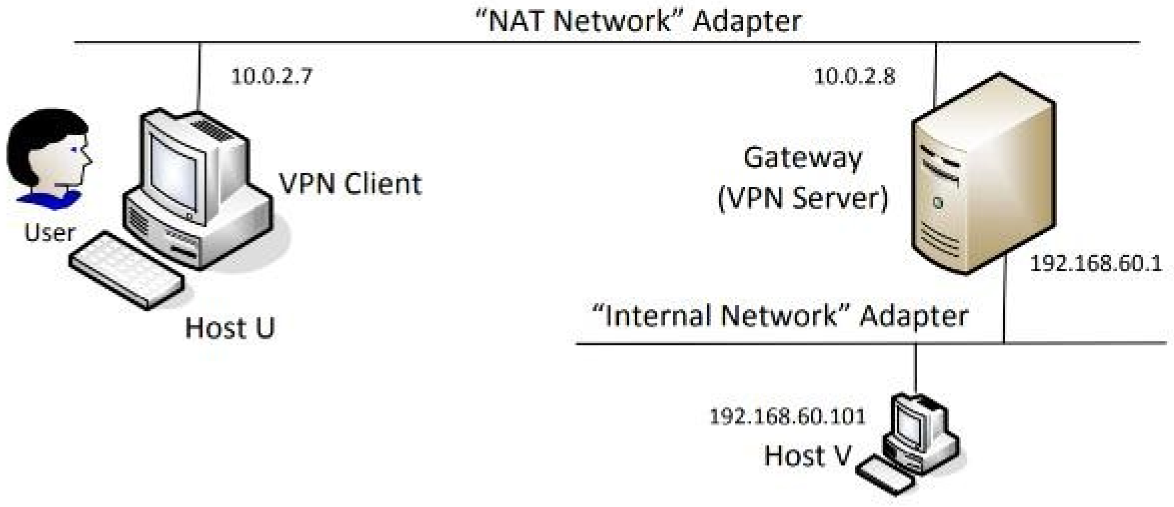
Rangineny sai kiran
CB.SC.P2CYS24011

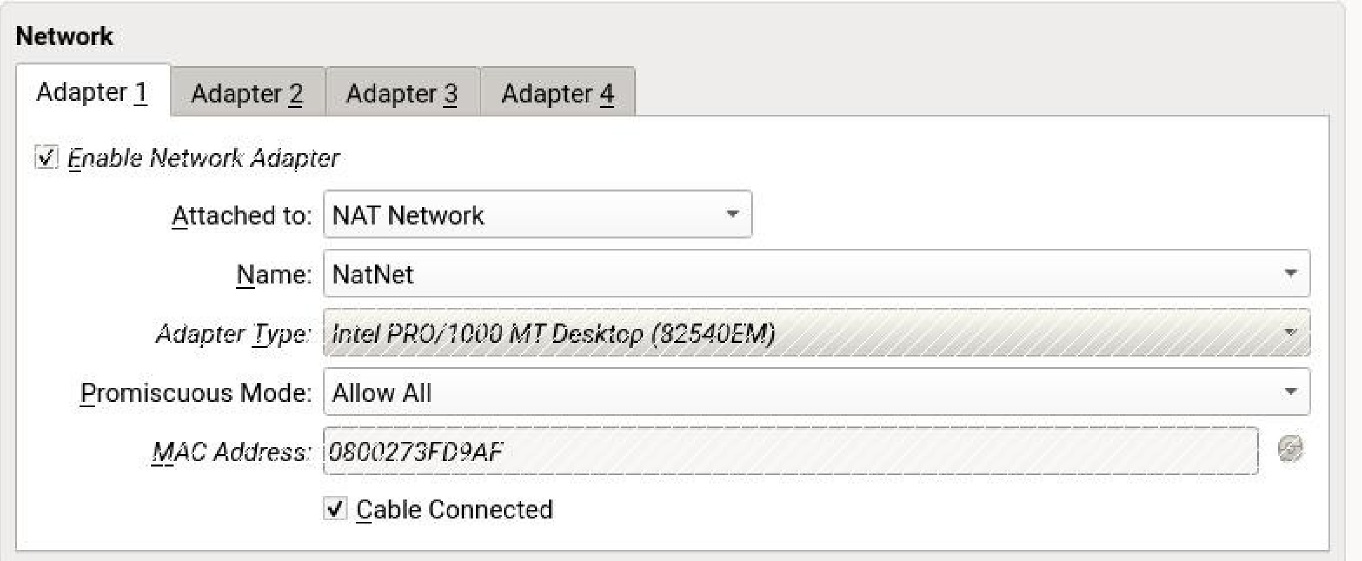
24CYS682 - Cyber Security Lab
Assignment – 9
Virtual Private Network Lab

Task 1 – VM Setup

We will create a VPN tunnel between a computer (client) and a gateway, allowing the computer to securely access a private network via the gateway. We need at least three VMs: VPN client (also serving as Host U), VPN server (the gateway), and a host in the private network (Host V). The network setup is depicted in the figure**.**



We need to establish a set up like above. In order to do this, the client and the server will have a NAT NETWORK



Network

☑ Enable Network Adapter

Attached to: NAT Network

Name: NatNet

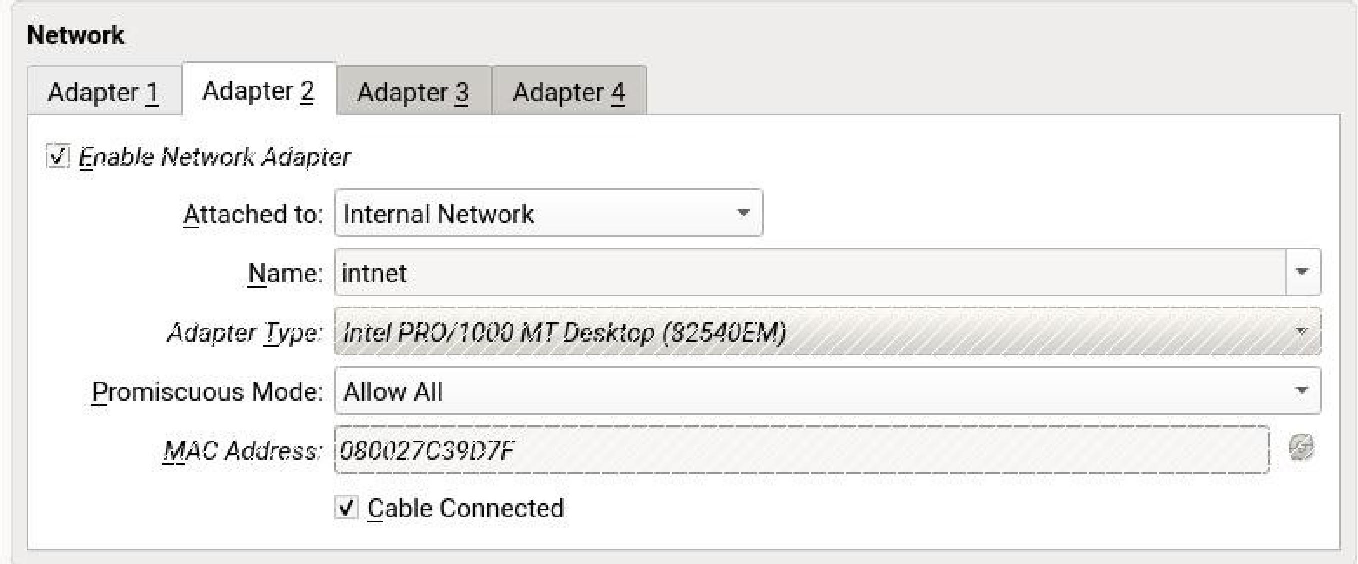
Adapter Type: Intel PRO/1000 MT Desktop (82540EM)

Promiscuous Mode: Allow All

MAC Address: 0800273FD9AF

☑ Cable Connected

However, the server and the host machines will be connected through an 'internal network' so that the client and the host have no connection.



Network

☑ Enable Network Adapter

Attached to: Internal Network

Name: intnet

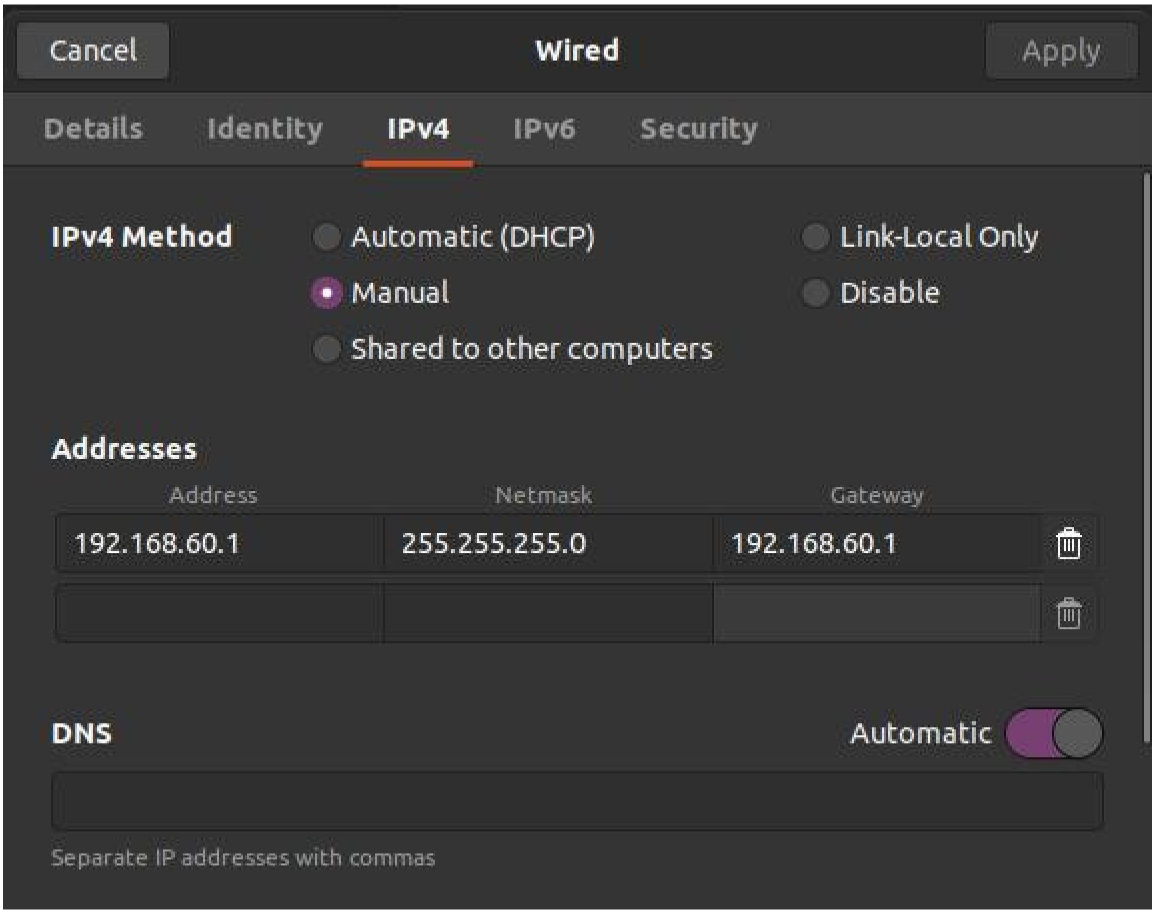
Adapter Type: Intel PRO/1000 MT Desktop (82540EM)

Promiscuous Mode: Allow All

MAC Address: 080027C39D7F

☑ Cable Connected

Server VM Configuration



IPv4 Method

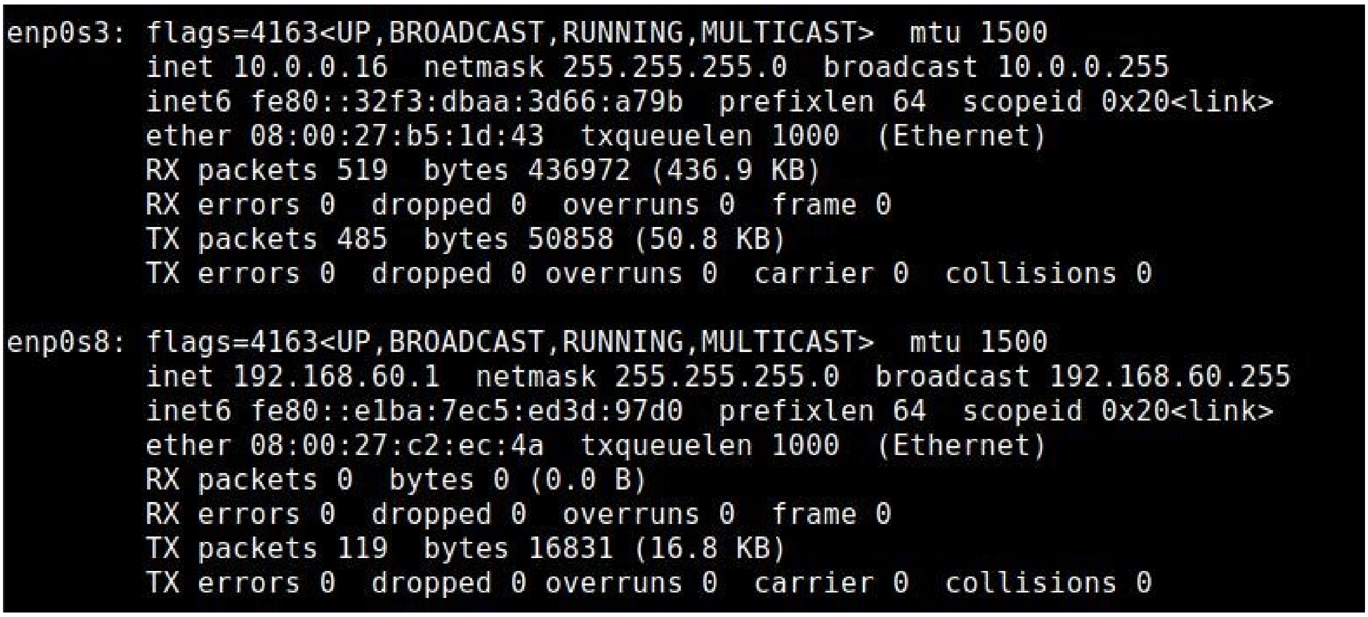
○ Automatic (DHCP)
○ Link-Local Only
● Manual
○ Disable
○ Shared to other computers

Addresses

Address	Netmask	Gateway
192.168.60.1	255.255.255.0	192.168.60.1

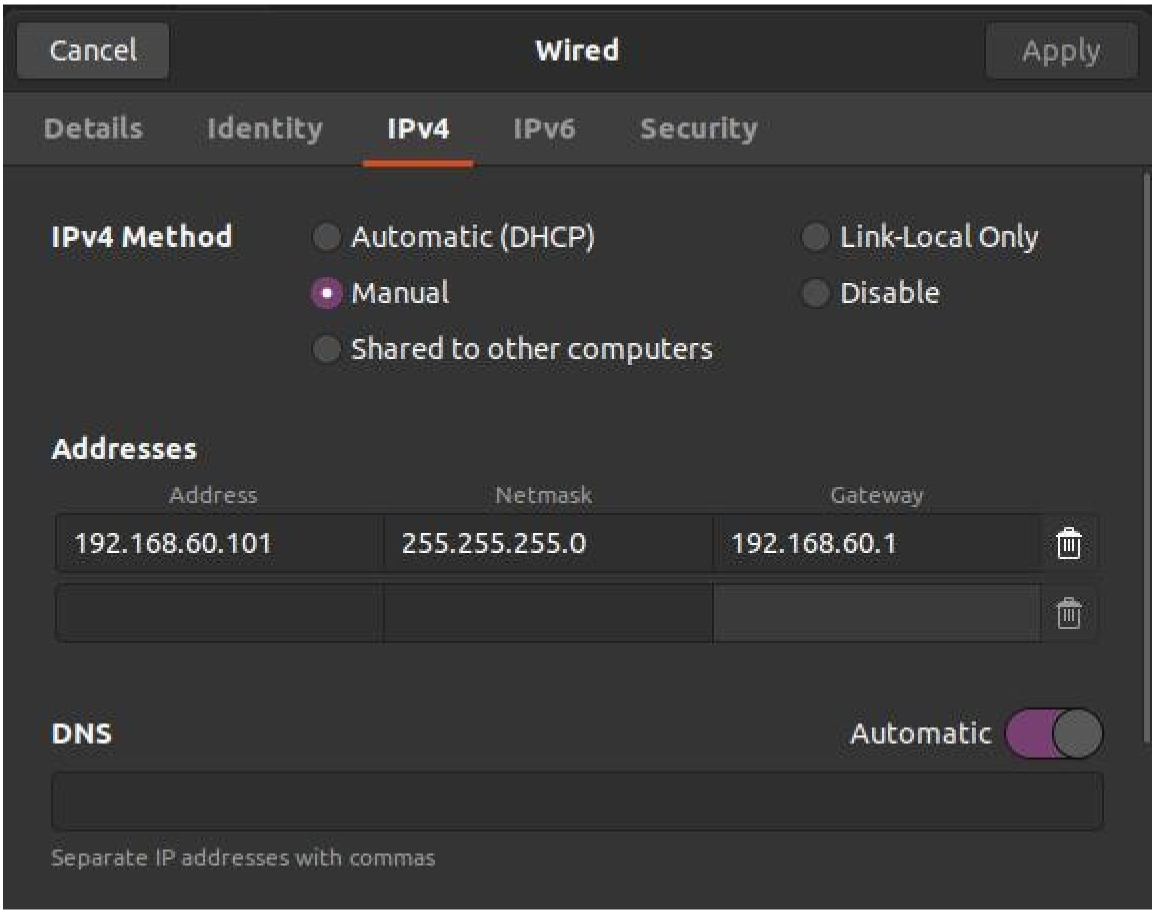
DNS — Automatic: on

Separate IP addresses with commas



```
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.16  netmask 255.255.255.0  broadcast 10.0.0.255
        inet6 fe80::32f3:dbaa:3d66:a79b  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:b5:1d:43  txqueuelen 1000  (Ethernet)
        RX packets 519  bytes 436972 (436.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 485  bytes 50858 (50.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.60.1  netmask 255.255.255.0  broadcast 192.168.60.255
        inet6 fe80::e1ba:7ec5:ed3d:97d0  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:c2:ec:4a  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 119  bytes 16831 (16.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Host VM Configuration



IPv4 Method

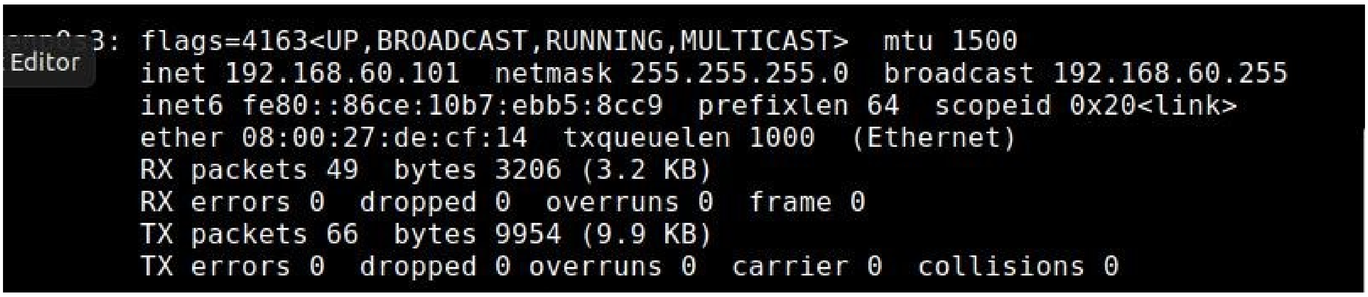
○ Automatic (DHCP) ○ Link-Local Only
● Manual ○ Disable
○ Shared to other computers

Addresses

Address	Netmask	Gateway
192.168.60.101	255.255.255.0	192.168.60.1

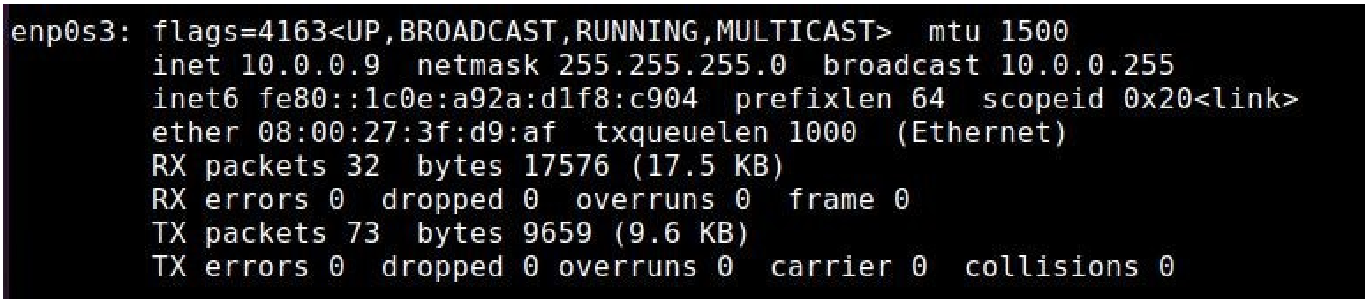
DNS Automatic

Separate IP addresses with commas



```
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.60.101  netmask 255.255.255.0  broadcast 192.168.60.255
        inet6 fe80::86ce:10b7:ebb5:8cc9  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:de:cf:14  txqueuelen 1000  (Ethernet)
        RX packets 49  bytes 3206 (3.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 66  bytes 9954 (9.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

VPN Client Configuration



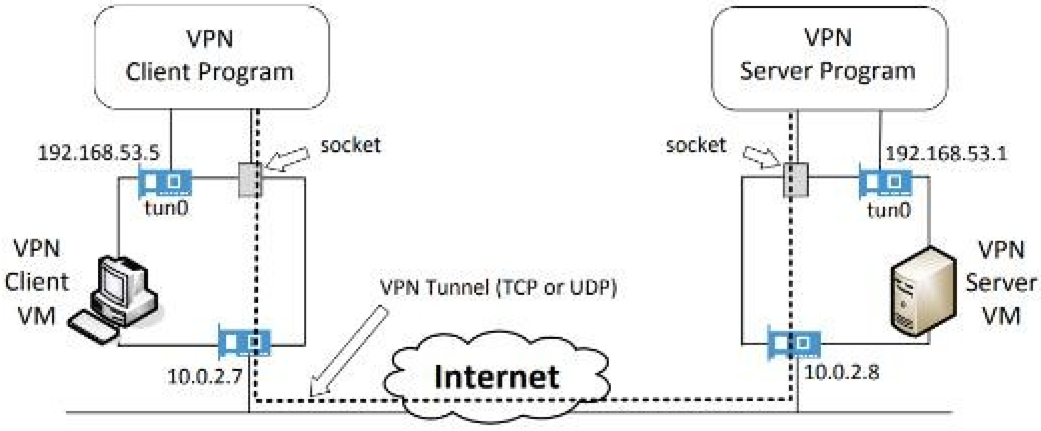
```
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.9  netmask 255.255.255.0  broadcast 10.0.0.255
        inet6 fe80::1c0e:a92a:d1f8:c904  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:3f:d9:af  txqueuelen 1000  (Ethernet)
        RX packets 32  bytes 17576 (17.5 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 73  bytes 9659 (9.6 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Therefore, based on the above screenshots, the following network connections are established:

- **VPN Client** – Adapter 1: NAT Network
- **VPN Server** – Adapter 1: NAT Network,

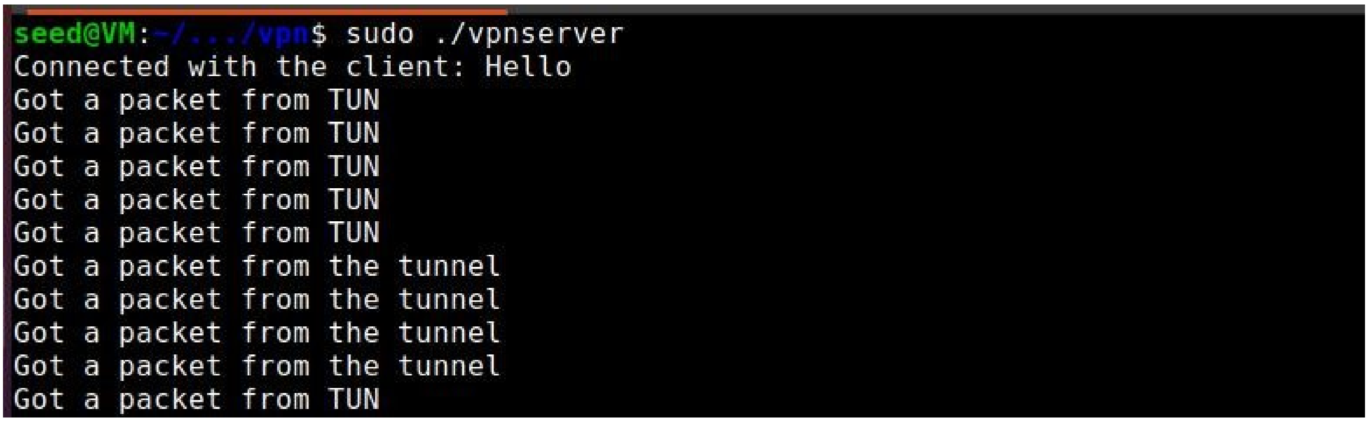
  – Adapter 2: Internal Network
- **Host V** – Adapter 1: Internal Network

Task 2: Creating a VPN Tunnel using TUN/TAP



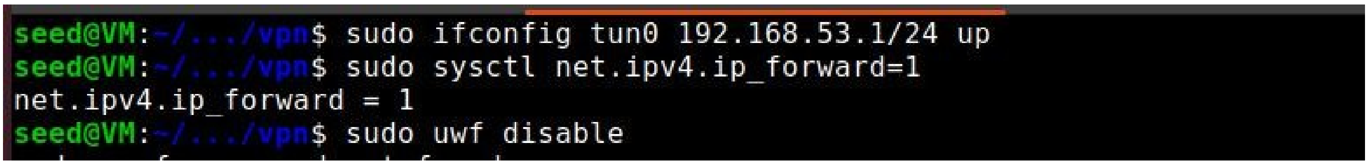
Step 1: Run VPN server

Run VPN Server and set it’s IP address of the interface Now we run the vpnserver.c code on the server machine.

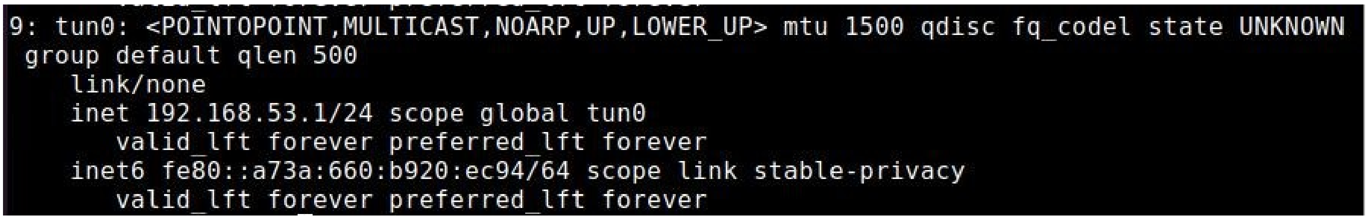


```
seed@VM:~/.../vpn$ sudo ./vpnserver
Connected with the client: Hello
Got a packet from TUN
Got a packet from TUN
Got a packet from TUN
Got a packet from TUN
Got a packet from TUN
Got a packet from the tunnel
Got a packet from the tunnel
Got a packet from the tunnel
Got a packet from the tunnel
Got a packet from TUN
```

Then we assign an IP address to the tun0 inferface and activate it. IP Address assigned: 192.168.53.1/24. We also enable port forwarding. Upon checking ifconfig : we have an established tunnel:



```
seed@VM:~/.../vpn$ sudo ifconfig tun0 192.168.53.1/24 up
seed@VM:~/.../vpn$ sudo sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
seed@VM:~/.../vpn$ sudo uwf disable
```



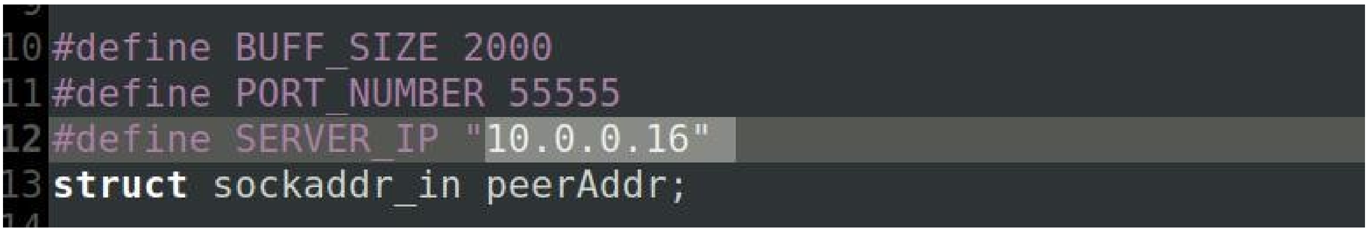
```
9: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UNKNOWN
 group default qlen 500
    link/none
    inet 192.168.53.1/24 scope global tun0
       valid_lft forever preferred_lft forever
    inet6 fe80::a73a:660:b920:ec94/64 scope link stable-privacy
       valid_lft forever preferred_lft forever
```

We can see that the tunnel is active. The VPN Server needs to forward packets to other destinations, so it needs to function as a gateway. We need to enable the IP forwarding for a computer to behave like a gateway.

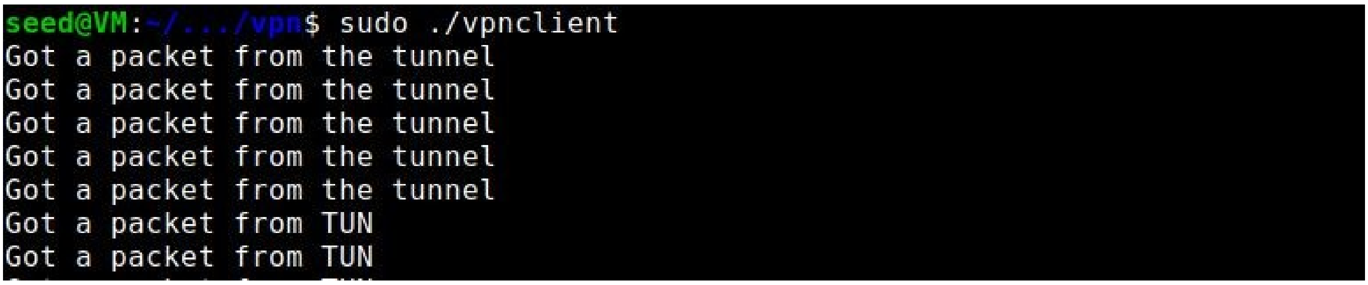
Step 2: Run VPN Client

Set server ip in client code.

Run VPN Client and set IP address of the interface Now we run the vpnclient.c code on the client machine.

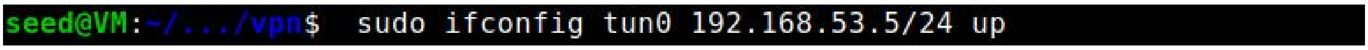


```
#define BUFF_SIZE 2000
#define PORT_NUMBER 55555
#define SERVER_IP "10.0.0.16"
struct sockaddr_in peerAddr;
```



```
seed@VM:~/.../vpn$ sudo ./vpnclient
Got a packet from the tunnel
Got a packet from the tunnel
Got a packet from the tunnel
Got a packet from the tunnel
Got a packet from the tunnel
Got a packet from TUN
Got a packet from TUN
```

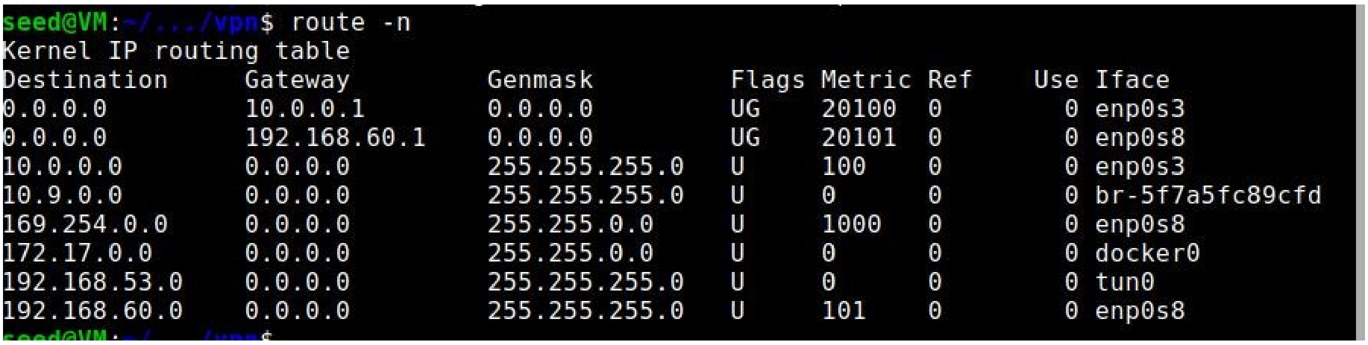
Then we assign an IP address to the tun0 inferface and activate it. IP Address assigned: 192.168.53.5/24



```
seed@VM:~/.../vpn$  sudo ifconfig tun0 192.168.53.5/24 up
```

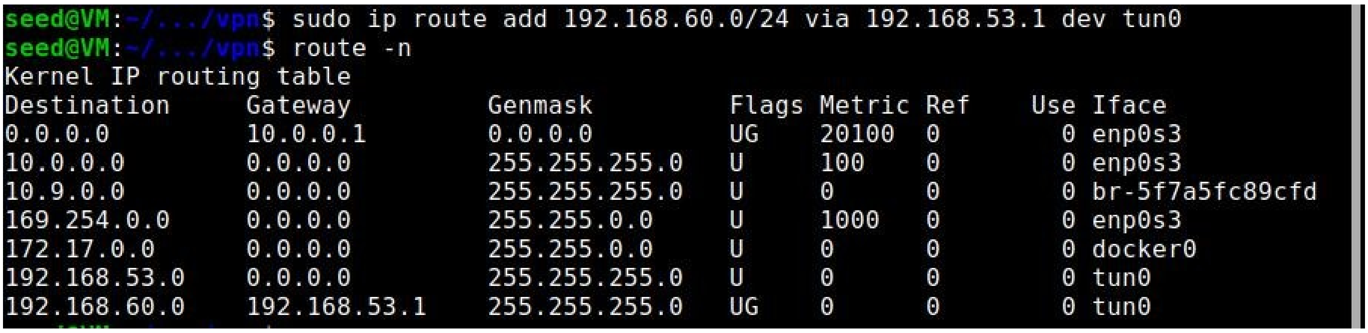
Step 3: setting up routing table in client and

server VPN Server routing table



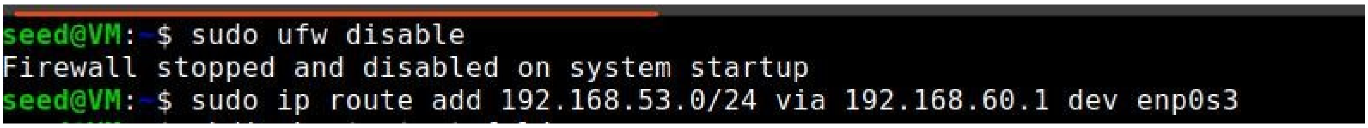
```
seed@VM:~/.../vpn$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.0.1        0.0.0.0         UG    20100  0        0 enp0s3
0.0.0.0         192.168.60.1    0.0.0.0         UG    20101  0        0 enp0s8
10.0.0.0        0.0.0.0         255.255.255.0   U     100    0        0 enp0s3
10.9.0.0        0.0.0.0         255.255.255.0   U     0      0        0 br-5f7a5fc89cfd
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 enp0s8
172.17.0.0      0.0.0.0         255.255.0.0     U     0      0        0 docker0
192.168.53.0    0.0.0.0         255.255.255.0   U     0      0        0 tun0
192.168.60.0    0.0.0.0         255.255.255.0   U     101    0        0 enp0s8
```

VPN Client routing table



```
seed@VM:~/.../vpn$ sudo ip route add 192.168.60.0/24 via 192.168.53.1 dev tun0
seed@VM:~/.../vpn$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.0.1        0.0.0.0         UG    20100  0        0 enp0s3
10.0.0.0        0.0.0.0         255.255.255.0   U     100    0        0 enp0s3
10.9.0.0        0.0.0.0         255.255.255.0   U     0      0        0 br-5f7a5fc89cfd
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 enp0s3
172.17.0.0      0.0.0.0         255.255.0.0     U     0      0        0 docker0
192.168.53.0    0.0.0.0         255.255.255.0   U     0      0        0 tun0
192.168.60.0    192.168.53.1    255.255.255.0   UG    0      0        0 tun0
```

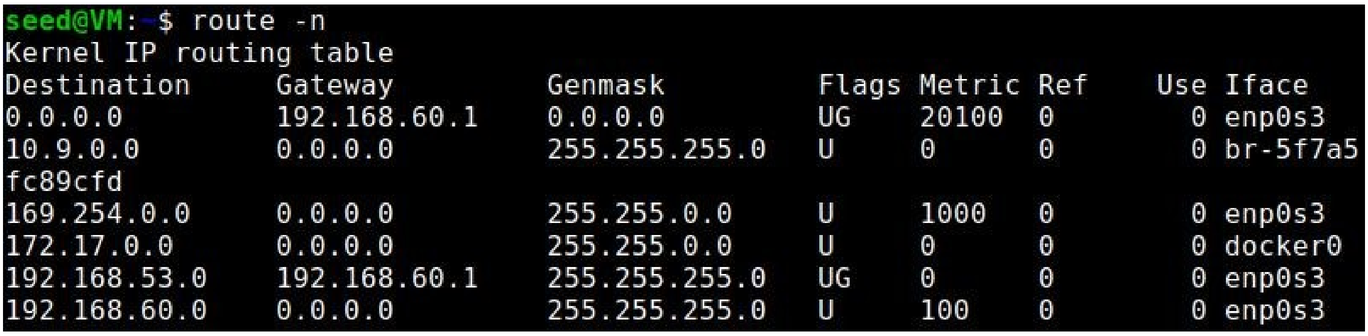
Step 4: Set up routing on HOST



```
seed@VM:~$ sudo ufw disable
Firewall stopped and disabled on system startup
seed@VM:~$ sudo ip route add 192.168.53.0/24 via 192.168.60.1 dev enp0s3
```

To set up routing on the Host, we first disable the firewall using `sudo ufw disable` to prevent any interference. Next, we add a route to direct traffic for the VPN network (192.168.53.0/24) through the correct gateway using `sudo ip route add 192.168.53.0/24 via`

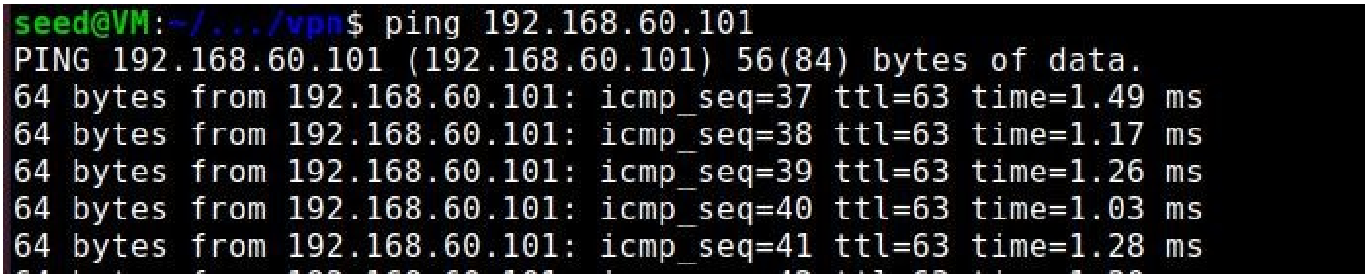
`192.168.60.1 dev enp0s3`. Finally, we verify the routing table with `route -n` to ensure the route has been correctly added.



```
seed@VM:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.60.1    0.0.0.0         UG    20100  0        0 enp0s3
10.9.0.0        0.0.0.0         255.255.255.0   U     0      0        0 br-5f7a5
fc89cfd
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 enp0s3
172.17.0.0      0.0.0.0         255.255.0.0     U     0      0        0 docker0
192.168.53.0    192.168.60.1    255.255.255.0   UG    0      0        0 enp0s3
192.168.60.0    0.0.0.0         255.255.255.0   U     100    0        0 enp0s3
```

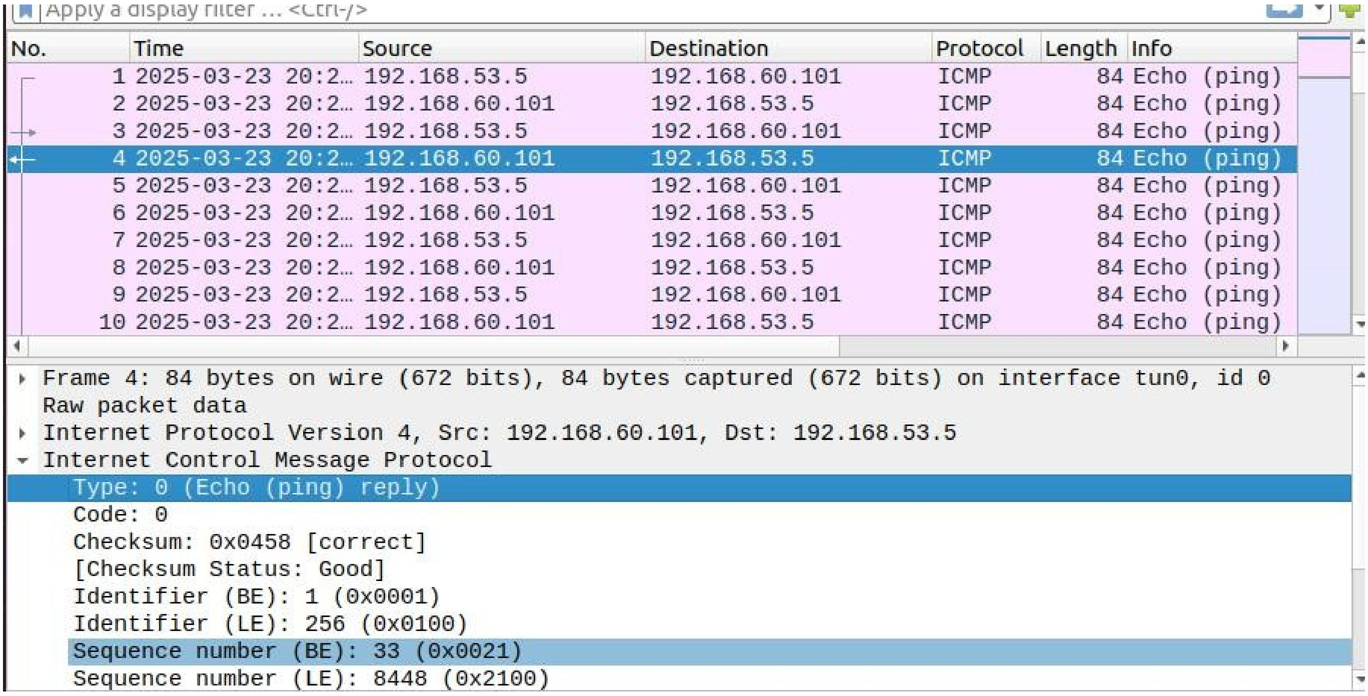
Step 5: Test the VPN tunnel (ping and telnet)

First we will perform the ping command to see if the VPN tunnel has been established:



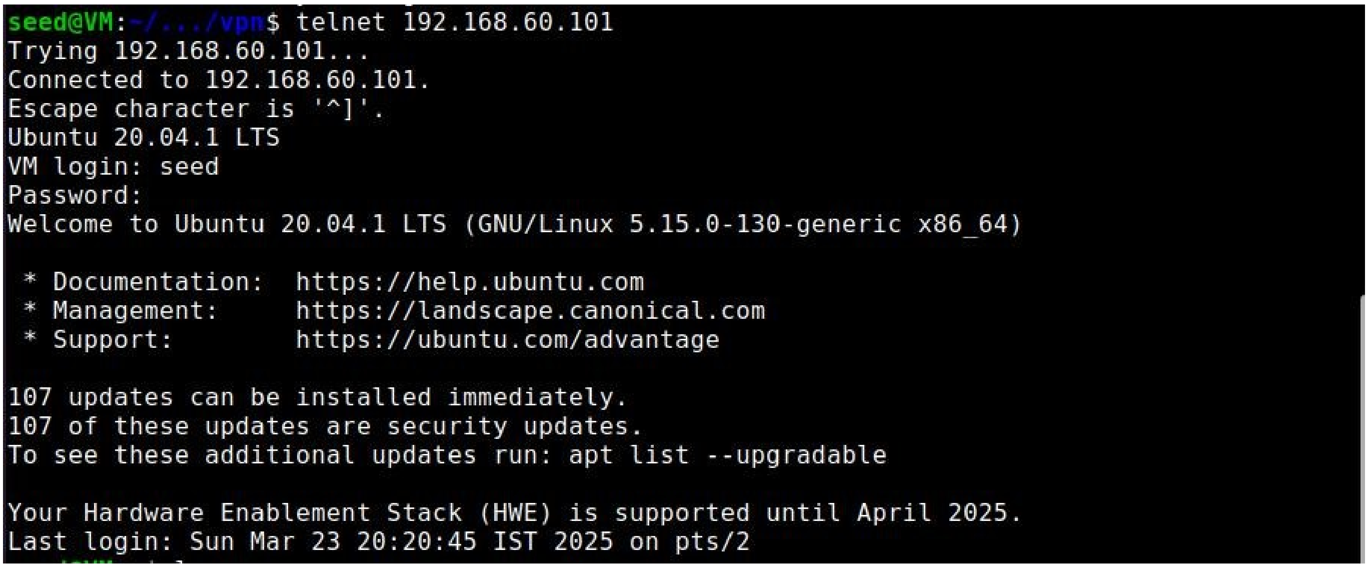
```
seed@VM:~/.../vpn$ ping 192.168.60.101
PING 192.168.60.101 (192.168.60.101) 56(84) bytes of data.
64 bytes from 192.168.60.101: icmp_seq=37 ttl=63 time=1.49 ms
64 bytes from 192.168.60.101: icmp_seq=38 ttl=63 time=1.17 ms
64 bytes from 192.168.60.101: icmp_seq=39 ttl=63 time=1.26 ms
64 bytes from 192.168.60.101: icmp_seq=40 ttl=63 time=1.03 ms
64 bytes from 192.168.60.101: icmp_seq=41 ttl=63 time=1.28 ms
```

We have successfully established connectivity, as confirmed by the ping response. The Wireshark screenshot provides a detailed view of the ICMP packet exchange, illustrating the communication between the source and destination over the VPN tunnel.



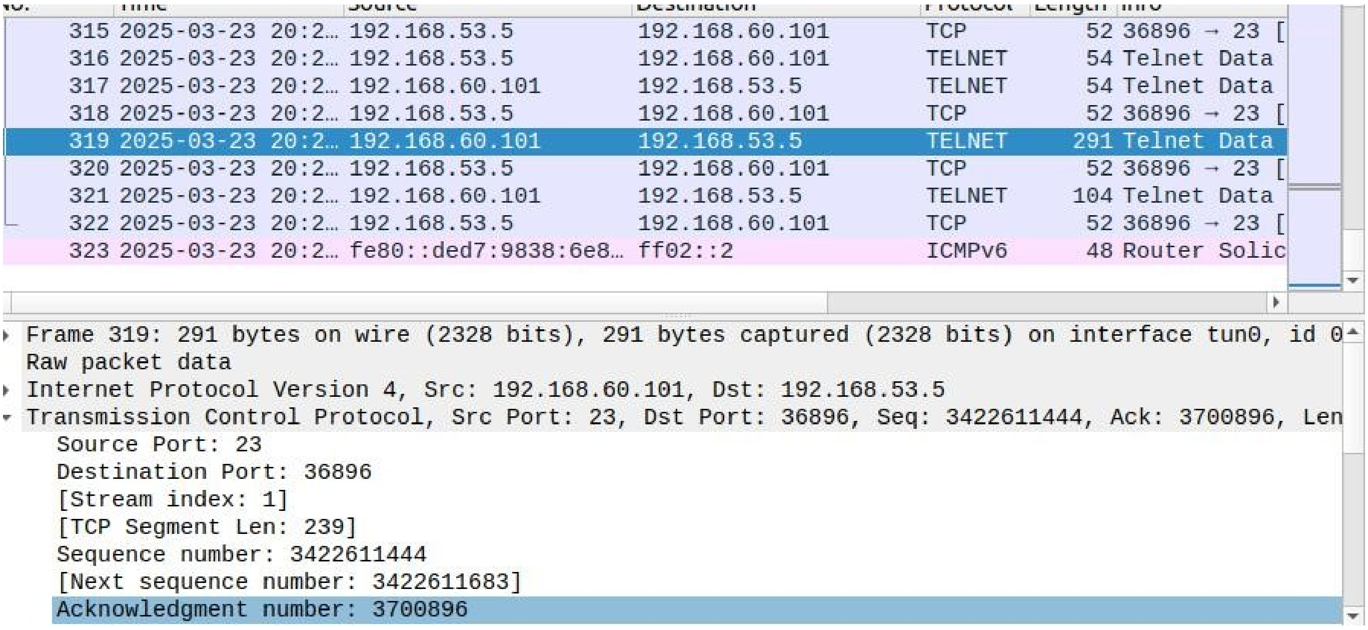
```
No.  Time               Source          Destination     Protocol Length Info
   1 2025-03-23 20:2… 192.168.53.5    192.168.60.101  ICMP         84 Echo (ping)
   2 2025-03-23 20:2… 192.168.60.101  192.168.53.5    ICMP         84 Echo (ping)
   3 2025-03-23 20:2… 192.168.53.5    192.168.60.101  ICMP         84 Echo (ping)
   4 2025-03-23 20:2… 192.168.60.101  192.168.53.5    ICMP         84 Echo (ping)
   5 2025-03-23 20:2… 192.168.53.5    192.168.60.101  ICMP         84 Echo (ping)
   6 2025-03-23 20:2… 192.168.60.101  192.168.53.5    ICMP         84 Echo (ping)
   7 2025-03-23 20:2… 192.168.53.5    192.168.60.101  ICMP         84 Echo (ping)
   8 2025-03-23 20:2… 192.168.60.101  192.168.53.5    ICMP         84 Echo (ping)
   9 2025-03-23 20:2… 192.168.53.5    192.168.60.101  ICMP         84 Echo (ping)
  10 2025-03-23 20:2… 192.168.60.101  192.168.53.5    ICMP         84 Echo (ping)

▸ Frame 4: 84 bytes on wire (672 bits), 84 bytes captured (672 bits) on interface tun0, id 0
  Raw packet data
▸ Internet Protocol Version 4, Src: 192.168.60.101, Dst: 192.168.53.5
▾ Internet Control Message Protocol
    Type: 0 (Echo (ping) reply)
    Code: 0
    Checksum: 0x0458 [correct]
    [Checksum Status: Good]
    Identifier (BE): 1 (0x0001)
    Identifier (LE): 256 (0x0100)
    Sequence number (BE): 33 (0x0021)
    Sequence number (LE): 8448 (0x2100)
```

From the Wireshark screenshot, we can observe that the packets with source `192.168.53.5` (Client - `tun0`) and destination `192.168.60.101` (Host VPN) represent tunnel traffic, while the remaining packets correspond to regular network traffic. Next, we will establish a Telnet connection to verify that the VPN tunnel is functioning correctly stablished



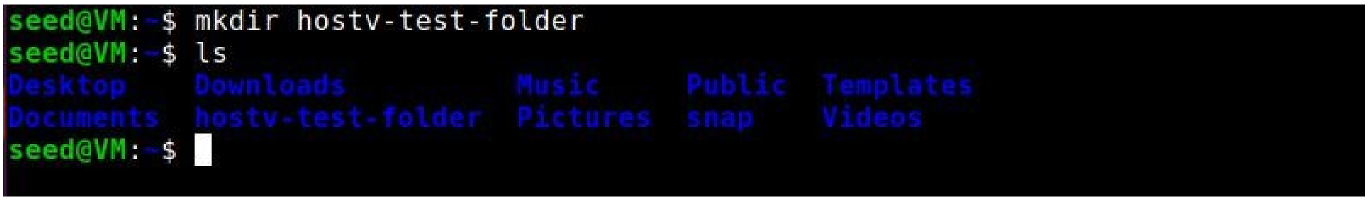
```
seed@VM:~/.../vpn$ telnet 192.168.60.101
Trying 192.168.60.101...
Connected to 192.168.60.101.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
VM login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.15.0-130-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

107 updates can be installed immediately.
107 of these updates are security updates.
To see these additional updates run: apt list --upgradable

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Sun Mar 23 20:20:45 IST 2025 on pts/2
```

We can see that we are successfully able to establish the telnet connection.
Wireshark screenshot to prove it:



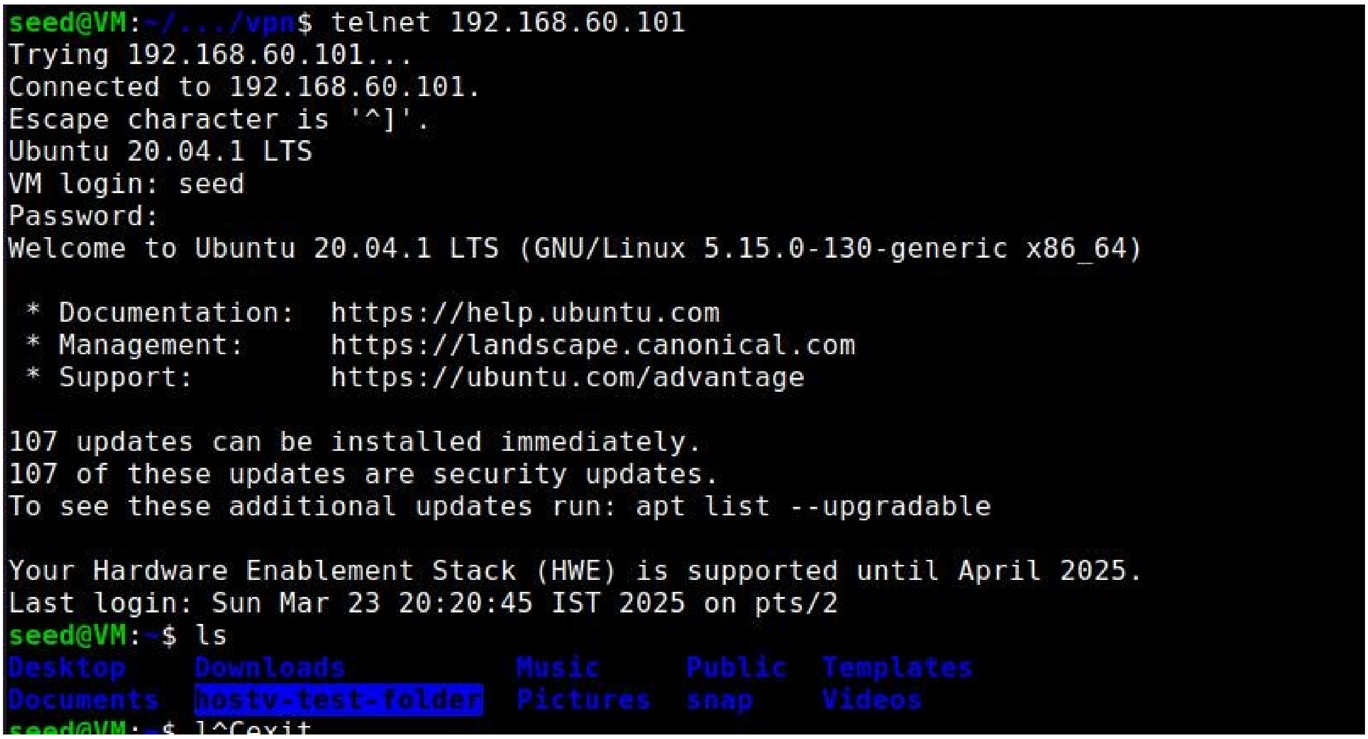
```
  315 2025-03-23 20:2… 192.168.53.5        192.168.60.101   TCP        52 36896 → 23 [
  316 2025-03-23 20:2… 192.168.53.5        192.168.60.101   TELNET     54 Telnet Data
  317 2025-03-23 20:2… 192.168.60.101      192.168.53.5     TELNET     54 Telnet Data
  318 2025-03-23 20:2… 192.168.53.5        192.168.60.101   TCP        52 36896 → 23 [
  319 2025-03-23 20:2… 192.168.60.101      192.168.53.5     TELNET    291 Telnet Data
  320 2025-03-23 20:2… 192.168.53.5        192.168.60.101   TCP        52 36896 → 23 [
  321 2025-03-23 20:2… 192.168.60.101      192.168.53.5     TELNET    104 Telnet Data
  322 2025-03-23 20:2… 192.168.53.5        192.168.60.101   TCP        52 36896 → 23 [
  323 2025-03-23 20:2… fe80::ded7:9838:6e8… ff02::2         ICMPv6     48 Router Solic

▸ Frame 319: 291 bytes on wire (2328 bits), 291 bytes captured (2328 bits) on interface tun0, id 0
  Raw packet data
▸ Internet Protocol Version 4, Src: 192.168.60.101, Dst: 192.168.53.5
▾ Transmission Control Protocol, Src Port: 23, Dst Port: 36896, Seq: 3422611444, Ack: 3700896, Len
    Source Port: 23
    Destination Port: 36896
    [Stream index: 1]
    [TCP Segment Len: 239]
    Sequence number: 3422611444
    [Next sequence number: 3422611683]
    Acknowledgment number: 3700896
```

From the screenshot, we can confirm that the VPN connection was successfully established. To further verify access, we executed the `ls` command on the VPN Host and created a new folder named `hostv-test-folder`, as shown in the screenshot



```
seed@VM:~$ mkdir hostv-test-folder
seed@VM:~$ ls
Desktop    Downloads          Music     Public  Templates
Documents  hostv-test-folder  Pictures  snap    Videos
seed@VM:~$ 
```

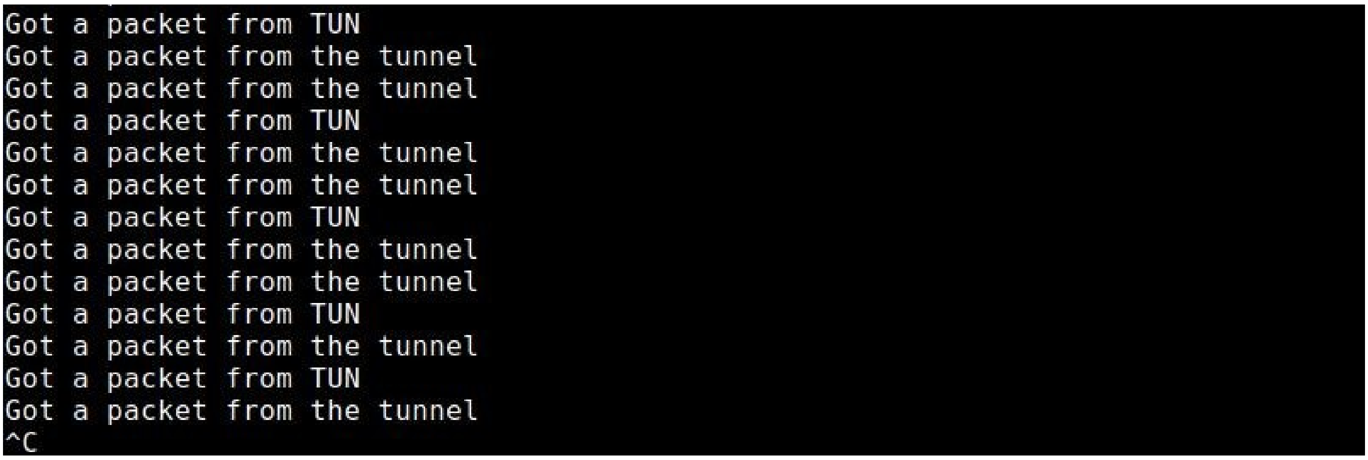
Now when we run ‘ls’ command on the telnet connection, we are able to notice that the new folder create is visible:



```
seed@VM:~/.../vpn$ telnet 192.168.60.101
Trying 192.168.60.101...
Connected to 192.168.60.101.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
VM login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.15.0-130-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

107 updates can be installed immediately.
107 of these updates are security updates.
To see these additional updates run: apt list --upgradable

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Sun Mar 23 20:20:45 IST 2025 on pts/2
seed@VM:~$ ls
Desktop    Downloads           Music     Public  Templates
Documents  hostv-test-folder  Pictures  snap    Videos
seed@VM:~$ ]^Cexit
```

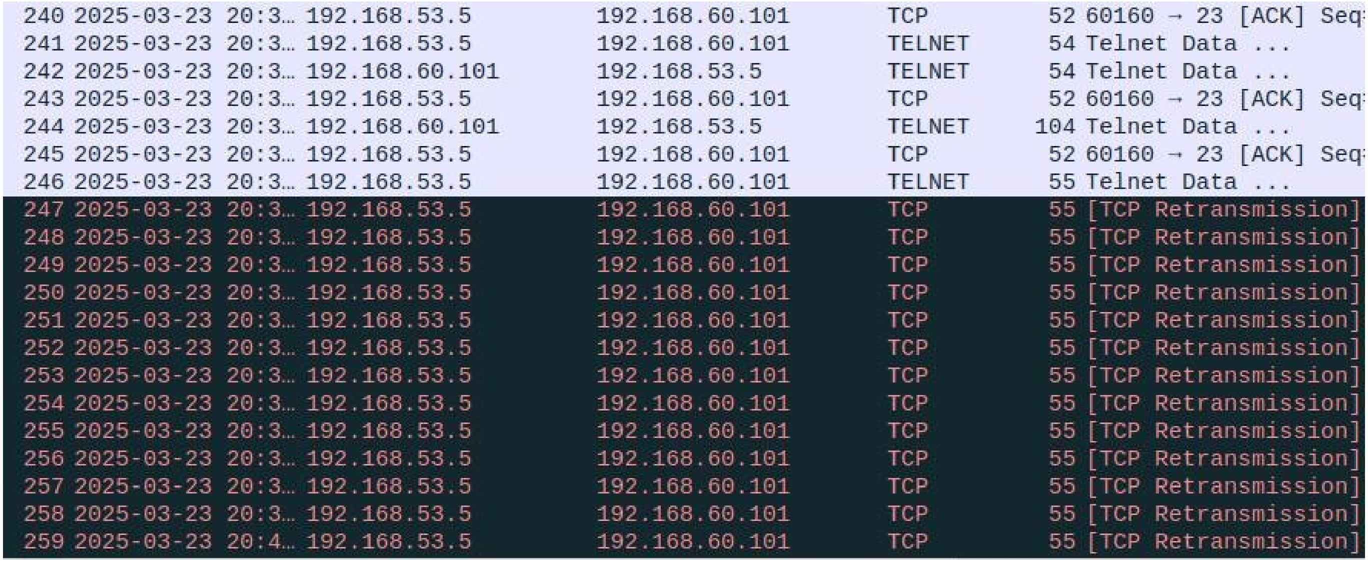
Step 6: Tunnel-Breaking Test

We disconnect the `vpnserver` program to intentionally break the VPN tunnel connection, as shown in the screenshot.



```
Got a packet from TUN
Got a packet from the tunnel
Got a packet from the tunnel
Got a packet from TUN
Got a packet from the tunnel
Got a packet from the tunnel
Got a packet from TUN
Got a packet from the tunnel
Got a packet from the tunnel
Got a packet from TUN
Got a packet from the tunnel
Got a packet from TUN
Got a packet from the tunnel
^C
```

Now, after disconnecting the VPN server, we are unable to execute the `ls` command over the Telnet connection. This confirms that the VPN tunnel was responsible for enabling communication, and without it, the connection is disrupted.



```
240 2025-03-23 20:3… 192.168.53.5      192.168.60.101   TCP         52 60160 → 23 [ACK] Seq
241 2025-03-23 20:3… 192.168.53.5      192.168.60.101   TELNET      54 Telnet Data ...
242 2025-03-23 20:3… 192.168.60.101    192.168.53.5     TELNET      54 Telnet Data ...
243 2025-03-23 20:3… 192.168.53.5      192.168.60.101   TCP         52 60160 → 23 [ACK] Seq
244 2025-03-23 20:3… 192.168.60.101    192.168.53.5     TELNET     104 Telnet Data ...
245 2025-03-23 20:3… 192.168.53.5      192.168.60.101   TCP         52 60160 → 23 [ACK] Seq
246 2025-03-23 20:3… 192.168.53.5      192.168.60.101   TELNET      55 Telnet Data ...
247 2025-03-23 20:3… 192.168.53.5      192.168.60.101   TCP         55 [TCP Retransmission]
248 2025-03-23 20:3… 192.168.53.5      192.168.60.101   TCP         55 [TCP Retransmission]
249 2025-03-23 20:3… 192.168.53.5      192.168.60.101   TCP         55 [TCP Retransmission]
250 2025-03-23 20:3… 192.168.53.5      192.168.60.101   TCP         55 [TCP Retransmission]
251 2025-03-23 20:3… 192.168.53.5      192.168.60.101   TCP         55 [TCP Retransmission]
252 2025-03-23 20:3… 192.168.53.5      192.168.60.101   TCP         55 [TCP Retransmission]
253 2025-03-23 20:3… 192.168.53.5      192.168.60.101   TCP         55 [TCP Retransmission]
254 2025-03-23 20:3… 192.168.53.5      192.168.60.101   TCP         55 [TCP Retransmission]
255 2025-03-23 20:3… 192.168.53.5      192.168.60.101   TCP         55 [TCP Retransmission]
256 2025-03-23 20:3… 192.168.53.5      192.168.60.101   TCP         55 [TCP Retransmission]
257 2025-03-23 20:3… 192.168.53.5      192.168.60.101   TCP         55 [TCP Retransmission]
258 2025-03-23 20:3… 192.168.53.5      192.168.60.101   TCP         55 [TCP Retransmission]
259 2025-03-23 20:4… 192.168.53.5      192.168.60.101   TCP         55 [TCP Retransmission]
```

As observed in the Wireshark screenshot, we are receiving a TCP redirect message, indicating that network traffic is being rerouted or that there is an issue with the established path. This suggests that after disconnecting the VPN, the Telnet connection is no longer able to reach its intended destination.