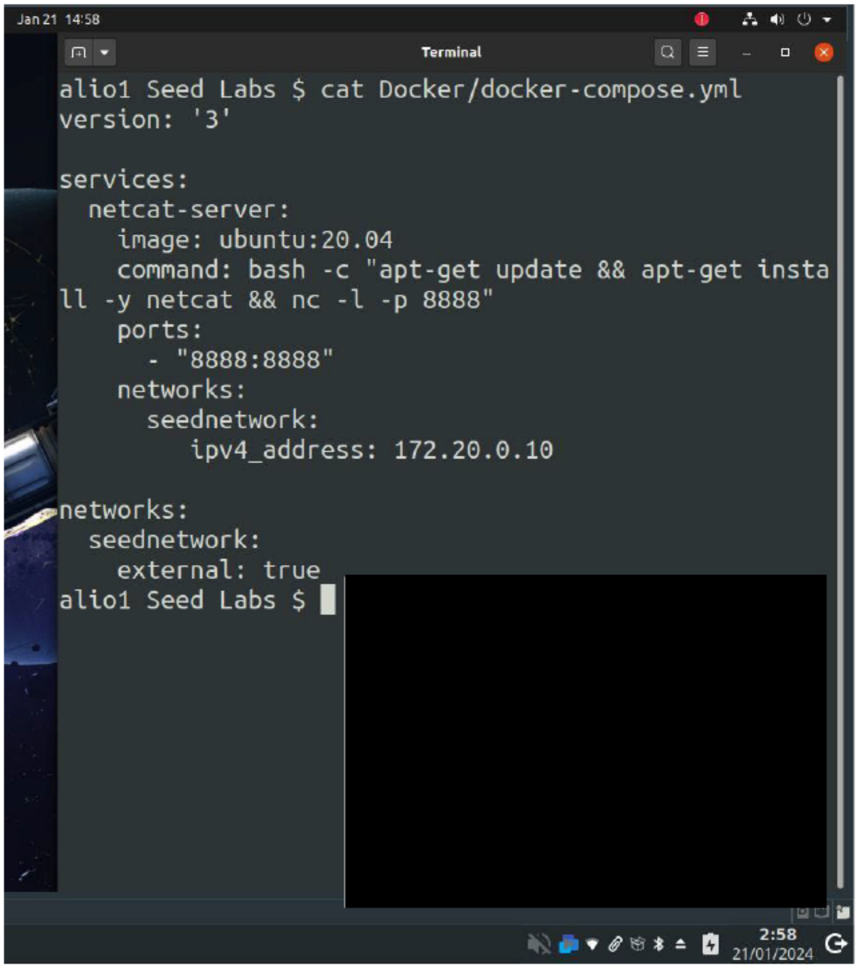
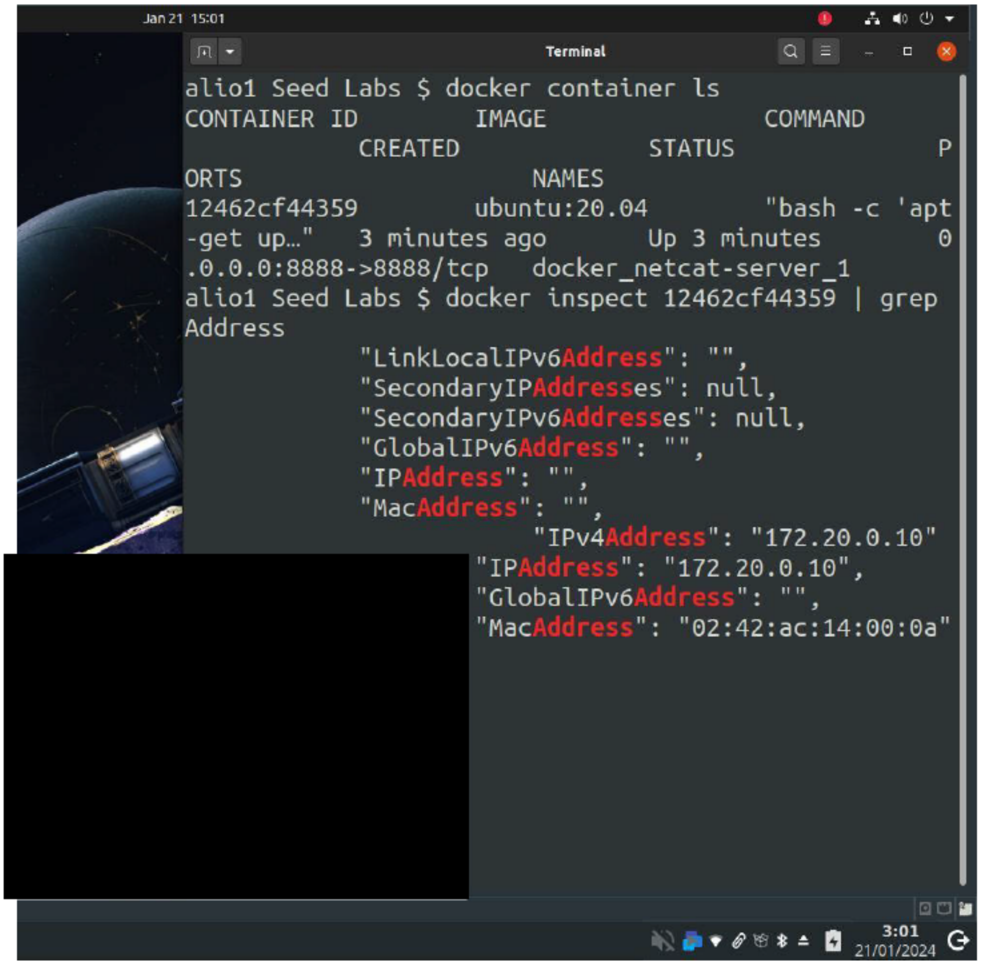
Tasks

Part 1: netcat

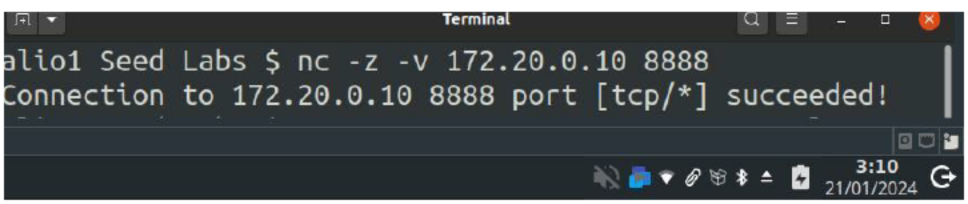
- A container was created using the ubuntu:20.04 image shown in the screenshot below. It installs netcat and runs the server using the nc –l command and runs on port 8080 with the addition of the –p 8080.



- Adding this container to the seednetwork was accomplished by utilizing the YAML shown in the screenshot above. Networks: seednetwork: ipv4_address gives it a static IP address. The second networks:seednetwork:external:true confirms it is an external network.
- To confirm this, I used the command: docker inspect 12462cf44359 | grep Address

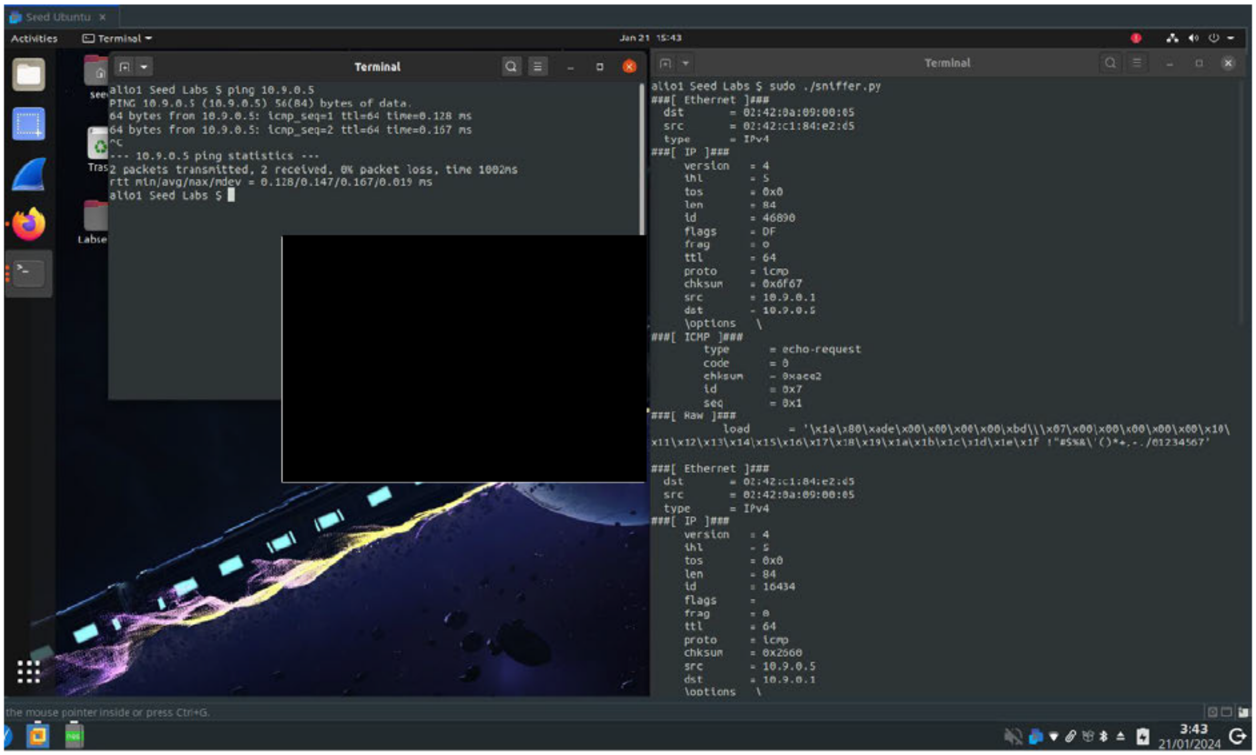


- To demonstrate the usage of nc commands I first tested reaching the nc server using nc –z –v. The –z tests connection without connecting and –v outputs verbosly.



Part 2: Packet Sniffing and Spoofing

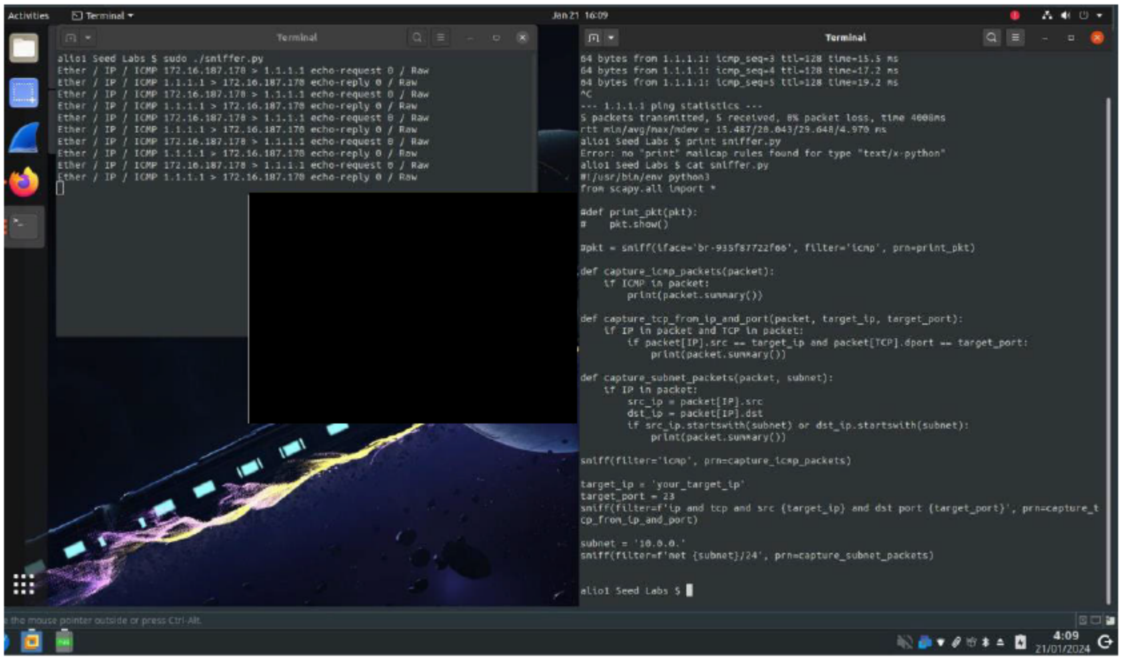
- Task 1.1: Sniffing packets
  - 1.1A: This task was accomplished with the python script provided in the book; the only modification was the interface name. See the screenshot below for ICMP / ping examples on the right terminal



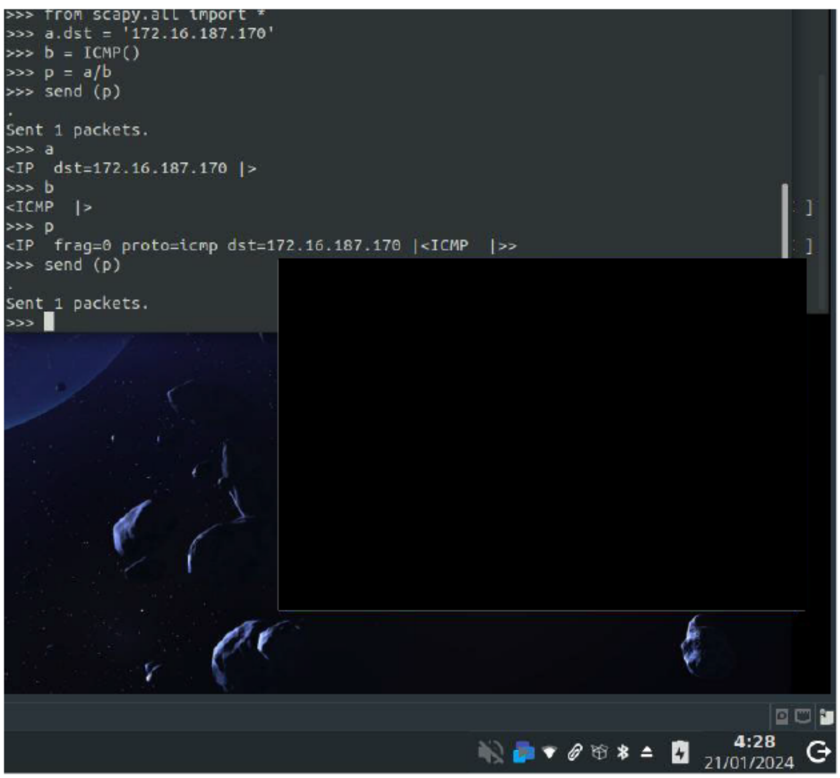
The second screenshot shows the output of running the script without sudo. The errors that were raised occur because tools like WireShark and our packet sniffer utilize sockets which require superuser privileges/permission

```
alio1 Seed Labs $ ./sniffer.py
Traceback (most recent call last):
  File "./sniffer.py", line 7, in <module>
    pkt = sniff(iface='br-935f87722f66', filter='icmp', prn=print_pkt)
  File "/usr/local/lib/python3.8/dist-packages/scapy/sendrecv.py", line 1036, in sniff
    sniffer._run(*args, **kwargs)
  File "/usr/local/lib/python3.8/dist-packages/scapy/sendrecv.py", line 906, in _run
    sniff_sockets[L2socket(type=ETH_P_ALL, iface=iface,
  File "/usr/local/lib/python3.8/dist-packages/scapy/arch/linux.py", line 398, in __init
__
    self.ins = socket.socket(socket.AF_PACKET, socket.SOCK_RAW, socket.htons(type))  # n
oqa: E501
  File "/usr/lib/python3.8/socket.py", line 231, in __init__
    _socket.socket.__init__(self, family, type, proto, fileno)
PermissionError: [Errno 1] Operation not permitted
alio1 Seed Labs $
```

- 1.1B: For this task, I modified the code to create three functions, one for each filter. Therefore, the code in the subsequent screenshot iterates through the packets, and sets key variables like target_ip and target_port for the second filter to meet the criteria and print using the built-in packet.summary() feature.

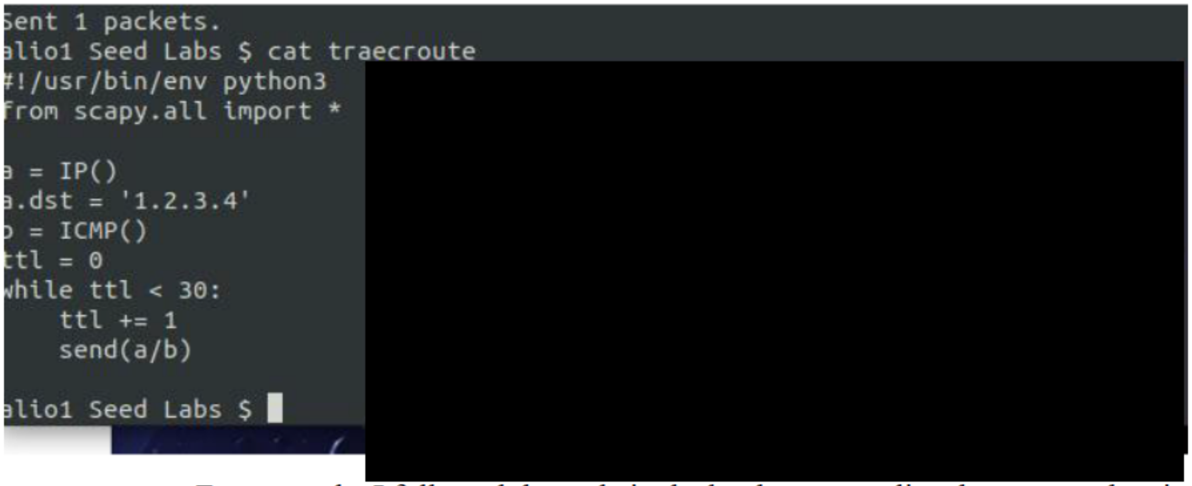


- Task 1.2: Spoofing ICMP packets
    - For this task, we sent a packet using Scapy. The only modifications needed to the code were adding the IP address. The screenshot below shows the successful sending of the ICMP request. As this is only one request, it is not captured by Wireshark, there would need to be more packets send to register

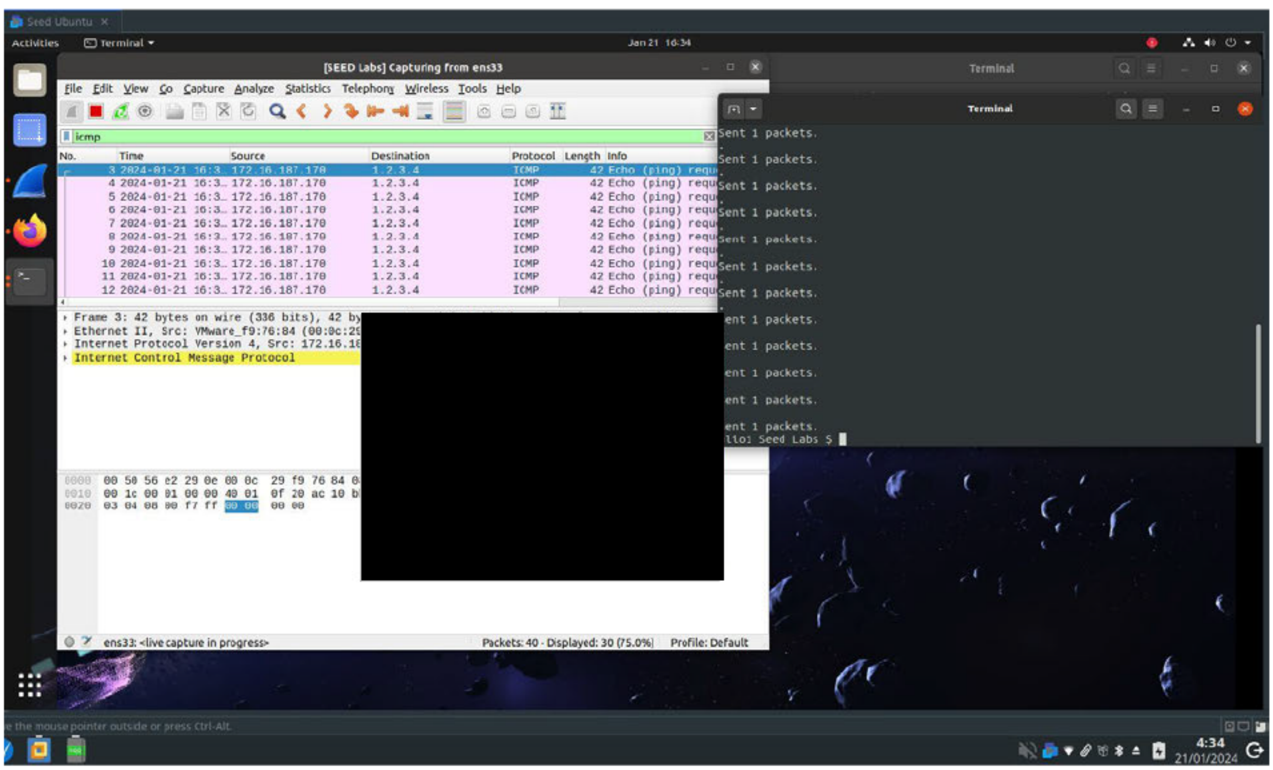


- Task 1.3: Traceroute

- For this step, I created a program to automate the process so that I did not have to increment TTL. I chose the typical default value for TTL in traceroute (being 30). In the first screenshot, I show the code, the second featured the output of *sudo ./traceroute*



- For my code, I followed the code in the book as an outline, however, when it came to setting TTL, I set it to zero and the run a while look that would iterate through TTL0 – TTL 29 to ensure that the traceroute would be seen by WireShark.



- Task 1.4: Sniffing and-then Spoofing

- To start this section, I had to research and found that it was a bit outside the scope of my abilities. So, I researched the question and found a great resource that goes through the problem. I have included the link here.
- What the code must do is identify ARP packets (which I believe the reason for pkt[#].type == 8) to start the Request and Reply. The subsequent section of code sets parameters to facilitate the REQUEST and REPLY section. By this, we set a source and destination, as well as ID, Sequence, and Load to fill the packet with necessary information. We can see the code working through print strings that show which step it is on.
    - We can make modifications to this code to return if it detects packets other than ICMP to expedite the process.