Task 1: VM Setup

Host U, on NAT Network as 10.0.2.4:

```
/bin/bash
/bin/bash 81x29
[09/07/21]seed@VM:~$ ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:3a:a0:7e
          inet addr:10.0.2.4  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::3cf9:3483:a5f3:2fb1/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:522 errors:0 dropped:0 overruns:0 frame:0
          TX packets:168 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:127150 (127.1 KB)  TX bytes:18742 (18.7 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:207 errors:0 dropped:0 overruns:0 frame:0
          TX packets:207 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:38264 (38.2 KB)  TX bytes:38264 (38.2 KB)

[09/07/21]seed@VM:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.2.1        0.0.0.0         UG    100    0        0 enp0s3
10.0.2.0        0.0.0.0         255.255.255.0   U     100    0        0 enp0s3
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 enp0s3
[09/07/21]seed@VM:~$
```

VPN Server/Gateway, on NAT Network at 10.0.2.5 and Internal network as 192.168.60.1:

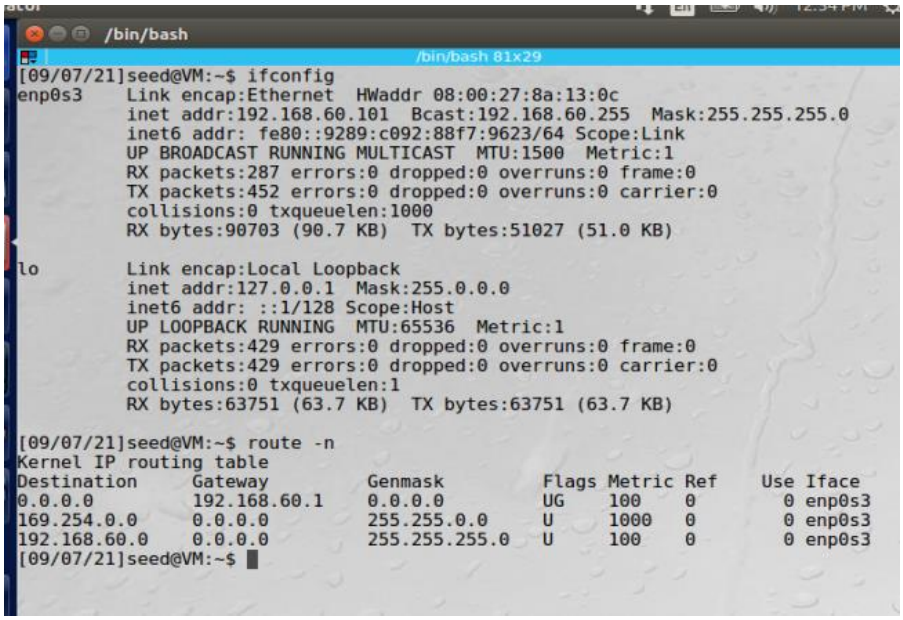
```
/bin/bash
/bin/bash 104x37
[09/07/21]seed@VM:~$ ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:59:41:98
          inet addr:10.0.2.5  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::4f2a:8ff4:445b:655d/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:84 errors:0 dropped:0 overruns:0 frame:0
          TX packets:328 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:8418 (8.4 KB)  TX bytes:36277 (36.2 KB)

enp0s8    Link encap:Ethernet  HWaddr 08:00:27:eb:a7:00
          inet addr:192.168.60.1  Bcast:192.168.60.255  Mask:255.255.255.0
          inet6 addr: fe80::27a8:be68:e97d:25ac/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:217 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:217 (217.0 B)  TX bytes:35327 (35.3 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:354 errors:0 dropped:0 overruns:0 frame:0
          TX packets:354 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:67923 (67.9 KB)  TX bytes:67923 (67.9 KB)

[09/07/21]seed@VM:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.2.1        0.0.0.0         UG    100    0        0 enp0s3
0.0.0.0         192.168.60.1    0.0.0.0         UG    101    0        0 enp0s8
10.0.2.0        0.0.0.0         255.255.255.0   U     100    0        0 enp0s3
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 enp0s3
192.168.60.0    0.0.0.0         255.255.255.0   U     100    0        0 enp0s8
[09/07/21]seed@VM:~$
```

Host V is on internal network only, at 192.168.60.101:



```
/bin/bash
/bin/bash 81x29
[09/07/21]seed@VM:~$ ifconfig
enp0s3    Link encap:Ethernet  HWaddr 08:00:27:8a:13:0c
          inet addr:192.168.60.101  Bcast:192.168.60.255  Mask:255.255.255.0
          inet6 addr: fe80::9289:c092:88f7:9623/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:287 errors:0 dropped:0 overruns:0 frame:0
          TX packets:452 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:90703 (90.7 KB)  TX bytes:51027 (51.0 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:429 errors:0 dropped:0 overruns:0 frame:0
          TX packets:429 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:63751 (63.7 KB)  TX bytes:63751 (63.7 KB)

[09/07/21]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    100    0        0 enp0s3
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 enp0s3
192.168.60.0    0.0.0.0         255.255.255.0   U     100    0        0 enp0s3
[09/07/21]seed@VM:~$
```

Observation: In this task we set up the network in preparation for the VPN tunnel. We have three machines running: Host U (10.0.2.4), VPN Server/Gateway (10.0.2.5/192.168.60.1) and Host V (192.168.60.101). Currently Host U and Host V cannot communicate with each other as seen in the ping messages above.

Explanation: The VPN Server/Gateway machine has two network interfaces configured so that we will be able to connect Host U to Host V after we establish our VPN tunnel. We are simulating that the hosts and the VPN server are connected over the internet by keeping them on separate networks.

Task 2: Creating a VPN Tunnel using TUN/TAP

Before running the client/server applications we must update the IP address of our VPN Server in the program:

```
/bin/bash
/bin/bash 79x29
  GNU nano 2.5.3              File: vpnclient.c

#include <fcntl.h>
#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <arpa/inet.h>
#include <linux/if.h>
#include <linux/if_tun.h>
#include <sys/ioctl.h>

#define BUFF_SIZE 2000
#define PORT_NUMBER 55555
#define SERVER_IP "10.0.2.5"
struct sockaddr_in peerAddr;

int createTunDevice() {
   int tunfd;
   struct ifreq ifr;
   memset(&ifr, 0, sizeof(ifr));

   ifr.ifr_flags = IFF_TUN | IFF_NO_PI;

   tunfd = open("/dev/net/tun", O_RDWR);
   ioctl(tunfd, TUNSETIFF, &ifr);
```

Step 1: Run VPN Server

First, we run the VPN server program on our Server VM, then configure tun0 which will be our VPN interface:

```
/bin/bash 81x14
[09/07/21]seed@VM:~/.../supportingfiles_lab7$ sudo ./vpnserver

/bin/bash 66x12
[09/07/21]seed@VM:~/Desktop$ sudo ifconfig tun0 192.168.53.1/24 up
[09/07/21]seed@VM:~/Desktop$
```

We can see our new tun0 interface in ifconfig:

```
/bin/bash
/bin/bash 81x29
          inet6 addr: fe80::27a8:be68:e97d:25ac/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:262 errors:0 dropped:0 overruns:0 frame:0
          TX packets:257 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:28905 (28.9 KB)  TX bytes:40453 (40.4 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:1145 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1145 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:141894 (141.8 KB)  TX bytes:141894 (141.8 KB)

tun0      Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-
00-00-00-00-00-00
          inet addr:192.168.53.1  P-t-P:192.168.53.1  Mask:255.255
.255.0
          inet6 addr: fe80::6762:8977:f7d3:1686/64 Scope:Link
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric
:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

[09/07/21]seed@VM:~$
```

Next we enable forwarding since our server will act as a gateway:

```
[09/07/21]seed@VM:~/Desktop$ sudo sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
[09/07/21]seed@VM:~/Desktop$
```

Step 2: Run the VPN Client

On Host U, we run the VPN Client program and point it to our server at 10.0.2.5, Then we configure our tun0 interface that will be used in the VPN:

```
/bin/bash
/bin/bash 66x11
[09/07/21]seed@VM:~/Desktop$ sudo ./vpnclient

/bin/bash 66x11
[09/07/21]seed@VM:~$ sudo ifconfig tun0 192.168.53.5/24 up
[09/07/21]seed@VM:~$
```

Step 3: Routing on the Client and Server VMs

In this screenshot from Host U, we first look at the routing table, then we add the route to 192.168.60.0/24 network via our tun0 interface. We again look at our routing table and see the new route:

```
/bin/bash 81x29
3;J
[09/07/21]seed@VM:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref
  Use Iface
0.0.0.0         10.0.2.1        0.0.0.0         UG    100    0
    0 enp0s3
10.0.2.0        0.0.0.0         255.255.255.0   U     100    0
    0 enp0s3
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0
    0 enp0s3
192.168.53.0    0.0.0.0         255.255.255.0   U     0      0
    0 tun0
[09/07/21]seed@VM:~$ sudo route add -net 192.168.60.0/24 tun0
[09/07/21]seed@VM:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref
  Use Iface
0.0.0.0         10.0.2.1        0.0.0.0         UG    100    0
    0 enp0s3
10.0.2.0        0.0.0.0         255.255.255.0   U     100    0
    0 enp0s3
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0
    0 enp0s3
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     0      0
    0 tun0
[09/07/21]seed@VM:~$
```

We can also see on our sever that our route is set up for the 192.168.60.0/24 network:

```
[09/07/21]seed@VM:~/Desktop$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.2.1        0.0.0.0         UG    100    0        0 enp0s3
0.0.0.0         192.168.60.1    0.0.0.0         UG    101    0        0 enp0s8
10.0.2.0        0.0.0.0         255.255.255.0   U     100    0        0 enp0s3
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 enp0s3
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     100    0        0 enp0s8
[09/07/21]seed@VM:~/Desktop$
```

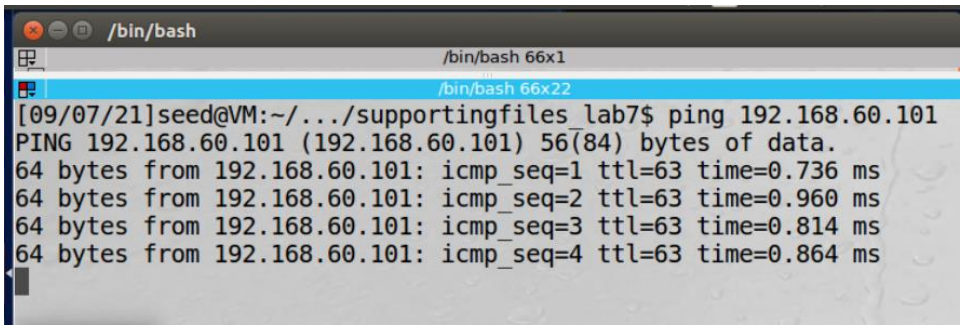
Step 4: Set Up Routing Host V

In the screenshot below we first see the Host V routing table, then we add the route for our VPN, 192.168.53.0/24 network:

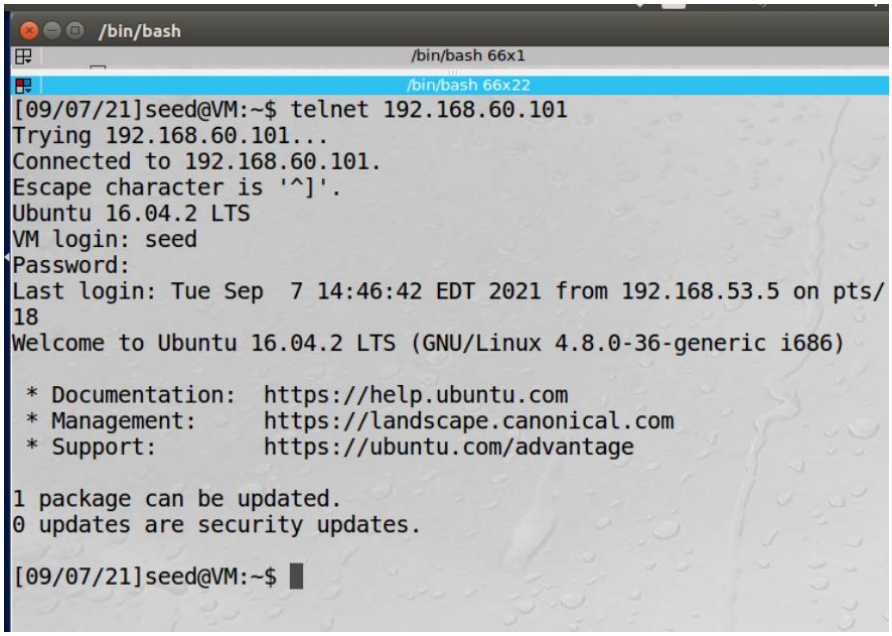
```
/bin/bash 81x29
[09/07/21]seed@VM:~$ route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.60.1    0.0.0.0         UG    100    0        0 enp0s3
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 enp0s3
192.168.60.0    0.0.0.0         255.255.255.0   U     100    0        0 enp0s3
[09/07/21]seed@VM:~$ sudo route add -net 192.168.53.0/24 gw 192.168.60.1 enp0s3
[09/07/21]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    100    0        0 enp0s3
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 enp0s3
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
[09/07/21]seed@VM:~$
```

Step 5: Test the VPN

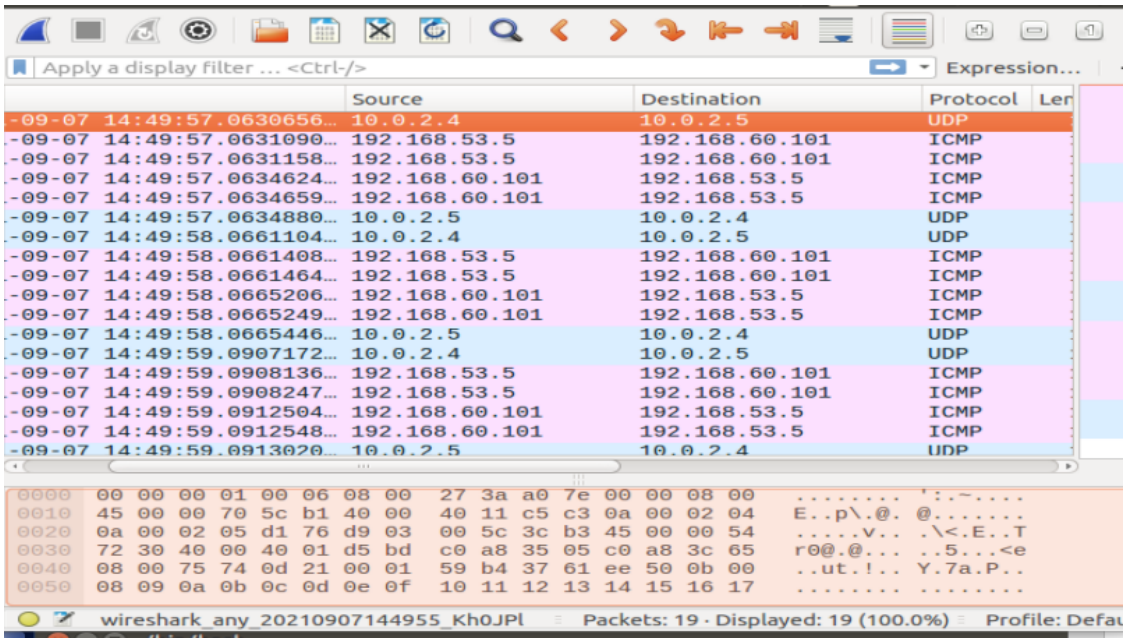
Now from Host U we test the connection by pinging Host V, and see we are connected:



And now we can connect via telnet as well:

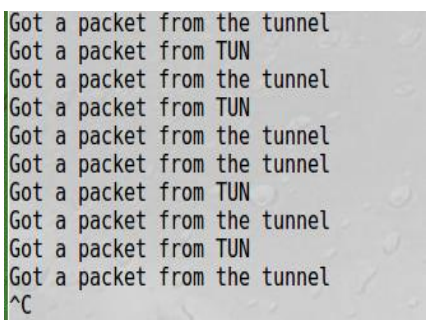


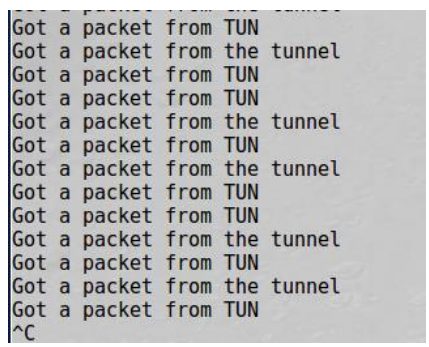
Looking at Wireshark, we can see that traffic routed between Host U and Host V uses the 192.168.53.0 VPN source, and the other traffic like when I ping 8.8.8.8 doesn't go through the VPN interface:



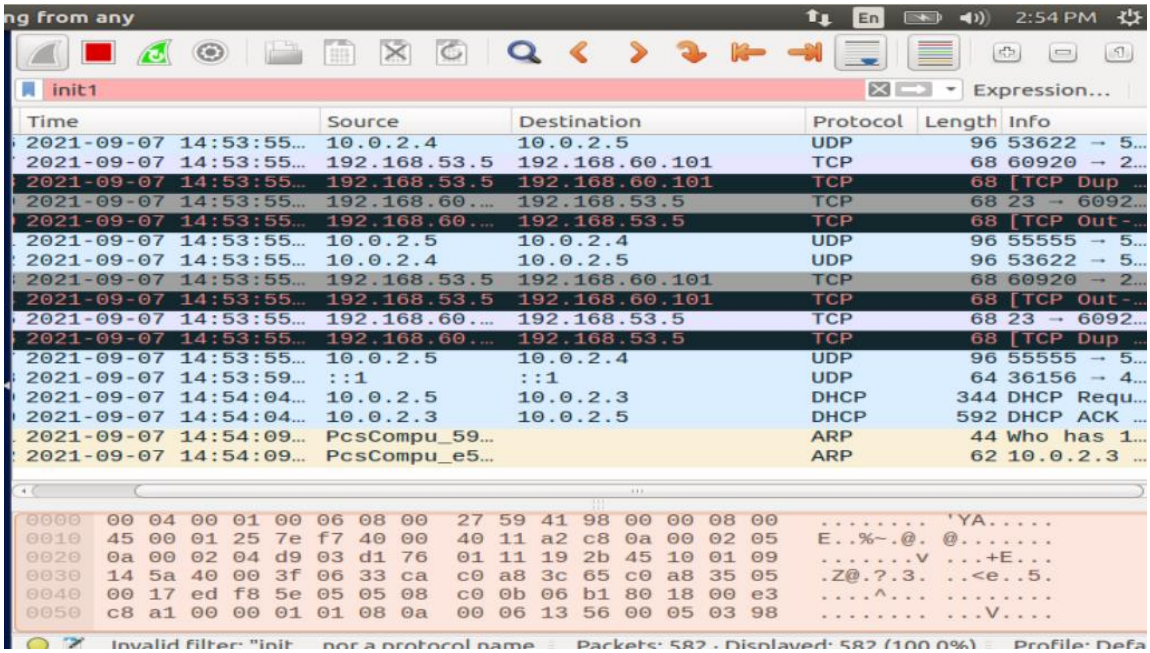
Step 6: Tunnel-Breaking Test

While Telnet is still active, we kill our client and server VPN programs:





After breaking the tunnel, we attempt to type commands in the Telent window, and we can see that the packets are not delivered but being put in a buffer:



Explanation: In this task we setup our VPN tunnel, tested it, broke the tunnel, and reestablished it, allowing us to communicate between Host U and Host V. We started by configuring our Server, we first run the VPN server program, then we configure the tun0 interface, and allow traffic forwarding, so that it can act as a gateway. The second step was to run the VPN Client program on Host U, and again we had to configure traffic destined to the 192.168.60.0/24 network to use our tun0 interface. Next we configure the Host V; we need to add the route so when it responds to Host U it knows to go through the VPN server. Then we can ping and telnet from Host U to Host V. As an experiment we tried to break the VPN to observe what happened with telnet packets. We saw that they were buffered and when we reestablished the VPN the packets that were buffered were sent.