

Task 1: VM Setup

The following is the Network Setup for this lab:

| Name | Role | IP Address | MAC Address |
|-------------|--------------------|----------------|-------------------|
| SEEDUbuntu | VPN Client/Host U | 10.0.2.7 | 08:00:27:b7:ba:af |
| SEEDUbuntu1 | Gateway/VPN Server | 10.0.2.8 | 08:00:27:cd:2d:fd |
| | | 192.168.60.1 | 08:00:27:50:12:67 |
| SEEDUbuntu2 | Host V | 192.168.60.101 | 08:00:27:98:60:5e |

The following shows the configuration on Host U/VPN Client:

```

[03/26/20]seed@VM:~$ ifconfig
enp0s3  Link encap:Ethernet  HWaddr 08:00:27:b7:ba:af
        inet addr:10.0.2.7  Bcast:10.0.2.255  Mask:255.255.255.0
        inet6 addr: fe80::4fd4:7bb8:663f:1798/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:58 errors:0 dropped:0 overruns:0 frame:0
        TX packets:65 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:9085 (9.0 KB)  TX bytes:7398 (7.3 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:67 errors:0 dropped:0 overruns:0 frame:0
        TX packets:67 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1
        RX bytes:21416 (21.4 KB)  TX bytes:21416 (21.4 KB)

```

The following shows the configuration on Host V:

```

[03/26/20]seed@VM:~$ ifconfig
enp0s3  Link encap:Ethernet  HWaddr 08:00:27:98:60:5e
        inet addr:192.168.60.101  Bcast:192.168.60.255  Mask:255.255.255.0
        inet6 addr: fe80::7424:492b:395a:416f/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:198 errors:0 dropped:0 overruns:0 frame:0
        TX packets:551 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:46584 (46.5 KB)  TX bytes:88900 (88.9 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:147 errors:0 dropped:0 overruns:0 frame:0
        TX packets:147 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1
        RX bytes:16826 (16.8 KB)  TX bytes:16826 (16.8 KB)

```

The following shows the configuration on the VPN Server:

```
Terminal
[03/26/20]seed@VM:~$ ifconfig
enp0s3  Link encap:Ethernet  HWaddr 08:00:27:cd:2d:fd
        inet addr:10.0.2.8  Bcast:10.0.2.255  Mask:255.255.255.0
        inet6 addr: fe80::3928:8afb:c6e0:2cd8/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:197 errors:0 dropped:0 overruns:0 frame:0
        TX packets:293 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:68271 (68.2 KB)  TX bytes:28387 (28.3 KB)

enp0s8  Link encap:Ethernet  HWaddr 08:00:27:50:12:67
        inet addr:192.168.60.1  Bcast:192.168.60.255  Mask:255.255.255.0
        inet6 addr: fe80::aaf4:3cb1:8dec:fdb2/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:212 errors:0 dropped:0 overruns:0 frame:0
        TX packets:393 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:44426 (44.4 KB)  TX bytes:67875 (67.8 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:254 errors:0 dropped:0 overruns:0 frame:0
        TX packets:254 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1
        RX bytes:33997 (33.9 KB)  TX bytes:33997 (33.9 KB)
```

Task 2: Creating a VPN Tunnel using TUN/TAP

Step 1: Run VPN Server

We first run the VPN server program `vpnserv` on the Server VM as follows:

```
Terminal
[03/26/20]seed@VM:~/.../vpn$ ls
Makefile  README  vpnclient.c  vpnserv.c
[03/26/20]seed@VM:~/.../vpn$ gcc -o vpnserv vpnserv.c
[03/26/20]seed@VM:~/.../vpn$ sudo ./vpnserv
```

After the program runs, a virtual TUN network interface is created on the system. This new interface is not yet configured, so we configure it by giving it an IP address and activating it. We use 192.168.53.1 for this interface. Since the VPN Server needs to forward packets between the private network and the tunnel, it also needs to function as a gateway, and this is achieved by enabling IP forwarding on the VPN Server. These steps are achieved as following:

```

[03/26/20]seed@VM:~$ sudo ifconfig tun0 192.168.53.1/24 up
[03/26/20]seed@VM:~$ ifconfig -a
enp0s3      Link encap:Ethernet  HWaddr 08:00:27:cd:2d:fd
            inet addr:10.0.2.8  Bcast:10.0.2.255  Mask:255.255.255.0
            inet6 addr: fe80::3928:8afb:c6e0:2cd8/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
            RX packets:449 errors:0 dropped:0 overruns:0 frame:0
            TX packets:574 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:130690 (130.6 KB)  TX bytes:50472 (50.4 KB)

enp0s8      Link encap:Ethernet  HWaddr 08:00:27:50:12:67
            inet addr:192.168.60.1  Bcast:192.168.60.255  Mask:255.255.255.0
            inet6 addr: fe80::aaf4:3cb1:8dec:fdb2/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
            RX packets:401 errors:0 dropped:0 overruns:0 frame:0
            TX packets:434 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:77174 (77.1 KB)  TX bytes:74353 (74.3 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:375 errors:0 dropped:0 overruns:0 frame:0
            TX packets:375 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1
            RX bytes:41465 (41.4 KB)  TX bytes:41465 (41.4 KB)

tun0
-00         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::7669:409f:7565:1029/64 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)

[03/26/20]seed@VM:~$ sudo sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
[03/26/20]seed@VM:~$ █

```

Step 2: Run VPN Client

After running the VPN Server program, we run the VPN client program on the Client VM. This program also creates a TUN interface and we configure the tun0 interface by assigning it an IP address of 192.168.53.5. The following shows the successful configuration:

```

Terminal
[03/26/20]seed@VM:~$ sudo ifconfig tun0 192.168.53.5/24 up
[03/26/20]seed@VM:~$ ifconfig -a
enp0s3      Link encap:Ethernet  HWaddr 08:00:27:b7:ba:af
            inet addr:10.0.2.7  Bcast:10.0.2.255  Mask:255.255.255.0
            inet6 addr: fe80::4fd4:7bb8:663f:1798/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
            RX packets:253 errors:0 dropped:0 overruns:0 frame:0
            TX packets:315 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:77309 (77.3 KB)  TX bytes:31283 (31.2 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:374 errors:0 dropped:0 overruns:0 frame:0
            TX packets:374 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1
            RX bytes:39987 (39.9 KB)  TX bytes:39987 (39.9 KB)

tun0
-00         Link encap:UNSPEC  HWaddr 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
            inet addr:192.168.53.5  P-t-P:192.168.53.5  Mask:255.255.255.0
            inet6 addr: fe80::e0ff:504f:1c9c:db67/64 Scope:Link
            UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
            RX packets:0 errors:0 dropped:0 overruns:0 frame:0
            TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:500
            RX bytes:0 (0.0 B)  TX bytes:144 (144.0 B)

[03/26/20]seed@VM:~$

```

Step 3: Set Up Routing on Client and Server VMs

After the above two steps, the tunnel is established. We then set up routing paths to direct the intended traffic to the tunnel on the client and server machine. On the client machine, we direct all the packets going to the private network (192.168.60.0/24) towards the tun0 interface, from where the packets will be forwarded through the VPN tunnel. We use the route command to add the following routing entry on the Client VM:

```

[03/26/20]seed@VM:~$ sudo route add -net 192.168.60.0/24 tun0
[03/26/20]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
[03/26/20]seed@VM:~$

```

Also, on both client and server machines, we ensure a routing entry that directs the traffic going to the 192.168.53.0/24 network towards the tun0 interface. The following shows the routing table at the VM Server:

```

Terminal
[03/26/20]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
192.168.53.0     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
[03/26/20]seed@VM:~$

```

Step 4: Set Up Routing on Host V

Now, in order to send the Host V replies to Host U through the VPN tunnel, we add a routing entry on Host V that routes the packets going to Host U's network to the VPN Server. From the VPN Server, this packet will then go through the VPN tunnel to the VPN Client and eventually to Host U. The following shows the route entry on Host V:

```

Terminal
[03/26/20]seed@VM:~$ sudo route add -net 192.168.53.0/24 gw 192.168.60.1 enp0s3
[03/26/20]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
10.0.2.0         0.0.0.0         255.255.255.0   U      0      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
[03/26/20]seed@VM:~$

```

Step 5: Test the VPN Tunnel

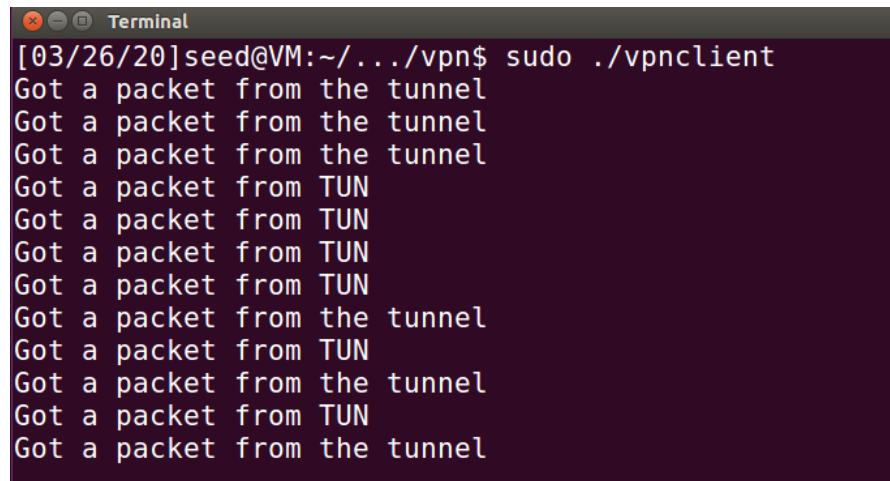
Now, in order to test the access from Host U to Host V via the tunnel, we run the “ping 192.168.60.101” command on Host U. The following shows that the ping is successful:

```

Terminal
[03/26/20]seed@VM:~$ 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=1 ttl=63 time=1.83 ms
64 bytes from 192.168.60.101: icmp_seq=2 ttl=63 time=1.21 ms
64 bytes from 192.168.60.101: icmp_seq=3 ttl=63 time=1.61 ms
^C
--- 192.168.60.101 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 1.214/1.554/1.830/0.257 ms

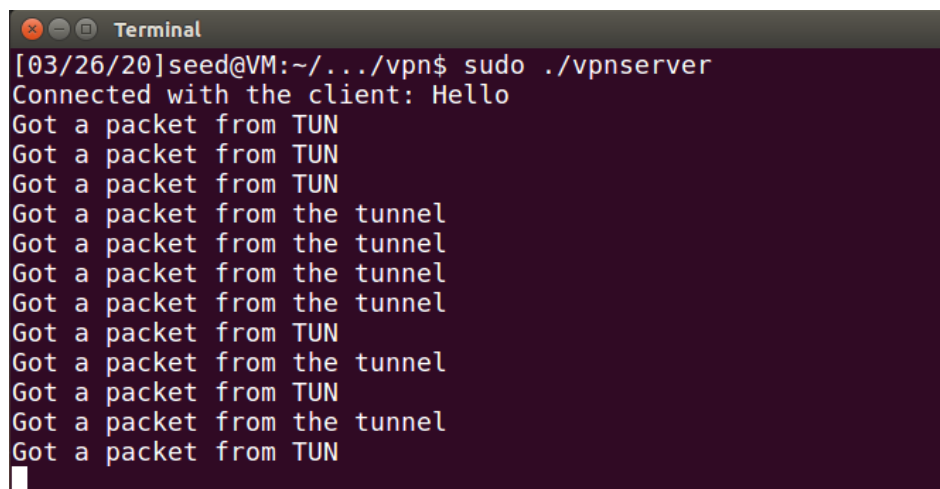
```


On sending a ping request, the client program reflects that it has received a packet on the TUN interface (ping request) and consecutively a packet from the tunnel (ping reply). The first 6 lines should be ignored as they appear while changing the tun0 interface's configuration.

A terminal window titled "Terminal" showing the output of the command `sudo ./vpncclient`. The output consists of 14 lines: the first 6 lines are "Got a packet from the tunnel", and the next 8 lines are "Got a packet from TUN".

```
[03/26/20]seed@VM:~/../vpn$ sudo ./vpncclient
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
Got a packet from TUN
Got a packet from TUN
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
```

On the VPN Server, the program reflects receiving a packet from the tunnel (ping request) and then a packet from TUN (ping reply).

A terminal window titled "Terminal" showing the output of the command `sudo ./vpnsrvr`. The output starts with "Connected with the client: Hello", followed by 12 lines: 6 "Got a packet from TUN" and 6 "Got a packet from the tunnel".

```
[03/26/20]seed@VM:~/../vpn$ sudo ./vpnsrvr
Connected with the client: Hello
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
Got a packet from the tunnel
Got a packet from TUN
Got a packet from the tunnel
Got a packet from TUN
```

The ping request comes from the Host U and the ping reply comes from Host V. A packet from the tunnel means a packet from the internet to the program (socket), and from TUN means a packet from the Host U or Host V to the VPN Client or Server, respectively. Also, on looking at the captured Wireshark traffic, we see that the first 4 packets are a part of a single ping (ping request to reply) and packets 2 and 3 are a part of the tunnel between VPN Client and Server, whereas packet 1 and 4 are the data of packets 2 and 3 respectively (the data part of the packet 2 starts with 45, confirming it as an IP packet).

On seeing a ping for 192.168.60.0/24 network, the Client machine routes the traffic to tun0 interface – 192.168.53.5. Hence, we see the tun0 interface's IP as source IP. Then, the running client program takes this packet from the tun interface and writes it to the socket interface that it

has started with the server. The server receives this packet and routes it to the private network to reach Host V. Host V then responds to the ping, and it reaches the VPN Client program from the Server through the established tunnel. The client program then re-routes it to the tun interface. This completes the entire ping from Host U to Host V via the VPN tunnel.

| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|-------------------------------|----------------|----------------|----------|--------|--|
| 1 | 2020-03-26 18:32:58.447778348 | 192.168.53.5 | 192.168.60.101 | ICMP | 100 | Echo (ping) request id=0x1130, seq=1/256, ttl=64 |
| 2 | 2020-03-26 18:32:58.447808054 | 10.0.2.7 | 10.0.2.8 | UDP | 128 | 52557 → 55555 Len=84 |
| 3 | 2020-03-26 18:32:58.449343454 | 10.0.2.8 | 10.0.2.7 | UDP | 128 | 55555 → 52557 Len=84 |
| 4 | 2020-03-26 18:32:58.449597586 | 192.168.60.101 | 192.168.53.5 | ICMP | 100 | Echo (ping) reply id=0x1130, seq=1/256, ttl=63 |
| 5 | 2020-03-26 18:32:58.449653862 | ::1 | ::1 | UDP | 64 | 48833 → 57065 Len=0 |
| 6 | 2020-03-26 18:32:59.449146215 | 192.168.53.5 | 192.168.60.101 | ICMP | 100 | Echo (ping) request id=0x1130, seq=2/512, ttl=64 |
| 7 | 2020-03-26 18:32:59.449192678 | 10.0.2.7 | 10.0.2.8 | UDP | 128 | 52557 → 55555 Len=84 |
| 8 | 2020-03-26 18:32:59.450263563 | 10.0.2.8 | 10.0.2.7 | UDP | 128 | 55555 → 52557 Len=84 |
| 9 | 2020-03-26 18:32:59.450340226 | 192.168.60.101 | 192.168.53.5 | ICMP | 100 | Echo (ping) reply id=0x1130, seq=2/512, ttl=63 |
| 10 | 2020-03-26 18:33:00.450507703 | 192.168.53.5 | 192.168.60.101 | ICMP | 100 | Echo (ping) request id=0x1130, seq=3/768, ttl=64 |
| 11 | 2020-03-26 18:33:00.450677450 | 10.0.2.7 | 10.0.2.8 | UDP | 128 | 52557 → 55555 Len=84 |
| 12 | 2020-03-26 18:33:00.452000969 | 10.0.2.8 | 10.0.2.7 | UDP | 128 | 55555 → 52557 Len=84 |
| 13 | 2020-03-26 18:33:00.452102468 | 192.168.60.101 | 192.168.53.5 | ICMP | 100 | Echo (ping) reply id=0x1130, seq=3/768, ttl=63 |

▶ Frame 2: 128 bytes on wire (1024 bits), 128 bytes captured (1024 bits) on interface 0
 ▶ Linux cooked capture
 ▶ Internet Protocol Version 4, Src: 10.0.2.7, Dst: 10.0.2.8
 ▶ User Datagram Protocol, Src Port: 52557, Dst Port: 55555
 ▼ Data (84 bytes)
 Data: 45000054a83f400040019faec0a83505c0a83c650000c66a...
 [Length: 84]

In a similar manner, a telnet is established between Host U and V:

```

Terminal
[03/26/20]seed@VM:~$ telnet 192.168.60.101
Trying 192.168.60.101...
Connected to 192.168.60.101.
Escape character is '^]'.
Ubuntu 16.04.2 LTS
VM login: seed
Password:
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.8.0-36-generic i686)

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

1 package can be updated.
0 updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

[03/26/20]seed@VM:~$

```

A similar approach is taken by telnet with the only difference being that the packet 1 and 4 now uses tcp/telnet protocol instead of icmp protocol. Packet 2 and 3 still remain the tunnel traffic and packet 1 and 4 are the data part of packet 2 and 3, respectively. The following show the same:

| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|-------------------------------|----------------|----------------|----------|--------|--|
| 1 | 2020-03-26 18:36:42.815900974 | 192.168.53.5 | 192.168.60.101 | TCP | 76 | 35828 → 23 [SYN] Seq=203408848 Win=29200 Len=0 MSS=1460 ... |
| 2 | 2020-03-26 18:36:42.815927398 | 10.0.2.7 | 10.0.2.8 | UDP | 104 | 52557 → 55555 Len=60 |
| 3 | 2020-03-26 18:36:42.816897749 | 10.0.2.8 | 10.0.2.7 | UDP | 104 | 55555 → 52557 Len=60 |
| 4 | 2020-03-26 18:36:42.816967876 | 192.168.60.101 | 192.168.53.5 | TCP | 76 | 23 → 35828 [SYN, ACK] Seq=1314295262 Ack=203408849 Win=2 ... |
| 5 | 2020-03-26 18:36:42.816987044 | 192.168.53.5 | 192.168.60.101 | TCP | 68 | 35828 → 23 [ACK] Seq=203408849 Ack=1314295263 Win=29312 ... |
| 6 | 2020-03-26 18:36:42.817000455 | 10.0.2.7 | 10.0.2.8 | UDP | 96 | 52557 → 55555 Len=52 |
| 7 | 2020-03-26 18:36:42.817355663 | 192.168.53.5 | 192.168.60.101 | TELNET | 95 | Telnet Data ... |
| 8 | 2020-03-26 18:36:42.817387014 | 10.0.2.7 | 10.0.2.8 | UDP | 123 | 52557 → 55555 Len=79 |
| 9 | 2020-03-26 18:36:42.818330589 | 10.0.2.8 | 10.0.2.7 | UDP | 96 | 55555 → 52557 Len=52 |
| 10 | 2020-03-26 18:36:42.818423270 | 192.168.60.101 | 192.168.53.5 | TCP | 68 | 23 → 35828 [ACK] Seq=1314295263 Ack=203408876 Win=29056 ... |
| 11 | 2020-03-26 18:36:42.875853732 | 10.0.2.8 | 10.0.2.7 | UDP | 108 | 55555 → 52557 Len=64 |
| 12 | 2020-03-26 18:36:42.875970159 | 192.168.60.101 | 192.168.53.5 | TELNET | 80 | Telnet Data ... |
| 13 | 2020-03-26 18:36:42.875987802 | 192.168.53.5 | 192.168.60.101 | TCP | 68 | 35828 → 23 [ACK] Seq=203408876 Ack=1314295275 Win=29312 ... |

▶ Frame 2: 104 bytes on wire (832 bits), 104 bytes captured (832 bits) on interface 0

▶ Linux cooked capture

▶ Internet Protocol Version 4, Src: 10.0.2.7, Dst: 10.0.2.8

▶ User Datagram Protocol, Src Port: 52557, Dst Port: 55555

▼ Data (60 bytes)

Data: 4510003cc12c4000400606c4c0a83505c0a83c658bf40017...

[Length: 60]

Step 6: Tunnel-Breaking Test

With the already established telnet connection from the previous task, we break the VPN tunnel by closing the vpnserver program. After doing that, we type characters in the telnet program and see that nothing is visible. Immediately the telnet program becomes unresponsive. On looking at the Wireshark trace, we see that packets are sent out, however since the server port is no more open and connected, the ICMP replies with an error message of Destination port unreachable.

| No. | Time | Source | Destination | Protocol | Length | Info |
|-----|-------------------------------|--------------|----------------|----------|--------|---|
| 1 | 2020-03-26 18:40:48.543487673 | 192.168.53.5 | 192.168.60.101 | TELNET | 69 | Telnet Data ... |
| 2 | 2020-03-26 18:40:48.543535959 | 10.0.2.7 | 10.0.2.8 | UDP | 97 | 52557 → 55555 Len=53 |
| 3 | 2020-03-26 18:40:48.544410324 | 10.0.2.8 | 10.0.2.7 | ICMP | 125 | Destination unreachable (Port unreachable) |
| 4 | 2020-03-26 18:40:48.544513546 | :::1 | :::1 | UDP | 64 | 48833 → 57065 Len=0 |
| 5 | 2020-03-26 18:40:48.755792449 | 192.168.53.5 | 192.168.60.101 | TCP | 69 | [TCP Keep-Alive] 35828 → 23 [PSH, ACK] Seq=203409042 Ack=1314 ... |
| 6 | 2020-03-26 18:40:48.758755057 | 10.0.2.7 | 10.0.2.8 | UDP | 97 | 52557 → 55555 Len=53 |
| 7 | 2020-03-26 18:40:48.759223137 | 10.0.2.8 | 10.0.2.7 | ICMP | 125 | Destination unreachable (Port unreachable) |
| 8 | 2020-03-26 18:40:49.055468357 | 192.168.53.5 | 192.168.60.101 | TCP | 69 | [TCP Keep-Alive] 35828 → 23 [PSH, ACK] Seq=203409042 Ack=1314 ... |
| 9 | 2020-03-26 18:40:49.056704152 | 10.0.2.7 | 10.0.2.8 | UDP | 97 | 52557 → 55555 Len=53 |
| 10 | 2020-03-26 18:40:49.058448629 | 10.0.2.8 | 10.0.2.7 | ICMP | 125 | Destination unreachable (Port unreachable) |
| 11 | 2020-03-26 18:40:49.485562906 | 192.168.53.5 | 192.168.60.101 | TCP | 69 | [TCP Keep-Alive] 35828 → 23 [PSH, ACK] Seq=203409042 Ack=1314 ... |
| 12 | 2020-03-26 18:40:49.485640961 | 10.0.2.7 | 10.0.2.8 | UDP | 97 | 52557 → 55555 Len=53 |
| 13 | 2020-03-26 18:40:49.486084629 | 10.0.2.8 | 10.0.2.7 | ICMP | 125 | Destination unreachable (Port unreachable) |

▶ Frame 1: 69 bytes on wire (552 bits), 69 bytes captured (552 bits) on interface 0

▶ Linux cooked capture

▶ Internet Protocol Version 4, Src: 192.168.53.5, Dst: 192.168.60.101

▶ Transmission Control Protocol, Src Port: 35828, Dst Port: 23, Seq: 203409042, Ack: 1314297056, Len: 1

▼ Telnet

Data: 1

This is plausible because the port closes when the server program closes, and the tunnel is now broken. We then start the vpnserver program again and configure the tun0 interface as before. After we do that, we see that VPN server program starts receiving packets from the tunnel and the tun interface. This is because, all the random characters typed on the telnet client are not lost but buffered. Only when the telnet server receives the typed character, it responds with the character and that is displayed on the telnet client machine. Since, the server did not receive anything, the client went blank.

```
[03/26/20]seed@VM:~/.../vpn$ sudo ./vpnserver
Connected with the client: E[0]
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 TUN
Got a packet from the tunnel
Got a packet from TUN
Got a packet from the tunnel
```


After reconnection, the buffered characters eventually reached the server and were reflected on the client side. This can be seen as follows:

```

Terminal
/home/seed
[03/26/20]seed@VM:~$ ps
  PID TTY          TIME CMD
 3303 pts/17    00:00:00 bash
 3385 pts/17    00:00:00 ps
[03/26/20]seed@VM:~$ ifconfig
enp0s3  Link encap:Ethernet  HWaddr 08:00:27:98:60:5e
        inet addr:192.168.60.101  Bcast:192.168.60.255  Mask:255.255.255.0
        inet6 addr: fe80::7424:492b:395a:416f/64  Scope:Link
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:301 errors:0 dropped:0 overruns:0 frame:0
        TX packets:669 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:55532 (55.5 KB)  TX bytes:99746 (99.7 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:491 errors:0 dropped:0 overruns:0 frame:0
        TX packets:491 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1
        RX bytes:33338 (33.3 KB)  TX bytes:33338 (33.3 KB)

[03/26/20]seed@VM:~$ ipiifcofn

```

Hence, due to TCP retransmission, even though the tunnel broke for a moment there, the data was still transmitted once the connection resumed. The following show the Wireshark trace:

| | | | | | |
|----|-------------------------------|----------------|----------------|--------|---|
| 64 | 2020-03-26 18:44:26.706600399 | 192.168.53.5 | 192.168.60.101 | TCP | 69 [TCP Keep-Alive] 35828 → 23 [PSH, ACK] Seq=203409042 Ack=... |
| 65 | 2020-03-26 18:44:26.706688840 | 10.0.2.7 | 10.0.2.8 | UDP | 97 52557 → 55555 Len=53 |
| 66 | 2020-03-26 18:44:26.703020443 | 10.0.2.8 | 10.0.2.7 | UDP | 97 55555 → 52557 Len=53 |
| 67 | 2020-03-26 18:44:26.703137502 | 192.168.60.101 | 192.168.53.5 | TELNET | 69 Telnet Data ... |
| 68 | 2020-03-26 18:44:26.703159366 | 192.168.53.5 | 192.168.60.101 | TELNET | 78 Telnet Data ... |
| 69 | 2020-03-26 18:44:26.703180993 | 10.0.2.7 | 10.0.2.8 | UDP | 106 52557 → 55555 Len=62 |
| 70 | 2020-03-26 18:44:26.704546791 | 10.0.2.8 | 10.0.2.7 | UDP | 106 55555 → 52557 Len=62 |
| 71 | 2020-03-26 18:44:26.704643144 | 192.168.60.101 | 192.168.53.5 | TELNET | 78 Telnet Data ... |
| 72 | 2020-03-26 18:44:26.748692223 | 192.168.53.5 | 192.168.60.101 | TCP | 68 35828 → 23 [ACK] Seq=203409053 Ack=1314297067 Win=266 Le... |
| 73 | 2020-03-26 18:44:26.748764998 | 10.0.2.7 | 10.0.2.8 | UDP | 96 52557 → 55555 Len=52 |

▶ Frame 68: 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on interface 0
 ▶ Linux cooked capture
 ▶ Internet Protocol Version 4, Src: 192.168.53.5, Dst: 192.168.60.101
 ▶ Transmission Control Protocol, Src Port: 35828, Dst Port: 23, Seq: 203409043, Ack: 1314297057, Len: 10
 ▼ Telnet
 Data: piifcofn