Lab 6:

Task 1 Screen shots:

Host U can communicate with VPN Server:

Text, letter

Description automatically generated

VPN Server can communicate with Host V:

Text, letter

Description automatically generated

Host U should not be able to communicate with Host V:

Text

Description automatically generated

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

Packets:

I have transmitted two packets from client-server container and captured them successfully:

Text

Description automatically generated

I have transmitted packets from host container and captured them successfully:

Text

Description automatically generated

Task 2.a: Name of the Interface:



Task 2.b: Set up the TUN Interface

Text

Description automatically generated

Task 2.c: Read from the TUN Interface

n Host U, ping a host in the 192.168.53.0/24 network:

Text

Description automatically generated

On Host U, ping a host in the internal network 192.168.60.0/24

Text

Description automatically generated

Task 2.d: Write to the TUN Interface:

After getting a packet from the TUN interface, if this packet is an ICMP echo request packet, construct a corresponding echo reply packet and write it to the TUN interface. Please provide evidence to show that the code works as expected.

Text

Description automatically generated

Instead of writing an IP packet to the interface, write some arbitrary data to the interface, and report your observation:

Code screenshot:

Text

Description automatically generated

Output screenshot:

Text

Description automatically generated with medium confidence

Run the tun server.py program on VPN Server, and then run tun client.py on Host U. To test whether the tunnel works or not, ping any IP address belonging to the 192.168.53.0/24 network:

Running tun\_client program on host u screenshot:

Text

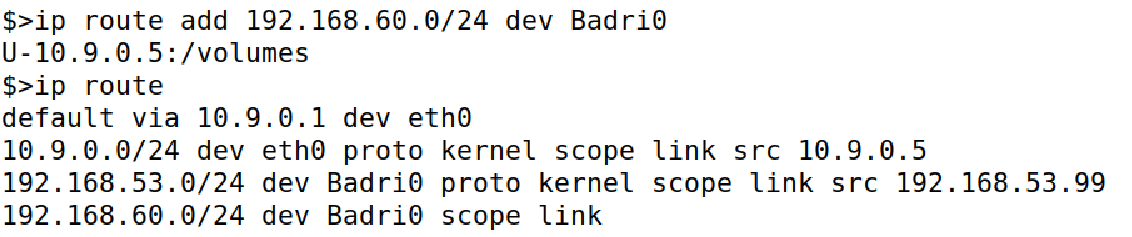
Description automatically generated

Vpn server screenshot:

Text

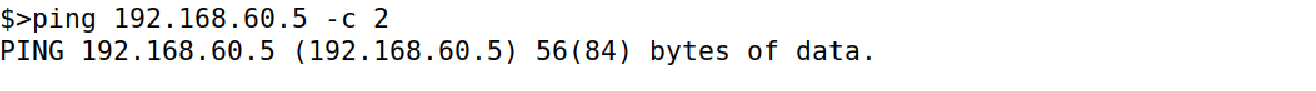
Description automatically generated

Adding ip route:



ping an IP address in the 192.168.60.0/24 network, the ICMP packets are received by tun server.py through the tunnel:

pining ip address:



Tun\_server.py screenshots:

Text

Description automatically generated

\*last two entries are proof that tun\_server received packets when we ping 192.168.60.5

Task 3: Send the IP Packet to VPN Server Through a Tunnel:

Running tun\_server:



Pining 192.168.60.5:

Text

Description automatically generated

it is sufficient to show (using Wireshark or tcpdump) that the ICMP packets have arrived at Host V:

A picture containing text, window

Description automatically generated

Task 5: Handling Traffic in Both Directions:

Code screenshot:

Client.py:

Text

Description automatically generated

Server.py:

Text

Description automatically generated

Once this is done, we should be able to communicate with Machine V from Machine U, and the VPN tunnel (un-encrypted) is now complete. Please show your wireshark proof using about ping and telnet commands.

Trying telent connection if everything is set properly telnet connection should be successful, please look into below screenshot for proof:

Screenshot 1:

Text

Description automatically generated

Screenshot 2:

Text

Description automatically generated

In your proof, you need to point out how your packets flow:

Text, letter

Description automatically generated

Task 6: Tunnel-Breaking Experiment:

Running telnet command:

Text

Description automatically generated

Stopping server program:

Text

Description automatically generated

Checking weather I can type on telnet interface, I’m not to type:



Task 7: Routing Experiment on Host V:

Text

Description automatically generated