# myServer.py will create a Kali Web Server

import socket

import sys

# We need to pass an empty string, so that all interfaces are available

HOST = ''

# You can choose any arbitrary port number

PORT = 8080

mySocket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

print('Socket has been created')

# Let us bind the socket to local host and port

try:

mySocket.bind((HOST, PORT))

except socket.error as msg:

print('Binding has failed. Error Code is : ' + str(msg[0]) + ' Message ' + msg[1])

sys.exit()

print('Socket bind is complete. Now we can proceed to make it listen...')

# Server is listening now on socket

mySocket.listen(10)

print('Socket is now listening')

# Let the server keep talking with the client

while 1:

# We are waiting to accept a connection - blocking call

connection, address = mySocket.accept()

print('Connected with ' + address[0] + ':' + str(address[1]))

mySocket.close()

---------------

python myServer.py

----------

# telnet command to run the localhost

telnet localhost 8080

------------

# How to connect to Google by using the socket programming in Python

# the first line refers to the socket, we need to import it from the library

import socket

import sys

try:

mySocket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

print("Socket successfully created")

except socket.error as err:

print("socket creation failed with error %s" % (err))

# default port for the socket

port = 80

try:

host\_ip = socket.gethostbyname('www.google.com')

except socket.gaierror:

# this means could not resolve the host

print("there was an error resolving the host")

sys.exit()

# connecting to the server

mySocket.connect((host\_ip, port))

print("the socket has successfully connected to google on port == %s" % (host\_ip))

----------------

Now we will see how we can build another Kali Linux local web server that listens to a certain port that we define (figure 7.3).

# first of all import the socket library import socket

# next we will create a socket object

mySocket = socket.socket()

print("Socket successfully created")

# let us reserve a port on our computer

# in our case it is 8080 but it can be anything like 12345

port = 8080

# Next we will bind to the port and we have not typed any IP in the ip field

# we keep an empty string; because, this makes the server listen to any request

# coming from other computers on the network

mySocket.bind(('', port))

print("socket bounded to %s" % (port))

# let us put the socket into listening mode

mySocket.listen(5)

print("socket is now listening")

# we can make it a forever loop until we interrupt it or an error occurs

while True:

# Establish connection with client.

c, addr = mySocket.accept()

print('Got a connection from this', addr)

# we can send a thank you message to the client.

c.send('Thank you for connecting')

# Close the connection with the client

c.close()

----------------

pg@kali:~$ cd PycharmProjects/kaliServer/

pg@kali:~/PycharmProjects/kaliServer$ ls

kaliServer.py serverClient.py venv

pg@kali:~/PycharmProjects/kaliServer$ python serverClient.py

Socket successfully created

socket bounded to 8080

socket is listening

('Got connection from', ('127.0.0.1', 51290))

--------------

pg@kali:~$ cd PycharmProjects/kaliServer/

pg@kali:~/PycharmProjects/kaliServer$ telnet localhost 8080

Trying ::1...

Trying 127.0.0.1...

telnet: Unable to connect to remote host: Connection refused

-------------

# running pycharm.sh file

./pycharm.sh

// You can also install it through terminal. Just type this command on your terminal.

#installing Pycharm through terminal

sudo apt-get install pycharm-community

--------------

su leafpad /etc/proxychains.conf

-------------

# Proxy DNS requests - no leak for DNS data

proxy\_dns

------

[ProxyList]

# add proxy here ...

# meanwile

# defaults set to "tor"

socks4 127.0.0.1 9050

socks5 127.0.0.1 9050

socks5 185.43.7.146 1080

socks5 75.98.148.183 45021

-----------

# ProxyList format

# type host port [user pass]

# (values separated by 'tab' or 'blank')

# Examples:

# socks5 192.168.67.78 1080 lamer secret

# http 192.168.89.3 8080 justu hidden

# socks4 192.168.1.49 1080

# http 192.168.39.93 8080

------------

# socks5 192.168.67.78 1080 lamer secret

------------

socks5 127.0.0.1 9050

-----------

// Open up your terminal and type:

service tor status

// It will fail if you don't start it. So type the following to start the service:

service tor start

// Now you can open up your browser through the terminal. Just type:

proxychains firefox www.duckduckgo.com

--------

cat /etc/resolv.conf

// It will show something like this:

# Generated by NetworkManager

nameserver 192.168.1.1

--------------

nameserver 208.67.222.222

nameserver 208.67.220.220

----------

nano /etc/dhcp/dhclient.conf

----------

prepend domain-name-servers 127.0.0.1;

-----------

prepend domain-name-servers 208.67.222.222 208.67.220.220;

----------

service network-manager restart

// root@kali:~# cd Downloads/

root@kali:~/Downloads# ls

VPNBook.com-OpenVPN-DE1.zip

// I have downloaded openvpn zipped file. Now, I am going to unzip it using the following command.

root@kali:~/Downloads# unzip VPNBook.com-OpenVPN-DE1.zip

Archive: VPNBook.com-OpenVPN-DE1.zip

inflating: vpnbook-de233-tcp80.ovpn

inflating: vpnbook-de233-tcp443.ovpn

inflating: vpnbook-de233-udp53.ovpn

inflating: vpnbook-de233-udp25000.ovpn

// Now, we can have a look what is inside the openvpn folder.

root@kali:~/Downloads# ls

VPNBook.com-OpenVPN-DE1.zip vpnbook-de233-udp25000.ovpn

vpnbook-de233-tcp443.ovpn vpnbook-de233-udp53.ovpn

vpnbook-de233-tcp80.ovpn

// Now issue this command with your Internet connection open:

openvpn vpnbook-de233-tcp443.ovpn

// Now you can check your name server again. It'll show two new addresses.

root@kali:/home/ss# nano /etc/dhcp/dhclient.conf

root@kali:/home/ss# service network-manager restart

root@kali:/home/ss# cat /etc/resolv.conf

# Generated by NetworkManager

search domain.name

nameserver 208.67.222.222

nameserver 208.67.220.220

root@kali:/home/ss#

--------------

command: ipconfig

---------

root@kali:~# macchanger -s eth0

Current MAC: xx:xx:xx:xx:16:ec (CADMUS COMPUTER SYSTEMS)

Permanent MAC: xx:xx:xx:xx:xx:ec (CADMUS COMPUTER SYSTEMS)

----------

root@kali:~# macchanger –h

----------

root@kali:~# macchanger -m mac=XX:XX:XX:XX:XX:XXeth0

-----------