COMPUTER NETWORK SECURITY

LAB-9

VPN Tunneling

NAME: VISHWAS M

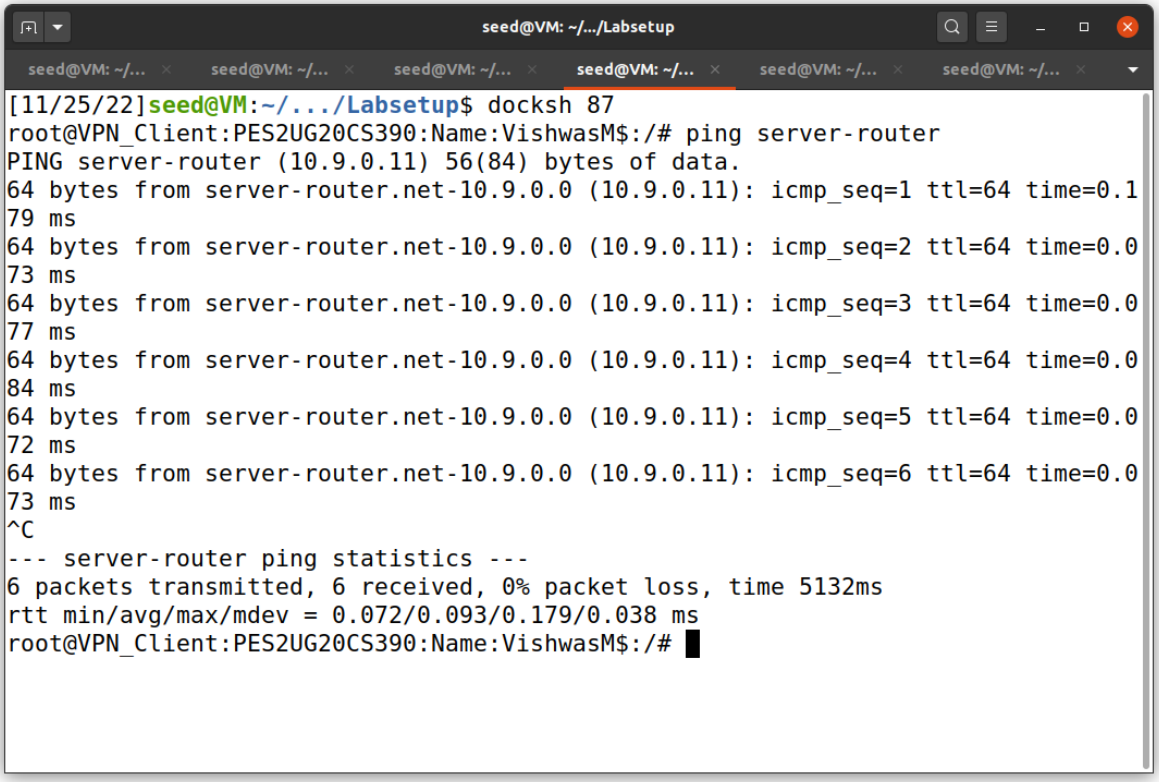
SRN: PES2UG20CS390

SEC: F

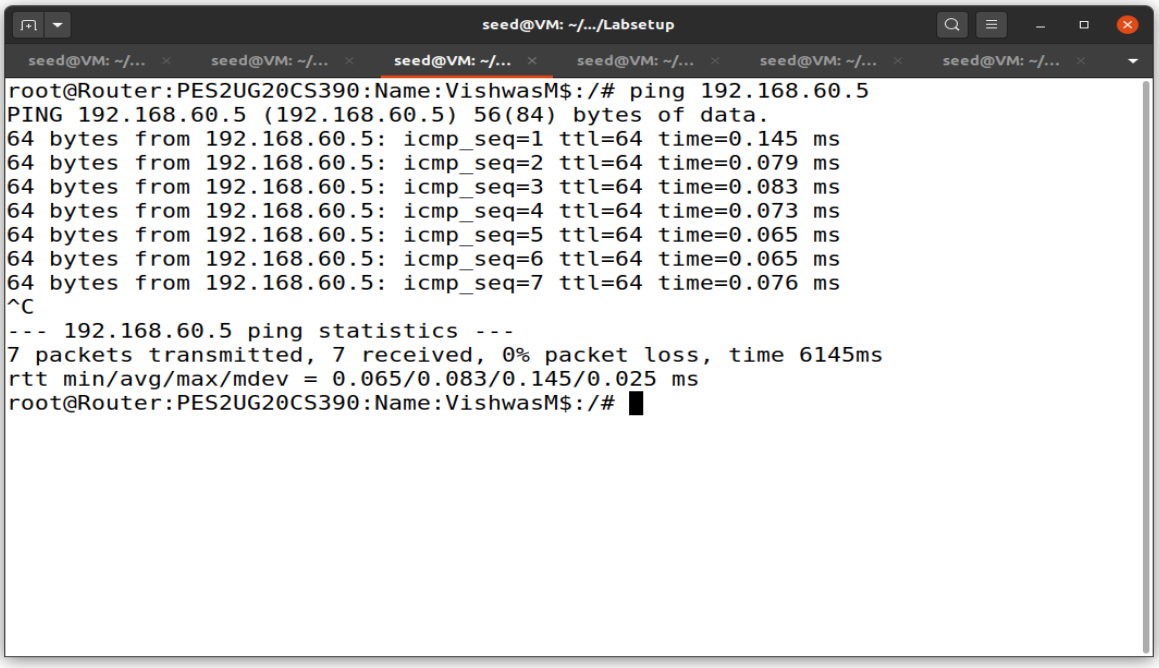
DATE:26/11/2022

Task1: Network Security

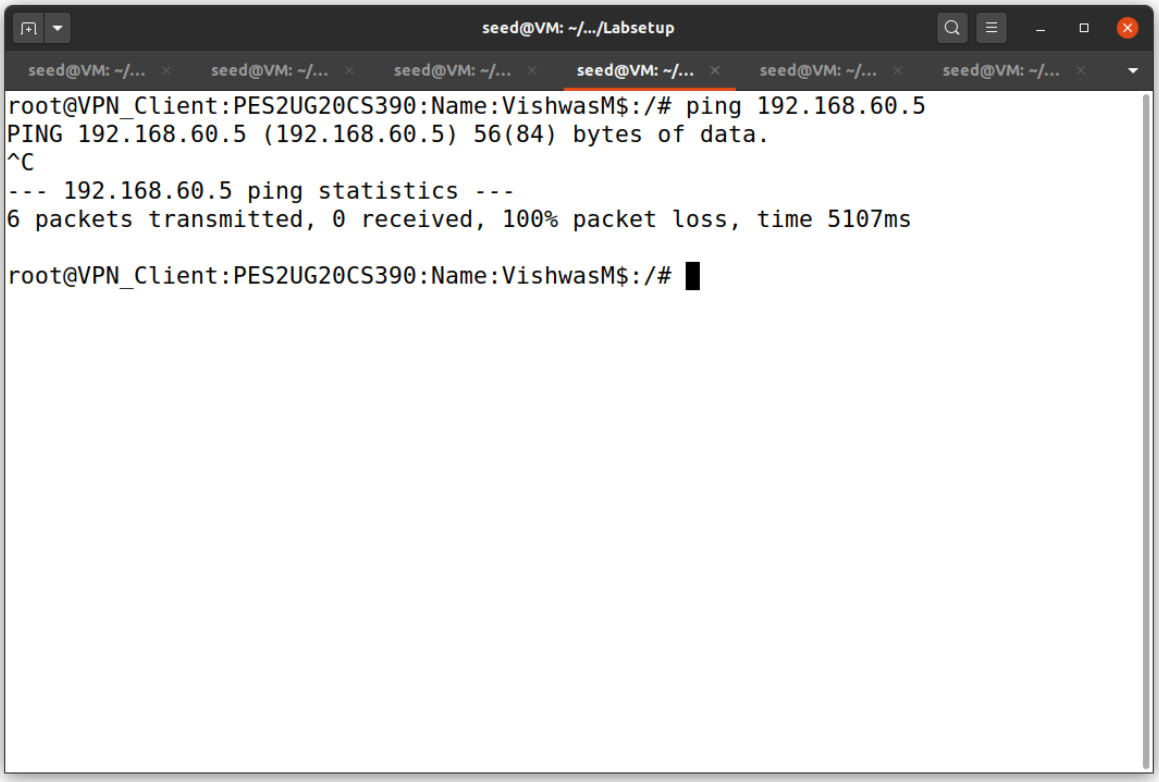
Host U - 10.9.0.5 can communicate with VPN Server (server-router)



VPN Server (server-router) can communicate with Host V (host-192.168.60.5)
On server-router

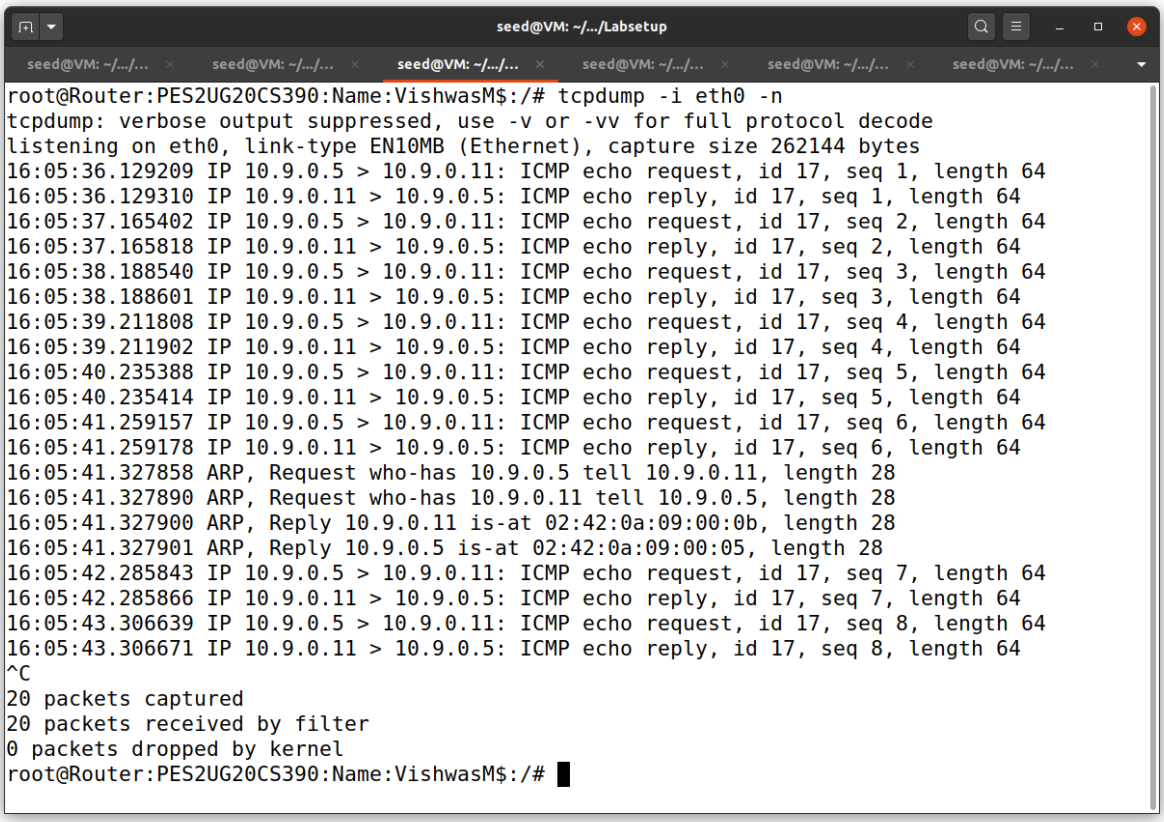


Host U (Client - 10.9.0.5) should not be able to communicate with Host V (host 192.168.60.5)
On Client 10.9.0.5

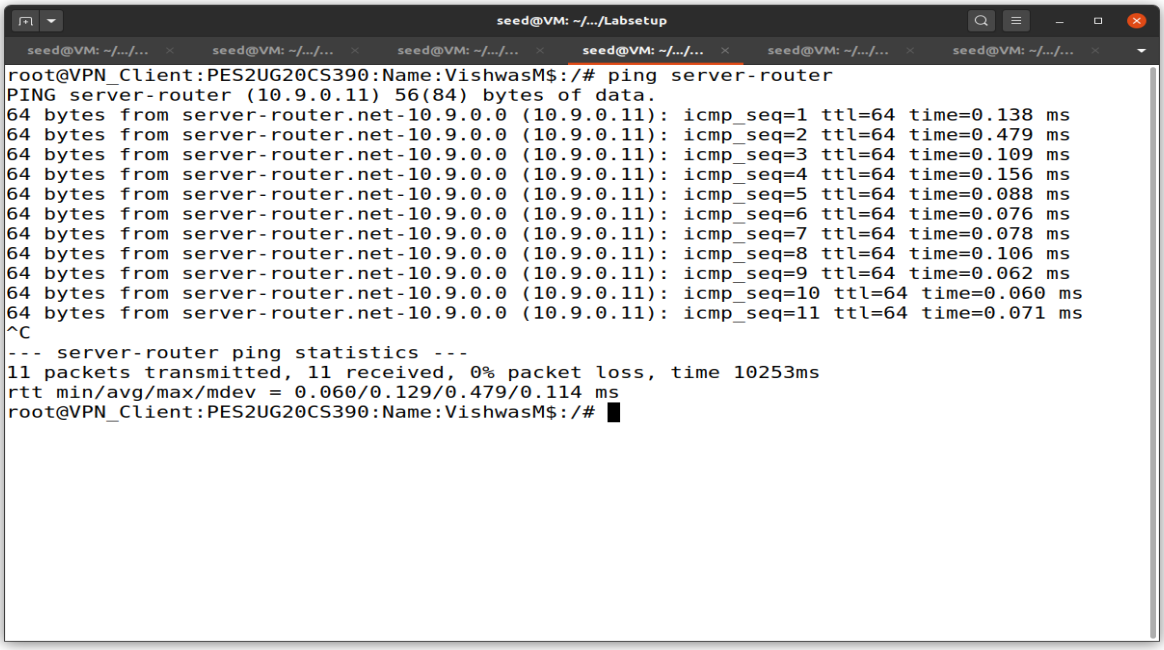


Run tcpdump on the router, and sniff the traffic on each of the networks. Show that you can capture packets.

On server-router run –

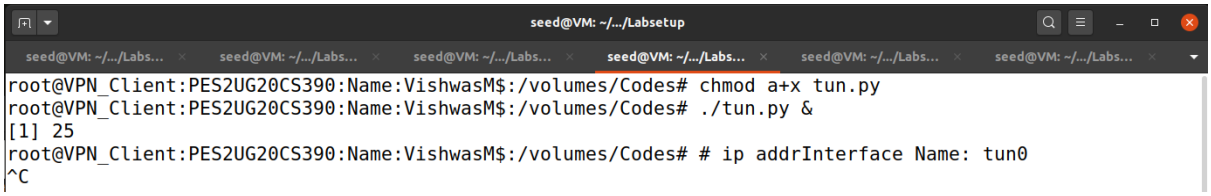


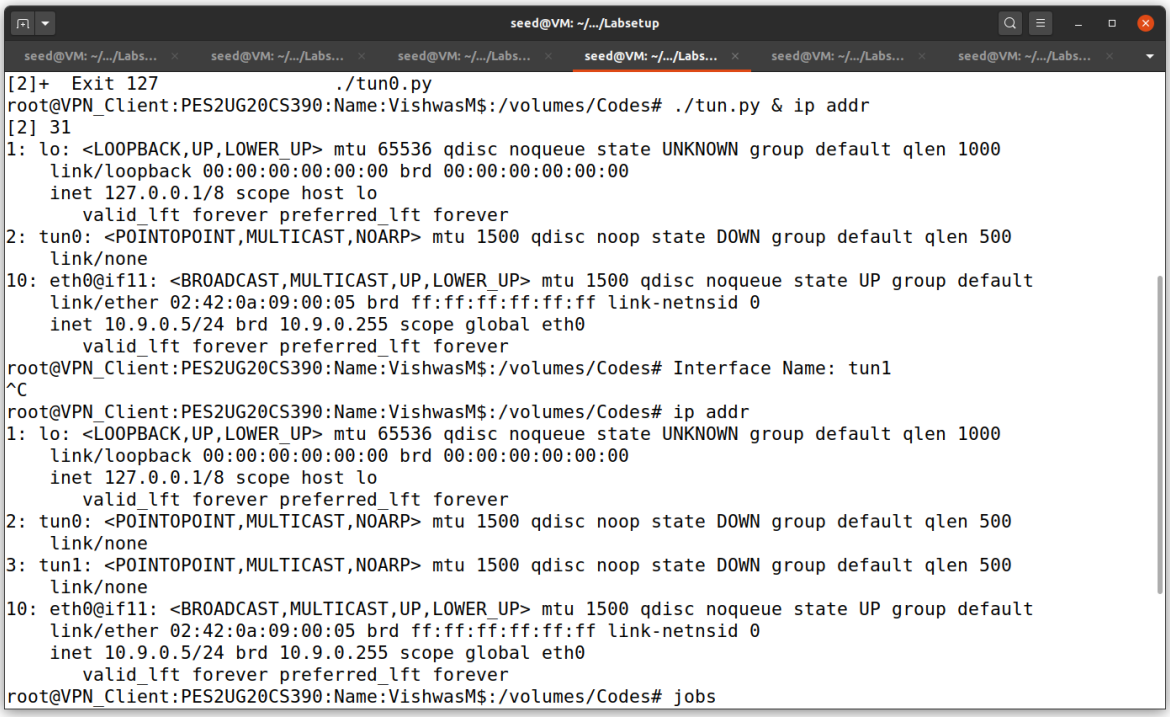
On Client - 10.9.0.5:



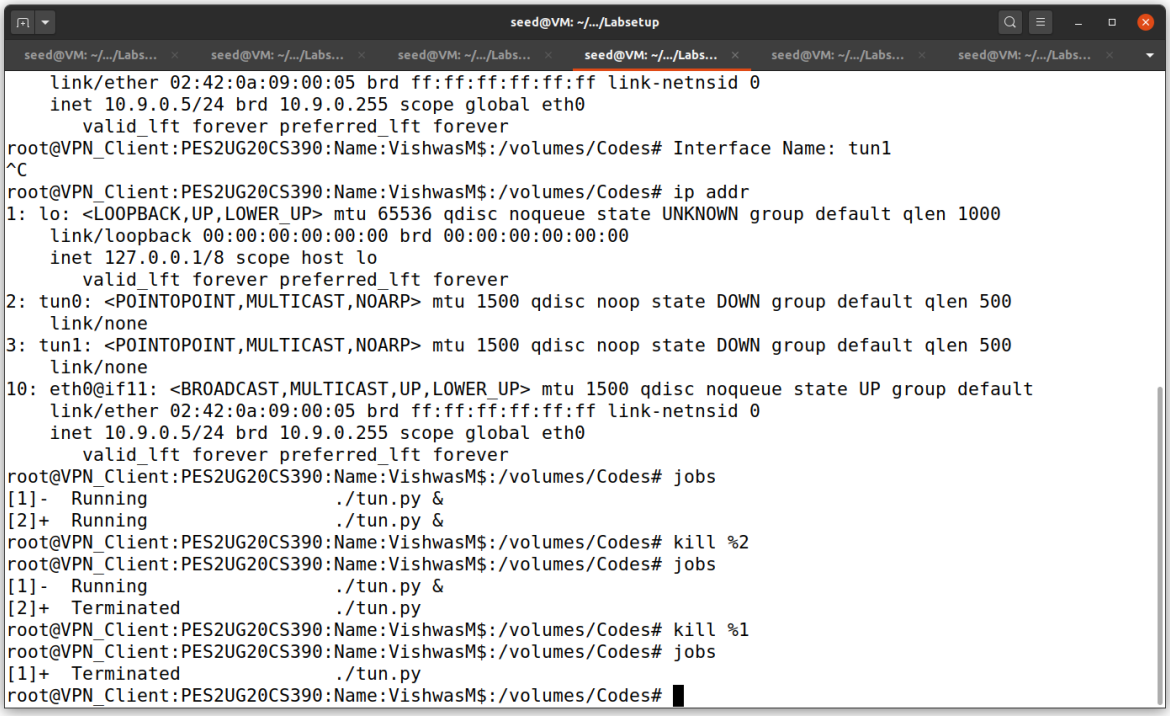
Task2: Create and Configure TUN Interface

Task 2.a: Name of the Interface

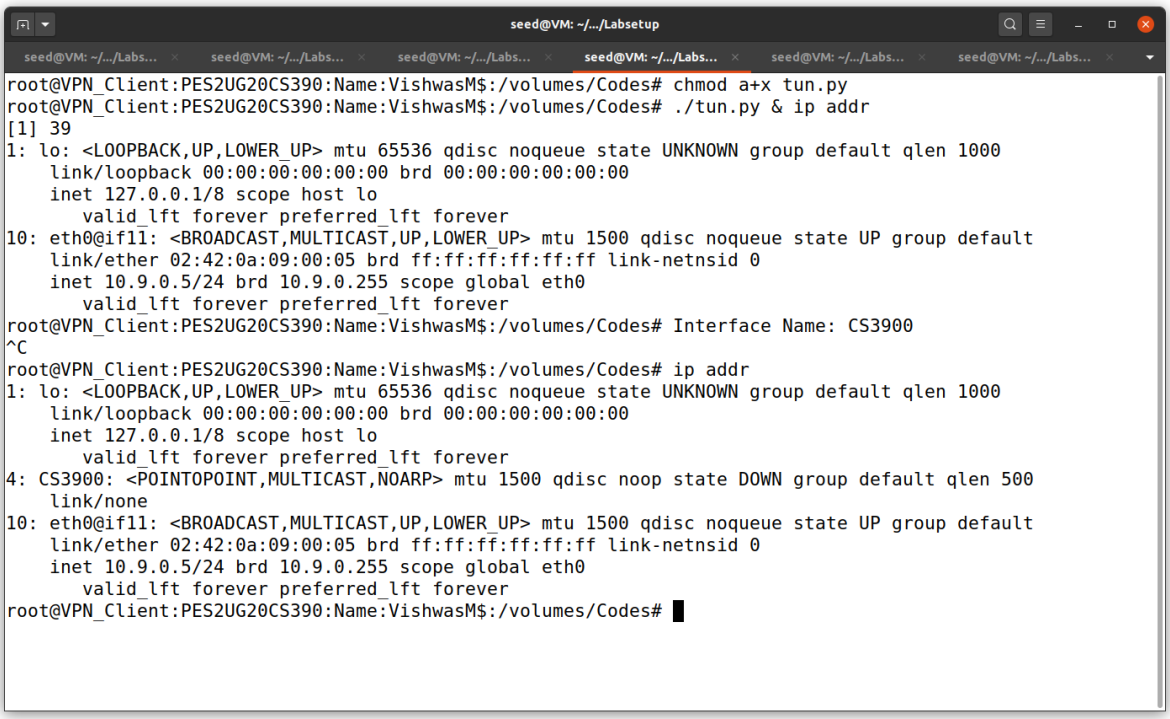




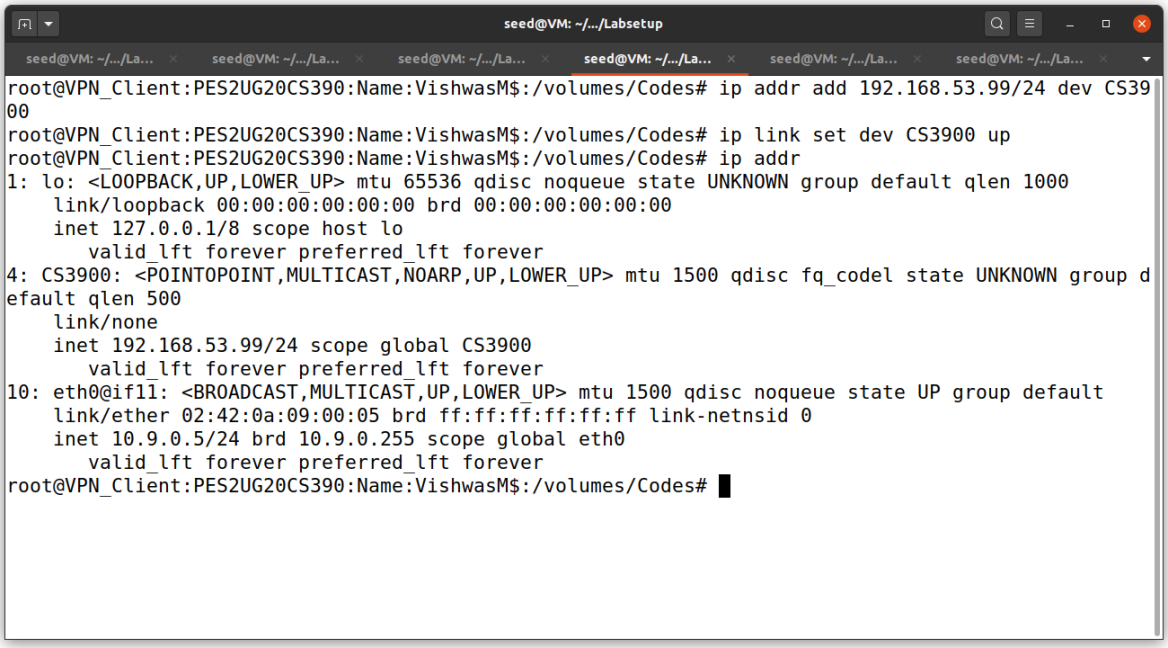
We use the cmd Jobs to check the TUN interface and we use kill command to kill the tunnel process.



Here I am changing the name of the TUN interface as ‘CS390’.



Task 2.b: Set up the TUN interface



First, we assigned an IP address to the TUN interface. Then we need to bring up the interface because it is still in down state.

Task 2.c: Read from the TUN interface

When we ping 192.168.53.5:

```
       valid_lft forever preferred_lft forever
3: CS3900: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UNKNOWN group default qlen 500
    link/none
    inet 192.168.53.99/24 scope global CS3900
       valid_lft forever preferred_lft forever
12: eth0@if13: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:0a:09:00:05 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.9.0.5/24 brd 10.9.0.255 scope global eth0
       valid_lft forever preferred_lft forever
root@VPN_Client:PES2UG20CS390:Name:VishwasM$:/volumes/Codes# ping 192.168.53.5
PING 192.168.53.5 (192.168.53.5) 56(84) bytes of data.
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
^CIP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.53.5 echo-request 0 / Raw
^C
--- 192.168.53.5 ping statistics ---
16 packets transmitted, 0 received, 100% packet loss, time 15405ms

root@VPN_Client:PES2UG20CS390:Name:VishwasM$:/volumes/Codes#
```

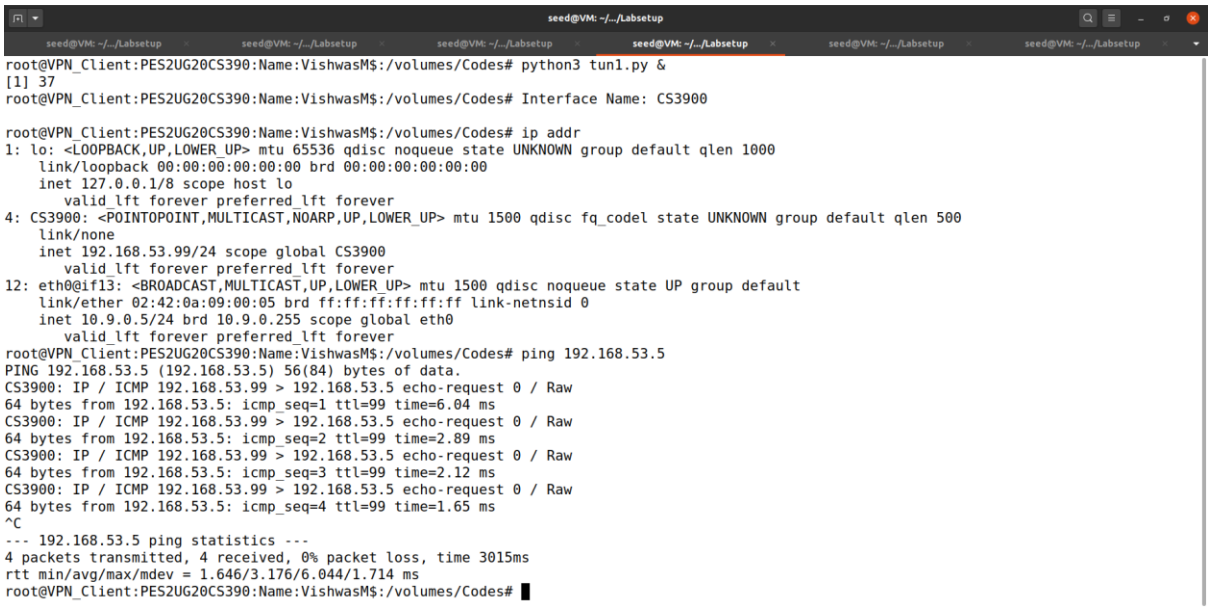
First, we had to change the code in tun.py. Then we had to run the updated python file. Then we are trying to read all the packets coming to TUN interface

When we ping 192.168.60.5:

```
root@VPN_Client:PES2UG20CS390:Name:VishwasM$:/volumes/Codes# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
^C
--- 192.168.60.5 ping statistics ---
7 packets transmitted, 0 received, 100% packet loss, time 6128ms

root@VPN_Client:PES2UG20CS390:Name:VishwasM$:/volumes/Codes#
```

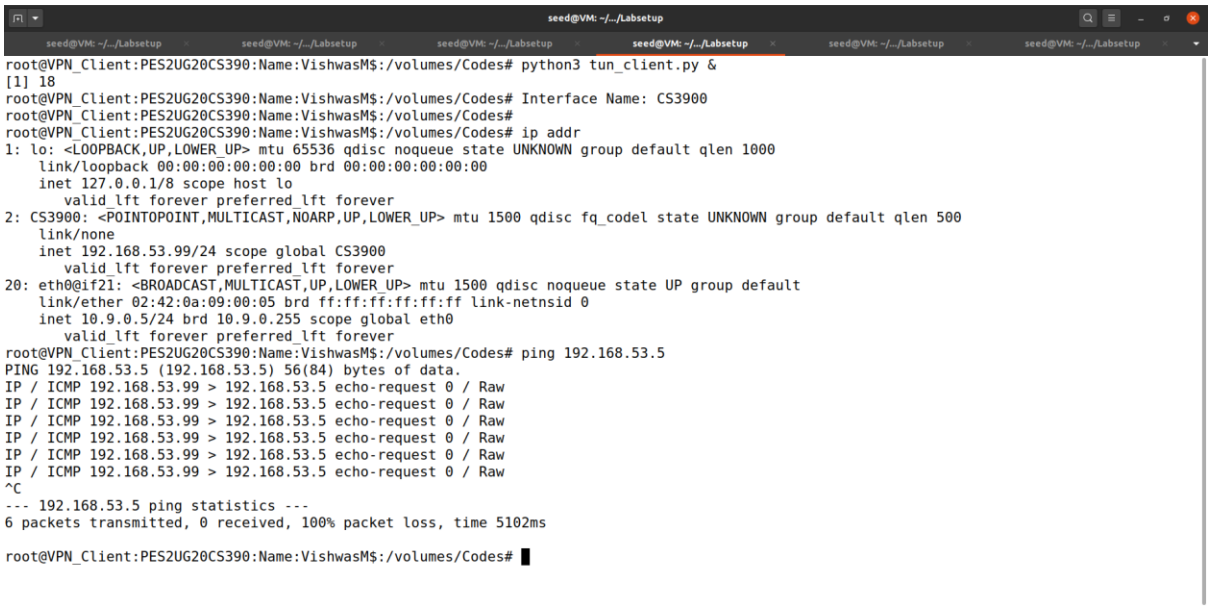
We don't get any output when we ping this ip addr because it is still not receiving any packets from the TUN interface.

Task 2.d: Write to the TUN interface



As we can see here both request and reply ICMP packets are seen which means that TUN interface is accepting as well as receiving the packets.

Task 3: Send the IP packet to VPN Server through a Tunnel



We can see that the packets are 100% lost in the above screenshot.

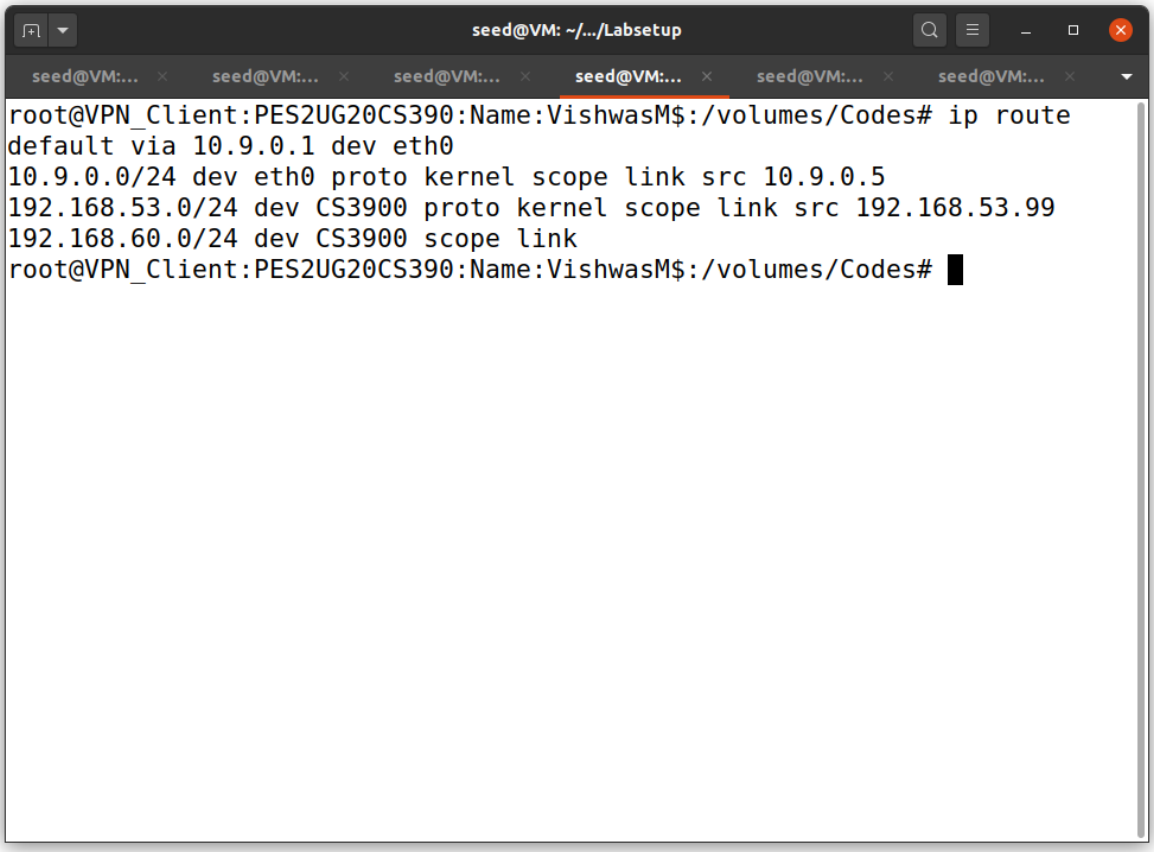
```
root@server-router:PES2UG20CS390:Name:VishwasM$:/volumes/Codes# python3 tun_server.py
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.53.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.53.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.53.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.53.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.53.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.53.5
```

In the server we can see how the packets are moving inside the tunnel as we can see in the above screenshot.

```
root@VPN_Client:PES2UG20CS390:Name:VishwasM$:/volumes/Codes# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
^C
--- 192.168.60.5 ping statistics ---
9 packets transmitted, 0 received, 100% packet loss, time 8198ms

root@VPN_Client:PES2UG20CS390:Name:VishwasM$:/volumes/Codes#
```

Here we are pinging to 192.168.60.5.

```
root@server-router:PES2UG20CS390:Name:VishwasM$:/volumes/Codes# python3 tun_server.py
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.53.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.53.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.53.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.53.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.53.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.53.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
```



We can see the IP routing routing table with the help of the above command mentioned in the above screenshot.

Task 4: Set up the VPN server

At Client:

```
root@VPN_Client:PES2UG20CS390:Name:VishwasM$:/volumes/Codes# python3 tun_client.py &
[2] 36
root@VPN_Client:PES2UG20CS390:Name:VishwasM$:/volumes/Codes# Interface Name: CS3901
RTNETLINK answers: File exists
^C
root@VPN_Client:PES2UG20CS390:Name:VishwasM$:/volumes/Codes# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
^C
--- 192.168.60.5 ping statistics ---
8 packets transmitted, 0 received, 100% packet loss, time 7151ms

root@VPN_Client:PES2UG20CS390:Name:VishwasM$:/volumes/Codes#
```

At Host1(192.168.60.5):

```
root@Host1:PES2UG20CS390:Name:VishwasM$:/# tcpdump -i eth0 -n
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 262144 bytes
06:58:21.242469 IP6 fe80::42:e7ff:fe8d:42e9.5353 > ff02::fb.5353: 0 [2q] PTR (QM)? _ipps._tcp.local.
PTR (QM)? _ipp._tcp.local. (45)
06:59:23.210223 IP6 fe80::42:e7ff:fe8d:42e9 > ff02::2: ICMP6, router solicitation, length 16
07:00:05.855168 ARP, Request who-has 192.168.60.5 tell 192.168.60.11, length 28
07:00:05.855280 ARP, Reply 192.168.60.5 is-at 02:42:c0:a8:3c:05, length 28
07:00:05.855317 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 46, seq 1, length 64
07:00:05.855404 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 46, seq 1, length 64
07:00:06.860082 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 46, seq 2, length 64
07:00:06.860186 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 46, seq 2, length 64
07:00:07.884631 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 46, seq 3, length 64
07:00:07.885646 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 46, seq 3, length 64
07:00:08.908483 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 46, seq 4, length 64
07:00:08.908510 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 46, seq 4, length 64
07:00:09.935049 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 46, seq 5, length 64
07:00:09.935094 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 46, seq 5, length 64
07:00:10.956844 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 46, seq 6, length 64
07:00:10.957286 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 46, seq 6, length 64
07:00:11.081176 ARP, Request who-has 192.168.60.11 tell 192.168.60.5, length 28
07:00:11.081498 ARP, Reply 192.168.60.11 is-at 02:42:c0:a8:3c:0b, length 28
07:00:11.980769 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 46, seq 7, length 64
07:00:11.980806 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 46, seq 7, length 64
07:00:13.005196 IP 192.168.53.99 > 192.168.60.5: ICMP echo request, id 46, seq 8, length 64
07:00:13.005237 IP 192.168.60.5 > 192.168.53.99: ICMP echo reply, id 46, seq 8, length 64
```

At Serve-Router:

```
    inet 192.168.53.1/24 scope global CS3900
       valid_lft forever preferred_lft forever
22: eth1@if23: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:c0:a8:3c:0b brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.168.60.11/24 brd 192.168.60.255 scope global eth1
       valid_lft forever preferred_lft forever
26: eth0@if27: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:0a:09:00:0b brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.9.0.11/24 brd 10.9.0.255 scope global eth0
       valid_lft forever preferred_lft forever
root@server-router:PES2UG20CS390:Name:VishwasM$:/volumes/Codes# 10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
10.9.0.5:60830 --> 0.0.0.0:9090
   Inside: 192.168.53.99 --> 192.168.60.5
root@server-router:PES2UG20CS390:Name:VishwasM$:/volumes/Codes#
```

Task 5: Handling Traffic in Both Directions

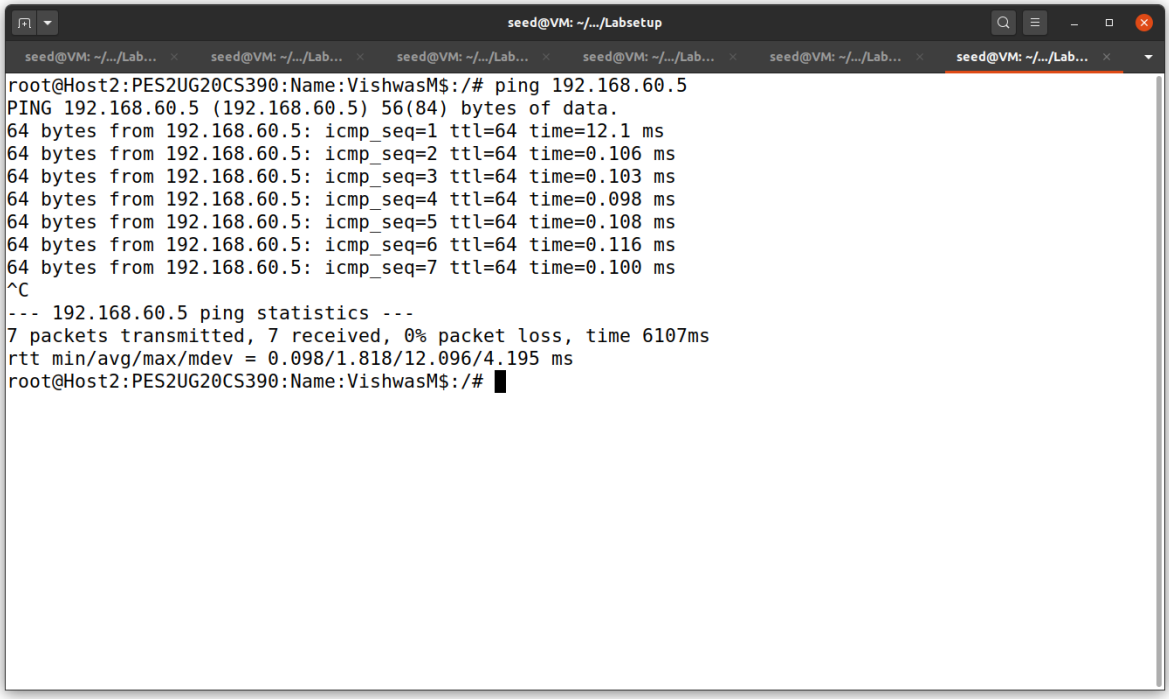
In server-router:

```
root@server-router:PES2UG20CS390:Name:VishwasM$:/volumes/Codes# python3 tun_server_select.py &
[1] 32
root@server-router:PES2UG20CS390:Name:VishwasM$:/volumes/Codes# Interface Name: CS3900

root@server-router:PES2UG20CS390:Name:VishwasM$:/volumes/Codes# From socket <==: 192.168.53.99 --> 19
2.168.60.5
From tun    ==>: 192.168.60.5 --> 192.168.53.99
From socket <==: 192.168.53.99 --> 192.168.60.5
From tun    ==>: 192.168.60.5 --> 192.168.53.99
From socket <==: 192.168.53.99 --> 192.168.60.5
From tun    ==>: 192.168.60.5 --> 192.168.53.99
From socket <==: 192.168.53.99 --> 192.168.60.5
From tun    ==>: 192.168.60.5 --> 192.168.53.99
From socket <==: 192.168.53.99 --> 192.168.60.5
From tun    ==>: 192.168.60.5 --> 192.168.53.99
From socket <==: 192.168.53.99 --> 192.168.60.5
From tun    ==>: 192.168.60.5 --> 192.168.53.99
From socket <==: 192.168.53.99 --> 192.168.60.5
From tun    ==>: 192.168.60.5 --> 192.168.53.99
From socket <==: 192.168.53.99 --> 192.168.60.5
From tun    ==>: 192.168.60.5 --> 192.168.53.99
From socket <==: 192.168.53.99 --> 192.168.60.5
From tun    ==>: 192.168.60.5 --> 192.168.53.99
From socket <==: 192.168.53.99 --> 192.168.60.5
From tun    ==>: 192.168.60.5 --> 192.168.53.99
```

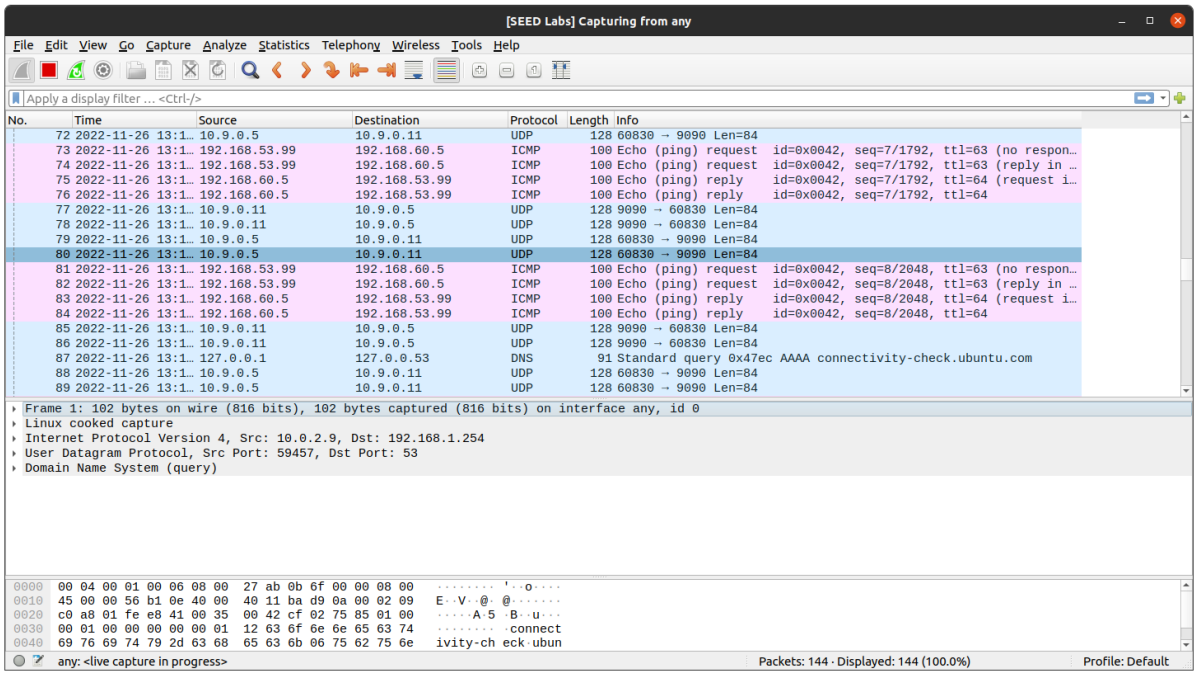
In VPN_Client:

```
root@VPN_Client:PES2UG20CS390:Name:VishwasM$:/volumes/Codes# python3 tun_client_select.py &
[3] 48
root@VPN_Client:PES2UG20CS390:Name:VishwasM$:/volumes/Codes# Interface Name: CS3902
RTNETLINK answers: File exists

root@VPN_Client:PES2UG20CS390:Name:VishwasM$:/volumes/Codes# IP / ICMP 192.168.53.99 > 192.168.60.5 e
cho-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
IP / ICMP 192.168.53.99 > 192.168.60.5 echo-request 0 / Raw
```

In another VPN_Client server:



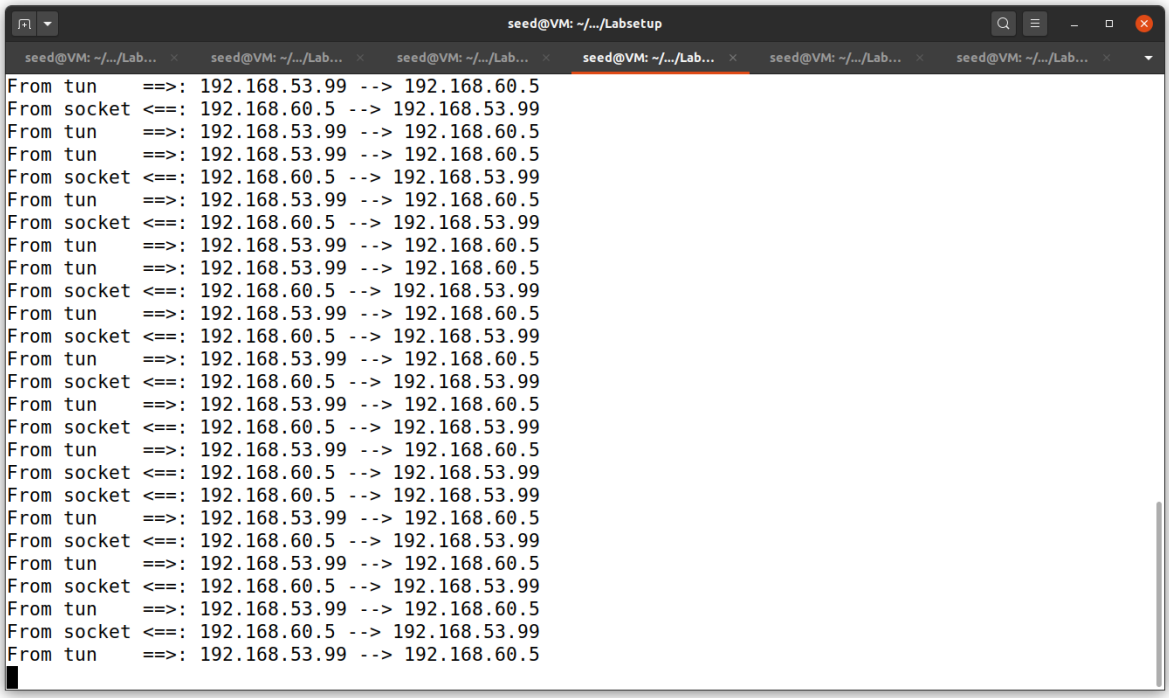
Wireshark capture:



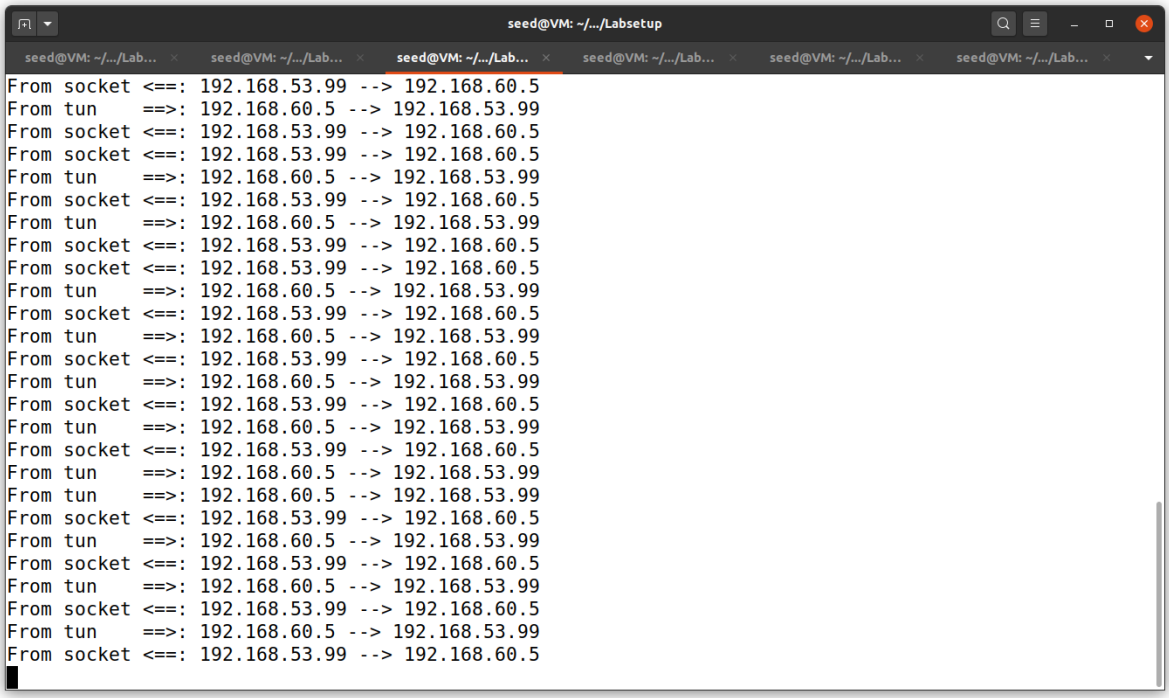
When we do a telnet connection:

In another VPN_Client where we establish telnet connection:

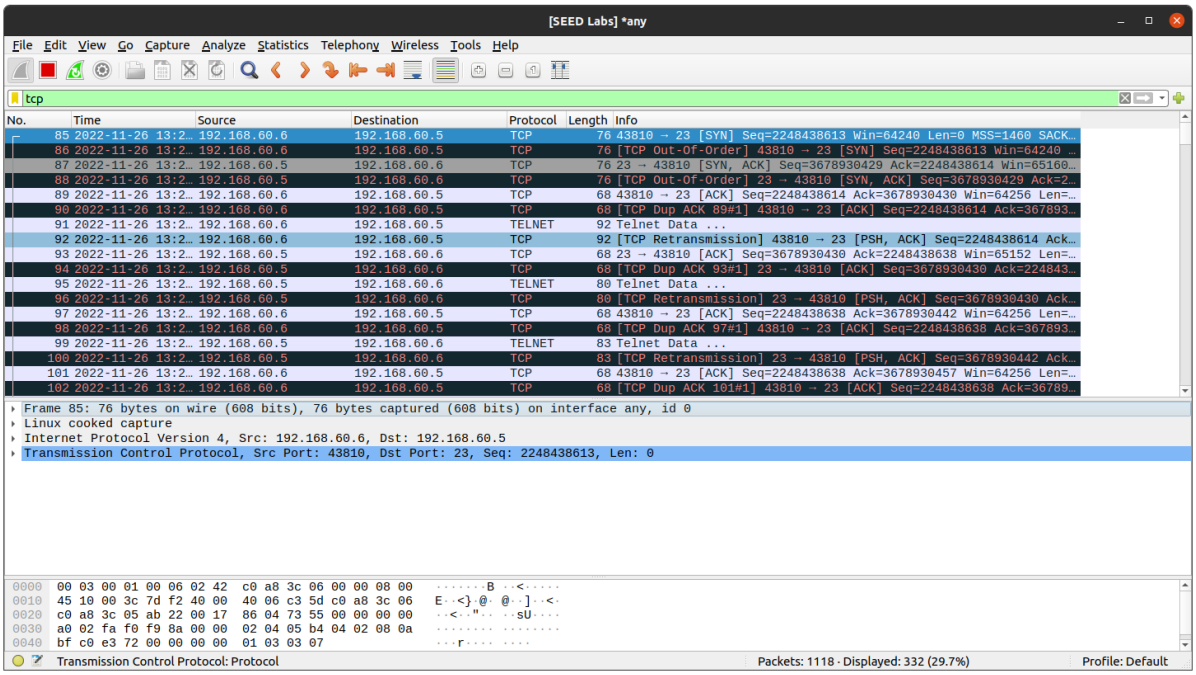
VPN_Client:



Server- router :



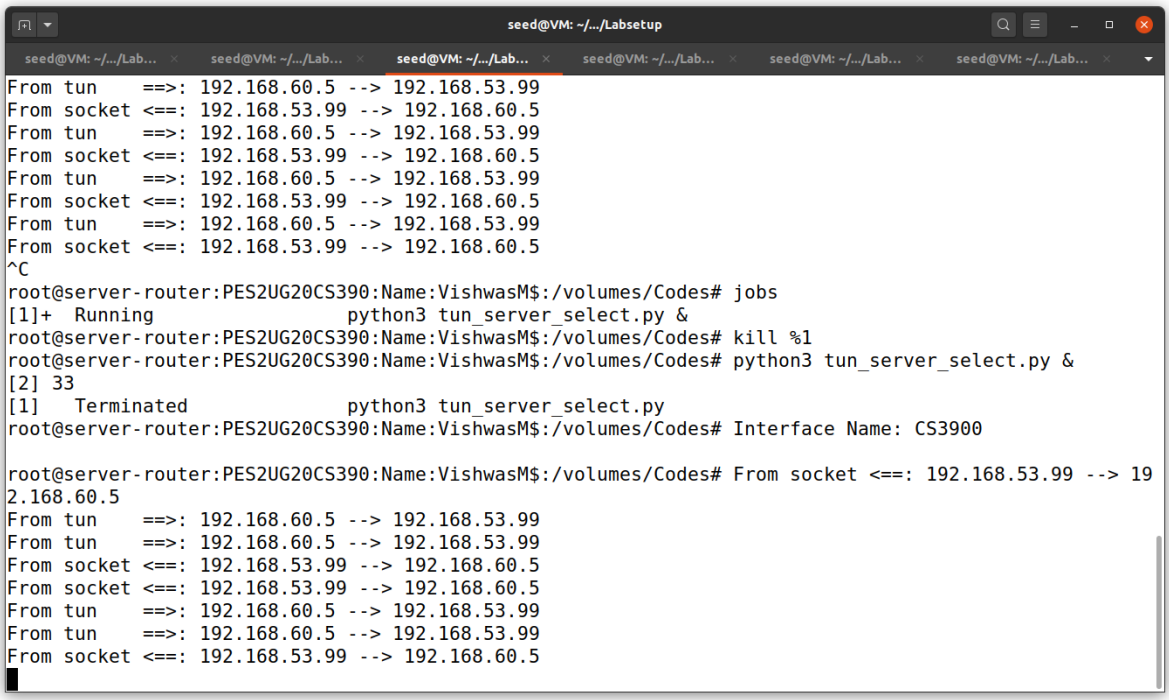
Wireshark Capture:



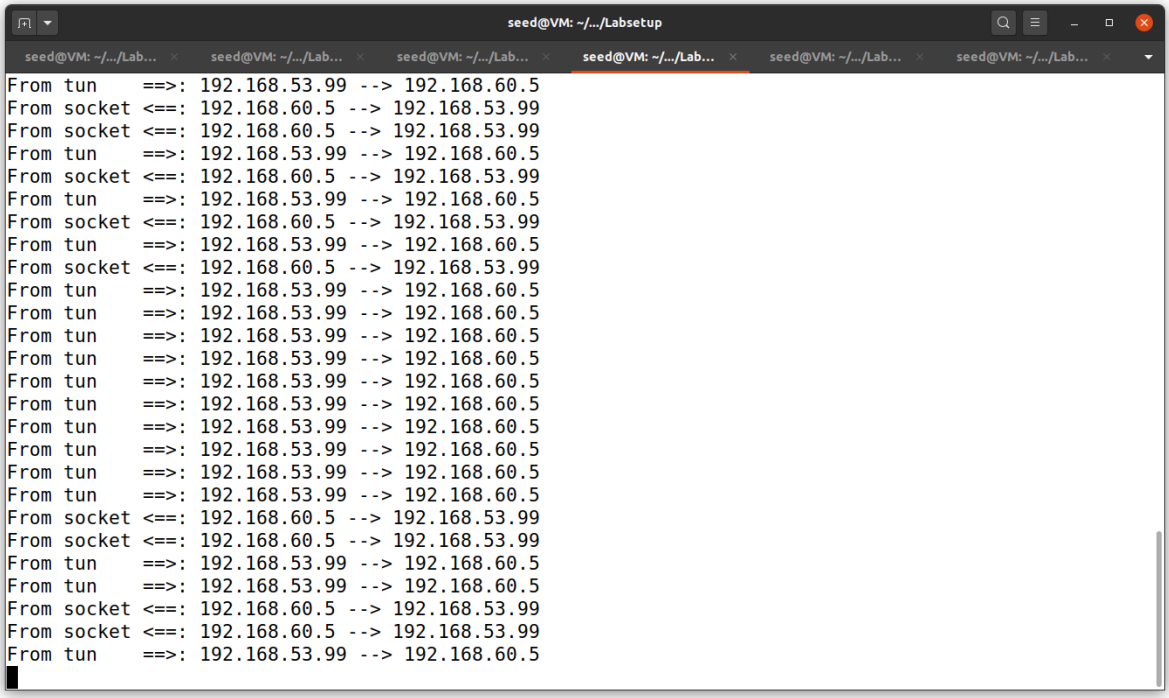
Task 6: Tunnel Breaking Experiment

Here we are breaking the Tunnel connection by pressing ctrl+C. When we try and press something in the client it will work as the screen gets stuck.

When we connect the tunnel again the word that we typed in terminal gets printed on the terminal which indicates that the tunnel is back and the connection is successful.

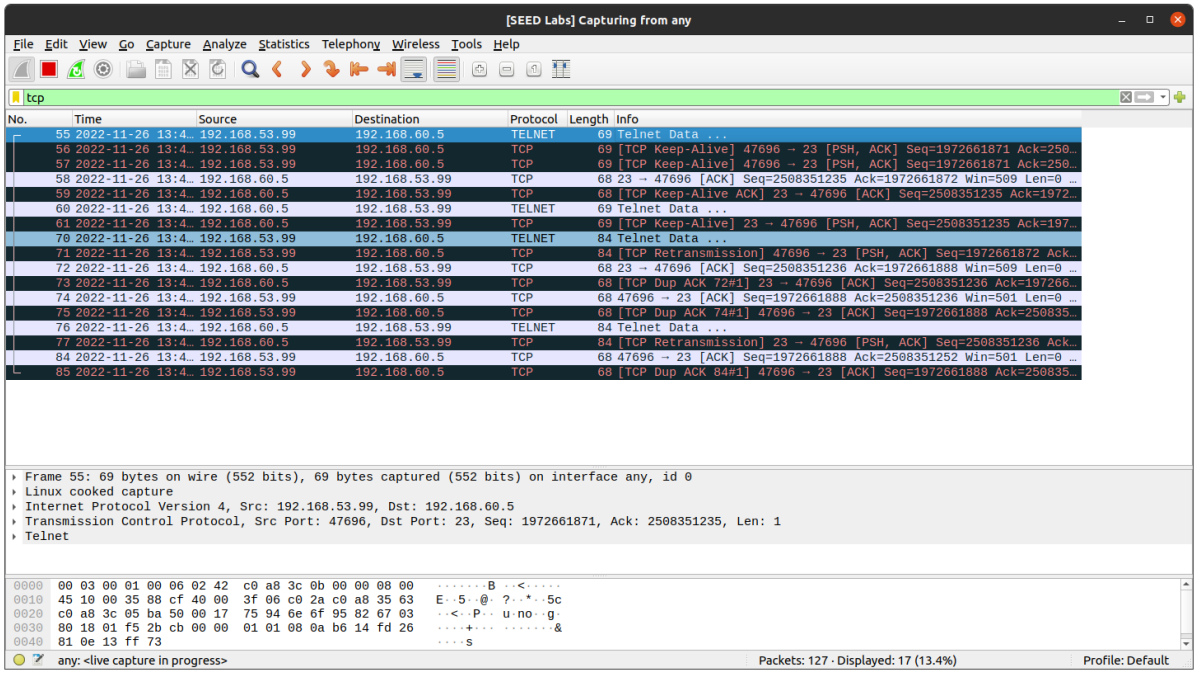


VPN_Client:



Server-Router:

Wireshark Capture:



Here we can observe that the connection which we just broke and connected in clearly depicted here.