50.020 Network Security Lab 8 | Wong Chi Seng 1002853

Setup VMs:

Server: 10.0.2.9

Client: 10.0.2.15

VPN Client/Server Setup:

First we have to ensure that firewalls are disabled before running the programs. Compile the programs with the IP of the Server in the SERVER IP field. Upon successful connection, a “hello” message will be printed along with packets containing handshake messages through the tunnel.

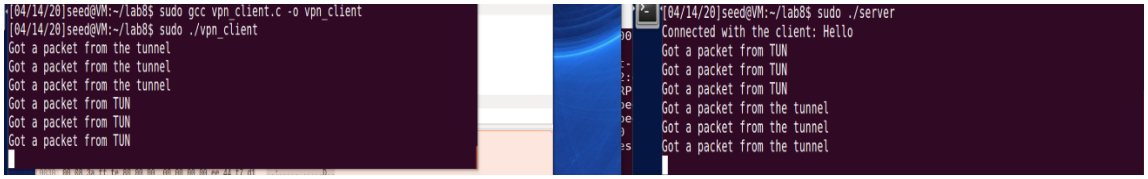

Figure 1 successful setup

The IP address of both client and server should have their new IP addresses assigned to them. The server having 192.168.53.1 and the client having 192.168.53.5.

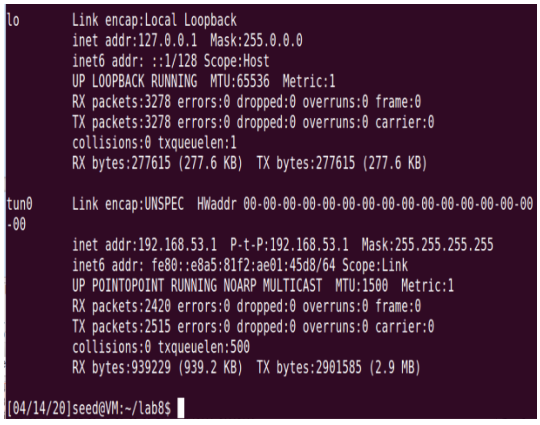

Figure 2 server

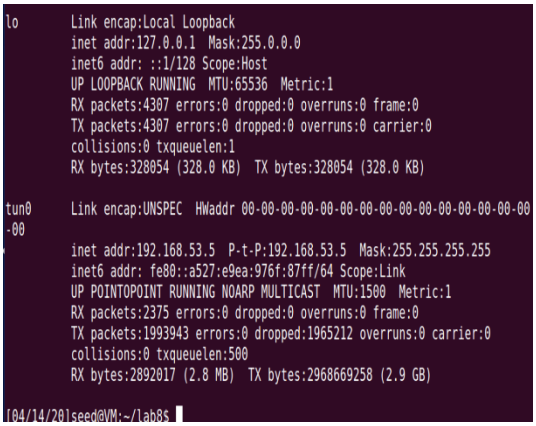

Figure 3 client

Routing

We add a route in both the client and server machines to route packets to the virtual IPs to the default gateway socket so that they can pass through the tunnel.

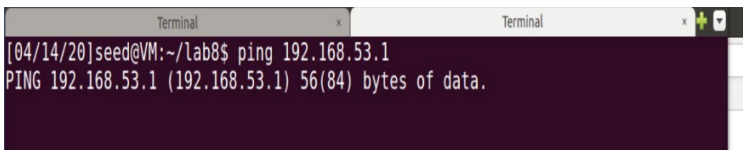

Figure 4 pre routing pinging

The screenshot below shows the routing table on the client's side after adding the route for the virtual IP addresses.

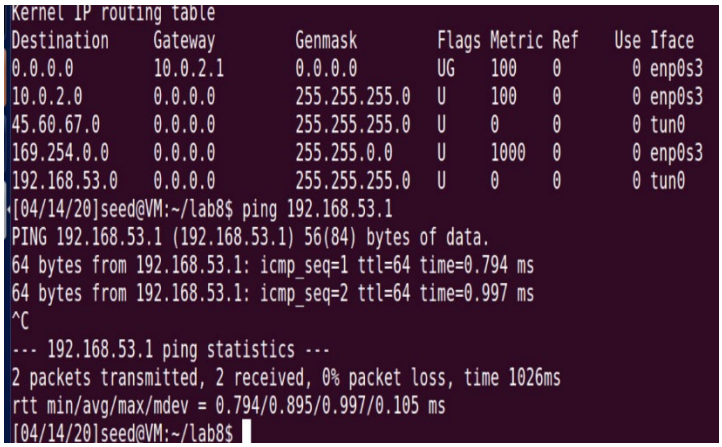

Figure 5 post routing pinging

Firewall configuration

For this task, I've chosen facebook's website as a target for blocking, taking note that it frequents between 157.240.7.0 and 157.240.13.0. We can obtain the IP addresses through a dig command.

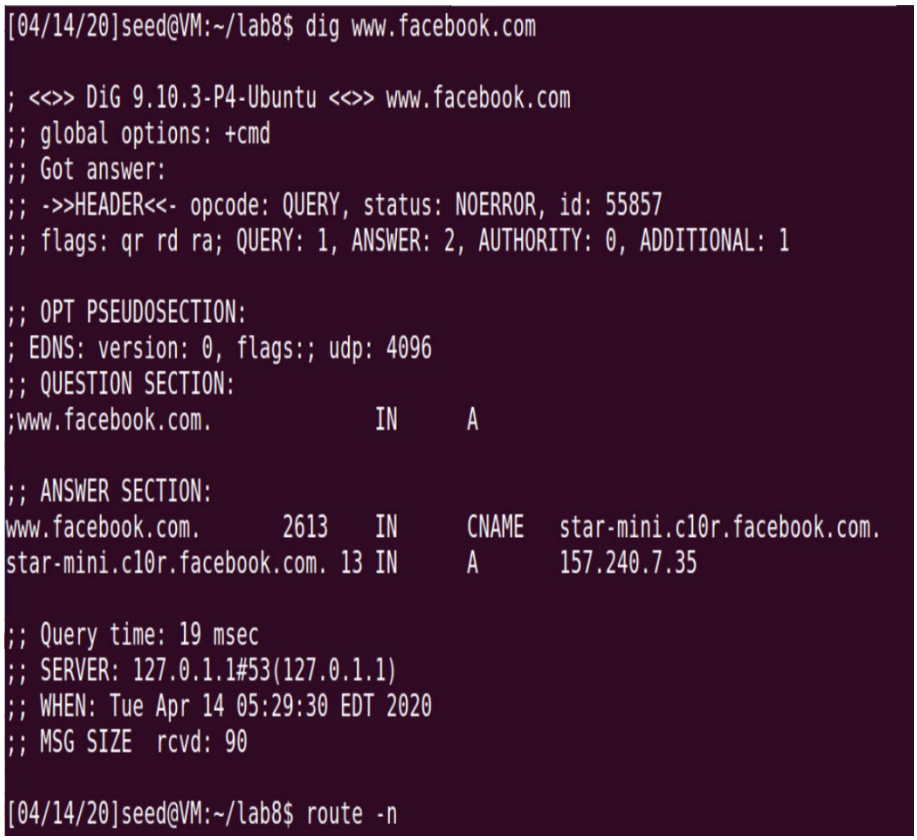

Figure 6 dig facebook

Before the firewall was configured, operations such as telnet and accessing websites were still able to be performed as shown in the screenshots:

```
[04/14/20]seed@VM:~/lab8$ ping 157.240.7.35
PING 157.240.7.35 (157.240.7.35) 56(84) bytes of data.
64 bytes from 157.240.7.35: icmp_seq=1 ttl=50 time=19.8 ms
64 bytes from 157.240.7.35: icmp_seq=2 ttl=50 time=17.4 ms
64 bytes from 157.240.7.35: icmp_seq=3 ttl=50 time=9.68 ms
64 bytes from 157.240.7.35: icmp_seq=4 ttl=50 time=9.18 ms
```

Figure 7 Successful ping

After adding the firewall rules,

```
[04/14/20]seed@VM:~/lab8$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
45.60.67.5                 DENY OUT    Anywhere on enp0s3
157.240.13.35              DENY OUT    Anywhere on enp0s3
157.240.7.35               DENY OUT    Anywhere on enp0s3

[04/14/20]seed@VM:~/lab8$
```

Figure 8 Block facebook

the ping operation failed

```
[04/14/20]seed@VM:~/lab8$ ping 157.240.7.35
PING 157.240.7.35 (157.240.7.35) 56(84) bytes of data.
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
^C
--- 157.240.7.35 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2037ms

[04/14/20]seed@VM:~/lab8$
```

Figure 9 Ping failed for facebook

We now add the blocking of telnet operations by blocking out the actual IP of the server so that we cannot telnet to it.

```
[04/14/20]seed@VM:~/lab8$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
45.60.67.5                 DENY OUT    Anywhere on enp0s3
157.240.13.35              DENY OUT    Anywhere on enp0s3
157.240.7.35               DENY OUT    Anywhere on enp0s3
10.0.2.9                   DENY OUT    Anywhere on enp0s3

[04/14/20]seed@VM:~/lab8$ telnet 10.0.2.9
Trying 10.0.2.9...
```

Figure 10 Blocking server IP

Tunnelling configuration

Facebook

First, we add the routing for traffic going out to the facebook IP. We can do this on the client side by routing traffic to the default gateway under the tun0 interface.

```
[04/14/20]seed@VM:~/lab8$ 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
157.240.13.0    0.0.0.0         255.255.255.0   U     0      0        0 tun0
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
```

Figure 11 route table with facebook IP subnet

After the routing is done, we can now send packets through the tunnel to the IP of the website. As seen in the screenshots, the traffic on the client is shown to pass straight to the IP of facebook as I disabled promiscuous mode. However, on the server, it is seen that facebook first sends the packets to the server via TCP and then the server sends these packets via UDP from 10.0.2.9 which is the client's machine. In the command line we can also see packets being sent through the tunnel to the tun interface and then reaching the application.

```
21834 2020-04-14 05:39:35.3269828… 10.0.2.15      10.0.2.9        IPv4     1514 Fragmented IP protocol (proto=U…
21835 2020-04-14 05:39:35.3269946… 10.0.2.15      10.0.2.9        UDP        62 36322 → 9001 Len=1500
21836 2020-04-14 05:39:35.3270121… 10.0.2.15      10.0.2.9        UDP       222 36322 → 9001 Len=180
21837 2020-04-14 05:39:35.3270154… 157.240.13.35  10.0.2.9        TCP        60 443 → 37282 [ACK] Seq=1036433 A…
21838 2020-04-14 05:39:35.3272849… 10.0.2.9       10.0.2.15       UDP        82 9001 → 36322 Len=40
21839 2020-04-14 05:39:35.3275407… 10.0.2.9       157.240.13.35   TLSv1.2   133 Application Data
21840 2020-04-14 05:39:35.3276296… 10.0.2.9       157.240.13.35   TCP      1514 [TCP segment of a reassembled P…
21841 2020-04-14 05:39:35.3276884… 10.0.2.9       157.240.13.35   TLSv1.2   194 Application Data
21842 2020-04-14 05:39:35.3279221… 157.240.13.35  10.0.2.9        TCP        60 443 → 37282 [ACK] Seq=1036433 A…
21843 2020-04-14 05:39:35.3280849… 10.0.2.9       10.0.2.15       UDP        82 9001 → 36322 Len=40
21844 2020-04-14 05:39:35.3316774… 157.240.13.35  10.0.2.9        TLSv1.2    89 Application Data
21845 2020-04-14 05:39:35.3317955… 10.0.2.9       10.0.2.15       UDP       117 9001 → 36322 Len=75
21846 2020-04-14 05:39:35.3322020… 10.0.2.15      10.0.2.9        UDP        82 36322 → 9001 Len=40
21847 2020-04-14 05:39:35.3322738… 10.0.2.9       157.240.13.35   TCP        54 37282 → 443 [ACK] Seq=163843683…
21848 2020-04-14 05:39:35.5278384… 157.240.13.35  10.0.2.9        TLSv1.2   222 Application Data
21849 2020-04-14 05:39:35.5286765… 10.0.2.9       10.0.2.15       UDP       250 9001 → 36322 Len=208
21850 2020-04-14 05:39:35.5292779… 10.0.2.15      10.0.2.9        UDP        82 36322 → 9001 Len=40
21851 2020-04-14 05:39:35.5293642… 10.0.2.9       157.240.13.35   TCP        54 37282 → 443 [ACK] Seq=163843683…
```

Figure 12 Server side

```
  564 2020-04-14 05:34:14.0187661… 157.240.13.19  192.168.53.5    TLSv1.2   257 Application Data, Application Data
  565 2020-04-14 05:34:14.0187907… 192.168.53.5   157.240.13.19   TCP        40 47946 → 443 [ACK] Seq=3143698737 Ack=1807956 …
  566 2020-04-14 05:34:20.8409913… 192.168.53.5   157.240.13.35   TLSv1.2   192 Application Data
  567 2020-04-14 05:34:20.8410553… 192.168.53.5   157.240.13.35   TCP      1500 [TCP segment of a reassembled PDU]
  568 2020-04-14 05:34:20.8410567… 192.168.53.5   157.240.13.35   TLSv1.2   119 Application Data
  569 2020-04-14 05:34:20.8411510… 192.168.53.5   157.240.13.35   TCP      1500 [TCP segment of a reassembled PDU]
  570 2020-04-14 05:34:20.8411520… 192.168.53.5   157.240.13.35   TCP      1500 [TCP segment of a reassembled PDU]
  571 2020-04-14 05:34:20.8411551… 192.168.53.5   157.240.13.35   TLSv1.2  1038 Application Data
  572 2020-04-14 05:34:20.8428213… 157.240.13.35  192.168.53.5    TCP        40 443 → 37282 [ACK] Seq=961389 Ack=1638393351 W…
  573 2020-04-14 05:34:20.8432427… 157.240.13.35  192.168.53.5    TCP        40 443 → 37282 [ACK] Seq=961389 Ack=1638394890 W…
  574 2020-04-14 05:34:20.8437057… 157.240.13.35  192.168.53.5    TCP        40 443 → 37282 [ACK] Seq=961389 Ack=1638397348 W…
```

Figure 13 Client side

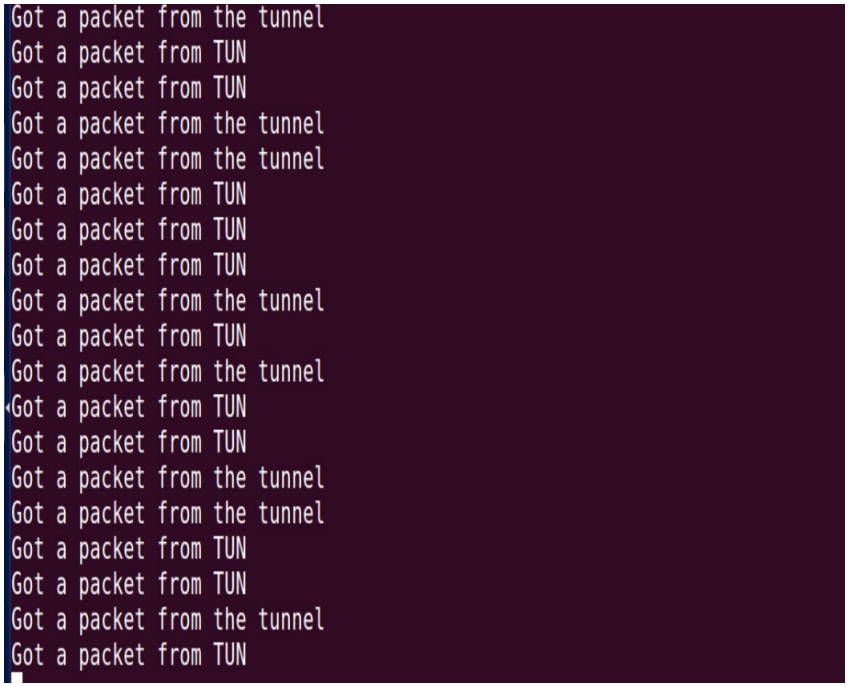

Figure 14 command line traffic

The packets are first being sent to the server as the packets will still go through the enp0s3 socket before being sent through the tunnel. The firewall on the client thus blocks the packets going to the enp0s3 socket but they are still able to flow through on the server. When the server transfers packets though the tunnel through UDP, the firewall is unable to block packets through that interface and hence the client is still able to receive the data.

Telnet

As shown previously, the firewall is already blocking traffic to 10.0.2.9 which is the server's IP address. When we try to establish a telnet connection, the module hangs. However, establishing a telnet connection to the server via the virtual IP address is successful, as yet again packets are being transferred to and from the server through the tunnel which is not blocked by the firewall. This can be seen in the wireshark trace captures. This is possible because we have already set up a routing for the data going through the subnet 192.168.53.0/24.

6	2020-04-14 05:45:52.5326973…	192.168.53.5	192.168.53.1	TCP	52 54236 → 23 [ACK] Seq=2766137491…
7	2020-04-14 05:45:52.5330692…	192.168.53.5	192.168.53.1	TELNET	79 Telnet Data ...
8	2020-04-14 05:45:52.5357136…	192.168.53.1	192.168.53.5	TCP	52 23 → 54236 [ACK] Seq=2315655305…
9	2020-04-14 05:45:57.5617514…	192.168.53.1	192.168.53.5	TELNET	64 Telnet Data ...
10	2020-04-14 05:45:57.5618175…	192.168.53.5	192.168.53.1	TCP	52 54236 → 23 [ACK] Seq=2766137518…
11	2020-04-14 05:45:57.5622777…	192.168.53.1	192.168.53.5	TELNET	91 Telnet Data ...
12	2020-04-14 05:45:57.5622911…	192.168.53.5	192.168.53.1	TCP	52 54236 → 23 [ACK] Seq=2766137518…
13	2020-04-14 05:45:57.5625105…	192.168.53.5	192.168.53.1	TELNET	127 Telnet Data ...
14	2020-04-14 05:45:57.5652474…	192.168.53.1	192.168.53.5	TCP	52 23 → 54236 [ACK] Seq=2315655356…
15	2020-04-14 05:45:57.5652718…	192.168.53.1	192.168.53.5	TELNET	55 Telnet Data ...
16	2020-04-14 05:45:57.5653404…	192.168.53.5	192.168.53.1	TELNET	55 Telnet Data ...

Figure 15 telnet client

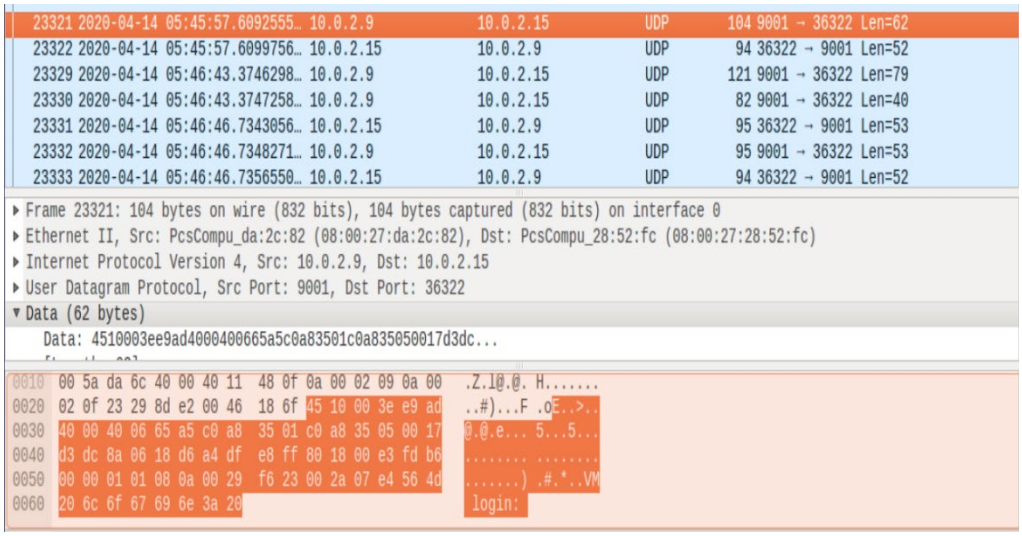

Figure 16 telnet server

As seen in the telnet server data, the login information is being sent to the IP address of 10.0.2.9 as that those are the sockets that the data flows through before reaching the virtual sockets. The data sent is also done through UDP which tells that the data is flowing through the VPN tunnel.