide to how ethical hacking really works.

Chapter 1: Mastering the Terminal Window

This chapter outlines the in and outs of the Linux system Terminal window and covers how to manage the file system like the pros. You will learn how to manage users and groups inside Kali, and you will see how to manipulate files and folders during your engagements and much more.

Chapter 2: Bash Scripting

Bash scripting is an essential skill for a penetration tester. In this chapter you will learn how to start to use programming principles such as variables, functions, conditions, loops, and much more.

Chapter 3: Network Hosts Scanning

This chapter teaches you how to conduct network scans like professionals. You will learn first about the basics of networking, and then you will delve deep into the port scanning techniques.

Chapter 4: Internet Information Gathering

This chapter discusses the passive information gathering phase in penetration testing. You will be introduced to how to deal with advanced search engine queries. Also, you will learn how to use Shodan and other tools to get the job done.

Chapter 5: Social Engineering Attacks

This chapter focuses on how to take advantage of human weakness to exploit organizations. You will learn about how to send phishing emails and steal credentials. On top of that, you will see how to use the Social Engineer Toolkit as a penetration tester. Finally, you will see how USB Rubber Ducky operates in similar SE attacks.

Chapter 6: Advanced Enumeration Phase

This chapter reviews how to handle the enumeration phase in a penetration testing engagement. Enumeration means collecting the necessary information that will allow us to exploit the specific service (e.g., FTP, SSH, etc.).

Chapter 7: Exploitation Phase

This chapter discusses some actual attacks and shows you how to get inside the systems. In the previous chapters, you had all the information about each service, and in this one, we will take this step further and exploit the vulnerabilities.

Chapter 8: Web Application Vulnerabilities

This chapter focuses on the basics of web application vulnerabilities. The goal is to allow you test web applications with ease during your engagements. Every company has a website these days, and it's crucial to understand this topic from A to Z.

Chapter 9: Web Penetration Testing and Secure Software Development Lifecycle

In this chapter, you will mainly learn about the methodology of web application penetration testing and how to use Burp Suite Pro. Finally, you will see how to implement a secure software development lifecycle (SSDLC) in an organization.

Chapter 10: Linux Privilege Escalation

This chapter focuses mainly on Linux operating system privilege escalation. The techniques in this chapter will allow you to gain root privileges on a com promised Linux OS.

Chapter 11: Windows Privilege Escalation

This chapter describes how to get administrator privileges on the compromised Windows OS. First you will learn about how to enumerate the Windows OS, and then you will see how to exploit the Windows system with practical examples.

Chapter 12: Pivoting and Lateral Movement

This chapter describes how to use the pivoting techniques to move laterally on the compromised network. In this chapter, you will learn how Windows hashes work under the hood and how to reuse admin credentials to get the job done.

Chapter 13: Cryptography and Hash Cracking

This chapter describes how to crack hashes during your engagements using Hashcat. Before starting on the cracking topic, you will learn about the basics of cryptography including hashing and encryption.

Chapter 14: Reporting

This chapter explains how to present professional penetration testing reports. Also, you will learn how to evaluate accurately the severity of your findings.

Chapter 15: Assembly Language and Reverse Engineering

This chapter will introduce you to the concept of reverse engineering using the assembly language. You will learn about the basics of the assembly language including registers, assembly instructions, memory segments, and much more.

Chapter 16: Buffer/Stack Overflow

This chapter will use what you learned in the previous chapter to exploit the stack using the buffer overflow technique.

Chapter 17: Programming with Python

This chapter discusses the basics of Python version 3. This programming language is the choice of hackers, so you should learn it too.

Chapter 18: Pentest Automation with Python

This chapter focuses on the automation of the penetration testing phases using the Python language. You will see a complete practical example that can use in your career.

Appendix A: Kali Linux Desktop at a Glance

This appendix focuses on how to manage the interface of the Kali Linux desktop environment. You will learn how to handle this operating system with ease and customize it to your liking.

Appendix B: Building a Lab Environment Using Docker

This appendix will delve deep with Docker, and you will see how images and containers work in practice. Both Docker and hypervisor technologies facilitate the creation of a live lab so we, penetration testers, can have fun with it.

Companion Download Files

As you work through the examples in this book, you may choose either to type in all the code manually or to use the source code files that accompany the book. All the source code used in this book is available for download from `www.wiley .com/go/kalilinuxpenbible`.

How to Contact the Publisher

If you believe you've found a mistake in this book, please bring it to our attention. At John Wiley & Sons, we understand how important it is to provide our customers with accurate content, but even with our best efforts an error may occur.

To submit your possible errata, please email it to our Customer Service Team at `wileysupport@wiley.com` with the subject line "Possible Book Errata Submission."

How to Contact the Author

We appreciate your input and questions about this book! Email the author at `gus.khawaja@guskhawaja.me`, or message him on Twitter at `@GusKhawaja`.

1

Mastering the Terminal Window

Kali Linux can be summarized in two simple words: terminal window. If you master your terminal window skills, then you will be one of those elite ethical hackers. In this chapter, you will learn all the essentials of the terminal window so you can start using Kali Linux like a boss.

If you already know how to manage the terminal window, please use this chapter as a reference, or maybe go over it quickly in case there is something new that you haven't learned before. The main goal of this chapter is not only

to show you the commands of Kali Linux but to help you deeply understand it through practical examples.

Kali Linux is a Debian-based operating system developed by Offensive Security, so if you're used to Ubuntu, for example, the commands in the terminal window will look the same since Debian and Kali share an equal distribution. Here's what this chapter covers:

- Kali Linux file system
- Terminal window basics
- Managing users and groups
- Manipulating files and folders
- Handling remote connections
- Kali Linux system management
- Dealing with networking in Kali Linux

Kali Linux File System

Understanding the structure of the file system in Kali Linux is crucial. The directory structure of your Kali OS is based on the Unix Filesystem Hierarchy Standard (FHS), and that's how the directories are structured inside Kali Linux. In Windows, the root directory is `C:\`, but in Kali Linux, it's a forward slash (`/`). Do not confuse the term *root directory* with the root user's home directory, which is `/root`, because they are two different things; the latter is the home directory for the root user. Speaking about the root user, it's essential to understand that this user is the equivalent to the Administrator user on Windows operating systems. In the Kali 2020.1 release, Offensive Security introduced the nonroot user by default, which means that you'll need to execute the `sudo` command if you want to run high-privilege tools.

To get a visual representation of the Kali Linux file system directories, open the terminal window and execute the `ls` command to list the contents of the root system directory. Take note that by default you will be in the user home directory. To change it, you must execute the `cd /` command:

```
kali@kali:~$ cd /
kali@kali:/$ ls
bin boot dev etc home initrd.img initrd.img.old lib lib32 lib64  libx32
lost+found media mnt opt proc root run sbin srv sys  tmp usr var vmlinuz
vmlinuz.old
```

- `/bin` (binaries): This directory holds Linux binaries like the `ls` command that we executed earlier.
- `/sbin` (system binaries): This directory contains system binary files that serve as administrative commands (e.g., `fdisk`).
- `/boot`: This directory contains the Linux bootloader files.

- `/dev` (devices): This directory contains the device configuration files (e.g., `/dev/null`).
- `/sys`: This is similar to `/dev`, which contains configurations about devices and drivers.
- `/etc` (etcetera): This directory contains all the administration system files (e.g., `/etc/passwd` shows all the system users in Kali).
- `/lib` (libraries): This directory contains the shared libraries for the binaries inside `/bin` and `/sbin`.
- `/proc` (processes): This directory holds the processes and kernel information files.

- `/lost+found`: As the name says, this directory contains the files that have been recovered.
- `/mnt` (mount): This directory contains the mounted directories (e.g., a remote file share).
- `/media`: This directory holds the removable media mounted directories (e.g., DVD).
- `/opt` (option): This directory is used for add-on software package instal lation. Also, it is used when installing software by users (e.g., hacking tools that you download from GitHub).
- `/tmp` (temporary): This is a temporary folder used temporarily; the con tents are wiped after each reboot. The `tmp` folder is a good place to down load your tools for privilege escalation once you get a limited shell.
- `/usr` (user): This directory contains many subdirectories. In fact, `/usr/share` is a folder that you need to memorize because most of the tools that you use in Kali Linux (e.g., Nmap, Metasploit, etc.) are stored there, and it contains the wordlists dictionary files (`/usr/share/wordlists/`).
- `/home`: This is the home for Kali Linux users (e.g., `/home/john/`).
- `/root`: This is the root user home directory.
- `/srv` (serve): This folder holds some data related to system server func tionalities (e.g., data for FTP servers).
- `/var` (variable): This folder holds variable data for databases, logs, and websites. For example, `/var/www/html/` contains the files for the Apache web server.
- `/run` (runtime): This directory contains runtime system data (e.g., cur rently logged-in users).

Terminal Window Basic Commands

There are lots of common commands that we use as penetration testers on a daily basis. Many of these commands will be listed in the upcoming sections or later in this book. In this section, you will see all the general standard tools that I personally use frequently. You will also learn the basic commands that are identified for general use.

First, to open the terminal window from the desktop, you can use the Ctrl+Alt+T key combination instead of opening the application from its icon using the mouse cursor.

If you want to get help for any command that you want to execute, just append `-h` or `--help` to it (some commands require you to use only one of them). For

example, if you want to see the different options for the `cat` command, just type `cat --help` in your terminal window to get all the help needed regarding this tool. In the next command (`cat -h`), you'll see that the `-h` option does not work for the `cat` command. Instead, I used the `--help` option. (The `cat` command is used frequently to display the contents of a text file in the terminal window.)

```
kali@kali:~$ cat -h
cat: invalid option -- 'h'
Try 'cat --help' for more information.
kali@kali:~$ cat --help
Usage: cat [OPTION]... [FILE]...
Concatenate FILE(s) to standard output.

With no FILE, or when FILE is -, read standard input.

 -A, --show-all equivalent to -vET
 -b, --number-nonblank number nonempty output lines, overrides -n
-e equivalent to -vE
 -E, --show-ends display $ at end of each line
 -n, --number number all output lines
 -s, --squeeze-blank suppress repeated empty output lines  -t
equivalent to -vT
 -T, --show-tabs display TAB characters as ^I
 -u (ignored)
 -v, --show-nonprinting use ^ and M- notation, except for LFD and TAB
--help display this help and exit
 --version output version information and exit

Examples:
 cat f - g Output f's contents, then standard input, then g's
contents.
 cat Copy standard input to standard output.

GNU coreutils online help:
<https://www.gnu.org/software/coreutils/> Full documentation at:
<https://www.gnu.org/software/coreutils/cat> or available locally
via: info '(coreutils) cat invocation'
```

To clear the terminal window text, execute the `clear` command or press Ctrl+L to get the job done.

To open a new terminal window tab, from your current terminal session press Ctrl+Shift+T.

To complete the input (e.g., a filename or a command name) automatically,

I use the Tab key. What if multiple files start with the same text? Then, if you hit Tab twice, the terminal window will display all the options in place. (The best way to understand this chapter is to open the terminal window and practice while reading the instructions.)

Let's look at an example. In my home directory, I have two files, `test.sh` and `test.txt`. Once I start typing `cat tes`, I hit Tab once, and it shows me `cat`

`test.`. This means I have multiple files with the same name. Then I hit Tab twice, and it shows me the list of files in the current directory. Finally, I can open the desired file, which is `test.txt`:

```
root@kali:~# cat test.
Test.sh test.txt
root@kali:~ cat test.txt
test
```

To stop the execution of any tool while it's running, you can use the Ctrl+C shortcut to stop it.

To exit the terminal window and close it, use the `exit` command or press Ctrl+D to get the job done.

To restart Kali Linux from the terminal window, you must use the `reboot` command, and to shut it down, you must use the `poweroff` command. Now, to get the list of executed recent commands, you'll have to use the `his tory` command.

In Linux, you must understand that we use a lot of redirection in the terminal window. For example, to save the output of the `ls` command into a file, I can redirect the output from the terminal window to a text file using the `>` (greater than) character:

```
kali@kali:~$ ls > ls_file.txt
kali@kali:~$ cat ls_file.txt
Desktop
Documents
Downloads
ls_file.txt
Music
Pictures
Public
Templates
Videos
```

Now, you can do the opposite by redirecting (printing) the text file contents into the terminal window using the `<` (less than) character:

```
kali@kali:~$ cat < ls_file.txt
Desktop
Documents
Downloads
ls_file.txt
Music
Pictures
Public
Templates
Videos
```

Another redirection that you need to be aware of is the commands pipe. In summary, you can combine the output of each command and send it to the next one using the | character:

```
$command 1 | command2 | command3 ...
```

For example, I will read a file, then sort the results, and finally use the `grep` command to filter out some text strings (the goal is to extract the files that start with the word *test*):

```
kali@kali:~$ cat ls_file.txt | sort | grep test
test.sh
test.txt
```

Tmux Terminal Window

Tmux is a particular terminal window that allows you to manage multiple windows in your current terminal session. The best way to explain it is through examples.

Starting Tmux

To start Tmux, you just type `Tmux` in your terminal window. At the bottom of your terminal window, you'll notice that a number and a name have been assigned to your opened window tab, as shown in Figure 1.1.

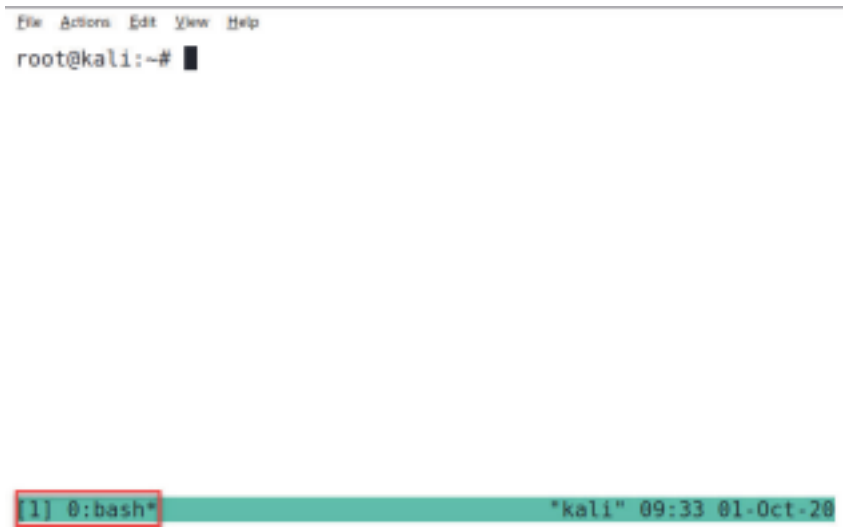


Figure 1.1: Tmux New Window

So what? Let's say you're in an engagement and you want to run Nmap in one window, plus run Metasploit in another one, and so on. This is where Tmux is handy, because you can work on multiple windows/sessions at the same time.

Tmux Key Bindings

In Tmux, you must use Ctrl+B to instruct it that you want to execute a Tmux action (command). In fact, the key combination Ctrl+B is the default one. You can always change the default configurations of Tmux in the configuration file. To change this behavior and assign Ctrl+A instead of Ctrl+B, then you must create the config file yourself for the first time. To get the job done, you have two options for creating a config file in Tmux. The first way is to add a user specific file called `~/.tmux.conf`, and the second way is to add a global file (to all users) under `/etc/tmux.conf`. In my case (for this example), I will add the configuration file under `/etc/tmux.conf` (and I will add the configurations for the key bindings in it):

```
root@kali:/# touch /etc/tmux.conf
root@kali:/# echo unbind C-b >> /etc/tmux.conf
root@kali:/# echo set -g prefix C-a >> /etc/tmux.conf
root@kali:/# echo bind C-a send-prefix >> /etc/tmux.conf
```

Tmux Session Management

In Figure 1.1, you can see that the name `bash` has been assigned automatically to your current session.

Window Rename

To rename the session, press Ctrl+B first (or Ctrl+A if you made the changes in the config files that we did previously). Then remove your fingers from the keyboard and press the comma (,) key on your keyboard. You should see that the prompt has changed to allow you to rename it. I will call it Window1; then press Enter after finishing the task:

```
(rename-window) Window1
```

Window Creation

At this stage, we have only one window, so let's create a second one by press ing Ctrl+B and then pressing the C key. Looking at the bottom, you'll see you have a new bash window, and Tmux has already highlighted the current tab with an asterisk (*), as shown in Figure 1.2.

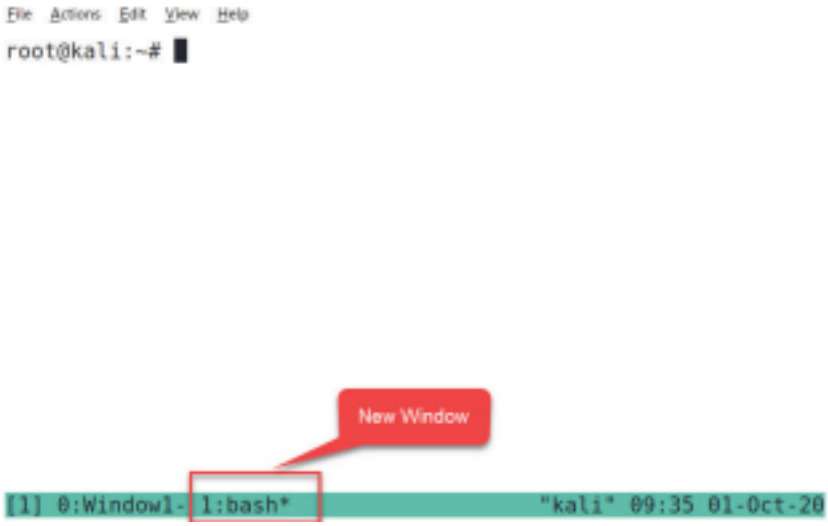


Figure 1.2: New Tmux Highlighted Tab

Splitting Windows

To split the selected tab into two subwindows side by side, as shown in Figure 1.3, you must press Ctrl+B and then enter the % character on your keyboard (remember that you need to press Shift+% or else it will be considered 5 on your keyboard).

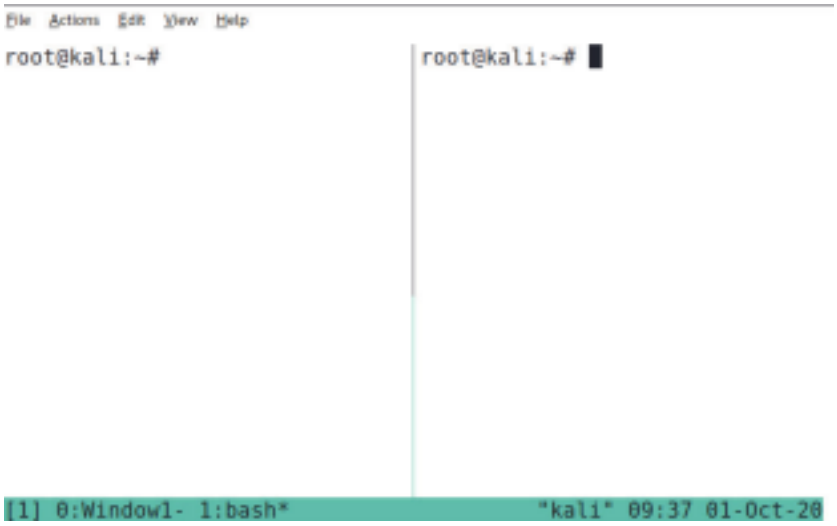


Figure 1.3: Tmux Vertical Windows Side by Side

Navigating Inside Tmux

Amazing, right? As you can see, the cursor is on the right pane (see Figure 1.3). To switch between panes (subwindows), press Ctrl+B and use the arrow keys on your keyboard (to change left, right, up, and bottom).

Next, go back to the Window1 session. To switch between windows, press Ctrl+B and then the number of the window (which is 0 according to this

example), and we should be back to the first window.

Now, divide the window into two sections, one over the other, as shown in Figure 1.4. To get this done, use Ctrl+B and then the double quote (“). Remember that you need to press Shift+“ or else that key produces a single quote.

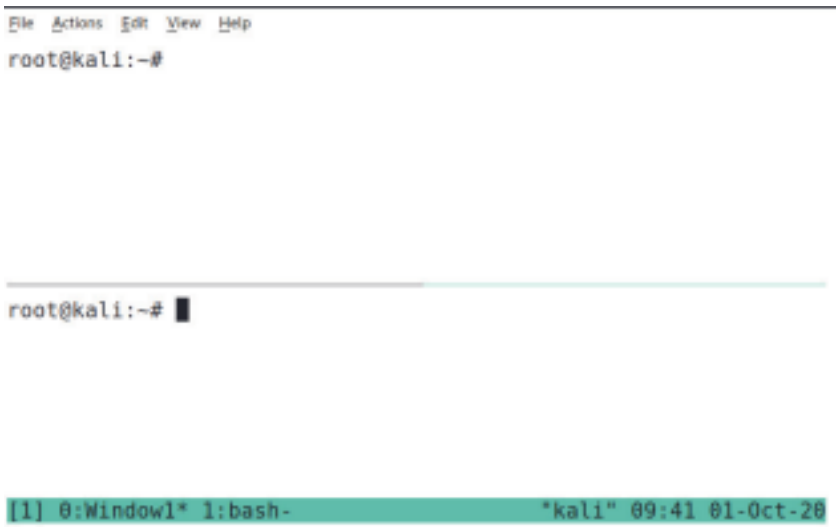


Figure 1.4: Tmux Horizontal Windows

The final tip for managing Tmux is for scrolling up and down inside a window or a pane session. In fact, you can’t use your mouse to scroll up and down in a Tmux session (the mouse scrolling is for commands history). Instead, you need to press Ctrl+B and then [on your keyboard, and after that you can use the up and down arrow keys on your keyboard to scroll. Once you’re done with the scrolling, press Esc or the Q key on your keyboard to go back to the normal mode.

To close a pane or a window, just use `exit` like with any regular terminal window session.

Tmux Commands Reference

Table 1.1 summarizes all the Tmux commands that you learned in this section. You can use it as a reference (this is just a quick guide so you can start using Tmux; if you want to go beyond basics, check the manual reference).

Table 1.1: Tmux Keyboard Shortcuts

To rename a window Ctrl+B+,

To open a new window Ctrl+B+C

To split windows vertically Ctrl+B+%

To split windows horizontally Ctrl+B+"

To navigate subwindows Ctrl+B+Left Arrow, Ctrl+B+Right Arrow To switch between

windows Ctrl+B+[window number] To scroll up Ctrl+B+[+Up Arrow

To scroll down Ctrl+B+[+Down Arrow

To escape the scrolling mode Esc

To close a pane/window Type **`exit`** (inside it)

Managing Users and Groups in Kali

Understanding the commands for managing users and groups is important because you'll use the information when you learn about privilege escalation later in the book. All the commands in this chapter will help you a lot in your engagements while using Kali Linux (as an OS for your pentests).

Figure 1.5 summarizes all the commands related to users' management/ security in Kali Linux.

Users Commands

Low-privilege users must prepend commands with `sudo` to execute system commands (and the low-privilege user must be in the `sudo` group to execute `sudo`). You will be asked for your account password if you want to use the `sudo` command. For example, if you want to execute the `fdisk` system tool to show the Kali-attached devices, use the following command:

```
root@kali:~# fdisk -l
Disk /dev/sda: 80 GiB, 85899345920 bytes, 167772160 sectors
Disk model: VMware Virtual S
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x7c02676c

Device Boot Start End Sectors Size Id Type
/dev/sda1 * 2048 165771263 165769216 79G 83 Linux
```



```
/dev/sda2 165773310 167770111 1996802 975M 5 Extended /dev/sda5
165773312 167770111 1996800 975M 82 Linux swap /  Solaris
```

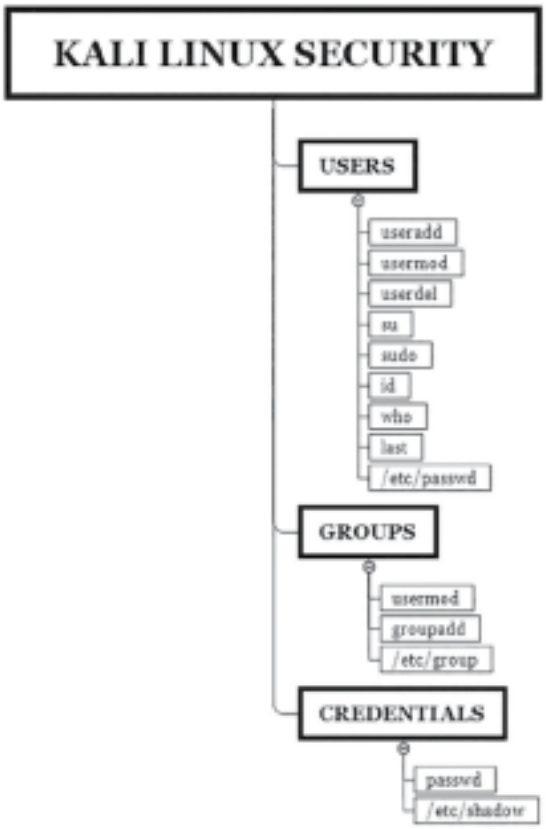


Figure 1.5: Kali Linux OS Security Commands

To add a new user to Kali (in this example, Gus is going to be the user), use the `useradd` command. Along with it you need to choose the `sudo` group with the `-G` option and the shell type with the `-s` option:

```
$useradd -m [user name] -G [group name] -s [shell type]
```

For our example, it looks like this:

```
root@kali:~# useradd -m Gus -G sudo -s /bin/bash
```

Next, let's give the new user a password using the `passwd` command: `$passwd [user name - that you want to change password]`

Here's how it looks in the terminal window:

```
root@kali:~# passwd Gus
```

Continues

(continued)

```
New password:
Retype new password:
passwd: password updated successfully
```

If you look closely at the top left, it's written `root@kali`; I know that this is confusing, but the structure of this part is in the following format:

```
username@hostname
```

To switch to the new user Gus that we created previously, we use the `su` command (pay attention to how the user has changed in the terminal window text and turned into `Gus@kali`):

```
$su [user name - that you want to switch to]

root@kali:~# su Gus
Gus@kali:/root$
```

To learn the capabilities of the current user with the `sudo` command, you need to execute `sudo -l` to get the correct information:

```
Gus@kali:~$ sudo -l

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

 #1) Respect the privacy of others.
 #2) Think before you type.
 #3) With great power comes great responsibility.

 [sudo] password for Gus:
Matching Defaults entries for Gus on kali:
 env_reset, mail_badpass,
 secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/
sbin\:/bin

User Gus may run the following commands on kali:
 (ALL : ALL) ALL
```

To view the current user information, use the `id` command:

```
Gus@kali:~$ id
uid=1001(Gus) gid=1001(Gus) groups=1001(Gus),27(sudo)
```

To list the currently logged on users, use `w` or `who` (with fewer details):

```
Gus@kali:~$ w
 10:44:06 up 19 min, 1 user, load average: 0.00, 0.00, 0.00
```



```
USER TTY FROM LOGIN@ IDLE JCPU PCPU WHAT root tty7 :0 10:24 19:55
2.36s 2.36s /usr/ lib/x
Gus@kali:~$ who
root tty7 2020-09-22 10:24 (:0)
```

To remove a user (the user that we will remove in this example is `test`), exe cute the `userdel` command:

```
$userdel [user name - that you want to delete]
Gus@kali:~$ sudo userdel test
```

To list the last logged in users in the Kali system, use the `last` command:

```
Gus@kali:~$ last
root tty7 :0 Tue Sep 22 10:24 still logged  in
reboot system boot 5.7.0-kali1-amd6 Tue Sep 22 10:24 still running root
```

```
tty8 :1 Tue Sep 22 10:21 - 10:23 (00:02) kali pts/1 tmux(1793).%0 Mon
Sep 21 12:16 - 10:23 (22:07) kali pts/2 tmux(1584).%0 Mon Sep 21 11:48
- 11:48 (00:00) kali tty7 :0 Mon Sep 21 10:50 - 10:23 (23:33) reboot
system boot 5.7.0-kali1-amd6 Mon Sep 21 10:50 - 10:23 (23:33) kali tty7
:0 Mon Jul 27 13:36 - 15:56 (02:20) reboot system boot 5.7.0-kali1-amd6
Mon Jul 27 13:36 - 15:57 (02:20) kali tty7 :0 Mon Jul 27 13:31 - crash
(00:05) reboot system boot 5.7.0-kali1-amd6 Mon Jul 27 13:30 - 15:57
(02:26) kali tty7 :0 Mon Jul 27 13:28 - crash (00:02) reboot system
boot 5.7.0-kali1-amd6 Mon Jul 27 13:28 - 15:57 (02:28)

wtmp begins Mon Jul 27 13:28:09 2020
```

Finally, take note that all the users in Kali are stored in a configuration file, `/etc/passwd`. Use the `cat` command to reveal its contents:

```
Gus@kali:~$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
```

The previous command will list all the users, even the system ones (the example just shows the first three). To filter the contents and limit the results for the human users, pipe the output using `|` in the `grep` command:

```
Gus@kali:~$ cat /etc/passwd | grep "/bin/bash"
root:x:0:0:root:/root:/bin/bash
postgres:x:119:124:PostgreSQL
administrator,,,:/var/lib/postgresql:/bin/ bash
kali:x:1000:1000:kali,,,:/home/kali:/bin/bash
Gus:x:1001:1001::/home/Gus:/bin/bash
```


Groups Commands

To add a new group in Kali Linux, use the `groupadd` command:

```
$groupadd [new group name]

Gus@kali:~$ sudo groupadd hackers
```

To join a user (which is Gus for this example) to the `hackers` group that we created earlier, execute the `usermod` command:

```
$usermod -aG [group name] [user name]

Gus@kali:~$ sudo usermod -aG hackers Gus
```

To list all the groups created in Kali Linux, open the file `/etc/group`. Again, use the `cat` command to get the job done (the following example shows only the first three):

```
Gus@kali:~$ cat /etc/group
root:x:0:
daemon:x:1:
bin:x:2:
 [...]
hackers:x:1002:Gus
```

Managing Passwords in Kali

You probably want your root user back like in the old days. To get this account back, you will need to set its password first. To change a user password, you have to use the `passwd` command:

```
Gus@kali:~$ sudo passwd root
New password:
Retype new password:
passwd: password updated successfully
```

Now to use the powerful root account, you have to use the `su` command to switch user:

```
Gus@kali:~$ sudo su root
root@kali:/home/Gus#
```


From now on, on the login screen, you can choose your root account instead of your nonroot user.

Finally, to list all the user's credentials in Kali Linux, you can reveal them in the file `/etc/shadow`. Use the `grep` command to get the user credentials for Gus:

```
root@kali:/# cat /etc/shadow | grep "Gus"
Gus:$6$Hb.QBfIoaCBTiqK$EUJ4ZdWmbsFqHMsPbMEz2df6FtWVf4J/
tMulxCoLQmfMlVWyqpMUHBGmHFulRknYHgSrFIF.hQTANgzJ6CQM8/:18527:0:99999:7:::
```

Let's simplify what you need to understand from the string. The delimiter that separates each section is the colon character (:).

Second, the `$6$` means that the password is hashed using SHA-512. Finally, the hashed password starts after `$6$` and right before the : delimiter:

```
Hb.QBfIoaCBTiqK$EUJ4ZdWmbsFqHMsPbMEz2df6FtWVf4J/
tMulxCoLQmfMlVWyqpMUHBGmHFulRknYHgSrFIF.hQTANgzJ6CQM8/
```

Files and Folders Management in Kali Linux

Your next challenge in the Linux operating system is to learn how to manage files and folders. By the end of this section, you will start using the files and directories on Kali like the pros.

Displaying Files and Folders

To list the files and subfolders inside any directory, use the `ls` command to get the job done (I use it a lot to get simpler output). But sometimes, the `ls` command by itself is not enough, so you may need to add a couple of options to get better output clarity. The first option that you can use is the `-a` command (all contents including hidden files), and the second option is the `-l` command (formatted list):

```
root@kali:~# ls
Desktop Documents Downloads Music Pictures Public Templates
Videos
root@kali:~# ls -la
total 144
```

```
drwx------ 14 root root 4096 Sep 22 10:24 .
drwxr-xr-x 19 root root 36864 Jul 27 15:41 ..
```

continues

(continued)

```
-rw------- 1 root root 155 Sep 22 10:23 .bash_history
-rw-r--r-- 1 root root 570 Jul 18 17:08 .bashrc
drwx------ 6 root root 4096 Sep 22 11:21 .cache
drwxr-xr-x 8 root root 4096 Sep 22 10:22 .config
drwxr-xr-x 2 root root 4096 Sep 22 10:21 Desktop
-rw-r--r-- 1 root root 55 Sep 22 10:21 .dmrc
drwxr-xr-x 2 root root 4096 Sep 22 10:21 Documents
drwxr-xr-x 2 root root 4096 Sep 22 10:21 Downloads
-rw-r--r-- 1 root root 11656 Jul 27 13:22 .face
lrwxrwxrwx 1 root root 11 Jul 27 13:22 .face.icon -> /root/.face
drwx------ 3 root root 4096 Sep 22 10:24 .gnupg
-rw------- 1 root root 306 Sep 22 10:24 .ICEauthority
drwxr-xr-x 3 root root 4096 Sep 22 10:21 .local
drwxr-xr-x 2 root root 4096 Sep 22 10:21 Music
drwxr-xr-x 2 root root 4096 Sep 22 10:21 Pictures
-rw-r--r-- 1 root root 148 Jul 18 17:08 .profile
drwxr-xr-x 2 root root 4096 Sep 22 10:21 Public
drwxr-xr-x 2 root root 4096 Sep 22 10:21 Templates
drwxr-xr-x 2 root root 4096 Sep 22 10:21 Videos
-rw------- 1 root root 98 Sep 22 10:24 .Xauthority
-rw------- 1 root root 5961 Sep 22 10:24 .xsession-errors
-rw------- 1 root root 6590 Sep 22 10:23 .xsession-errors.old
root@kali:~#
```

Take note that filenames that start with a dot character before their names mean that they are hidden (e.g., `.bash_history`). Also, at the far left before the permissions, the letter `d` means it's a directory and not a file. Finally, you can list another directory's contents differently than the current one by specifying the path of the destination folder:

```
$ls -la [destination directory path]
```

Permissions

For the permissions, the same principle applies to a file or a directory. To simplify it, the permissions are divided into three categories:

- Read (`r`): 4
- Write (`w`): 2
- Execute (`x`): 1

The permissions template applies the following pattern:

```
[User:r/w/x] [group:r/w/x] [everyone:r/w/x]
```

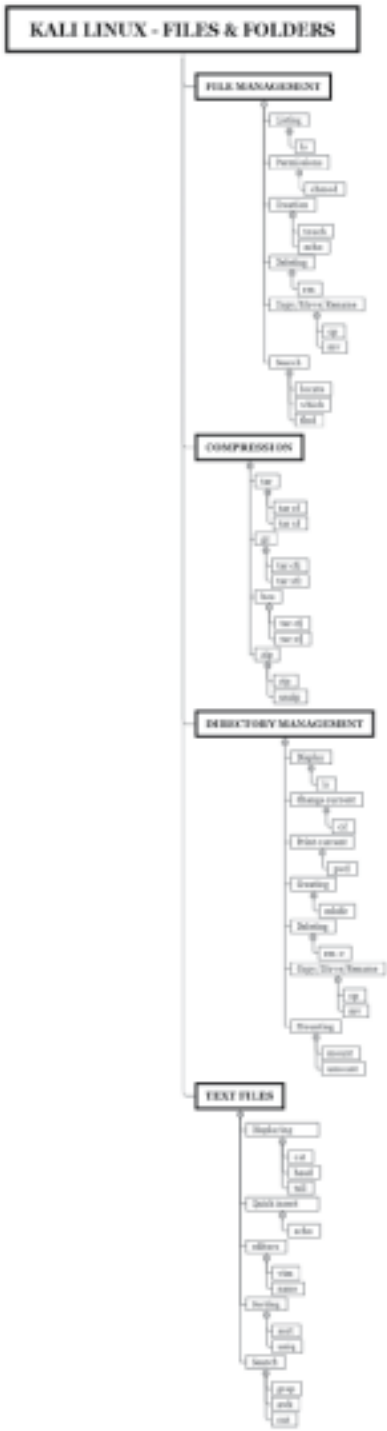

KALI LINUX - FILES & FOLDERS
FILE MANAGEMENT
COMPRESSION
DIRECTORY MANAGEMENT
TEXT FILES

Figure 1.6: Kali Linux – Files and Folders Commands

Let's look at a practical example. Lat's say you created a simple shell script that prints "test" (using the `echo` command) and that you wanted display its permissions (take note that this example uses the root user inside the terminal window):

```
root@kali:~# echo 'echo test' > test.sh
root@kali:~# ls -la | grep 'test.sh'
-rw-r--r-- 1 root root 10 Sep 22 11:25 test.sh
root@kali:~#
```

From the previous output results, we can see the following:

- For the root user, you can read and write because of `rw` at the beginning.
- For the root group, they can only read this file.
- For everyone else on the system, they can only read as well.

Let's say you want to execute this file, since you're the one who created it and you're the master root. Do you think you'll be able to do it (according to the previous permissions for the root user)?

```
root@kali:~# ./test.sh
bash: ./test.sh: Permission denied
```

The dot in the previous example means the current directory.

Indeed, the root has no permission to execute it, right? To change the per missions of the previous file based on the formula (`r`=4, `w`=2, and `x`=1), use this:

User:4+2+1=**7**; Group:4+2+1=**7**; Everyone:4

Then, use the `chmod` command to get the job done (this time, you should be able to execute the shell script):

```
$chmod [permissions numbers] [file name]

root@kali:~# chmod 774 test.sh
root@kali:~# ls -la | grep 'test.sh'
-rwxrwxr-- 1 root root 10 Sep 22 11:25 test.sh
root@kali:~# ./test.sh
test
root@kali:~#
```

There is another shortcut for this, which allows the execution of a file instead of calculating the numbers of each. We just need to add `+x` to the `chmod` command (but be careful because when you execute this one, you will be giving the exe cution permission to everyone as well):

```
$chmod +x [file name]
```



```
root@kali:~# chmod +x test.sh
root@kali:~# ls -la | grep 'test.sh'
-rwxrwxr-x 1 root root 10 Sep 22 11:25 test.sh
```

Manipulating Files in Kali

To simply create an empty file in Linux, you can use the `touch`

command: `$touch [new file]`

To insert text quickly into a file, you can use the `echo` command. Later in this chapter, you will learn how to edit text files with a text editor:

```
$echo 'text to add' > [file name]
```

To know a file type in a Linux system, you must use the `file` command: `$file [file name]`

Let's assemble all the commands together in the terminal window:

```
root@kali:~# touch test.txt
root@kali:~# echo test > test.txt
root@kali:~# file test.txt
test.txt: ASCII text
```

To copy a file in Kali, you must use the `cp` command to get the job done: `$ cp [source file path] [destination file path]`

```
root@kali:~# cp test.txt /home/kali
root@kali:~# ls /home/kali
Desktop Downloads Music Public test.sh Videos
Documents ls_file.txt Pictures Templates test.txt
```

To move a file that is equivalent to cut in Windows OS, you must use the `mv` command:

```
$mv [source file path] [destination file path]

root@kali:~# mv test.txt Documents/
root@kali:~# ls Documents/
test.txt
```

To delete the file that we just copied earlier in the `kali` home directory, use the `rm` command:

```
$rm [file path - that you want to delete]
root@kali:~# rm /home/kali/test.txt
```


To rename the previous file, we use the same `mv` command that we used to move a file:

```
$mv [original file name] [new file name]
root@kali:~/Documents# mv test.txt hello.txt
root@kali:~/Documents# ls
hello.txt
```

Searching for Files

There are multiple ways to search for files in Kali; the three common ones are the `locate`, `find`, and `which` commands.

You can use the `locate` command to locate a file that you're looking for quickly. You need to know that the `locate` command stores its data in a database, so when you search, you will find your results faster.

First, you will need to update the database for the `locate` command using

the `updatedb` command:

```
$updatedb
```

Now, we can start searching using the `locate` command:

```
$locate [file name]
root@kali:/# locate test.sh
/home/kali/test.sh
/usr/share/doc/socat/examples/readline-test.sh
/usr/share/doc/socat/examples/test.sh
```

You can use the `-n` switch for the `locate` command to filter out the number of output results. This option is handy if you know that the results will be enormous:

```
$locate -n [i] [search file criteria]
root@kali:/# locate *.conf -n 3
/etc/adduser.conf
/etc/ca-certificates.conf
/etc/debconf.conf
```

Use the `grep` command to get more granular results.

To find an application path, use the `which` command. This command will use the `$PATH` environment variable to find the results that you're looking for. As an example, to find where Python is installed, you can do the following:

```
$which [application name]

root@kali:/# which python
/usr/bin/python
```


It's important to understand that a Linux system will use `$PATH` to execute binaries. If you run it in the terminal window, it will display all the directories where you should save your programs/scripts (if you want to execute them without specifying their path):

```
root@kali:/# $PATH
bash: /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin: No
such file or directory
```

Let's look at a practical example; I saved the `test.sh` file in my home directory. Since the home folder is not in the `$PATH` variable, this means that I can execute it only if I specify the path or else it will fail:

```
root@kali:~# test.sh
bash: test.sh: command not found
root@kali:~# ./test.sh
test
```

Another useful command to find files with more flexible options is the `find` command. The advantage of using the `find` tool is that it allows adding more granular filters to find what you're looking for. For example, to find `file1.txt` under the root home directory, use this:

```
root@kali:~# find /root -name "file1.txt"
/root/temp/file1.txt
```

Let's say you want to list the large files (1GB+) in your system:

```
root@kali:~# find / -size +1G 2> /dev/null
/proc/kcore
```

Appending `2> /dev/null` to your command will clean the output results and filter out errors.

The following is a convenient find filter that searches for `setuid` files in Linux for privilege escalation (you will learn all the details in Chapter 10, "Linux Privilege Escalation"):

```
$ find / -perm -u=s -type f 2>/dev/null
```

Files Compression

There are multiple ways (compression algorithms) to compress files; the ones that I will cover in this section are the `.tar`, `.gz`, `.bz2`, and `.zip` extensions.

Here's the list of commands to compress and extract different types of archives: **Tar Archive**

To compress using tar extension:

```
$tar cf compressed.tar files
```

To extract a tar compressed file:

```
$tar xf compressed.tar
```

Gz Archive

To create compressed.tar.gz from files:

```
$tar cfz compressed.tar.gz files
```

To extract compressed.tar.gz:

```
$tar xfz compressed.tar.gz
```

To create a compressed.txt.gz file:

```
$gzip file.txt > compressed.txt.gz
```

To extract compressed.txt.gz:

```
$gzip -d compressed.txt.gz
```

Let's extract the `rockyou.txt.gz` file that comes initially compressed in Kali: `root@kali:~# gzip -d /usr/share/wordlists/rockyou.txt.gz`

Bz2 Archive

To create compressed.tar.bz2 from files:

```
$tar cfj compressed.tar.bz2 files
```

To extract compressed.tar.bz2:

```
$tar xfj compressed.tar.bz2
```

Zip Archive

To create compressed.zip from files:

```
$zip compressed.zip files
```

To extract compressed.zip files:

```
$unzip compressed.zip
```


Manipulating Directories in Kali

To print the current working directory, you must use the `pwd` command to get the job done (don't mix up the `pwd` command with `passwd` command; they're two different things):

```
$pwd
```

To change the current working directory, you must use the `cd` command: `$cd [new directory path]`

You can use `..` to traverse one upward directory. In fact, you can add as much as you want until you get to the system root folder, `/`:

```
root@kali:~/Documents# pwd
/root/Documents
root@kali:~/Documents# cd ../../
root@kali:/# pwd
/
```

As a final hint, for the `cd` command, you can use the `~` character to go directly to your current user home directory:

```
$cd ~
```

To create a directory called `test` in the root home folder, use the `mkdir` command: `$mkdir [new directory name]`

To copy, move, and rename a directory, use the same command for the file commands. Sometimes you must add the `-r` (which stands for recursive) switch to involve the subdirectories as well:

```
$cp -r [source directory path] [destination directory path]
$mv -r [source directory path] [destination directory path]
$mv -r [original directory name] [new directory name]
```

To delete a folder, you must add the `-r` switch to the `rm` command to get the job done:

```
$rm -r [folder to delete path]
```

Mounting a Directory

Let's see a practical example of how to mount a directory inside Kali Linux. Let's suppose you inserted a USB key; then mounting a directory is necessary to access your USB drive contents. This is applicable if you disabled the auto mount feature in your settings (which is on by default in the Kali 2020.1 release).

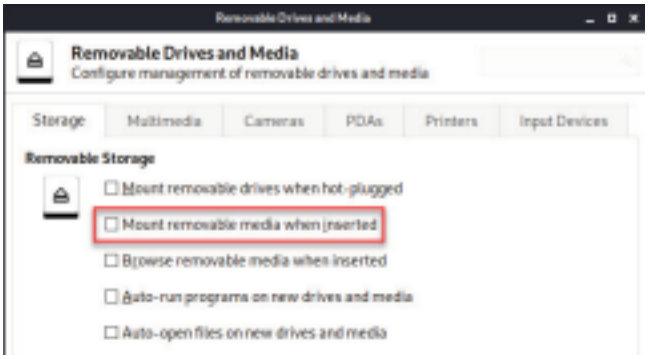


Figure 1.7: USB Mount

To mount a USB drive, follow these steps:

1. Display the disk list using the `lsblk` command.
2. Create a new directory to be mounted (this is where you will access the USB stick drive).
3. Mount the USB drive using the `mount` command.

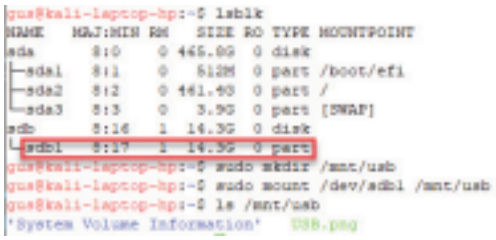


Figure 1.8: Mount Using the Command Line

Now, to eject the USB drive, use the `umount` command to unmount the directory: `root@kali-laptop-hp:~# umount /mnt/usb`

Managing Text Files in Kali Linux

Knowing how to handle files in Kali Linux is something that you'll often encounter during your engagements. In this section, you will learn about the most common commands that you can use to get the job done.

There are many ways to display a text file quickly on the terminal window. 90 percent of the time, I use the `cat` command for this purpose. What if you want to display a large text file (e.g., a password's dictionary file)? Then you have three choices: the `head`, `tail`, and `more` and `less` commands. It is important to note that you can use the `grep` command to filter out the results

that you're

looking for. For example, to identify the word *gus123* inside the `rockyou.txt` dictionary file, you can do the following:

```
root@kali:/usr/share/wordlists# cat rockyou.txt | grep gus123
gus123
angus123
gus12345
 [...]
```

The `head` command will display 10 lines in a text file starting from the top, and you can specify how many lines you want to display by adding the `-n` option:

```
$head -n [i] [file name]

root@kali:/usr/share/wordlists# head -n 7 rockyou.txt
123456
12345
123456789
password
iloveyou
princess
1234567
```

The `tail` command will display the last 10 lines in a file, and you can specify the number of lines as well using the `-n` switch:

```
$tail -n [i] [file name]

root@kali:/usr/share/wordlists# tail -n 5 rockyou.txt
 xCvBnM,
ie168
abygurl69
a6_123
*7!Vamos!
```

To browse a large file, use the `more` command. You need to press Enter or the spacebar on your keyboard to step forward. Pressing the B key will let you go backward. Finally, to search for text, press the **/** (forward slash) and the Q key to quit:

```
$more [file name]
```

`less` is like the `more` command; it allows you to view the contents of a file and navigate inside it as well. The main difference between `more` and `less` is that the `less` command is faster than the `more` command because it does not load the entire file at once, and it allows you to navigate inside the file using the Page Up/Down keys as well:

```
$less [file name]
```


To sort a text file, simply use the `sort` command:

```
$sort [file name] > [sorted file name]

root@kali:~/temp# cat file1.txt
```

```
5
6
4
root@kali:~/temp# sort file1.txt >file1_sorted.txt
root@kali:~/temp# cat file1_sorted.txt
4
5
6
```

To remove duplicates in a text file, you must use the `uniq` command: `$uniq [file name] > [no duplicates file name]`

```
root@kali:~/temp# cat file2.txt
5
6
4
4
5
5
5
root@kali:~/temp# uniq file2.txt > file2_uniq.txt
root@kali:~/temp# cat file2_uniq.txt
5
6
4
5
```

Later in this book, you will learn how to use the `sort` and `uniq` commands together to create a custom passwords dictionary file.

Vim vs. Nano

For the terminal window, we have two popular text editors, vim and nano. Most of the time, you can tackle four tasks in text editors:

- Open/create the text file
- Make text changes
- Search for text
- Save and quit

Nano is easier than vim. It's up to you to choose any of them; it's a matter of preference.

To open/create a text file, use these commands:

- `$vim [`*`text filename`*`]`
- `$nano [`*`text filename`*`]`

Once the text file is opened, you will need to start making your changes:

- In nano, you can just enter your text freely.
- In vim, you need to press I on your keyboard to enter insert mode. If

you want to search for a specific word inside your file, use these commands:

- In nano, press Ctrl+W.
- In vim, it depends which mode you're in.
  - If you're in insert text mode, then hit the Esc key and then press **/** followed by the word that you want to search for.
  - If you're in normal mode, then just press **/** followed by the word that you want to search for.

Finally, it's time to save and quit your text editor:

- In nano, press Ctrl+O to save, press the Enter key to execute the save task, and then press Ctrl+X to exit.
- In vim, make sure that you are in normal mode first (if you're not, then press the Esc key to go back in normal mode) and then use `:wq`. The `w` is for "write," and the `q` is to quit.

Searching and Filtering Text

One more thing to learn in the world of text files is the search mechanism. There are so many ways to search and filter out text, but the popular ones are as follows:

- `grep`
- `awk`
- `cut`

You've seen me using the `grep` command a lot. This filter command is structured in the following way:

```
$grep [options] [pattern] [file name]
```

Let's say you want to search for the word *password* in all the files starting from the root system (`/`).

```
root@kali:/# grep -irl "password" /
/boot/grub/i386-pc/zfscrypt.mod
```

continues

(continued)

```
/boot/grub/i386-pc/normal.mod
/boot/grub/i386-pc/legacycfg.mod
```

Here's what the options mean:

- `-i`: To ignore case and include all the uppercase/lowercase letters
- `-r`: To search recursively inside subfolders
- `-l`: To print the filenames where the filter matches

As another example, let's say you want to count the number of occurrences of the word *password* in the dictionary file `rockyou.txt`:

```
root@kali:/# cd /usr/share/wordlists/
root@kali:/usr/share/wordlists# grep -c "password"
rockyou.txt 3959
```

The `awk` command is an advanced tool for filtering text files, and it uses the following pattern:

```
$awk /[search criteria]/ [options] [file name]
```

For example, let's say you want to search for the text *root* inside the `/etc/passwd` file:

```
root@kali:/# awk '/root/' /etc/passwd
root:x:0:0:root:/root:/bin/bash
nm-openvpn:x:125:130:NetworkManager
OpenVPN,,,:/var/lib/openvpn/chroot:/ usr/sbin/nologin
```

Let's take the challenge one more step further. Let's say you want to extract the password of the root in the `/etc/shadow` file (you can print the whole thing first so you can visualize the difference of before and after):

```
 root@kali:/# awk '/root/' /etc/shadow
root:$6$uf2Jy/R8HS5Tx$Vw1wHuBV7unq1hImYGTJdNrRwMwRtf0yd/
aSH0zOhhdzWofAT5WUSduQTjWj8AbdmT62rLbcs6kP3xwdiLk.:18414:0:99999:7::
: root@kali:/# awk -F ':' '/root/{print $2}' /etc/shadow
$6$uf2Jy/R8HS5Tx$Vw1wHuBV7unq1hImYGTJdNrRwMwRtf0yd/
aSH0zOhhdzWofAT5WUSduQTjWj8AbdmT62rLbcs6kP3xwdiLk.
```

We know that the shadow file is using the `:` delimiter to separate the sections, so we use `-F ':'` to get the job done. Then, we tell the tool to print only the second part of the delimiter `{print $2}`, which is the hashed password contents.

Another popular way to extract substrings is the `cut` command. In the following example, we use the `cat` command to open the shadow file; then we use the `grep` command to filter out the root account, and finally, we use the `cut` command to extract the password:

```
root@kali:/# cat /etc/shadow | grep "root" | cut -d ":" -f 2
$6$uf2Jy/R8HS5Tx$Vw1wHuBV7unq1hImYGTJdNrRwMwRtf0yd/
aSH0zOhhdzWofAT5WUSduQTjWj8AbdmT62rLbcs6kP3xwdiLk.
```


Remote Connections in Kali

There are two common ways to connect remotely to other operating systems. For Windows, it is the Remote Desktop Protocol (RDP), and for Linux, it's the Secure Shell (SSH). In the next sections, I will explain how to use each protocol to connect remotely to an OS (Windows or Linux).

Remote Desktop Protocol

RDP is used to connect remotely to a Windows OS. Let's suppose that during your engagement you encountered a remote desktop port 3389 open on a Windows host (e.g., during your port scanning phase). Then, you will need to

try to connect to it with some basic credentials (e.g., a username of Administrator and a password of password123). There are many times during your engagements where you want to connect remotely to a Windows system to get the job done (from Kali Linux). In this case, you will need to use the `rdesktop` command.

```
$rdesktop [Windows host IP address] -u [username in windows] -p
 [password in windows]
```

You can also omit the password and enter it later. See the example in Figure 1.9.

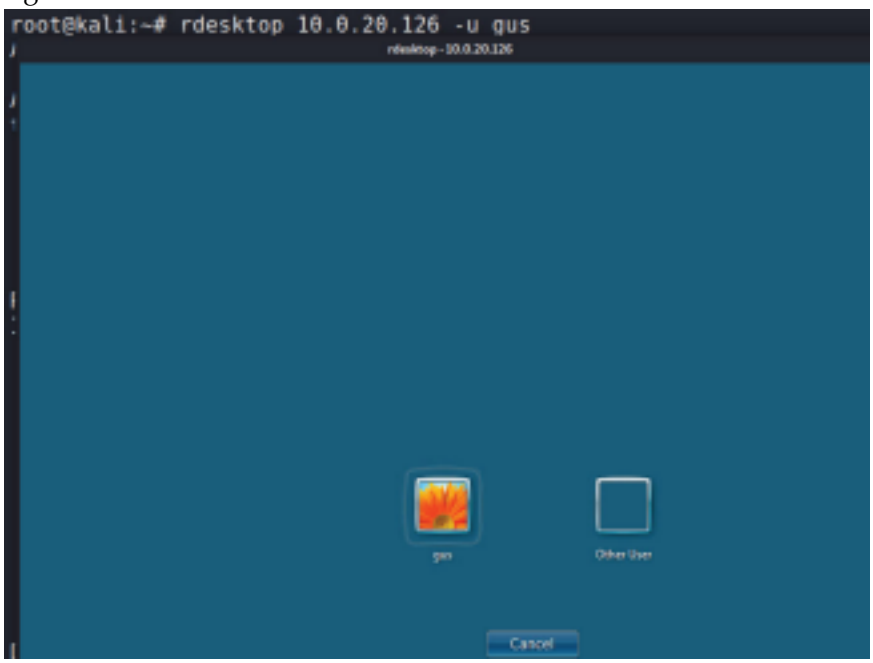


Figure 1.9: "Windows Login"

Secure Shell

The SSH protocol is a secure connection that allows you to execute commands remotely on a Linux host (in this case, Kali). By default, the SSH is a TCP protocol that works on port 22 by default. There are two ways to connect to a remote SSH server:

- Using a username/password credentials
- Using public/private keys (passwordless)

SSH with Credentials

Let's start first with the method that uses the password. By default, all the user accounts except the root account can log in remotely to SSH:

```
$ssh username@kaliIP
```

Figure 1.10 shows a root user who is not allowed to log in to Kali Linux remotely as well as a regular user (`kali`) who is able to log in remotely using SSH. In Figure 1.10, I'm using MobaXterm on Windows OS to connect

remotely using SSH to the Kali VM.

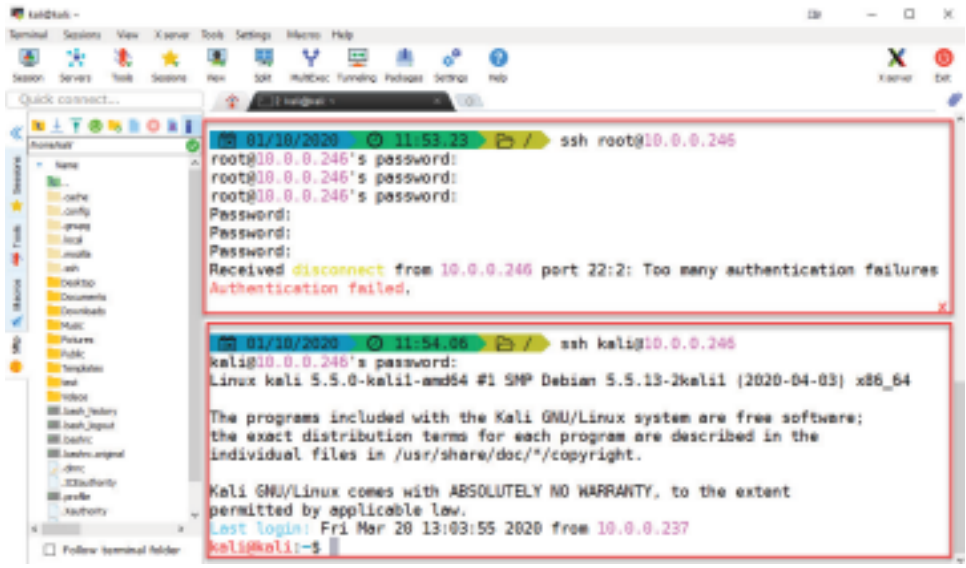


Figure 1.10: SSH with MobaXterm on Windows

To allow the root user to log in remotely to SSH, you will need to edit the configuration file of SSH under this directory:

```
/etc/ssh/sshd_config
```


Make sure to add the following line to the SSH configuration file: `PermitRootLogin Yes`

Now, we can try to connect to our Kali host remotely using the root account (it should work this time after the latest changes):

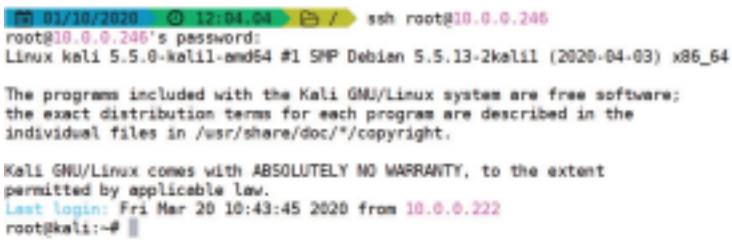


Figure 1.11: SSH Root Connection

Before you start using the SSH service on your Kali Linux, you will need to start the SSH service first. To do this, you will need to execute the following command:

```
$service ssh start
```

If you want to stop it later, use the following command:

```
$service ssh stop
```

If you want the SSH server to persist (automatically start) even after you

reboot your system, then you will need to execute the following command:

```
$systemctl enable ssh
```

If you forgot the status (started or stopped) of your SSH server, then execute the following command to get the results shown in Figure 1.12:

```
$service ssh status
```

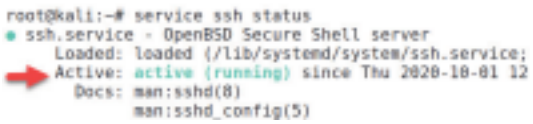


Figure 1.12: SSH Service Status

By default, the port number of SSH is 22, and if the remote Linux server has changed to another port, then you will need to specify it in your connection command:

```
$ssh username@kaliIP -p [port number]
```


Passwordless SSH

Using a public key and a private key, a remote user can log in using SSH. This method is more secure than the password way because no one will be able to use the brute-force technique to enter your server remotely.

There is a lot of misconception when it comes to the public/private keys mechanism. In the next steps, I developed an example from scratch so you can visualize how things happen in reality:

Here's the client machine information:

- OS: Ubuntu Desktop Linux V20
- IP:10.0.0.186

Here's the Kali Linux SSH Server host information:

- OS: Kali Linux 2020.1
- IP:10.0.0.246

First, we will generate a public key and a private key on our client host (Ubuntu). Why? The goal is to perform the following steps:

1. Generate a private key (`/home/[username]/.ssh/id_rsa`) on the client machine because it's the one that can decrypt the public key. If someone steals your public key, they can't hack into the remote host since they don't have the private key file.
2. Generate a public key (`/home/[username]/.ssh/id_rsa.pub`) on the client machine. We need to send a copy of the public key to the server. After that, the server will store the client's public key in a file called `authorized_keys`.

Let's start! On our client Ubuntu host, generate the public and private keys (Figure 1.13):

```
$ssh-keygen -t rsa -b 4096
```

The previous command used two arguments:

- `-t rsa`: The `t` stands for the type of the key to generate. RSA is the most common one, but you have other options as well (`dsa`, `ecdsa`, `ecdsa-sk`, `ed25519`, `ed25519-sk`, and `rsa`).
- `-b 4096`: The `b` option specifies the number of bits in the key to create. In our case (RSA key), the minimum size is 1,024 bits, and the default is 3,072 bits.

Take note that while performing the earlier steps, we've been asked to enter a passphrase. This password will be used to add more security when you log in remotely to SSH.

Figure 1.13: SSH Key Generation

Let's check out the folder where these files were saved on the client's host machine (`/home/gus/.ssh/`):

```
gus@ubuntu:~/.ssh$ ls -la
total 16
drwx------ 2 gus gus 4096 Oct 1 10:03 .
drwxr-xr-x 15 gus gus 4096 Oct 1 09:57 ..
-rw------- 1 gus gus 3369 Oct 1 10:03 id_rsa
-rw-r--r-- 1 gus gus 736 Oct 1 10:03 id_rsa.pub
```

Now we're ready to send a copy of the public key file `id_rsa.pub` to the Kali host machine. You can send it in multiple ways (e.g., by e-mail, SFTP, SCP, etc.) There is an easy, secure method using the SSH client package that comes with the SSH tool:

```
$ssh-copy-id username_on_kalihost@kaliIP
```

In the following example, we will use the root username and password (also, you will be asked for the password of this account) to copy the public key file:

```
gus@ubuntu:~/.ssh$ ssh-copy-id root@10.0.0.246
The authenticity of host '10.0.0.246 (10.0.0.246)' can't be established.
ECDSA key fingerprint is
SHA256:TA8zjlhAspZEc/3WZjyWRQBxzPfwJXE2X98JsMGnz6U. Are you sure you want
to continue connecting (yes/no/[fingerprint])? yes /usr/bin/ssh-copy-id:
```

```
INFO: attempting to log in with the new key(s), to filter  out any that
are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if
you are  prompted now it is to install the new keys
Password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'root@10.0.0.246'"
and check to make sure that only the key(s) you wanted were added.
```


Now, let's verify that the authorized key has really been added on the Kali host machine:

```
root@kali:~/.ssh# cat authorized_keys
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAACAQDNfvP6zEnKn55pY5hN8N34m
yD1XwwhS9JisvcR0qtXzM2957h9xeQMVVrUASA/xdwRObUak7wARZl+F
Y3pby5k+askzIgPIfqvU0lZJEpBtjobk6SdBha122pR3a72+Vh7f9hdg
GQoqXeF3pyXfYOhFEJZ0s0SCFGc/MfI38pBrXCgzHXS28QxzpZnIg3/
IwAcBIjbPYnszWSDqHplSFMpETPbHvPwUMU3RDGpvSgoscfyWchXzb97lViSk
/ zD2TbN2eSbm8k8txxIIZHq7LrAYHB8smvlFEHK6CNvIU+HU0NvvcwXmXvi
SCGcMAsNxzvEzEJf4U6RDhzbL85Id43VghhDYp1I7/D4euxPfs+Xt/qj6qaL4T66+
KvfML3loCRg9zBo0z6sZbOGOUu6iMYguVW/lTqC+Hui/SZUV9Zt3Z2/c/hC8r8+9/
SsauWXtFNC4mRTLKyeEluIdLe9USgxwtHB3uD7BgYNaC1hbgXsGdM1CoDrQS4TOLMai
q4gpIZE80dKFJTw3+EbIIj7SEPTKC6BmWZluOfYjkHDJ19qLKEGWuWqfwp6U9CW+i4f5cLo
M
Fssafqs/uSw/u0FA6jt+ykMZ7jvbYJhHmOa4dOGrOd9PyGw8/MM2qVo2VrATvk12oIQWZwd
F A8Fj1oKaGK1pFcngR+At10jL2y1mI4fJw== gus@ubuntu
```

Next, I will edit the SSH config file (`/etc/ssh/sshd_config`) again on Kali to allow only public key authentication:

```
PubkeyAuthentication yes
PasswordAuthentication no
```

To make sure that the changes are well propagated, it's better to restart the SSH server on Kali using this command:

```
$service ssh restart
```

It's time to test the SSH connection and see if it works remotely:

```
gus@ubuntu:~/.ssh$ ssh root@10.0.0.246
Linux kali 5.5.0-kali1-amd64 #1 SMP Debian 5.5.13-2kali1 (2020-04-03)
x86_64

The programs included with the Kali GNU/Linux system are free
software; the exact distribution terms for each program are described
in the individual files in /usr/share/doc/*/copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the
extent permitted by applicable law.
Last login: Thu Oct 1 12:04:15 2020 from 10.0.0.222
root@kali:~#
```

Kali Linux System Management

Since you will be using Kali Linux as a penetration testing arsenal, then you must know how to handle its system, including how to start an Apache web

server or check its status. The examples are endless. Don't worry, we will cover the most common scenarios that you'll encounter as a penetration tester later.

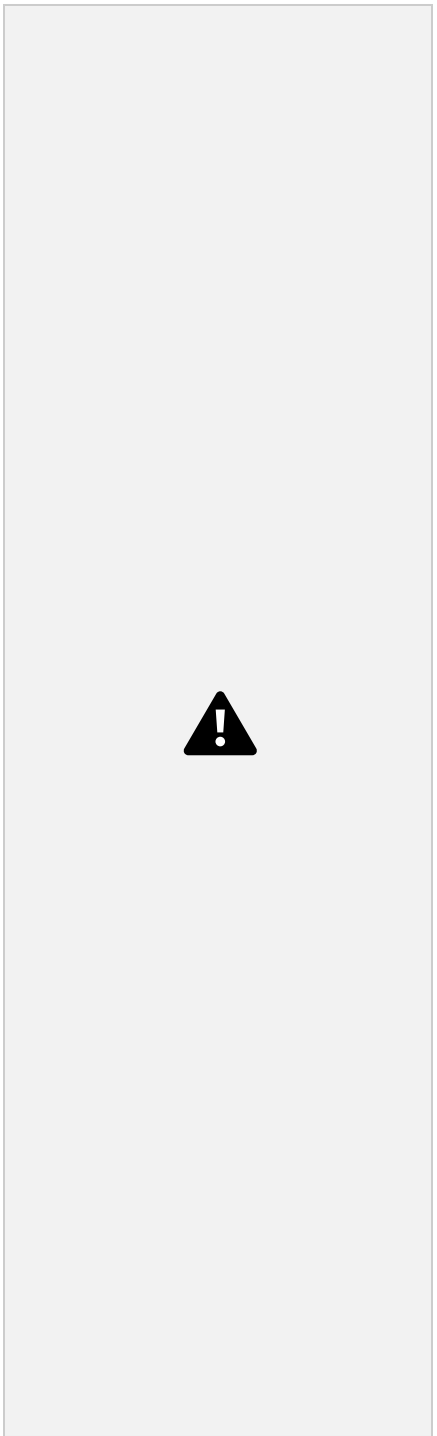


Figure 1.14: Kali System Management Commands

Linux Host Information

To display the hostname of Kali Linux, you simply execute the `hostname` command in your terminal window:

```
$hostname

root@kali:/# hostname
kali
```

What if you want to change your Kali hostname? Then you will need to edit its configuration file `/etc/hostname` (enter the desired computer name and don't forget to save and reboot your host).

Linux OS Information

Knowing the OS information for a Linux host is crucial for privilege escalation. That's how you will know if the version used is vulnerable to privilege escala tion (we will talk more about this topic in Chapter 10).

To display the operating system information of a Linux OS (which is Kali Linux in our case), I use the `uname` command, and along with it I display the contents of the `/etc/issue` configuration file:

```
$uname -a
$cat /etc/issue

root@kali:/# uname -a
Linux kali 5.6.0-kali2-amd64 #1 SMP Debian 5.6.14-2kali1 (2020-06-10)
x86_64 GNU/Linux
root@kali:/# cat /etc/issue
Kali GNU/Linux Rolling \n \l
```

Linux Hardware Information

From time to time, you will probably use special commands related to your PC or VM hardware.

To get the CPU information of your Linux host, you need to open `/proc/cpuinfo`:

```
root@kali:/# cat /proc/cpuinfo
processor : 0
vendor_id : GenuineIntel
cpu family : 6
model : 158
model name : Intel(R) Core(TM) i7-8700 CPU @ 3.20GHz
stepping : 10
```



```
microcode : 0xd6
cpu MHz : 3192.001
cache size : 12288 KB
 [...]
```

To get the RAM information of your Kali host, then you will need to open

the configuration file `/proc/meminfo`:

```
root@kali:/# cat /proc/meminfo
MemTotal: 8676820 kB
MemFree: 6183876 kB
MemAvailable: 7781928 kB
Buffers: 55444 kB
Cached: 1739668 kB
SwapCached: 0 kB
 [...]
```

To display the attached devices (e.g., disk drives, partitions, etc.), then you have a choice of two commands: either `fdisk` (which displays more information) or `lsblk`:

```
$fdisk -l

root@kali:/# fdisk -l
Disk /dev/sda: 80 GiB, 85899345920 bytes, 167772160 sectors
Disk model: VMware Virtual S
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x4a6f3195

Device Boot Start End Sectors Size Id Type
/dev/sda1 * 2048 163579903 163577856 78G 83 Linux
/dev/sda2 163581950 167770111 4188162 2G 5 Extended /dev/sda5
163581952 167770111 4188160 2G 82 Linux swap /  Solaris

$lsblk

root@kali:/# lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda 8:0 0 80G 0 disk
└sda1 8:1 0 78G 0 part /
└sda2 8:2 0 1K 0 part
└sda5 8:5 0 2G 0 part [SWAP]
sr0 11:0 1 1024M 0 rom
```


To display the list of USB devices (e.g., mouse, keyboard, USB stick, etc.), then you have to execute the `lsusb` command:

```
$lsusb

root@kali:/# lsusb
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub Bus
002 Device 004: ID 0e0f:0008 VMware, Inc. VMware Virtual USB Mouse
Bus 002 Device 003: ID 0e0f:0002 VMware, Inc. Virtual USB Hub
Bus 002 Device 002: ID 0e0f:0003 VMware, Inc. Virtual Mouse
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
```

To display all the mounted directories into the file system, then you will need to execute the `mount` command:

```
$mount

root@kali:/# mount
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
udev on /dev type devtmpfs (rw,nosuid,noexec,relatime,size=4308020k,nr_
```

```
inodes=1077005,mode=755)
 [...]
```

Managing Running Services

Services are servers that can run on your Kali Linux box, such as SSH, web, FTP, etc.

One of the common tasks in penetration testing is to run a web server on your Kali so you can transfer files to your victim machines (I will go into more details later in this book) after getting a remote shell. So, for example, to start the web server on your Kali Linux (for your information, that's not the only way to start a service, but it's my favorite because it's easy to memorize):

```
root@kali:/# service apache2 start
```

Here are the remaining commands that you will need to know about managing services:

To Get the status of a service (started, stopped):

$service [service name] status

$systemctl status [service name]

To start a service:

$service [service name] start

$systemctl start [service name]

To stop a service server:

$service [service name] stop

$systemctl stop [service name]

To restart a service:

$service [service name] restart

$systemctl restart [service name]

To enable a service to start on boot automatically:

$systemctl enable [service name]

To disable a service from automatically starting at boot:

$systemctl disable [service name]

Package Management

The first thing that you need to know before you update your Kali Linux system is that the configuration file for the Kali repository is located at `/etc/apt/ sources.list`:

```
root@kali:/# cat /etc/apt/sources.list
#
```

```
# deb cdrom:[Kali GNU/Linux 2020.2rc1 _Kali-last-snapshot_ - Official
amd64 DVD Binary-1 with firmware 20200505-14:58]/ kali-rolling contrib
main non-free

#deb cdrom:[Kali GNU/Linux 2020.2rc1 _Kali-last-snapshot_ - Official
amd64 DVD Binary-1 with firmware 20200505-14:58]/ kali-rolling contrib
main non-free

deb http://http.kali.org/kali kali-rolling main non-free contrib #
deb-src http://http.kali.org/kali kali-rolling main non-free contrib
```

To update your Kali Linux system (like Windows Update), execute the `update` command first and then the `upgrade` command. Take note, these two commands will use the earlier configuration file to download and install the necessary files:

```
$apt update
$apt upgrade -y
```

We're using the `-y` option in the upgrade command to ignore the prompts where it asks for input. In other words, we're just saying "yes" in advance. What is the difference between the `upgrade` and `update` commands? That's a confusing beginner question, and I'm here to help you start using these two

commands with confidence. In summary, the `update` command only updates the package list with the latest versions, but it does not install or upgrade the package. On the other hand, the `upgrade` command will upgrade and install the latest version of packages that were already installed (using the `update` command).

Now, to use these commands together, you will have to use the `&&` in bet ween, which will eventually run the first command, and when it's done, it will run the second:

```
$apt update && apt upgrade -y
```

To fully upgrade from one release to another, execute the `full-upgrade` command along with the `update` command.

```
$apt update && apt full-upgrade -y
```

Now, to list all the installed software packages on Kali Linux, you'll have to use the `dpkg` command:

```
$dpkg -l
```

What about installing a new software (package) on Kali? There are two common ways that I use most of the time. The first one is the `apt install` command, and the second one is `dpkg` (I use the latter only when I download a file that ends with `.deb` extension).

```
$apt install [package name] -y
$dpkg -i [filename.deb]
```

In some software packages, they will require you to use the `configure/make` installation way, if that's the case, then use the following commands (you must be inside the application directory):

```
$./configure && make && make install
```

If you want to remove an existing application from your Kali system, then you use the `apt remove` command:

```
$apt remove [package name]
```

How do we find a package name? Let's say you want to install something that is not already installed on Kali. Then you can search the repository packages using the following command:

```
$apt-cache search keyword
```

Finally, if you want to install a package and you're not sure if the name exists in the repository, then you can use the `apt-cache show` command:

```
$apt-cache show [software name]
```



```
root@kali:/# apt-cache show filezilla
Package: filezilla
Version: 3.49.1-1
Installed-Size: 6997
Maintainer: Adrien Cunin <adri2000@ubuntu.com>
Architecture: amd64
 [...]
```

Process Management

One of my favorite terminal window tools to list all the running processes on Kali is called `htop`. By default, it's not installed on Kali, so to install it, we use the `apt install` command:

```
root@kali:/# apt install htop -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

Once it's installed, you can run the `htop` command:

```
$htop
```

As you can see in Figure 1.15, we're running Nmap in another terminal window, and it has a process ID (PID) equal to 1338.

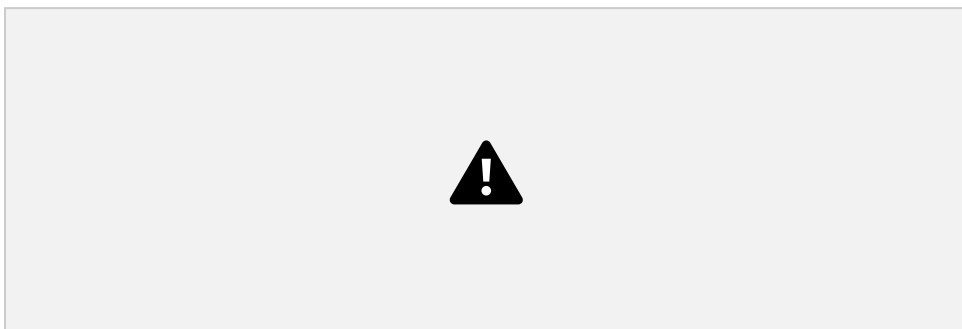


Figure 1.15: HTOP

Another way to get the list of currently running processes is by using the `ps` command:

```
$ps -Au

 -A: To select all the processes (if you want to list only the
processes that belongs to the current user then use the -x option
instead)
       -u: shows more info (e.g., CPU, MEM, etc) than the default output
```


To kill a process, you will need to identify its PID first; then you can use the `kill` command to get the job done:

```
$kill [PID]
```

If the system doesn't allow you to kill it, then you must force it to close using the `-9` switch:

```
$kill -9 [PID]
```

Networking in Kali Linux

In this section, you will get the chance to understand the basics of networking in Kali Linux. Later in the book we will come back to more advanced topics regarding networking, so make sure to understand and grasp the contents in this section.

Figure 1.16: Kali Networking Commands

Network Interface

You must be a pro in networking to survive in the penetration testing career. It's one of the pillars of the job if you're going to execute network infrastructure penetration tests.

PC hosts have internal IP addresses to connect with the network, and they have a public IP address to communicate with the outside world. The latter is the mission of your home router, and you don't manage it locally on your local host. On the other hand, you must maintain the internal network IP addresses, which are either static (you define it) or automatically assigned by a DHCP server (which is generally your home router).

IPv4 Private Address Ranges

Internal IP addresses (aka private IP addresses) for IPv4 have multiple ranges: classes A, B, and C.

- Class A: 10.0.0.0 to 10.255.255.255 or 10.0.0.0/8 (up to 16,777,214 hosts)
- Class B: 172.16.0.0 to 172.31.255.255 or 172.16.0.0/12 (up to 1,048,574 hosts)

- Class C: 192.168.0.0 to 192.168.255.255 or 192.168.0.0/24 (up to 254 hosts)

The biggest range is class A for corporations, but you can use it at home. (No one will stop you from doing that, and guess what? I use it myself for my home network.) The second, class B, is for small/midrange/big companies (depending on the number of hosts). The third is class C; this range is limited but is suitable for home users and small office/home office (SOHO) environments.

Let's take a quick look at our Kali host IP address. To get the information about our network interface, execute the popular `ifconfig` command (take note that there has been a shift to use the `ip addr` command lately instead of `ifconfig`).

According to Figure 1.17, we have two network interfaces. The first one on the top, `eth0`, is the Ethernet adapter that connects my Kali host with the internal network. If we had a second Ethernet adapter, it would be `eth1`. (Take note that if you're using a wireless adapter on your host, then you will see `wlan0`, `wlan1`, etc.)

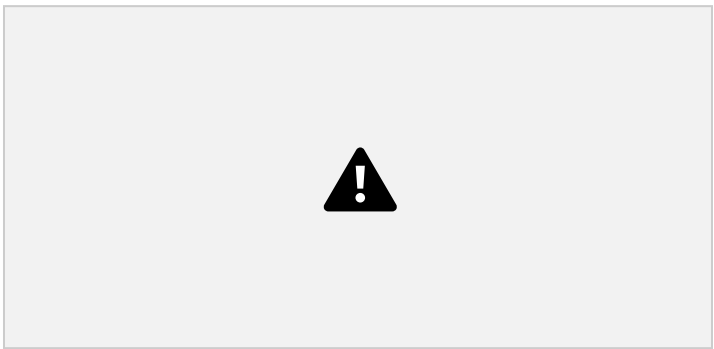


Figure 1.17: Kali Network Interfaces

There are two important facts to understand about our Ethernet adapter `eth0`. First, `inet 10.0.0.246` represents the Kali host IP address that was assigned automatically by the DHCP server. The second part is the netmask, which means that we're using a /24 subnet; in other words, we only need 254 hosts to be assigned on this IP range.

The second interface is `lo`, which represents a local loopback; you will never touch this since the network infrastructure will need it to operate correctly. There are two common other interfaces that you will encounter; the first one is the wireless interface if you're connected wirelessly instead of the wire. The second is the VPN interface, if you're connected to a remote VPN server.

Static IP Addressing

If you want to assign a fixed IP address to your Kali host, you will need to edit the configuration file `/etc/network/interfaces`. In the following new config uration, shown in Figure 1.18, add these three main components:

- Static IP address (it's going to be 10.0.0.20 in my case; in your case, it has

to match your private IP address range)

- Subnetmask or CIDR (/24 means 255.255.255.0)
- Router/gateway IP address (my router IP address is 10.0.0.1; yours could be different)

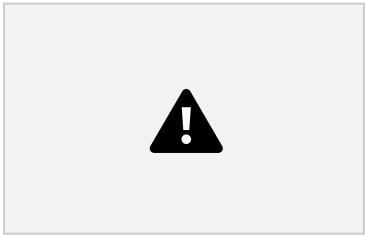


Figure 1.18: Static IP Configs

After you save your changes, make sure to reboot your Kali machine to get this new fixed IP address up and running. To test the connectivity to the outside world (after rebooting), try to ping the popular Google's DNS server on 8.8.8.8 (if for any reason you want to reverse your changes, just go back to the config file and remove/comment the new lines), as shown in Figure 1.19.

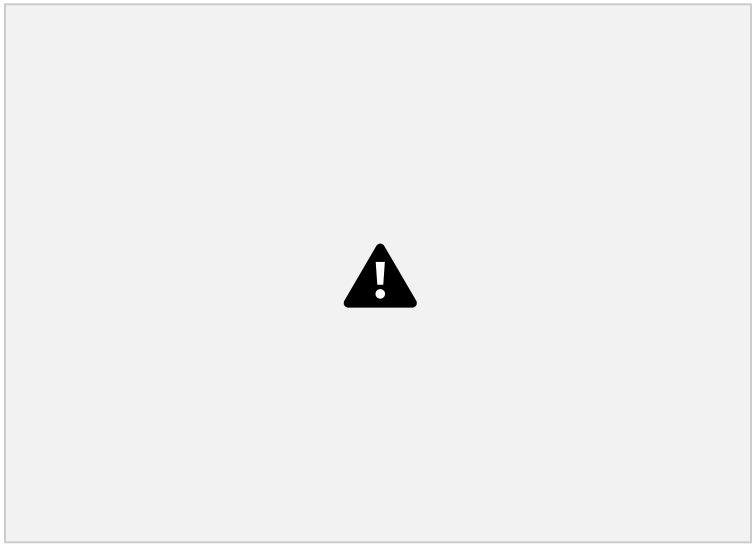


Figure 1.19: Testing Internet Connection

Take note that we're using 10.0.0.0 network as our main VLAN (virtual net work). In fact, we have multiple VLANs in our home network. For example, we have a VLAN for IoT devices, but why? It's because we want IoT devices to be on a separate network (10.0.50.0/24) without interfering with my main production hosts.

Another example is the Guests VLAN. This network is for people who con nect to the wireless guest access point, and they will be assigned in the

10.0.20.0 address range.

Companies implement the same concept. Ideally, they have a development environment that is different than the production environment network VLAN.

DNS

The Domain Name System (DNS) translates domain names into IP addresses. For example, instead of typing `https://172.217.13.132`, you simply type `https://google.com`. The question is, how did I come up with the IP address? Use the `host` command on your terminal window:

```
$host [domain name]

root@kali:/# host google.com
google.com has address 172.217.13.174
google.com has IPv6 address 2607:f8b0:4020:806::200e
google.com mail is handled by 40 alt3.aspmx.l.google.com.
google.com mail is handled by 30 alt2.aspmx.l.google.com.
google.com mail is handled by 10 aspmx.l.google.com.
google.com mail is handled by 50 alt4.aspmx.l.google.com.
google.com mail is handled by 20 alt1.aspmx.l.google.com.
```

The DNS is divided into two categories: public and private (like the IP addresses). The Google DNS address is public so that anyone connected to the internet can reach Google's website.

On the other hand, we can have private DNS for our local intranet. This can be set up using a DNS server (e.g., Microsoft Windows Server) or your router if it has a built-in DNS server. In my home network, I defined a domain called `ksec.local`. Each host on the network will have a domain name that corresponds to its IP address. For example, my file server domain name is `ds-server.ksec.local` (because the server hostname is `ds-server`), and the router/DNS server will manage all the DNS A records (an A record is a mapping between IPv4 addresses and domain names):

```
root@kali:~# host ds-server.ksec.local
ds-server.ksec.local has address 10.0.0.177
```

If you specify a nonexisting DNS record, you will get an error message (this is useful to brute-force the DNS records):

```
root@kali:~# host hello.ksec.local
Host hello.ksec.local not found: 3(NXDOMAIN)
```


Take note that you can add your own static DNS records inside your Kali host. The file is located at `/etc/hosts`, and here you can redirect any domain name to any live IP address. (This is how DNS poisoning works; the hacker will manipulate the A records to point to his server IP address.)

```
root@kali:~# cat /etc/hosts
127.0.0.1 localhost
127.0.1.1 kali

# The following lines are desirable for IPv6 capable hosts
::1 localhost ip6-localhost ip6-loopback
```

```
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

You'll learn more about this subject later in this book, and you will learn how DNS brute-forcing and zone transfers work.

Established Connections

To display the active network connections on your Kali host, you must use the `netstat` command tool to get the job done. You'll use this command in your post-exploitation phase to check how the Linux host is communicating with its network.

On our Kali host, we have started the SSH (port 22) and the web (port 80) services; the `netstat` tool will allow us to see them listening for incoming con nections:

```
root@kali:~# netstat -antu
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address Foreign Address State  tcp 0 0
0.0.0.0:22 0.0.0.0:* LISTEN  tcp6 0 0 :::80 :::* LISTEN  tcp6 0 0 :::22
:::* LISTEN  udp 0 0 10.0.0.185:68 10.0.0.1:67
ESTABLISHED
```

It's essential to understand what each option means:

- `-a/--all`: Display all the sockets. Take note that this option is very ver bose; thus, we need to combine it with the following options (to filter the output).
- `-n/--numeric`: Do not resolve names. In the previous command, you saw that the IP address is followed by the port number. If I don't use the `-n` option, then the tool will try to figure out the service name (for example, for 80, it's going to be HTTP instead).

- `-t/--tcp`: Display TCP connections.
- `-u/--udp`: Display UDP connections.

File Transfers

There are so many ways to transfer files in Kali Linux. First, to download files from the internet/intranet, you have two tools in your arsenal: `wget` and `curl`. In the following example, we use both of the tools to download a password text file from one of my local web servers:

```
$wget [URL]
root@kali:~# wget http://ubuntu.ksec.local/passwords.txt
--2020-10-01 13:32:02-- http://ubuntu.ksec.local/passwords.txt
Resolving ubuntu.ksec.local (ubuntu.ksec.local)... 10.0.0.186
Connecting to ubuntu.ksec.local
(ubuntu.ksec.local)|10.0.0.186|:80...  connected.
HTTP request sent, awaiting response... 200 OK
Length: 0 [text/plain]
Saving to: 'passwords.txt.1'

passwords.txt.1 [ <=>
```

```
] 0 --.-KB/s in 0s

2020-10-01 13:32:02 (0.00 B/s) - 'passwords.txt.1' saved [0/0]
$curl -O [URL]
root@kali:~# curl -O http://ubuntu.ksec.local/passwords.txt
 % Total % Received % Xferd Average Speed Time Time Time Current  Dload
Upload Total Spent Left Speed 100 32 100 32 0 0 16000 0 --:--:--
--:--:-- --:--:-- 16000
```

If you want to download files from GitHub, then you can use the `git` command: `$git clone [git project URL]`

Another way to securely transfer files using the SSH protocol is the `scp` command tool. It's important to understand that you will need the SSH ser vice to be started for this process to work properly. As usual, you see a practical example of how the workflow of copying works from source to destination.

First, we will start the SSH server on a remote Ubuntu Linux host, and this is where you're going to download my files. (By default SSH server is not installed

on Ubuntu. To get the job done, execute the command `$sudo apt install openssh-server -y`.) In this example, we are downloading a `passwords.txt` file from the remote Ubuntu server:

```
gus@ubuntu:~$ ls
Desktop Downloads passwords.txt Public Videos
Documents Music Pictures Templates
```

To get the job done of downloading the file, use the `scp` command with the following pattern (the dot at the end means that we are copying the file to our current directory in Kali):

```
$scp [remote-username@remote-ip:/remote-path] [destination local path]

root@kali:~# scp gus@ubuntu.ksec.local:/home/gus/passwords.txt
. gus@ubuntu.ksec.local's password:
passwords.txt
100% 17 16.7KB/s 00:00
```

Next, we will try to push a file called `test.txt` from my Kali to the remote SSH server (we will copy the file on the user's home directory in Ubuntu) using the `scp` command again:

```
$scp [file local path] [remote-username@remote-ip:/remote-path]

root@kali:~# scp /root/test.txt
gus@ubuntu.ksec.local:/home/gus gus@ubuntu.ksec.local's
password:
test.txt
100% 5 0.4KB/s 00:00
```

Later in this book, you will see even more ways to transfer files such as Samba, FTP, etc. For the time being, you just learned the most common ways that you need to be aware of.

Summary

With so many commands to learn in this chapter, it's overwhelming, right? The secret of mastering the usage of the terminal window is through practice. It will take a while to get familiar with the terminal window, but once you're in, you will fall in love with it.

Your role is focused on penetration testing, and the goal of this chapter is to make it easy for you to handle the system of Kali Linux. This chapter presented the necessary tools and commands that you will encounter during an engage ment. In the end, you're not a Linux system admin, but in cybersecurity, you will need to think out of the box.

2

Bash Scripting

In the previous chapter, you learned lots of commands in Linux. Now, let's take your skills to the next level in the command-line tools. In this chapter, you will see how to create scripted commands using Bash based on what you have learned so far.

Why Bash scripting? The universality of Bash gives us, penetration testers, the flexibility of executing powerful terminal commands without the need to install a compiler or an integrated development environment (IDE). To develop a Bash script, all you need is a text editor, and you're good to go.

When should you use Bash scripts? That's an important question to tackle before starting this chapter! Bash is not meant for developing sophisticated tools. If that's what you would like to do, you should use Python instead (Python fundamentals are covered later in this book). Bash is used for quick, small tools that you implement when you want to save time (e.g., to avoid repeating the same commands, you just write them in a Bash script).

This chapter will not only teach you the Bash scripting language, it will go beyond that to show you the ideology of programming as well. If you're new

to programming, this is a good starting point for you to understand how program ming languages work (they share a lot of similarities).

Here's what you're going to learn in this chapter:

- Printing to the screen using Bash
- Using variables

- Using script parameters
- Handling user input
- Creating functions
- Using conditional `if` statements
- Using `while` and `for` loops

Basic Bash Scripting

Figure 2.1 summarizes all the commands, so you can use it as a reference to grasp all the contents of this chapter. In summary, basic Bash scripting is divided into the following categories:

- Variables
- Functions
- User input
- Script output
- Parameters

Printing to the Screen in Bash

There are two common ways to write into the terminal command-line output using Bash scripting. The first simple method is to use the `echo` command that we saw in the previous chapter (we include the text value inside single quotes or double quotes):

```
$echo 'message to print.'
```

The second method is the `printf` command; this command is more flexible than the `echo` command because it allows you to format the string that you want to print:

```
$printf 'message to print'
```

The previous formula is too simplified; in fact, `printf` allows you to format strings as well (not just for printing; it's more than that). Let's look at an example: if we want to display the number of live hosts in a network, we can

use the following pattern:

```
root@kali:~# printf "%s %d\n" "Number of live hosts:" 15
Number of live hosts: 15
```

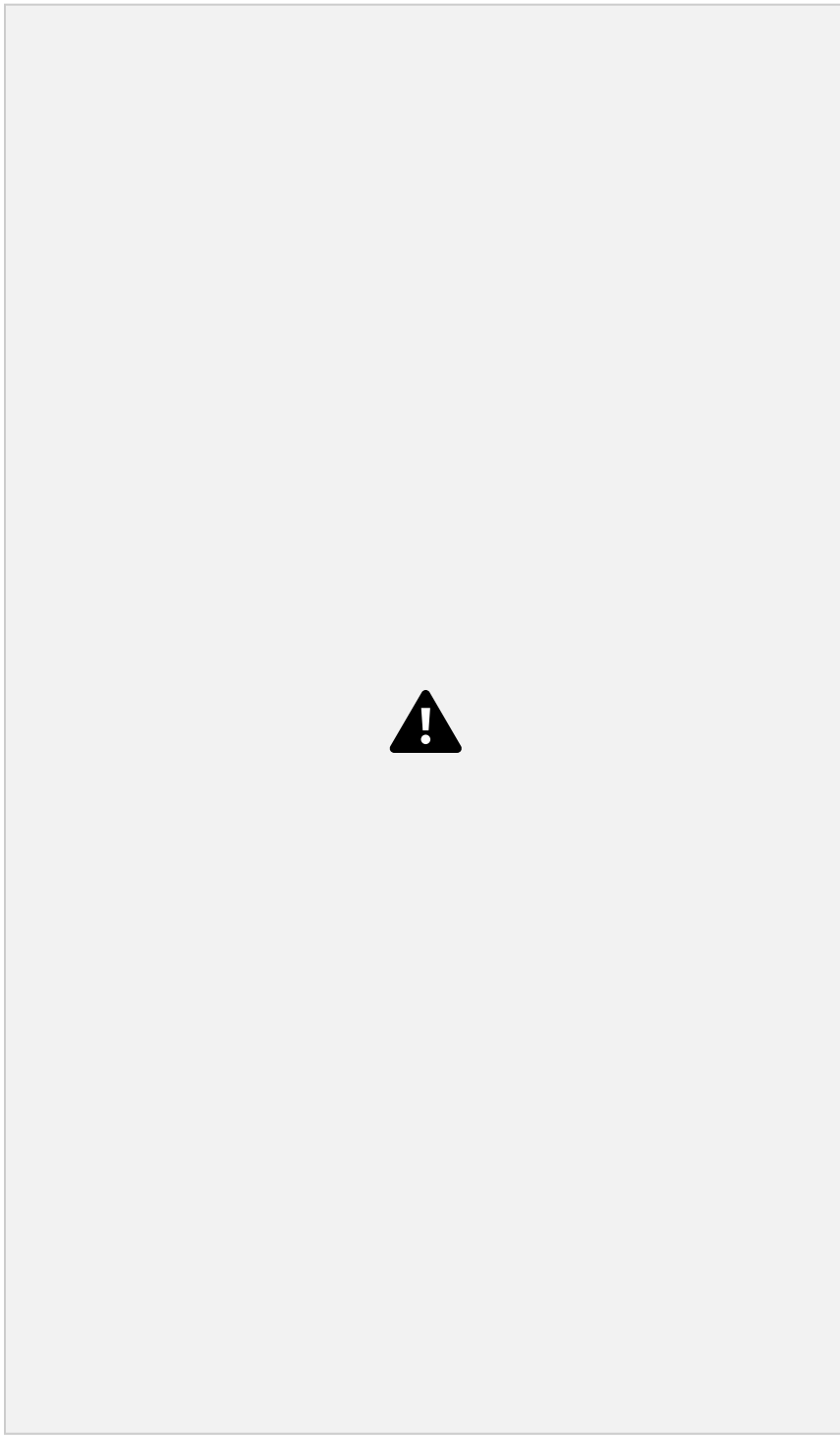



Figure 2.1: Bash Scripting

Let's divide the command so you can understand what's going on:

- `%s`: Means we're inserting a string (text) in this position
- `%d`: Means we're adding a decimal (number) in this position
- `\n`: Means that we want to go to a new line when the print is finished

Also, take note that we are using double quotes instead of single quotes. Double quotes will allow us to be more flexible with string manipulation than the single quotes. So, most of the time, we can use the double quotes for `printf` (we rarely need to use the single quotes).

To format a string using the `printf` command, you can use the following patterns:

- `%s`: String (texts)
- `%d`: Decimal (numbers)
- `%f`: Floating-point (including signed numbers)
- `%x`: Hexadecimal
- `\n`: New line
- `\r`: Carriage return
- `\t`: Horizontal tab

Variables

What is a variable, and why does every programming language use it anyway? Consider a variable as a storage area where you can save things like strings and numbers. The goal is to reuse them over and over again in your program, and this concept applies to any programming language (not just Bash scripting). To declare a variable, you give it a name and a value (the value is a string by default). The name of the variable can only contain an alphabetic character or underscore (other programming languages use a different naming convention). For example, if you want to store the IP address of the router in a variable, first you will create a file `var.sh` (Bash script files will end with `.sh`), and inside the file, you'll enter the following:

```
#!/bin/bash
#Simple program with a variable

ROUTERIP="10.0.0.1"

printf "The router IP address: $ROUTERIP\n"
```


Let's explain your first Bash script file:

- `#!/bin/bash` is called the *Bash shebang*; we need to include it at the top to tell Kali Linux which interpreter to use to parse the script file (we will

use the same concept in Chapter 18, "Pentest Automation with Python," with the Python programming language). The `#` is used in the second line to indicate that it's a comment (a comment is a directive that the creator will leave inside the source code/script for later reference).

- The variable name is called `ROUTERIP`, and its value is 10.0.0.1.
- Finally, we're *printing* the value to the output screen using the `printf` function.

To execute it, make sure to give it the right permissions first (look at the fol lowing output to see what happens if you don't). Since we're inside the same directory (`/root`), we will use `./var.sh` to execute it:

```
root@kali:~# ./var.sh
bash: ./var.sh: Permission denied
root@kali:~# chmod +x var.sh
root@kali:~# ./var.sh
The router IP address: 10.0.0.1
```

Congratulations, you just built your first Bash script! Let's say we want this script to *run automatically* without specifying its path anywhere in the system. To do that, we must add it to the `$PATH` variable. In our case, we will add `/opt` to the `$PATH` variable so we can save our custom scripts in this directory.

First, open the `.bashrc` file using any text editor. Once the file is loaded, scroll to the bottom and add the line highlighted in Figure 2.2.

Figure 2.2: Export Config

The changes will append `/opt` to the `$PATH` variable. At this stage, save the file and close all the terminal sessions. Reopen the terminal window and copy the script file to the `/opt` folder. From now on, we don't need to include its path; we just execute it by typing the script name `var.sh` (you don't need to re-execute the `chmod` again; the execution permission has been already set):

```
root@kali:~# cp var.sh /opt/
root@kali:~# cd /opt
root@kali:/opt# ls -la | grep "var.sh"
```

Continues

(continued)

```
-rwxr-xr-x 1 root root 110 Sep 28 11:24 var.sh
root@kali:/opt# var.sh
The router IP address: 10.0.0.1
```

Commands Variable

Sometimes, you might want to execute commands and save their output to a var iable. Most of the time, the goal behind this is to manipulate the contents

of the command output. Here's a simple command that executes the `ls` command and filters out the filenames that contain the word *simple* using the `grep` command. (Don't worry, you will see more complex scenarios in the upcoming sections of this chapter. For the time being, practice and focus on the fundamentals.)

```
#!/bin/bash
LS_CMD=$(ls | grep 'simple')
printf "$LS_CMD\n"
```

Here are the script execution results:

```
root@kali:/opt# simplels.sh
simpleadd.sh
simplels.sh
```

Script Parameters

Sometimes, you will need to supply parameters to your Bash script. You will have to separate each parameter with a space, and then you can manipulate those params inside the Bash script. Let's create a simple calculator (`simpleadd .sh`) that adds two numbers:

```
#!/bin/bash
#Simple calculator that adds 2 numbers

#Store the first parameter in num1 variable
NUM1=$1
#Store the second parameter in num2 variable
NUM2=$2
#Store the addition results in the total variable
TOTAL=$(($NUM1 + $NUM2))

echo '#########################'
printf "%s %d\n" "The total is =" $TOTAL
echo '#########################'
```

You can see in the previous script that we accessed the first parameter using the `$1` syntax and the second parameter using `$2` (you can add as many param eters as you want).