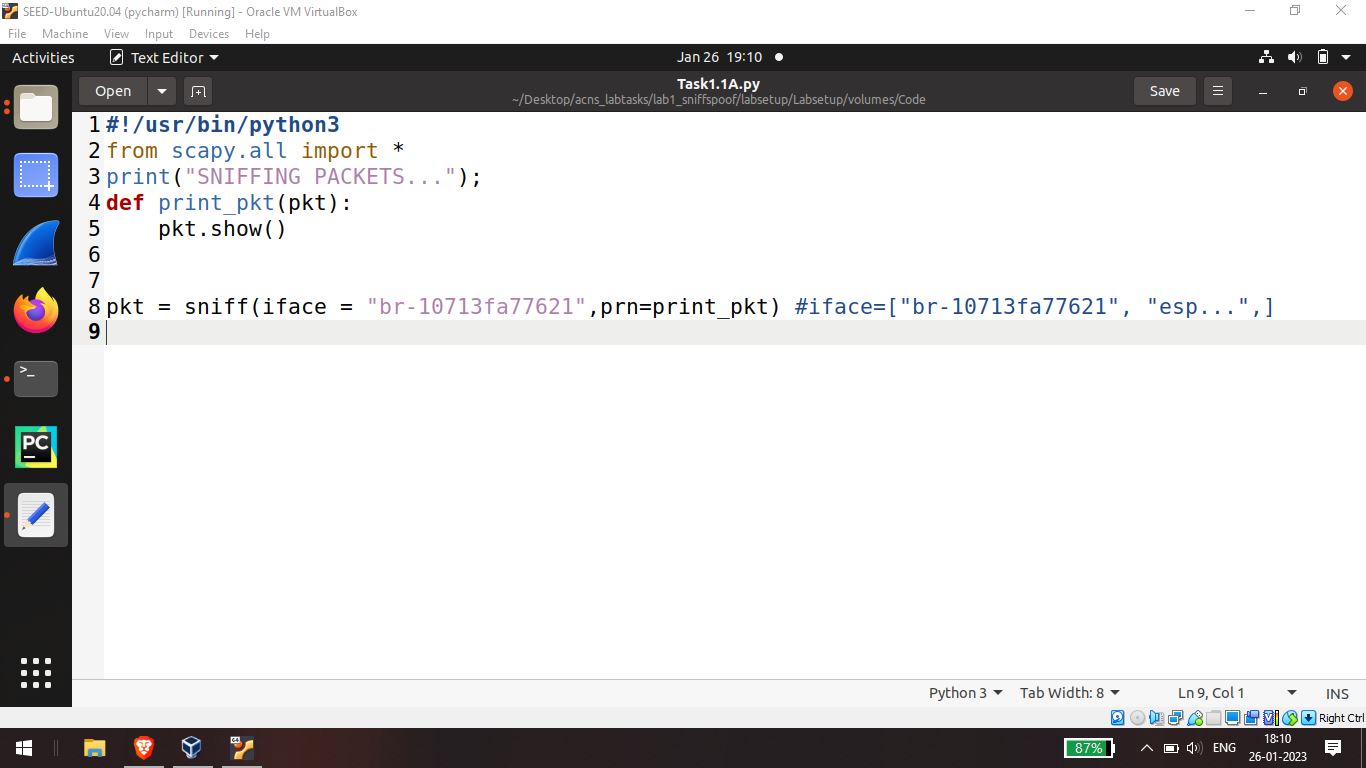
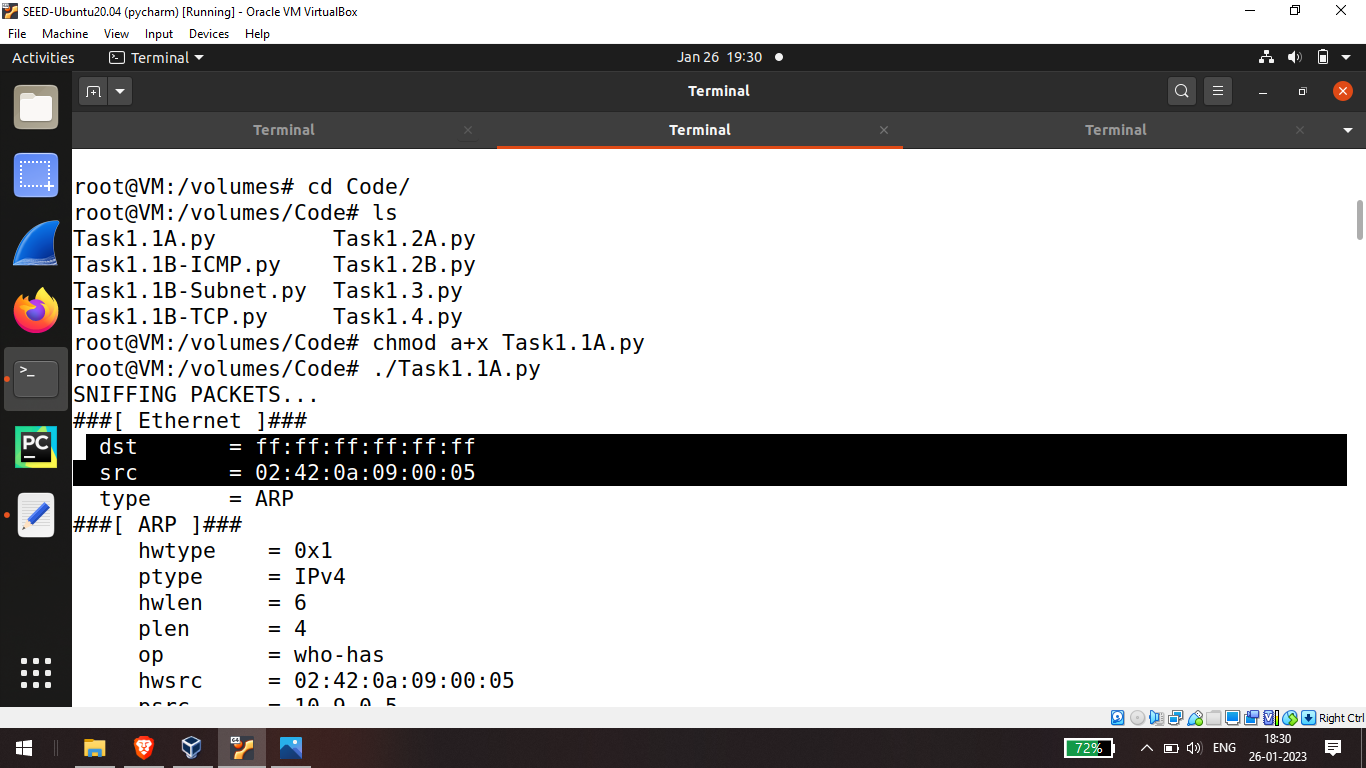
Task 1.1a

Sniffing packets:

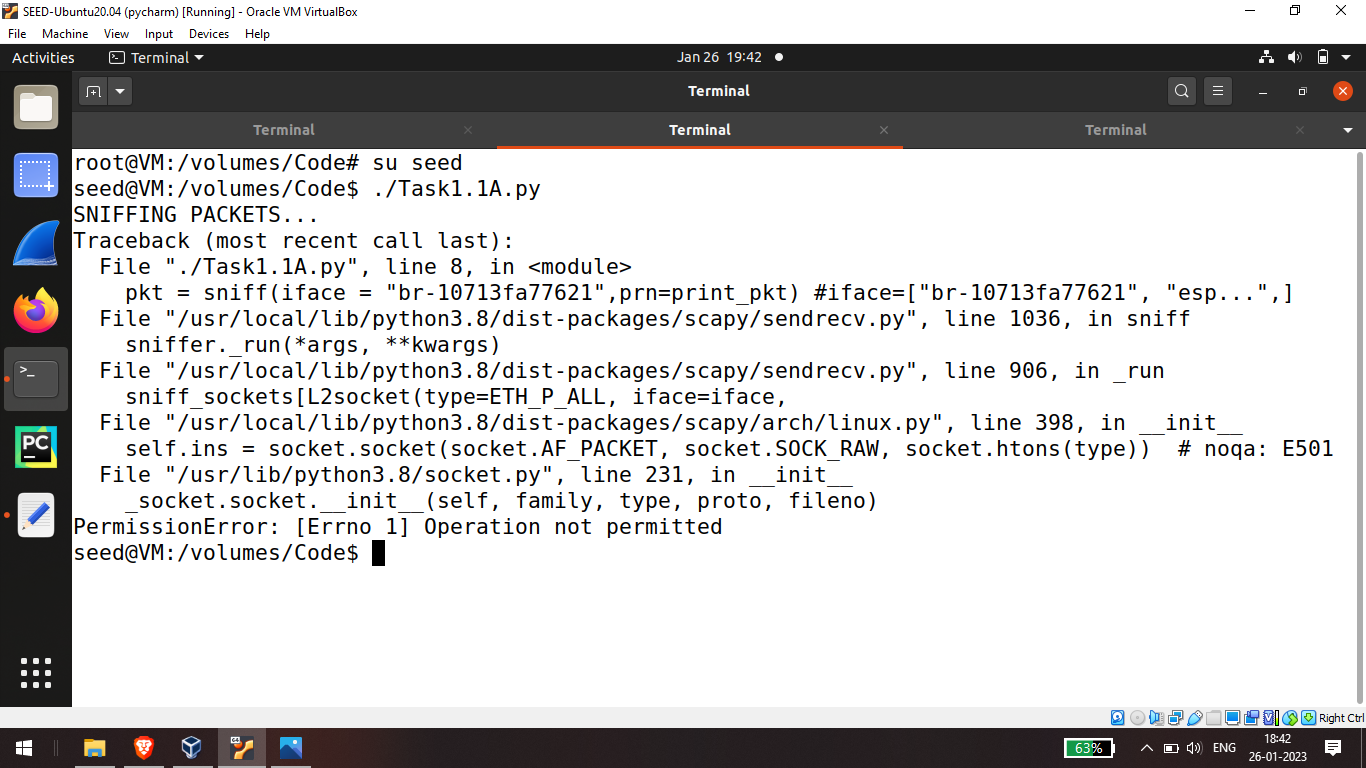


Task 1.1

captured now here the destination which has all f's indicate a broadcasted packet src contains the source mac address you may find all the packets in the network corresponding to arp ICMP



let's try this without root privilege



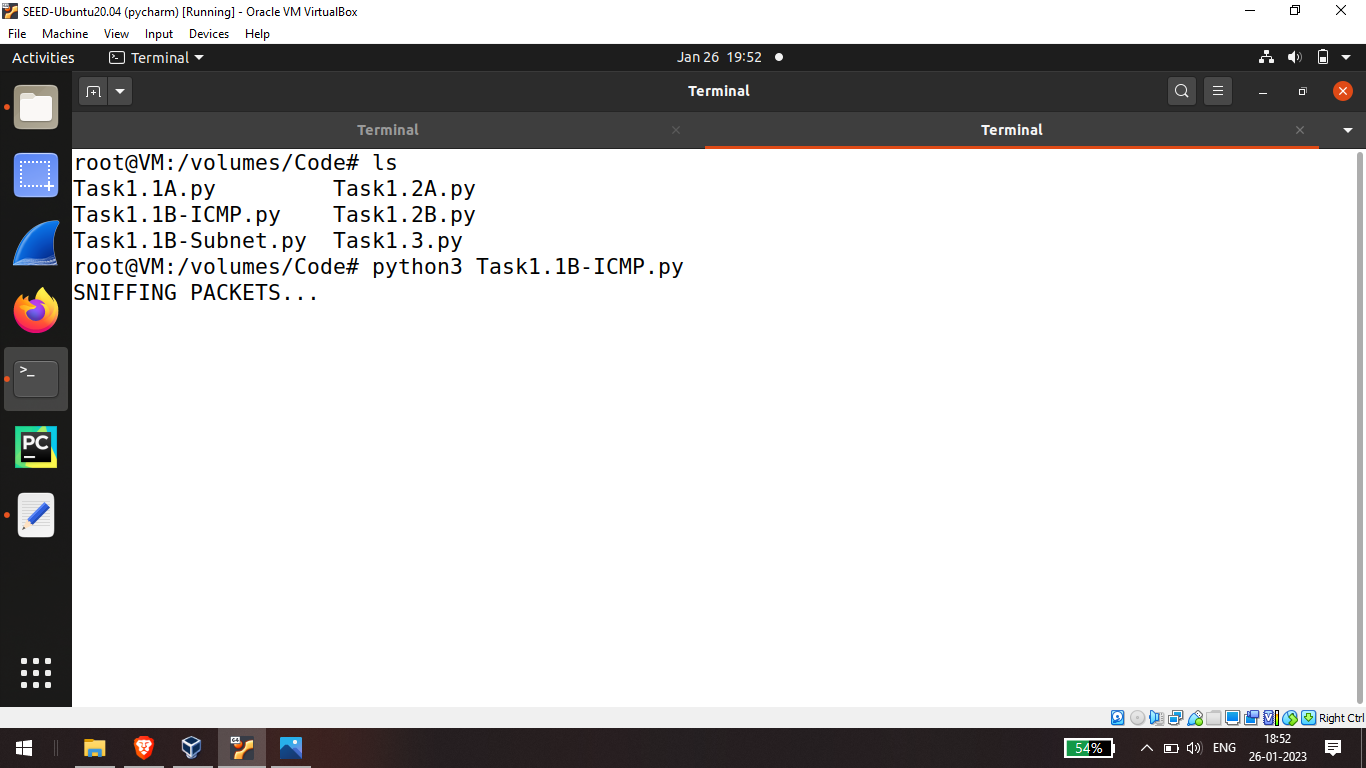
you will find the roots change to seed user now try executing it yes you get an error the line sniffing packets indicate the program is running but the sniffed method fails due to a permission error it says operation not

permitted so we require the root privilege to sniff all the packets without root privilege we will fail to sniff the packets

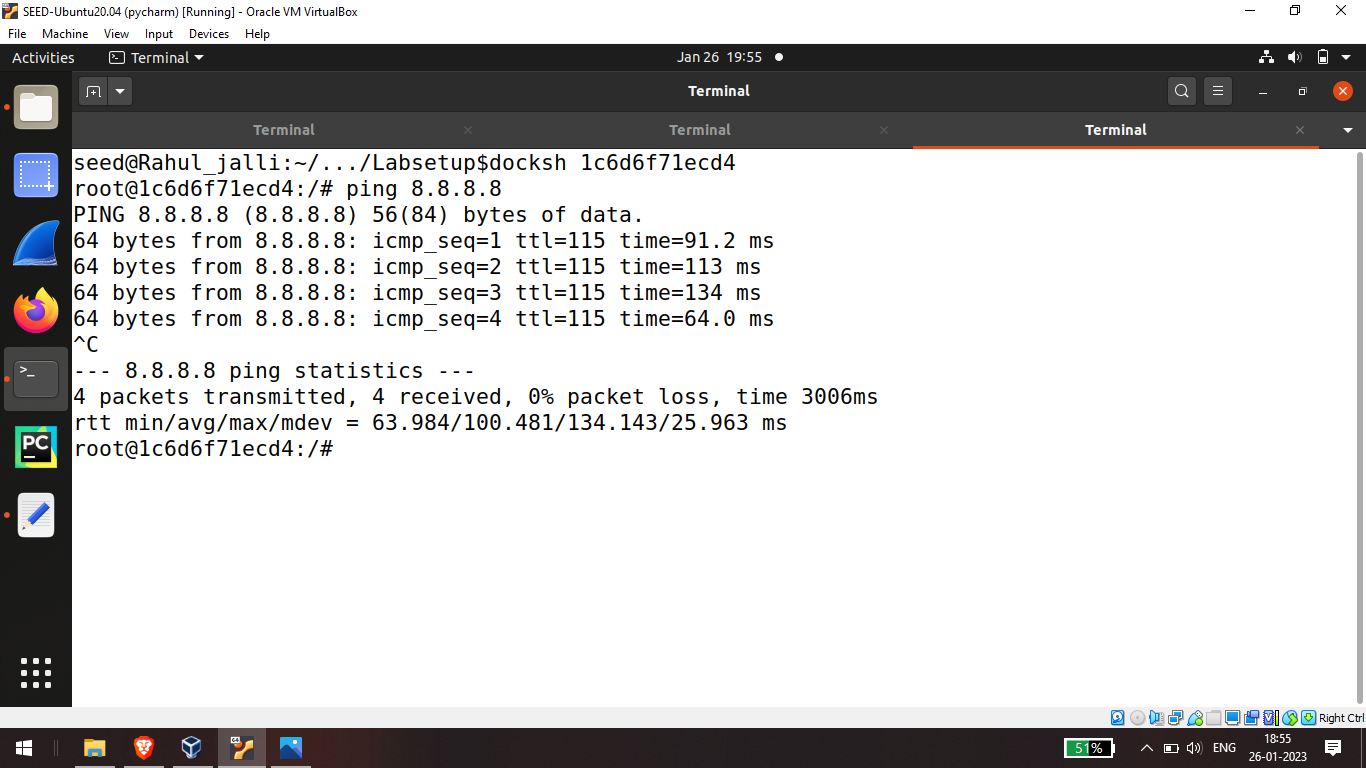
Task 1.1b

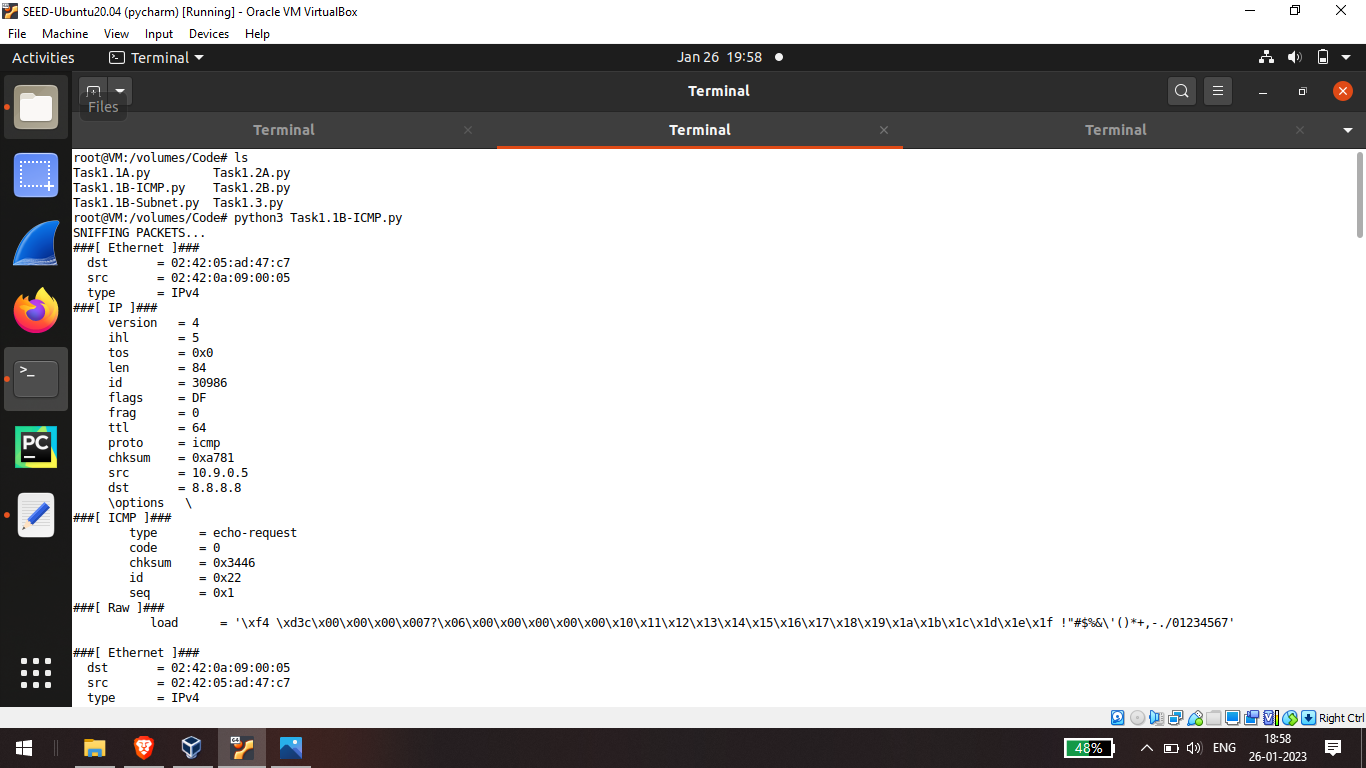


here the line filter icmp means to sniff only the icmp ip packets



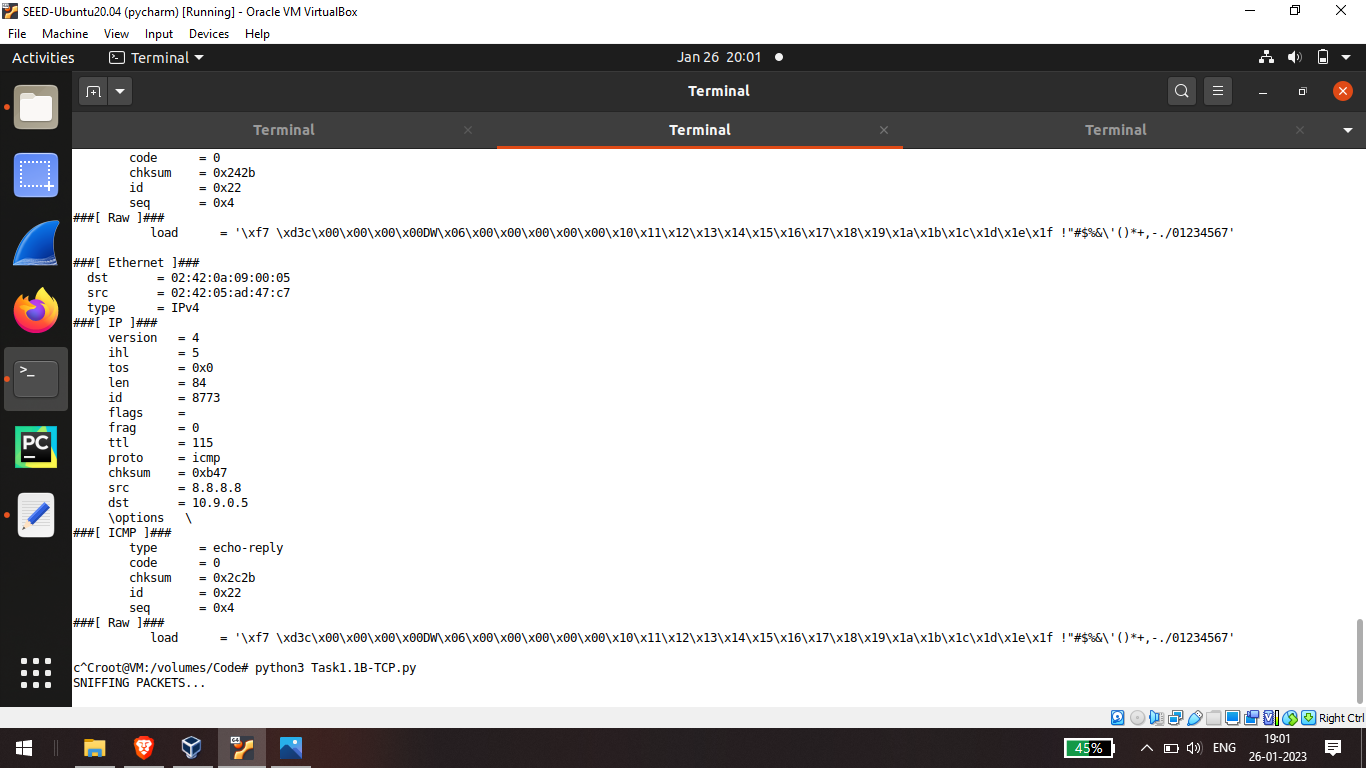
in the attacker terminal we will see only ip and icmp packets are captured while previously we had all the packets now we have the filtered packets

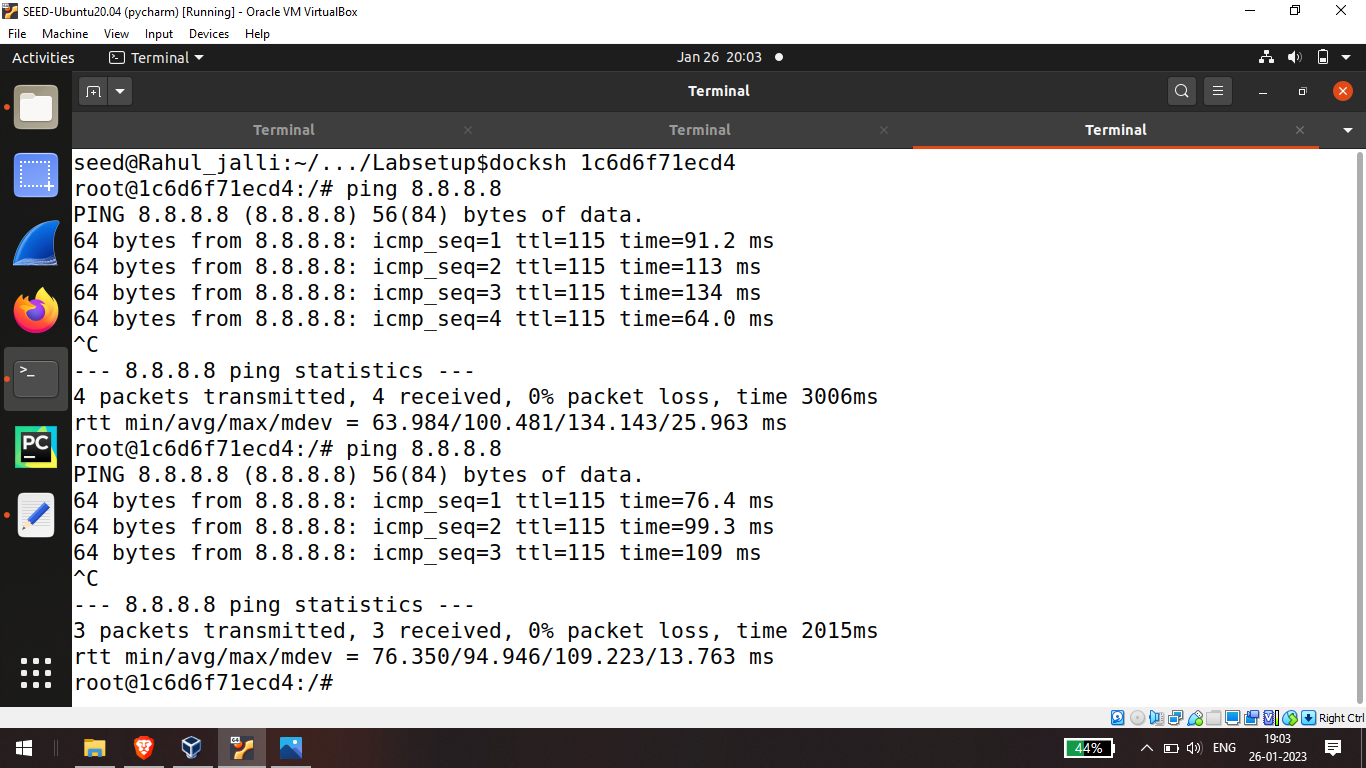




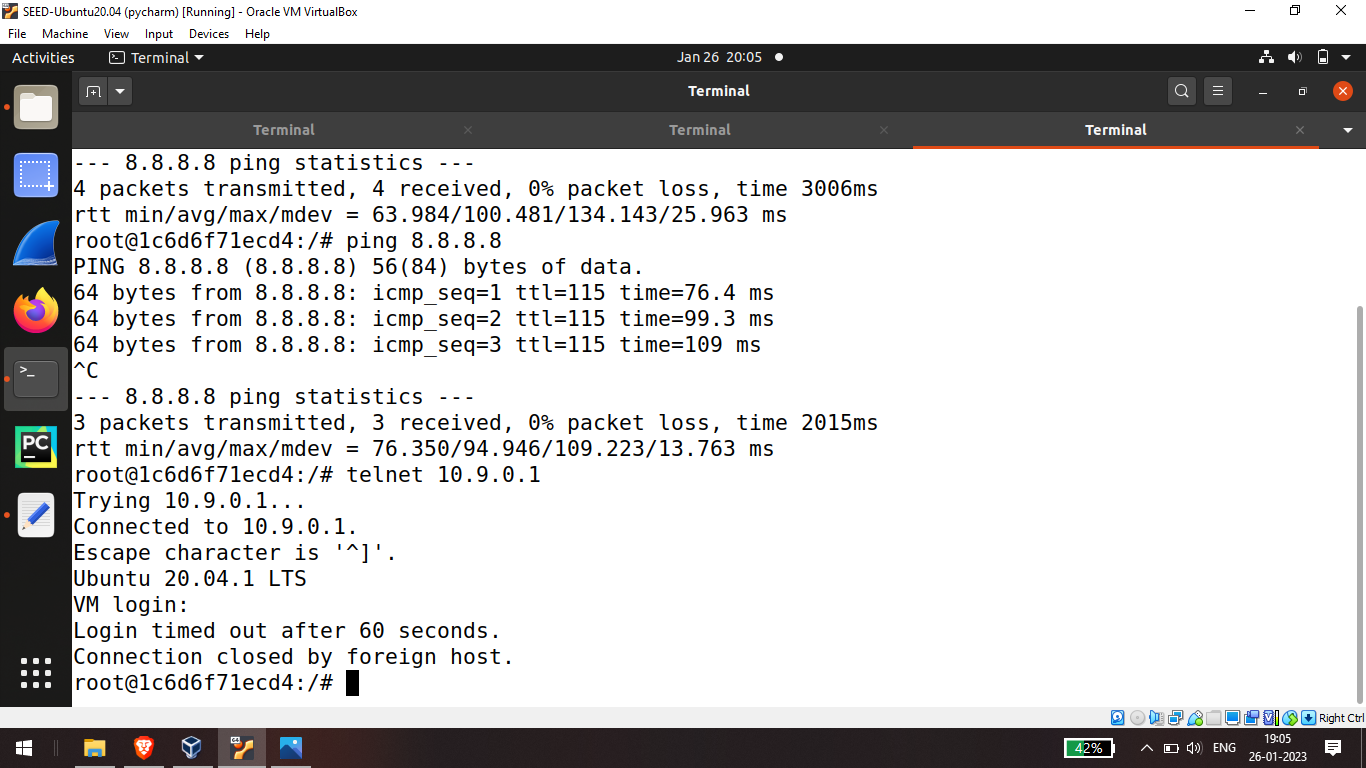
Task 1.1B tcp filter

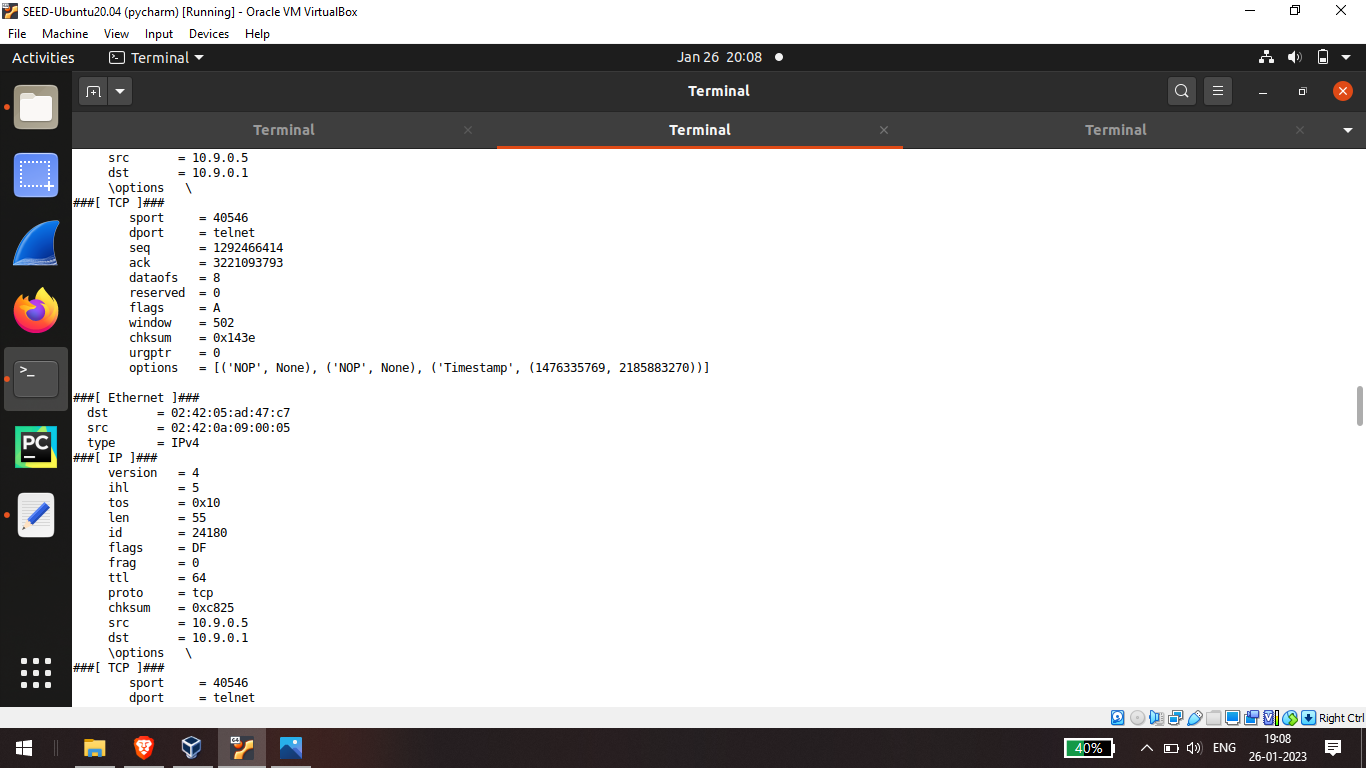
Now let's try to capture tcp but nothing will capture if we ping 8.8.8.8 as there are no tcp packets





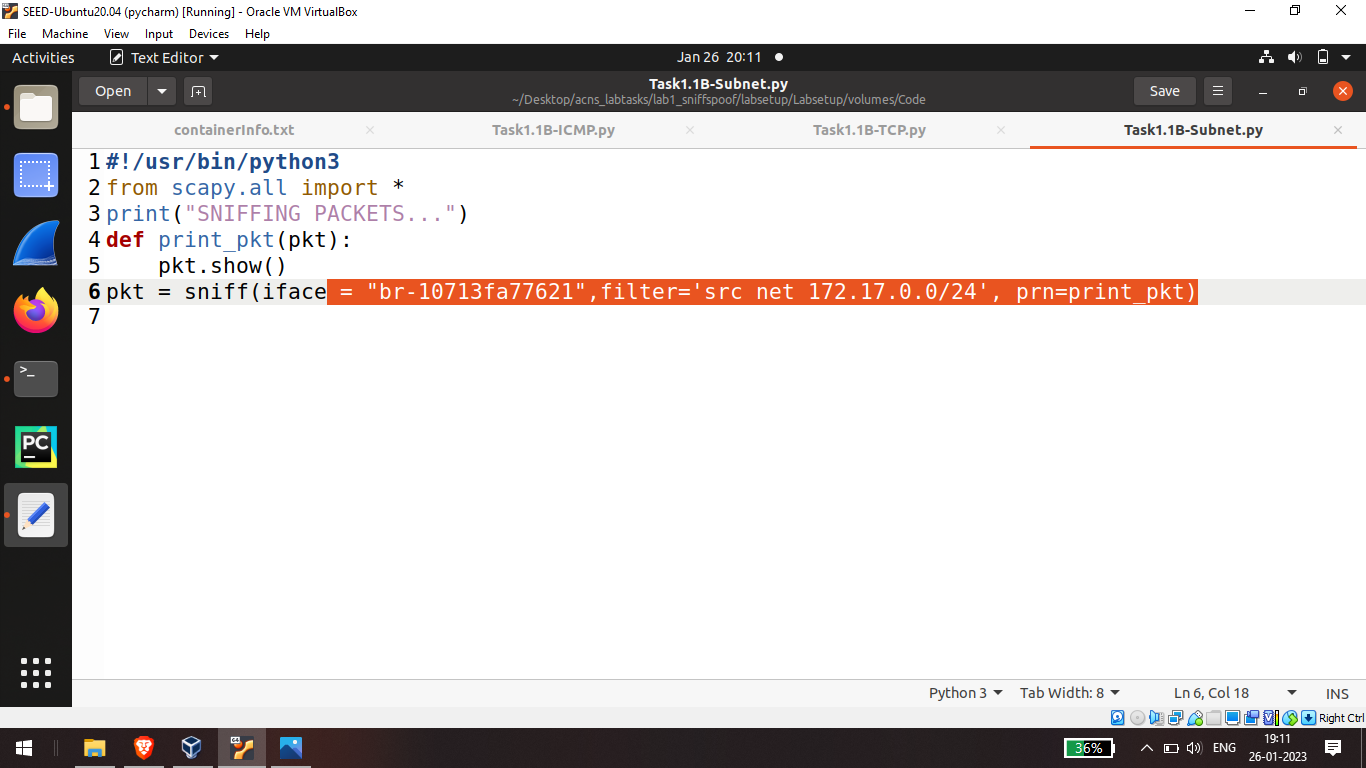
this is to open a tcp request to the attacker now you will see these tcp packets are captured in the trackers terminal and the depot telnet means port 23 this means simply capture the tcp packets

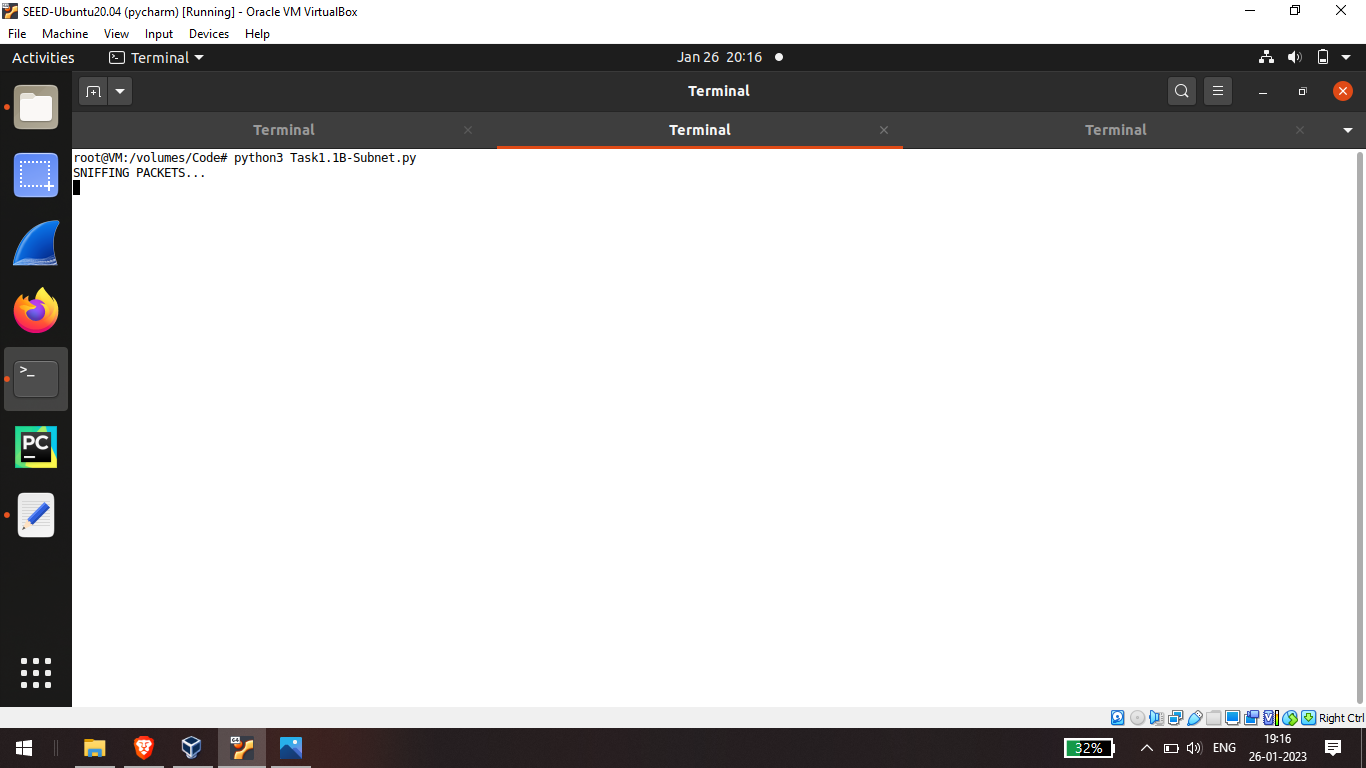




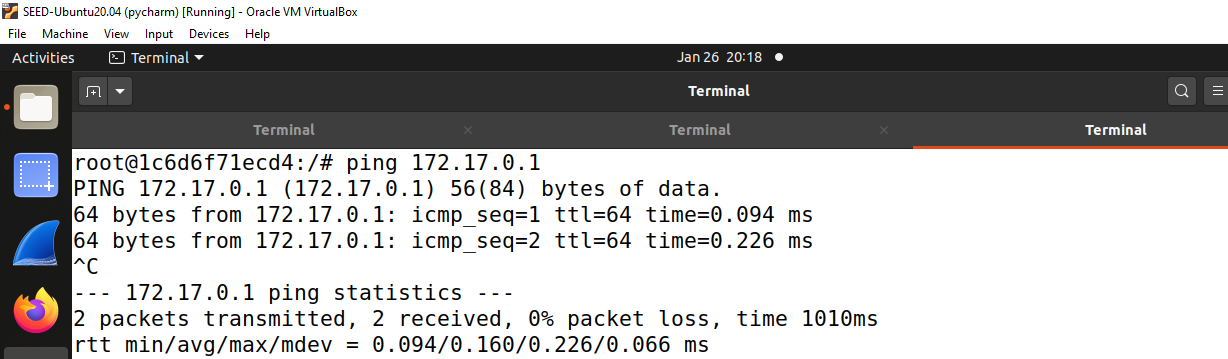
Task 1.1 subnet:

the filter is set to source IP which starts from 172.17.0.0/24 so to find how to get this go to the attackers terminal and type ifconfig here in docker 0 you will find this inet to be 172.17.0.

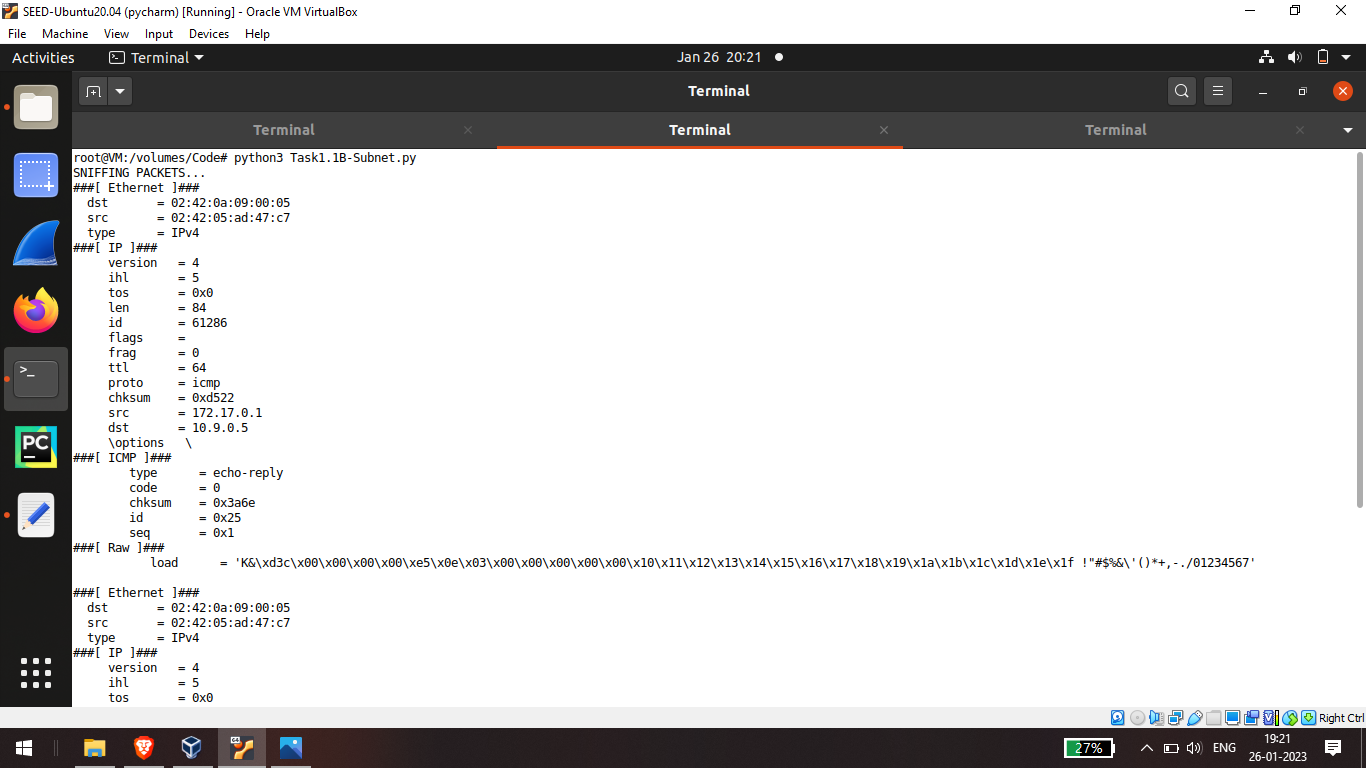




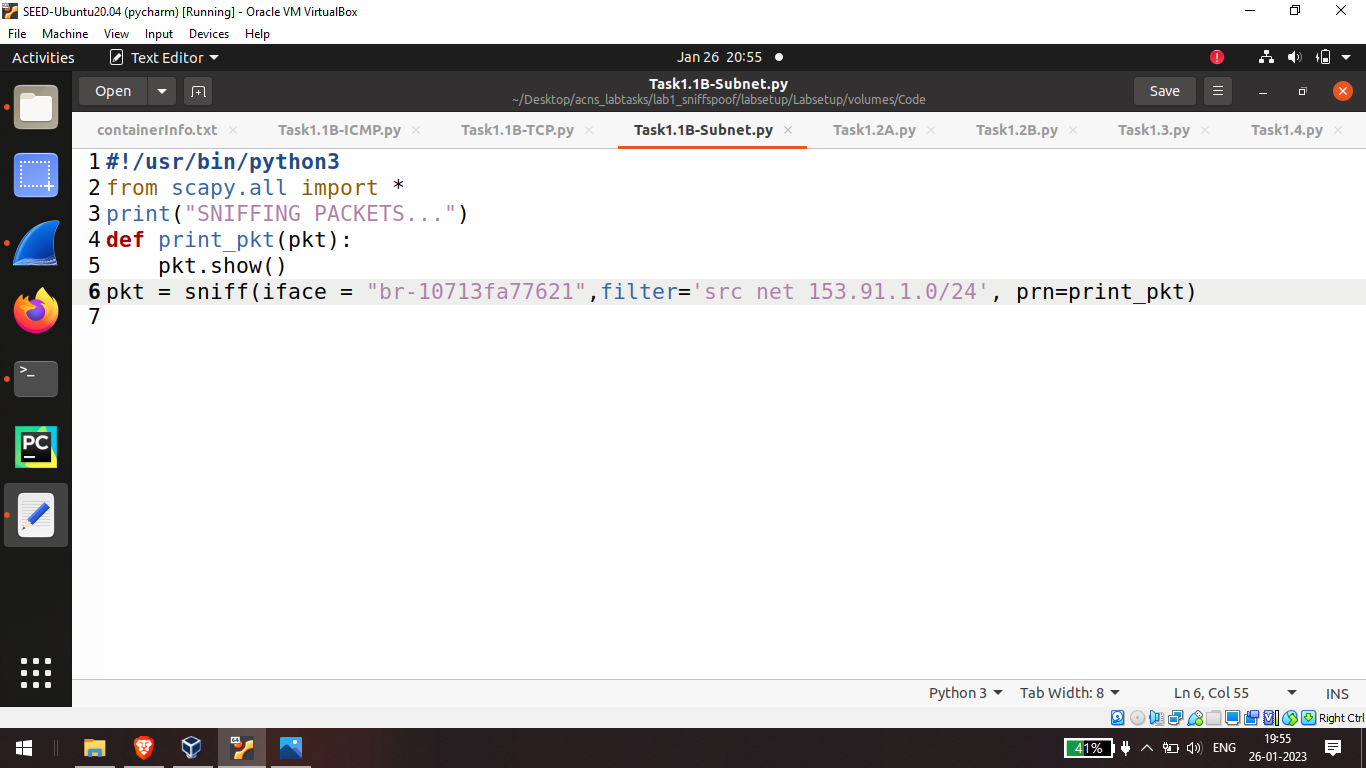
And lets ping 172.17.0.1 for packets

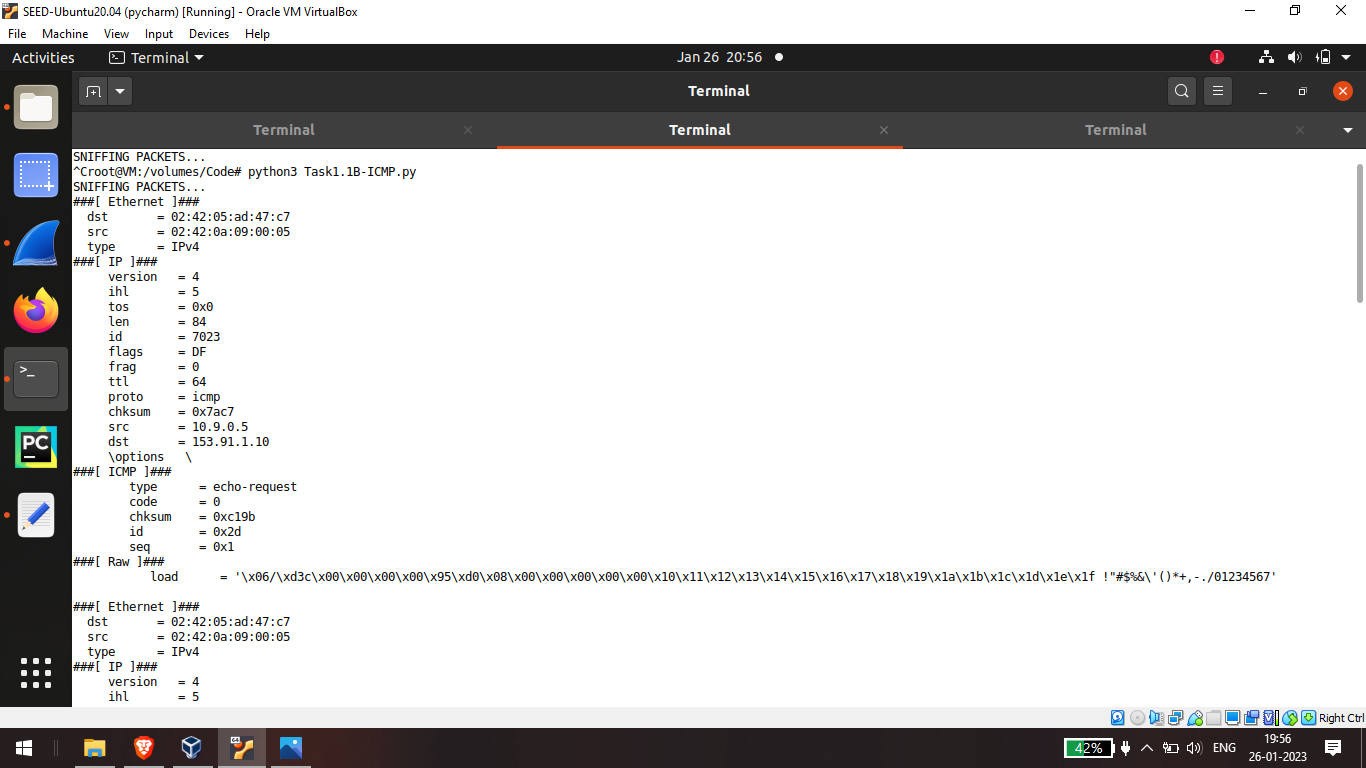


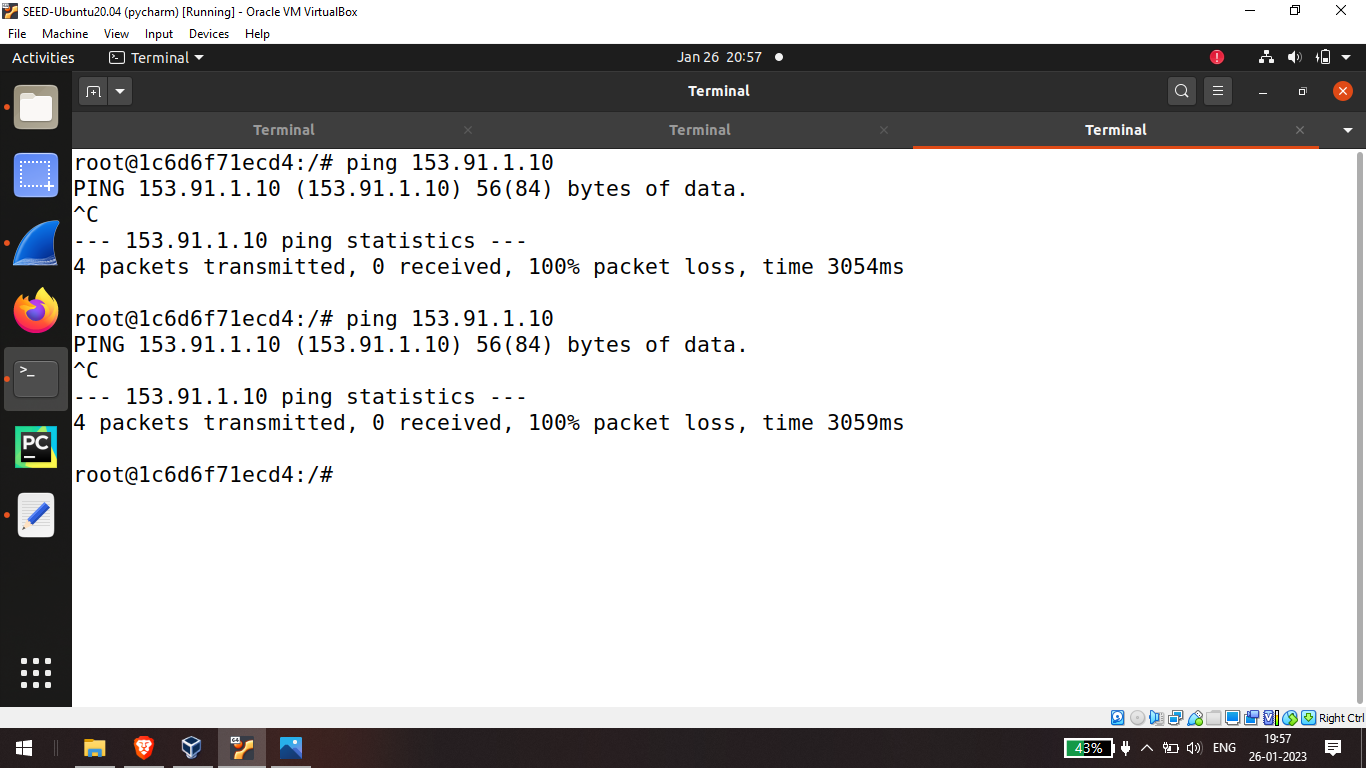
And the packets are captured



Task 1.1 Tcp: UCM server

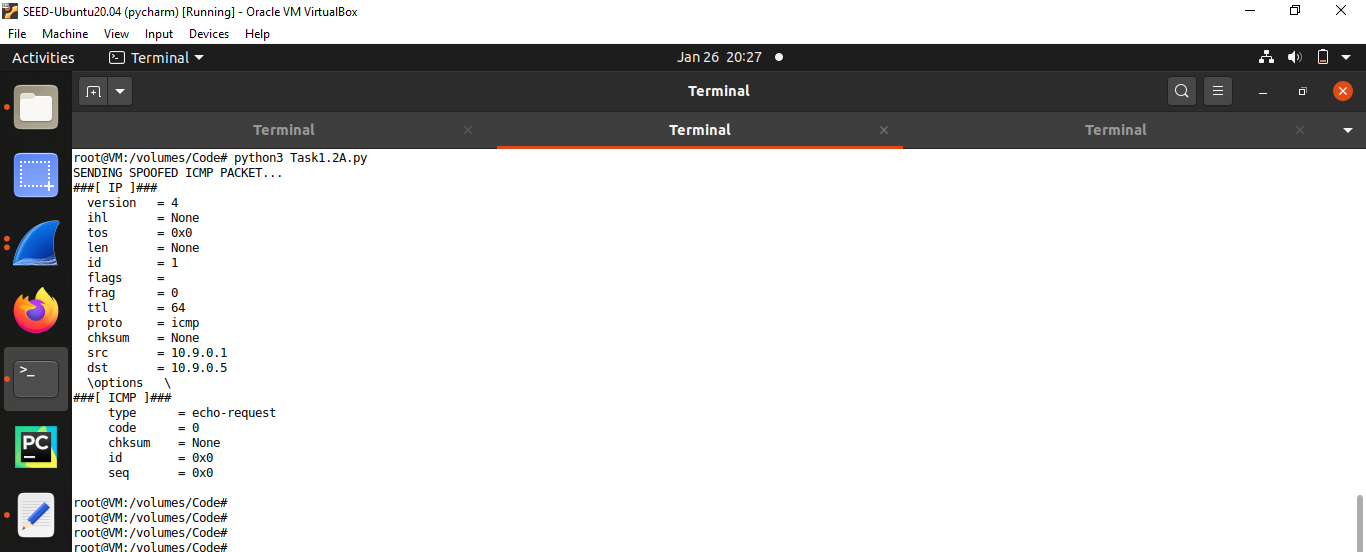






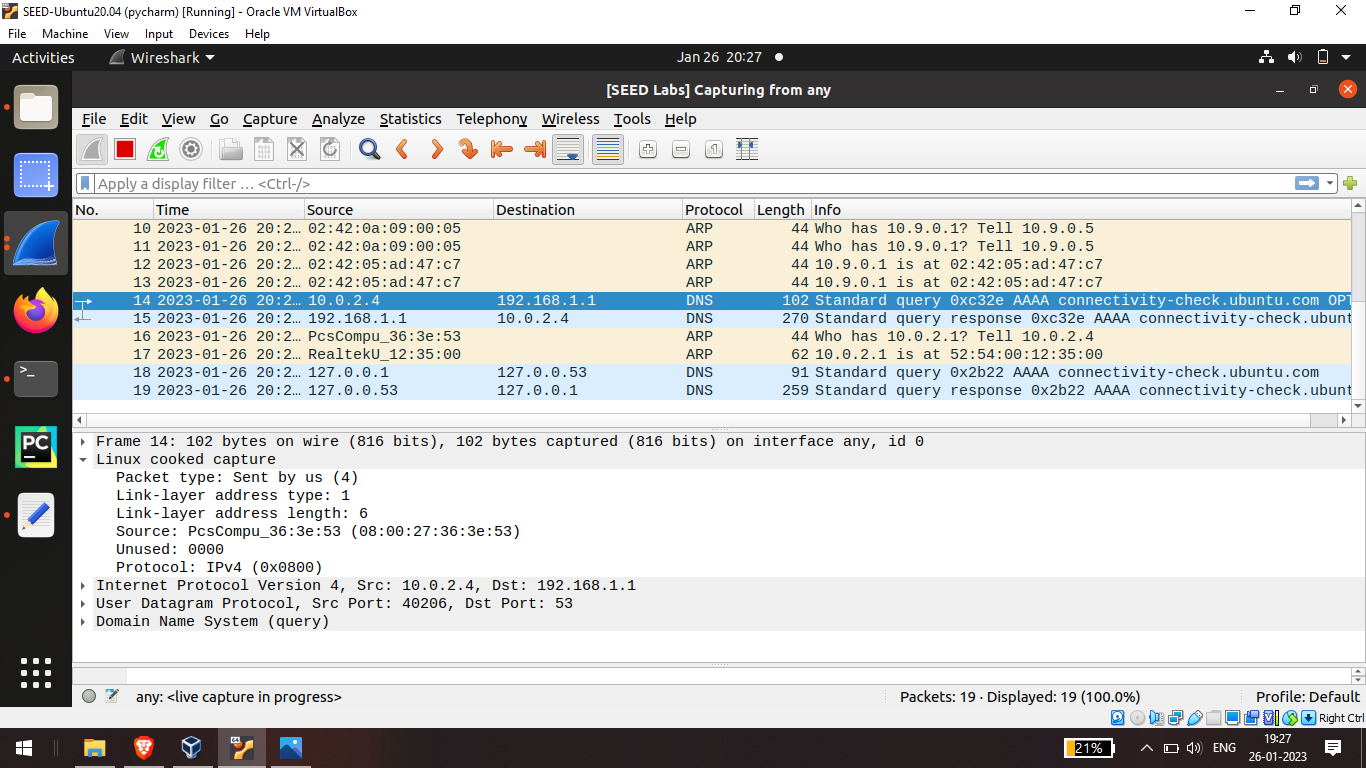
Task 1.2: Spoofing ICMP Packets

ip method defines an ip layer and set the source address to attacker ip and the destination to host aip now icmp method is created and packet is constructed the division operator is overloaded by scapi library that adds icmp as a payload of ip packet dot show displays the data the send method sends the packets in the network



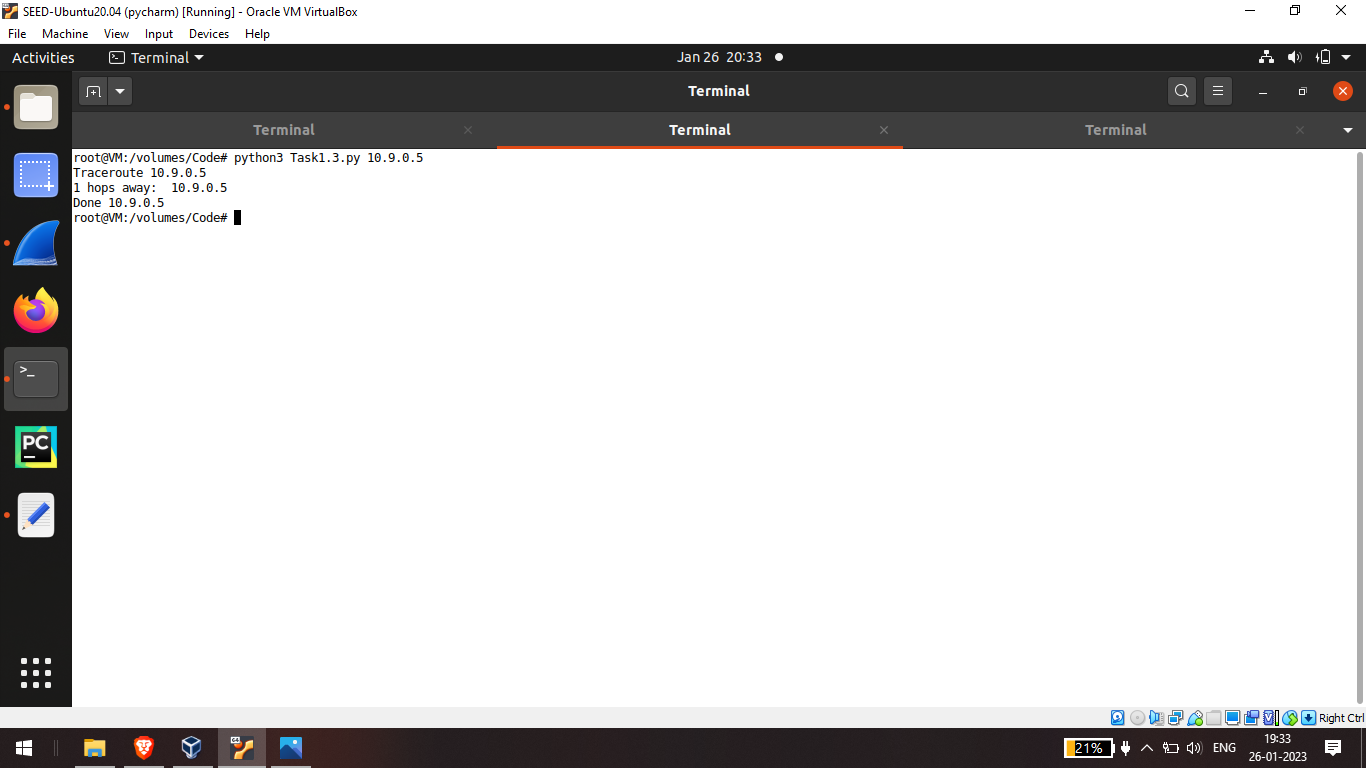
The captured packets can be seen in wire shark

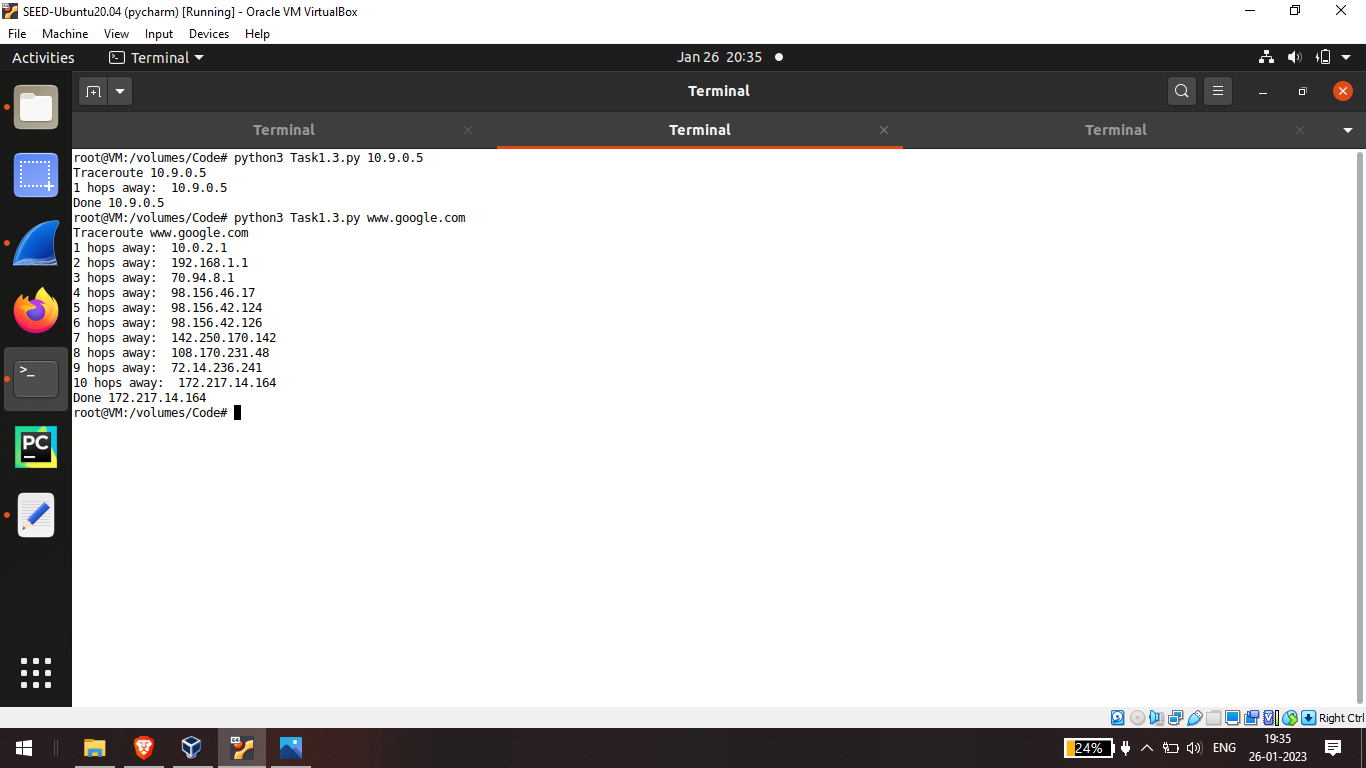
so this program shows how to send packets manually in the network traffic



Task 1.3: Traceroute

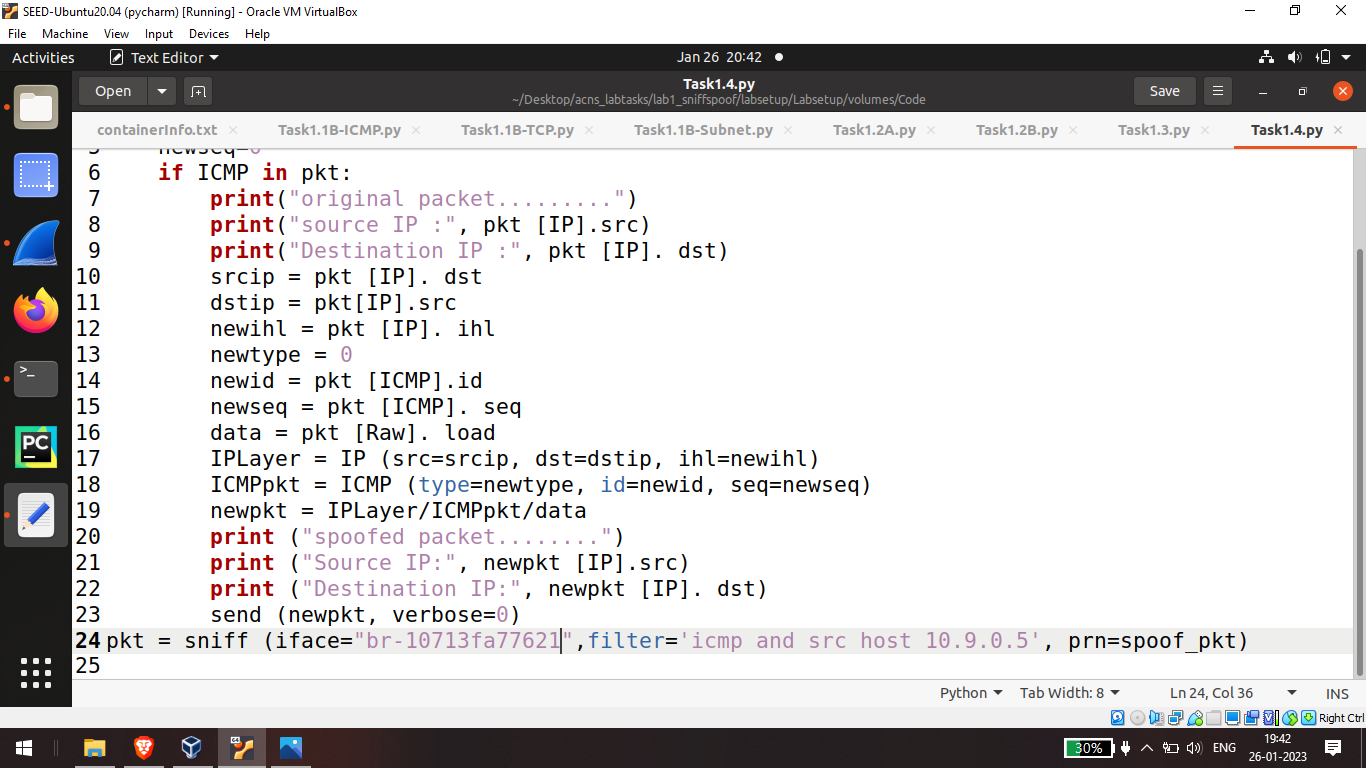
traceroute implementation using scapi it takes hostname or ip address as an input argument ttl is time to leave factor this is used to measure the distance between source and the destination

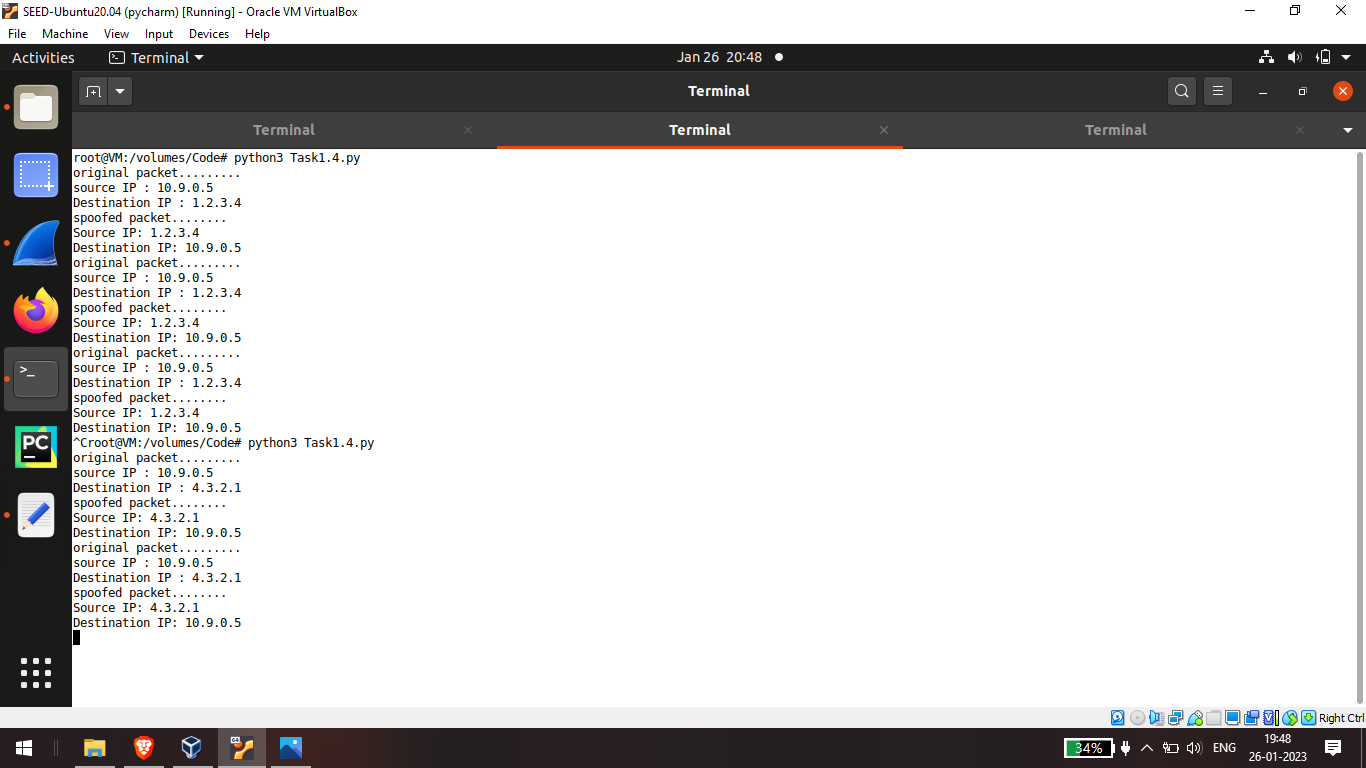


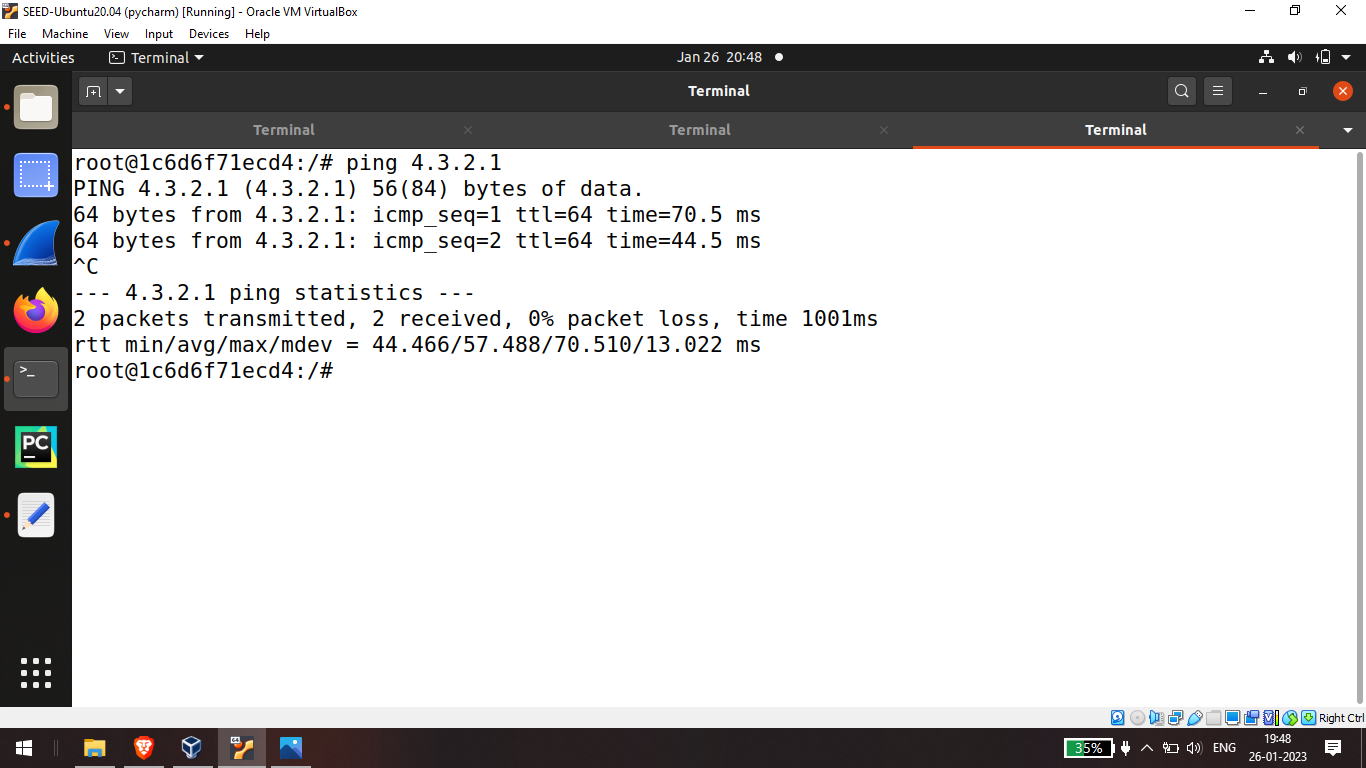


Task 1.4: Sniffing and-then Spoofing

spoof method that takes a sniff packet as an argument this method spokes only icmp packets we retrieve source ip and destination ip from the sniff packet and create a new ip packet the packet's destination is stored as source ip and packet source ip is stored as destination ip this is done so that we can send a reply

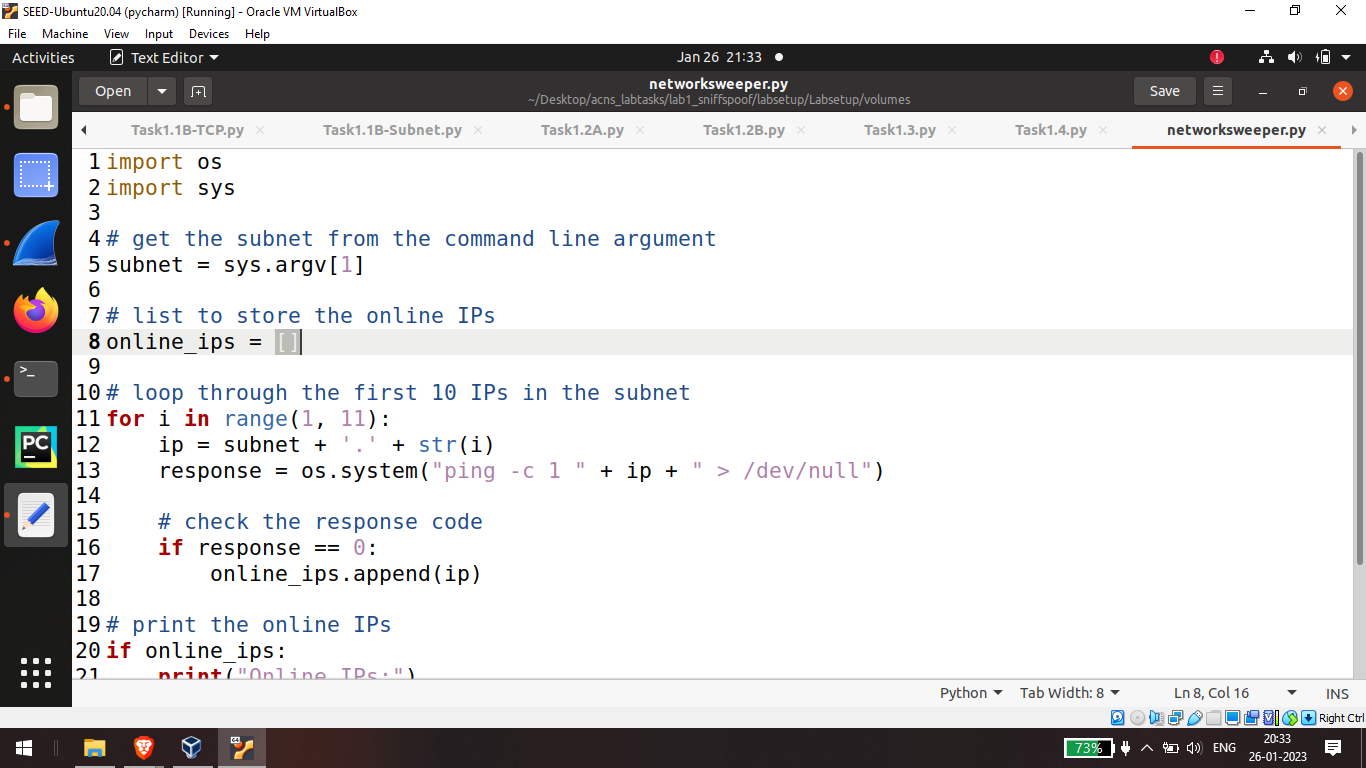




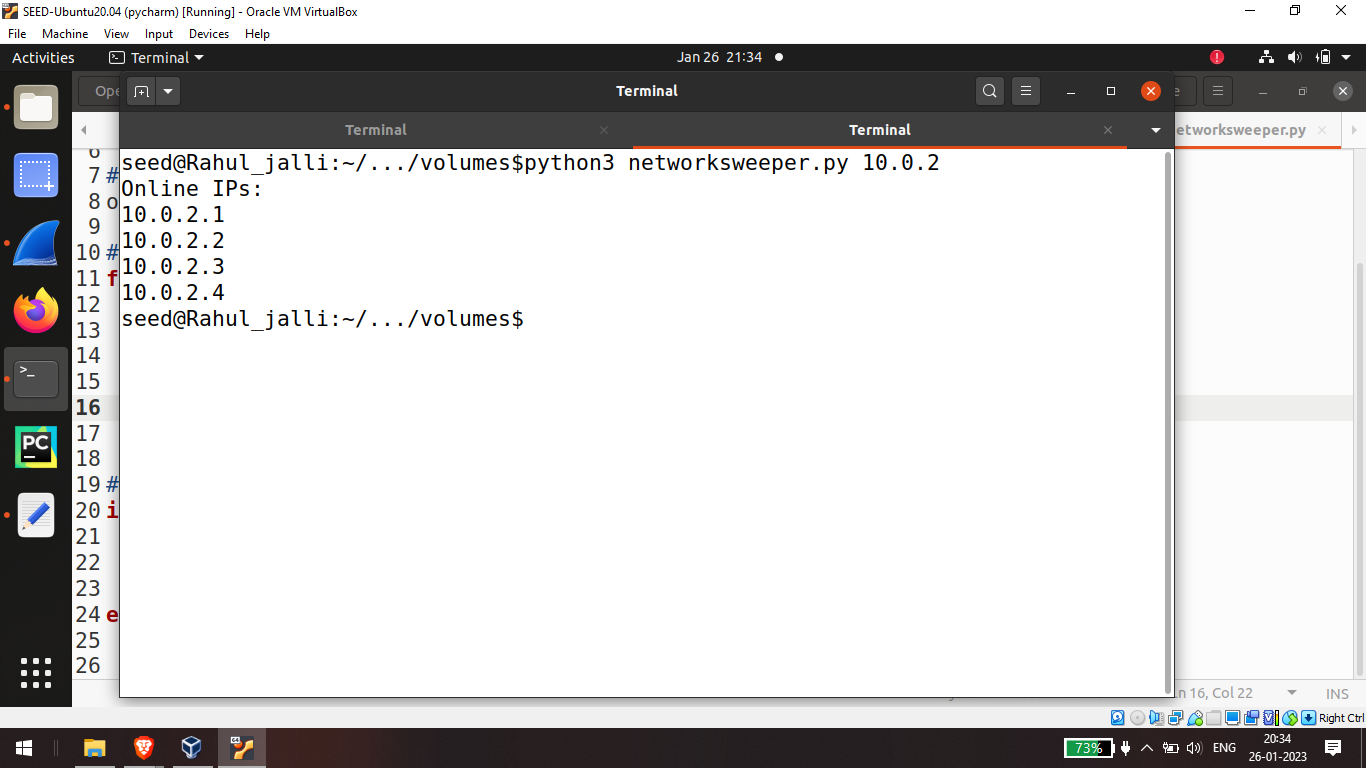


Extra task: Network Sweeper

Python program to implement a simple network sweeper to ping hosts on a given subnet.



Output



Extra task2: multiport scanner

This script will ask the user to enter the IP address and two ports which define the port range the script will scan

