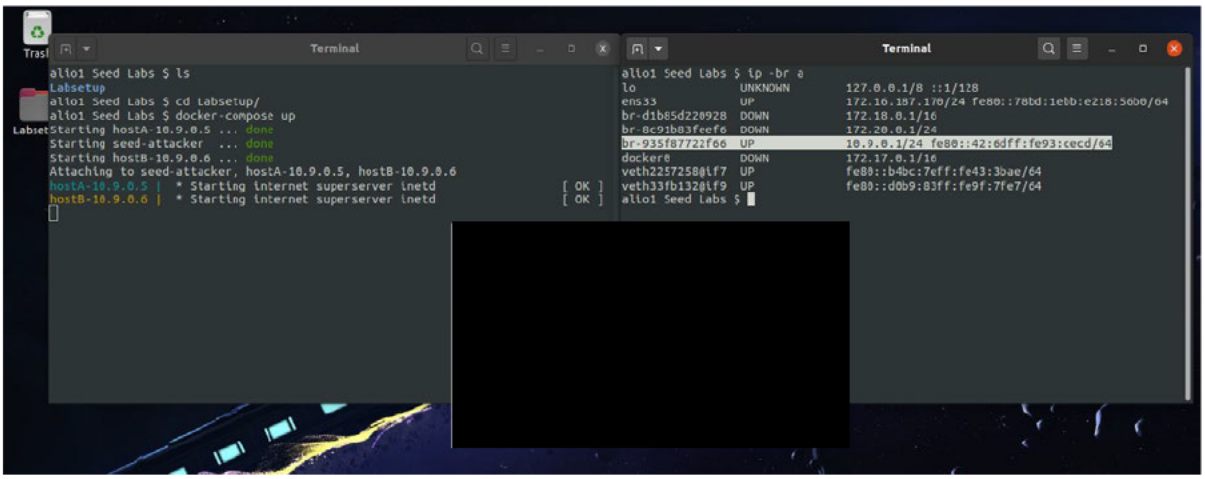
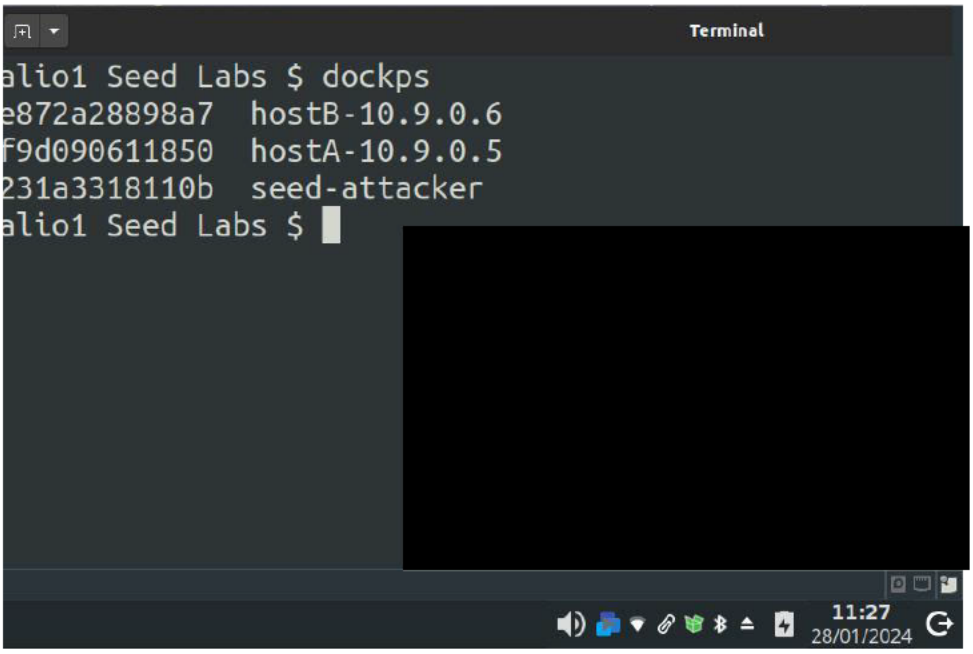
Tasks

Task 0 – Setup

The first task was to set up the container network, this was done by unzipping the requested file and running *docker build* and then *docker up*. The following two screenshots show the containers being built, the network configuration using *ip -br a* and the containers running using *dockps*





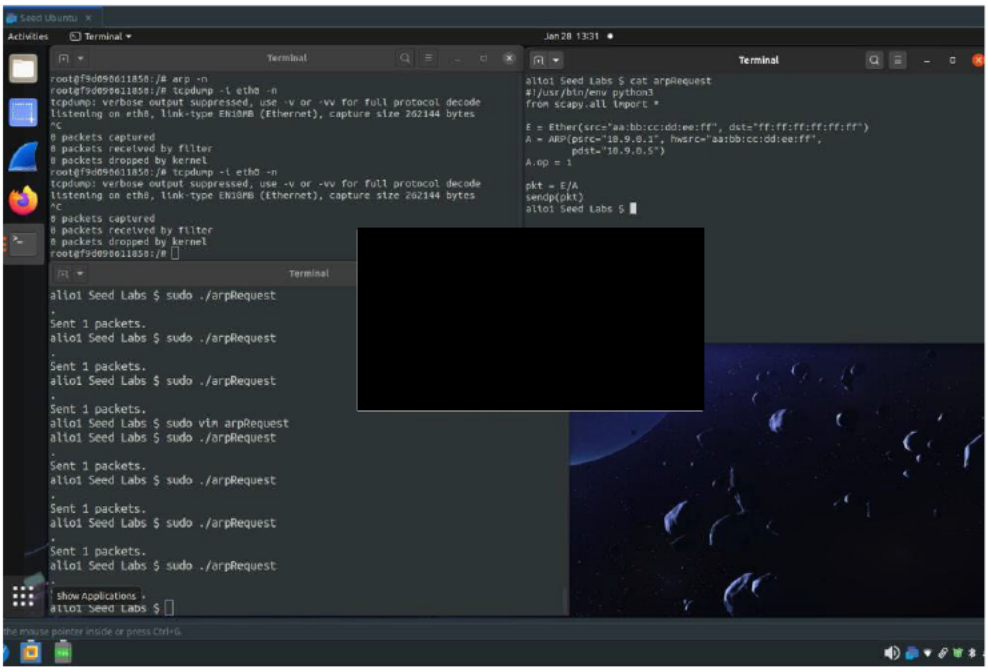
Task 1 – ARP Cache Poisoning

The first task for this lab has us create an ARP Cache Poisoning python script in a few scenarios. These are outlined below.

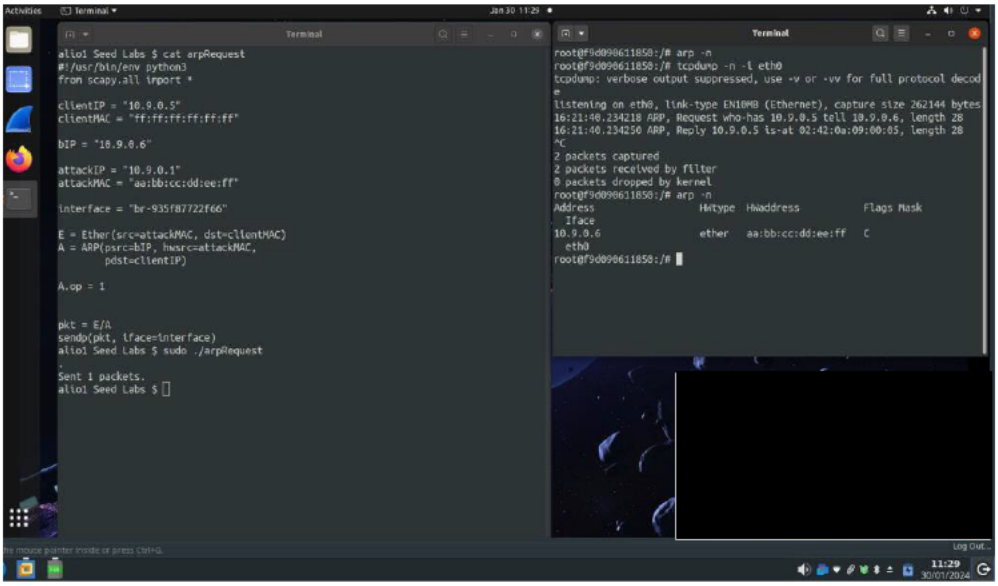
Task 1.A (using ARP request)

The first task asked us to construct an ARP request packet to map B's IP address (10.9.0.6) to M's MAC address. I opted to use my VM as the means of facilitating the attacks. We do this by construct an ARP Request (ARP code 1) where the destination IP is A (10.9.0.5), we set the

destination MAC address to the broadcast address (ff:ff:ff:ff:ff:ff) so that it will broadcast to all devices on the network



This proved unsuccessful, the device would not pick up the packet regardless of code configuration unless the attacking device would ping prior to sending the spoofed packet. However, by adding the line specificying interface (iface=interface) I was able to send the packet on the right interface and poison the cache.

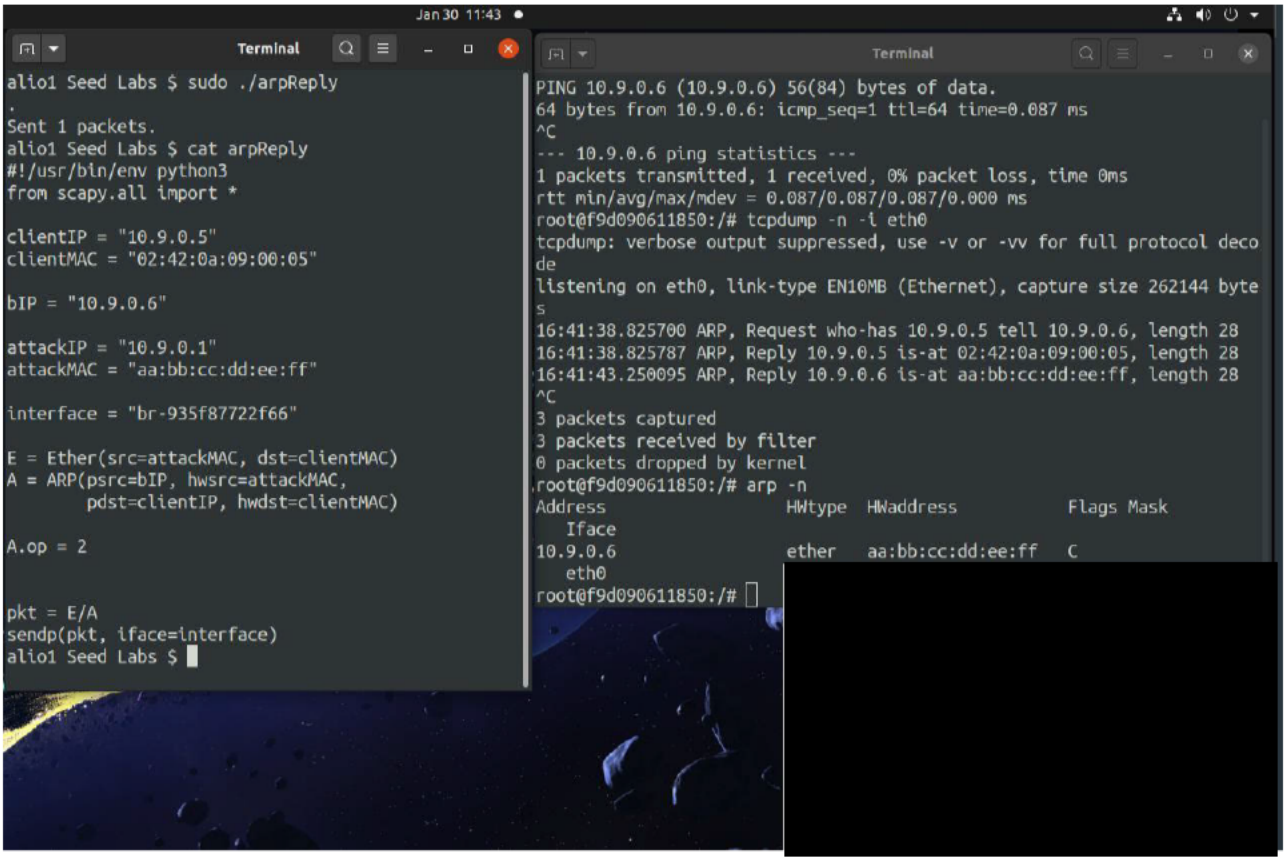


Task 1.B (using ARP reply)

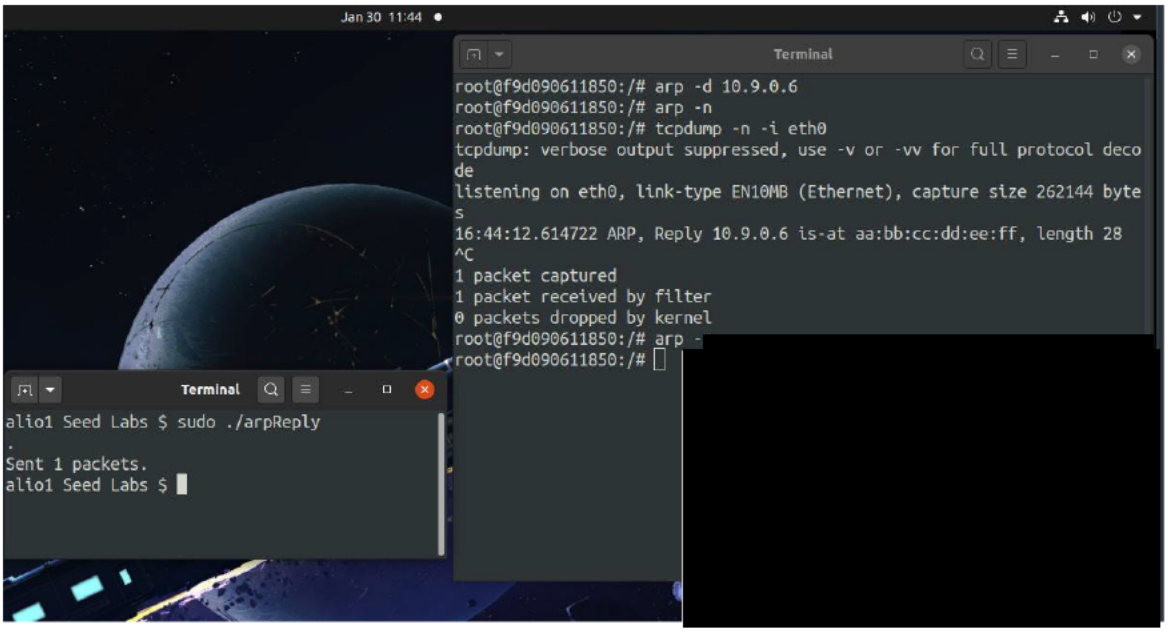
For this step I utilized a similar code but included the ARP code of 2, as well as including the destination MAC address. I got the MAC address for hostA using *docker inspect hostA* and received the following:

```
"MacAddress": "02:42:0a:09:00:05",
```

After this, attempting to ping created an entry within the arp cache, we can then poison it using the code snippet featured in the screenshot below, which allows shows the successful poisoning.

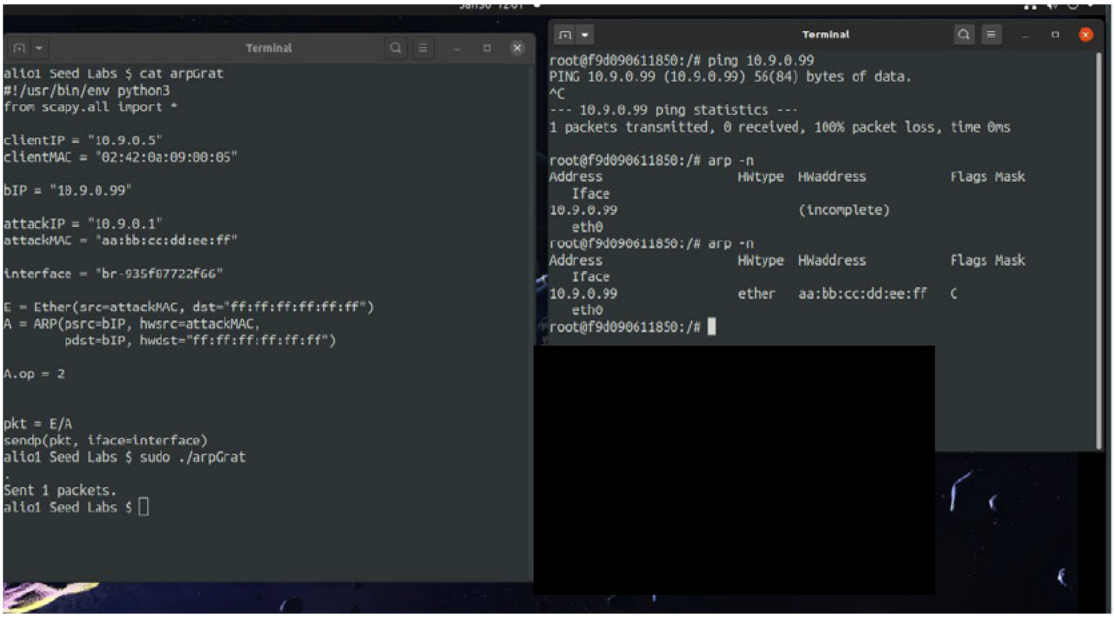


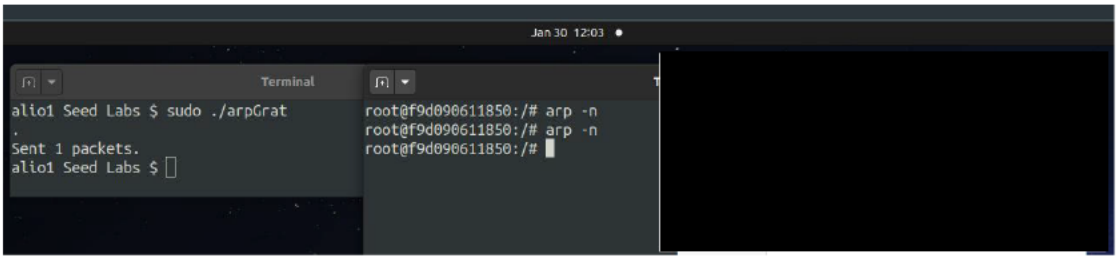
However, without a previous ping to create the arp cache, the packet was ignored by machine A. As shown in the screenshot below.



Task 1.C (using ARP gratuitous message)

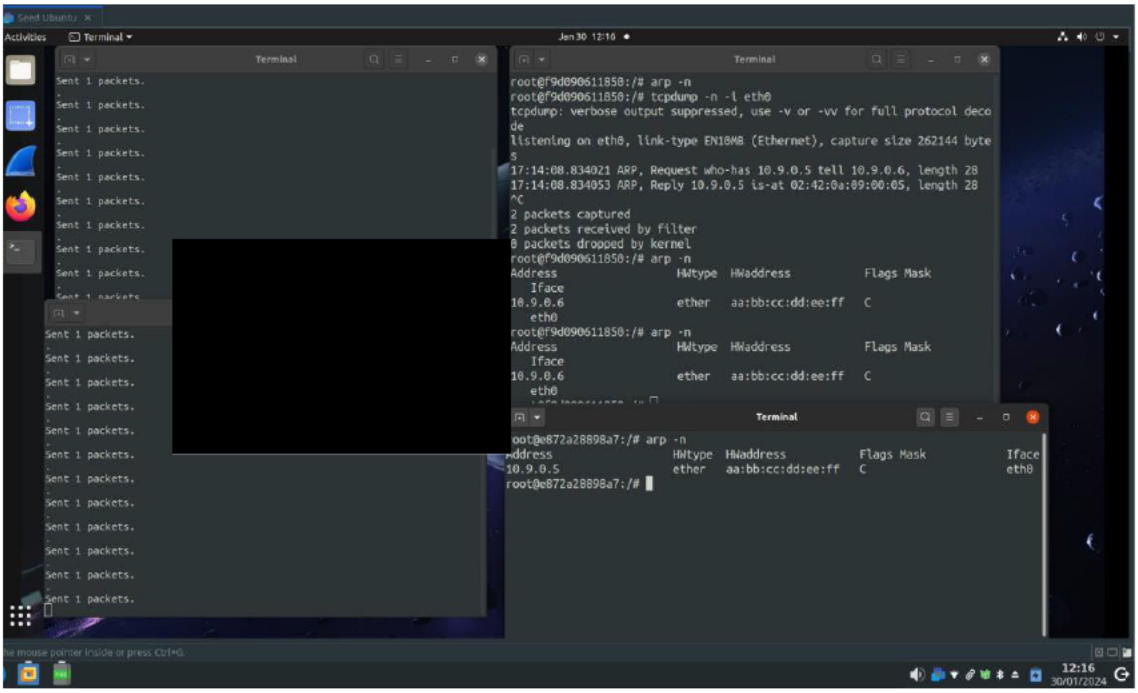
As discussed in the book, creating a blank entry allows the gratuitous message to poison the cache, however if there is no cache, this packet will be ignored, as shown in the second screenshot.

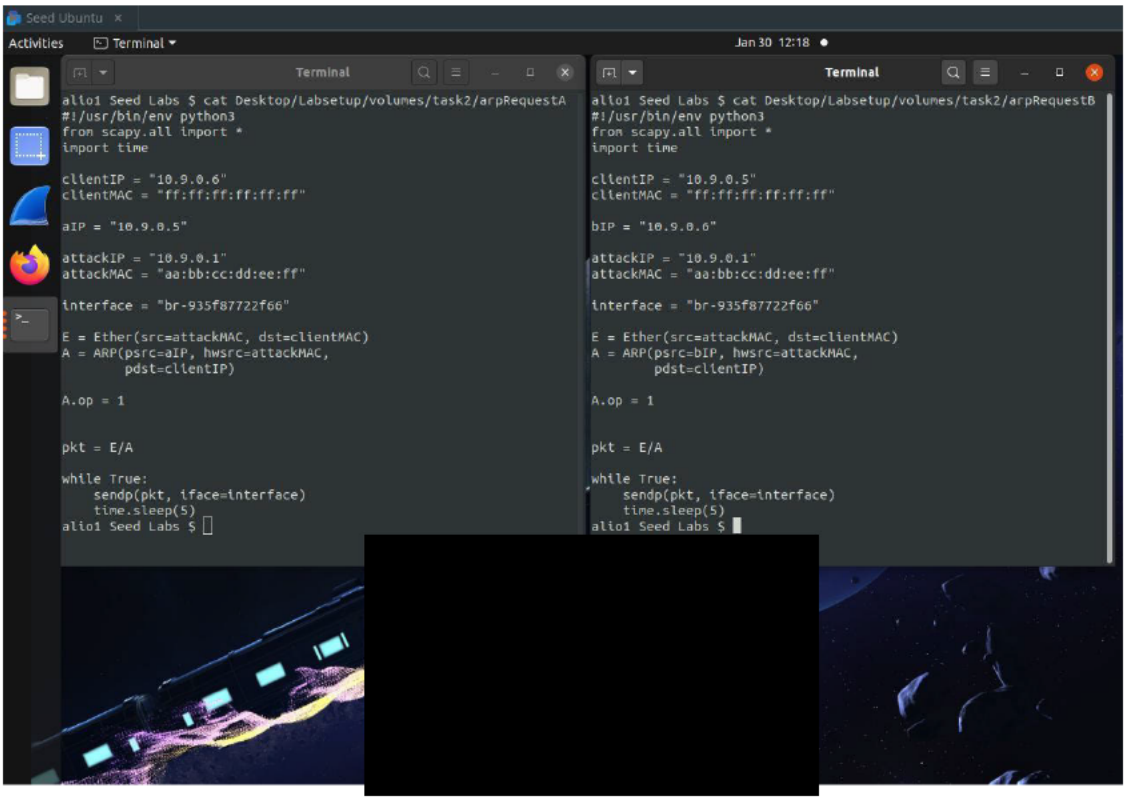




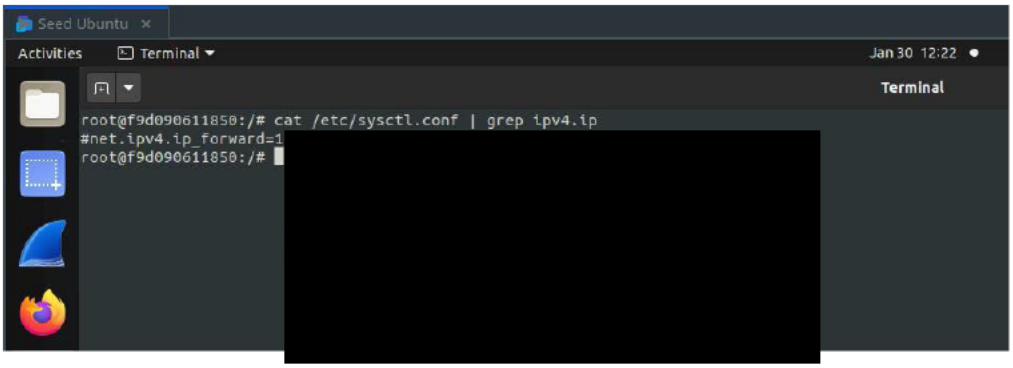
Task 2 – AITM Attack on Telnet using ARP Cache Poisoning

For this attack, I started by modifying the arpRequest script, and making a copy to poison the ARP caches of A and B respectively, the outcome display below. With the time library added, set a delay, sending an ARP request every 5 seconds until interrupted.

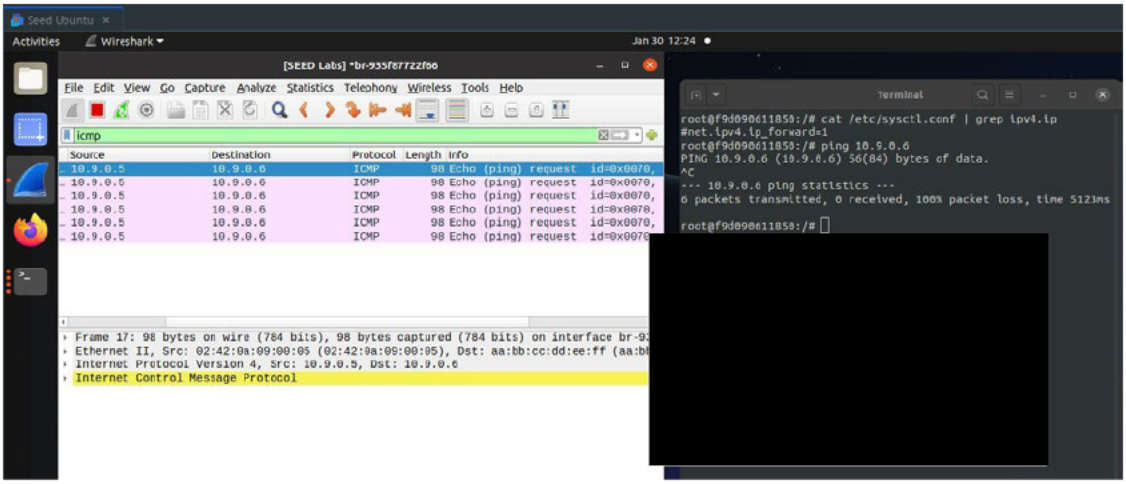




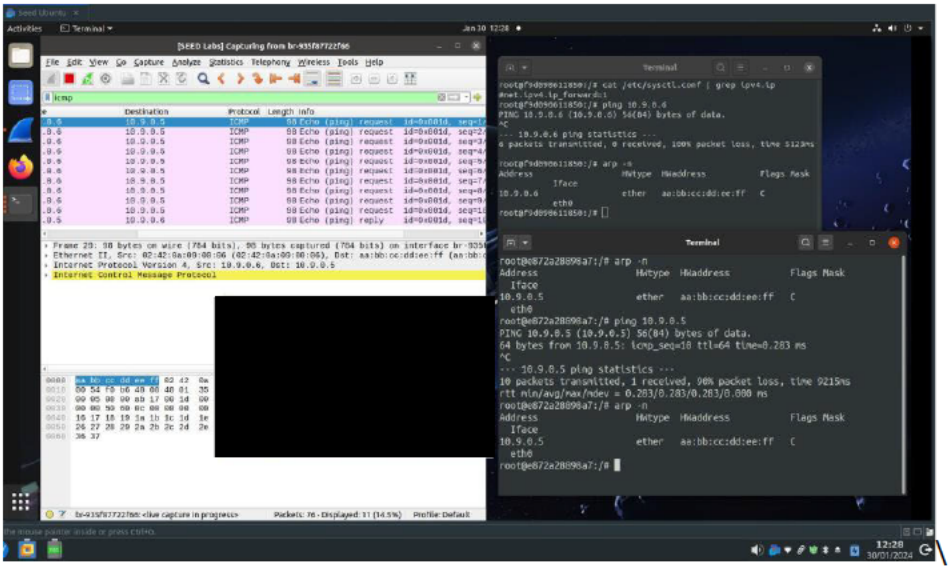
After this I ensured IP forwarding was off, looking at /etc/sysctl.conf



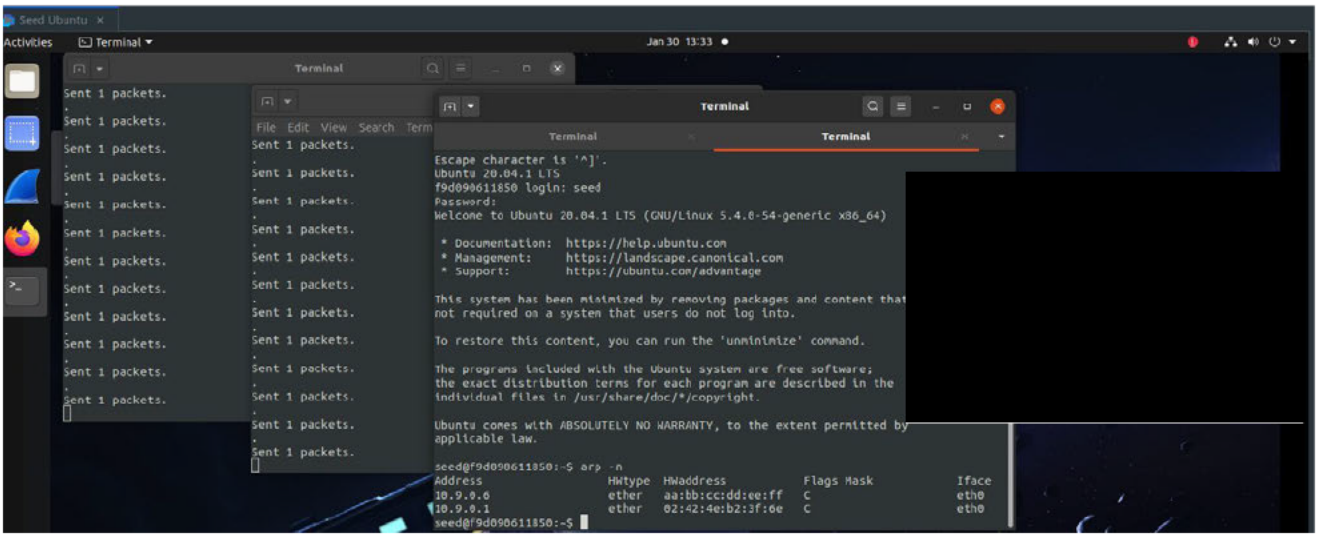
With IP forwarding off, these ICMP requests are caught on WireShark but do not get a response

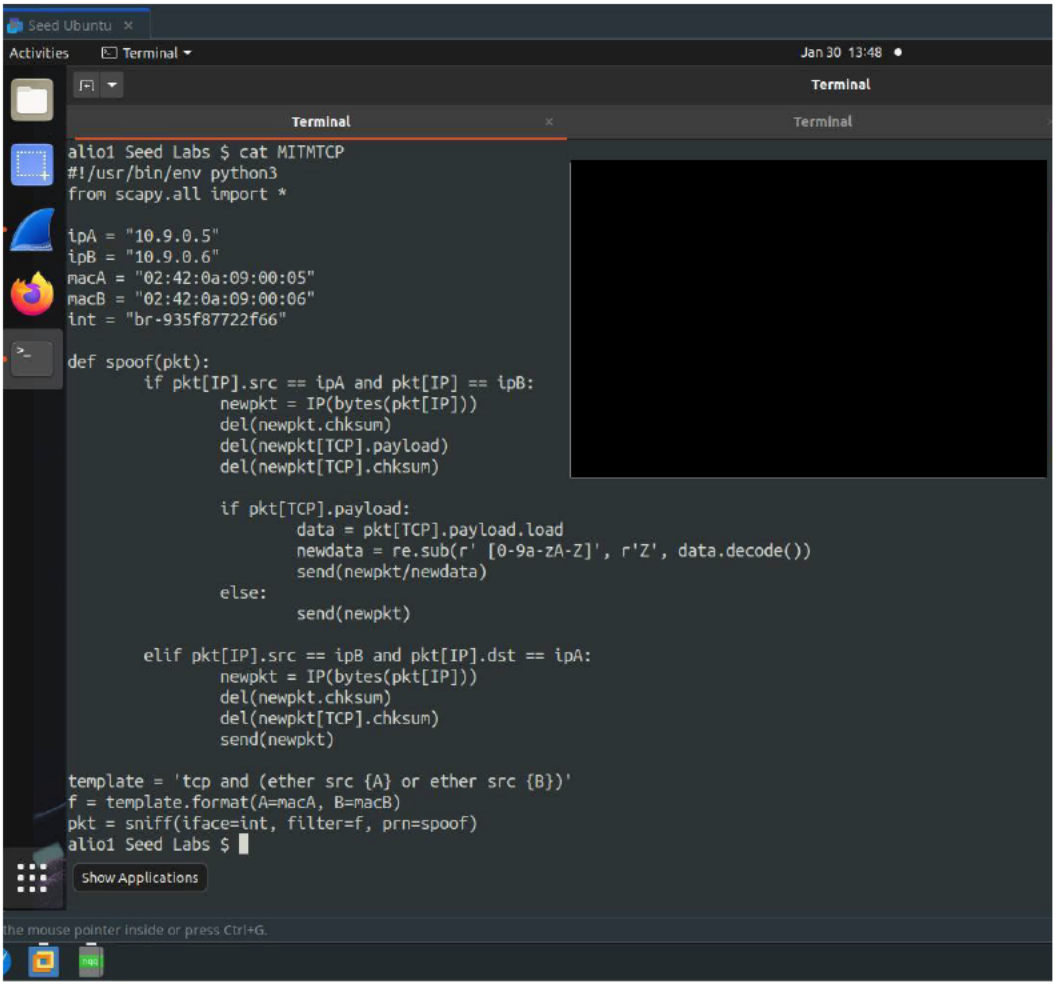


With ip.forwading turn on, on Machine H we have the following output. Where there was no reply for about 10 pings but was responded to eventually, the ARP cache still shows the poisoned IP address.

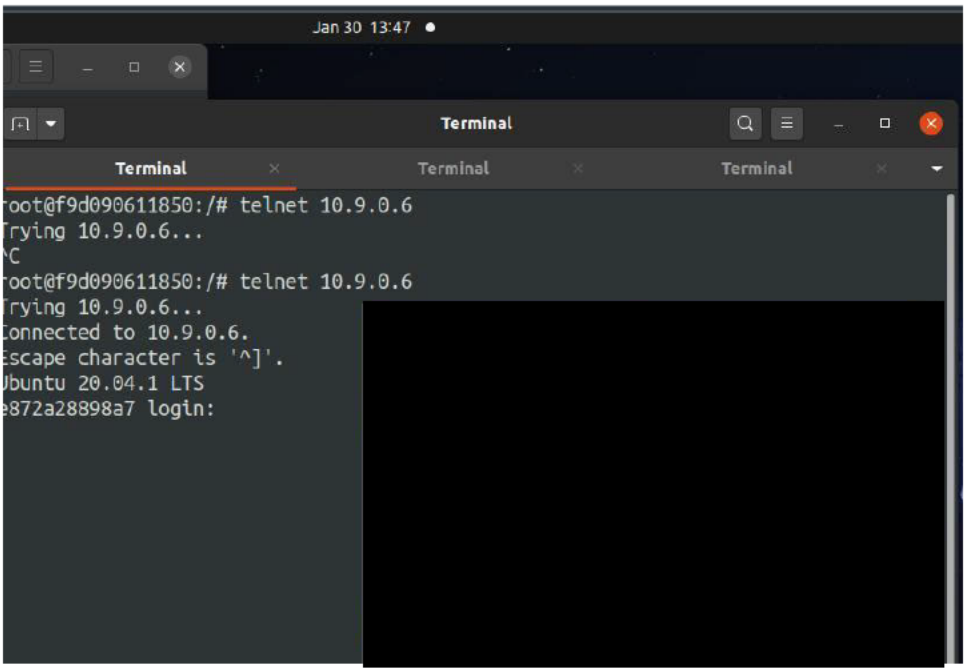


The last step for this Task was to Telnet from A to B and intercept, accomplished by modifying the skeletal structure presented in the PDF as the following code. The first screenshot shows the successful Telnet session. The second shows the new code.



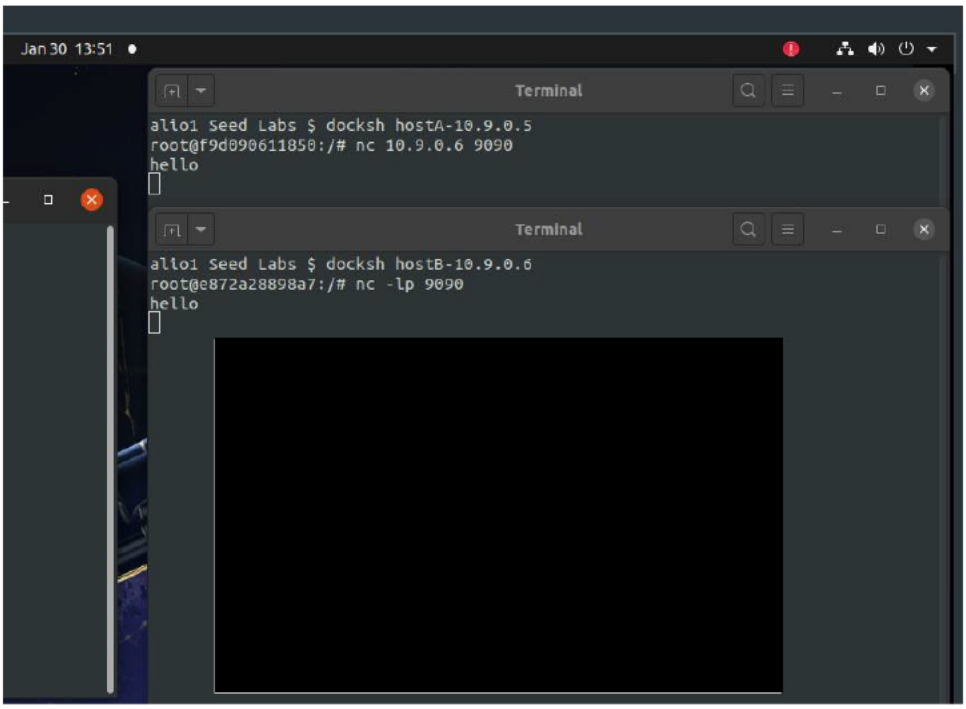


After this, the expected outcome was to turn inputs into Zs. This will be developed upon in Task 5. However, upon establishing the TelNet connection, it froze.

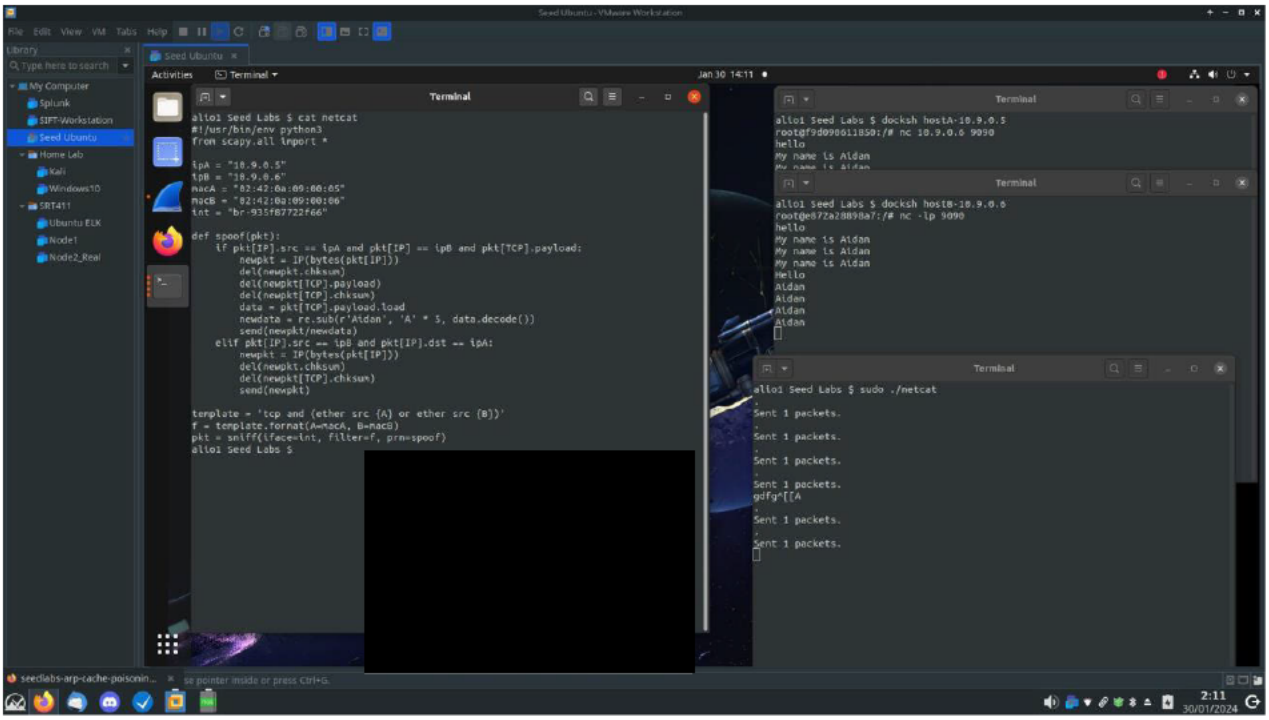


Task 3 – AITM Attack on Netcat using ARP Cache Poisoning

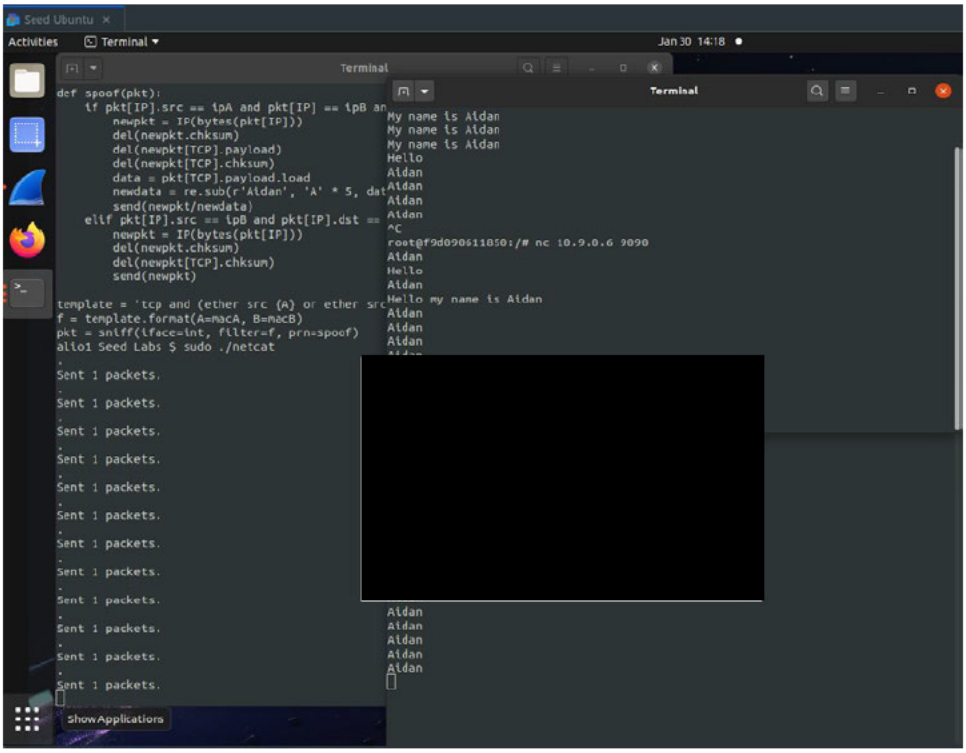
The final textbook task asked us to utilize a similar pattern with the usage of netcat. A connection was established to showcase connectivity



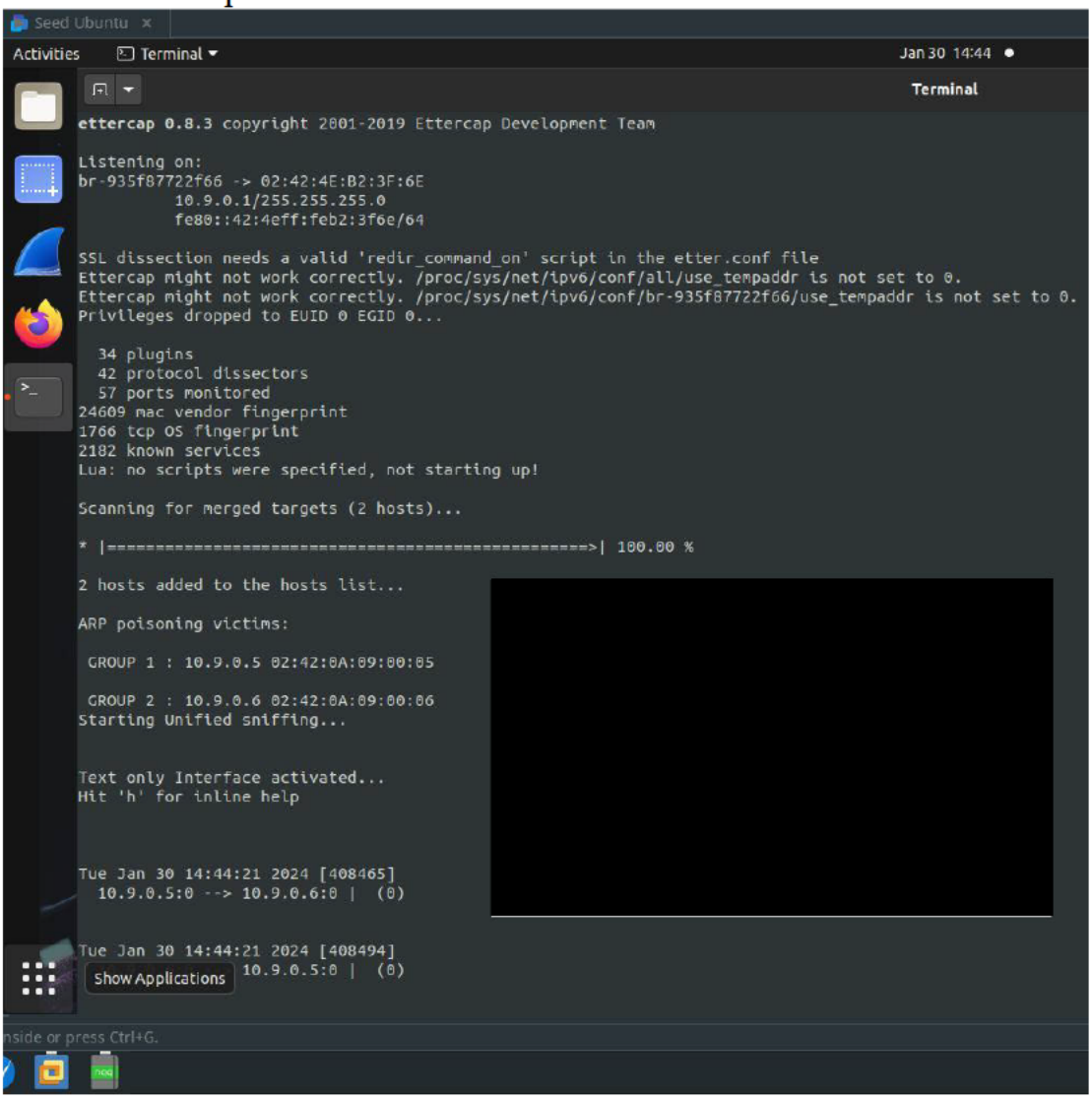
I modified the script used in the last part of Task 2, however, I was unable to intercept the packets at first. As shown in this screenshot.



I turned off IPv4 forwarding, the netcat froze momentarily, and then I had this output. I am unsure of why the output completely froze. This will require more investigation.



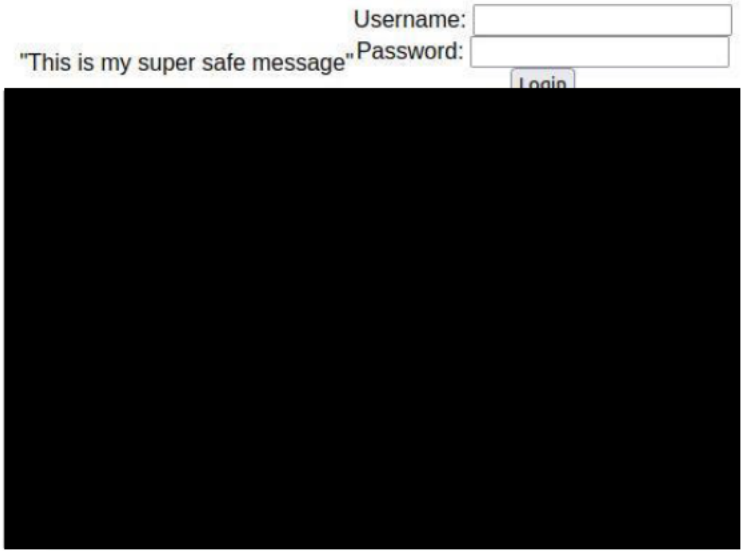
Task 4 – Ettercap

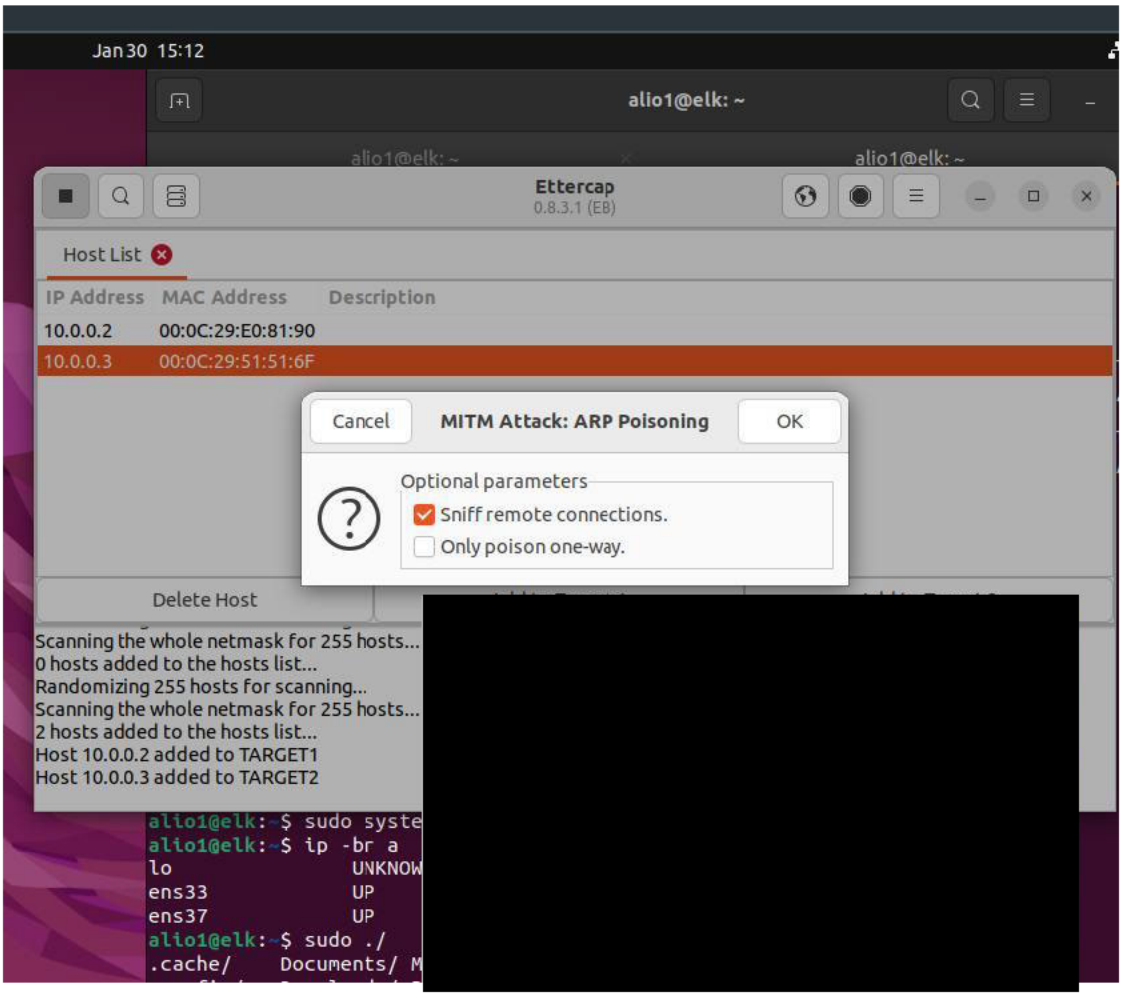


This task was accomplished by utilizing the ettercap command of *sudo ettercap –T –i br-935f87722f66 –M arp //10.9.0.5// //10.9.0.6//* which starts ettercap in the specificed CLI, listening on interface of our docker network. Moreover, the –M enables an AITM attack and arp specifies the protocol. Lastly, the IP addressess encased in backslashes add those IP addresses to the list of hosts to listen for.

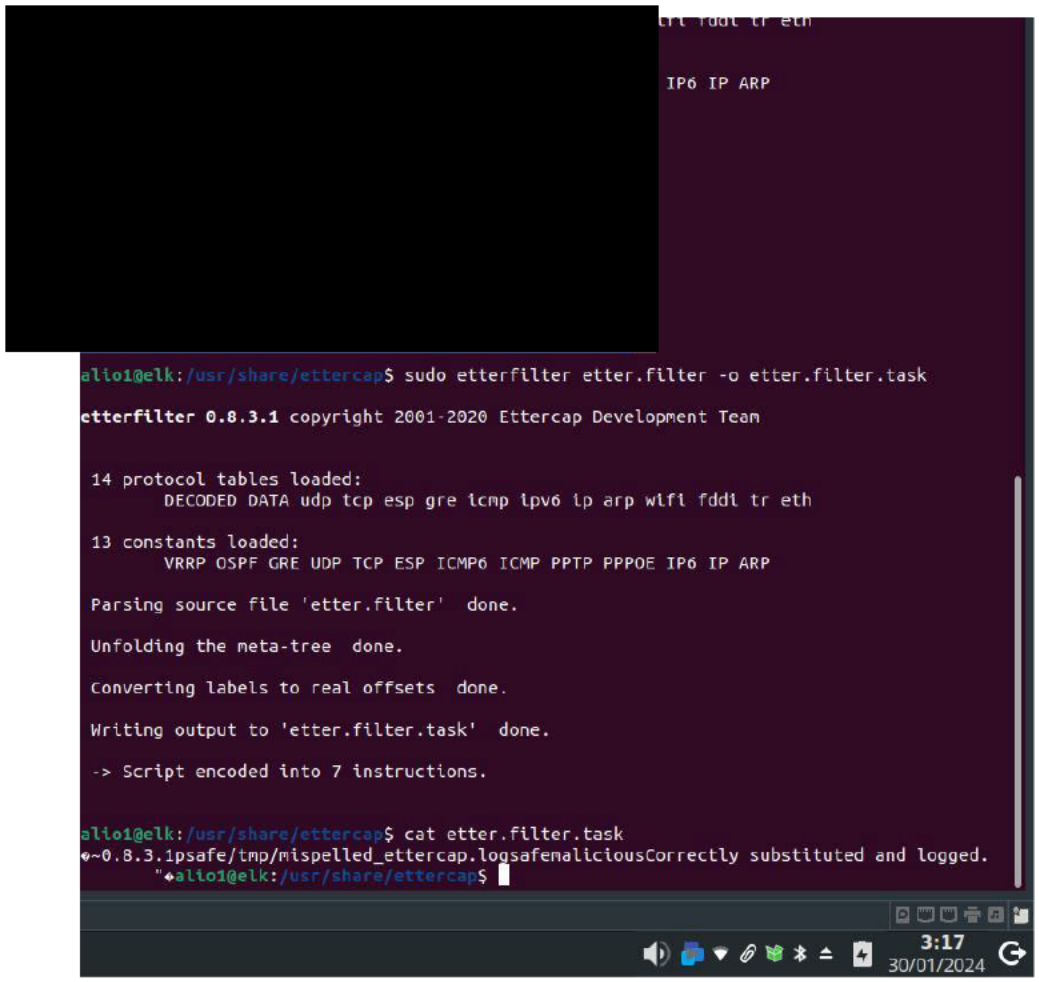
Task 5 – Ettercap Docker Tutorial

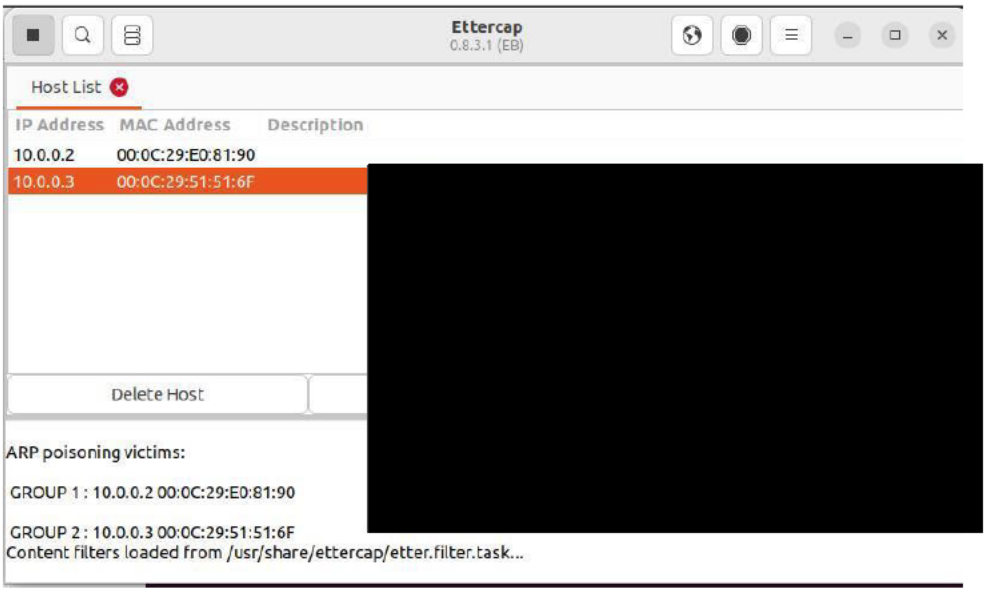
For this task, I utilized an Ubuntu VM that already had an Apache webserver loaded on port 80. I added a modified message that I intend to change.

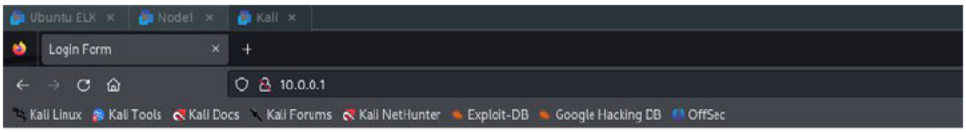

Username:
"This is my super safe message"
Password:
Login

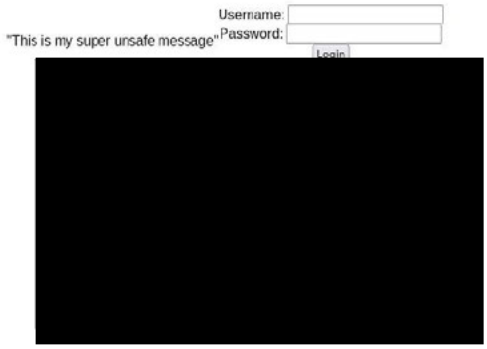

Jan 30 15:12
alio1@elk: ~
alio1@elk: ~
alio1@elk: ~
Ettercap
0.8.3.1 (EB)
Host List
IP Address
MAC Address
Description
10.0.0.2
00:0C:29:E0:81:90
10.0.0.3
00:0C:29:51:51:6F
Cancel
MITM Attack: ARP Poisoning
OK
Optional parameters
Sniff remote connections.
Only poison one-way.
Delete Host
Scanning the whole netmask for 255 hosts...
0 hosts added to the hosts list...
Randomizing 255 hosts for scanning...
Scanning the whole netmask for 255 hosts...
2 hosts added to the hosts list...
Host 10.0.0.2 added to TARGET1
Host 10.0.0.3 added to TARGET2
alio1@elk: $ sudo syste
alio1@elk:~$ ip -br a
lo UNKNOW
ens33 UP
ens37 UP
alio1@elk:~$ sudo ./
.cache/ Documents/ M

For the most part, I followed the instruction, except I chose to filter out the word “safe” to [redacted].









Task 6 – Challenge

I have opted out of doing this challenge; however, I will explain what my thought process would be.

The goal of this challenge was to replace inputs with a command like *sudo rm –rf /dev*. This command could be anything, but I figured this would be disruptive. To facilitate this, we would take the MITM script utilized earlier that replaced text to Z. This utilized regex to replace any characters with Z. We could utilize this resource to create the Regex to replace inputs into strings (source) or we could utilize Python's replace function to replace any input with a string, and then any secondary input with enter, which would force the removal of the directory on the receiving machine.