Princess Sumaya University for Technology
جامعة الأميرة سميّة للتكنولوجيا

Bypassing firewalls using VPNs

Name: Yara Al-Mhameed 20220822

Bahauldeen Qabani 20220106

Supervisor: Nour Kattawi

Contents

Table of figures

Introduction:

In This assignment we will show how firewalls work in blocking the traffic to specific addresses and how VPN servers bypass the firewall using the method of rerouting traffic through a tunnel. We used a simple example with 2 VMs one acting as a client and the other as a server.

Task 1: VM Setup

First we setup 2 different virtual machines one acting as a Client (clientvpn) and the other as a Server (servervpn), we then connected both VMs to a NAT network and assigned them IP Addresses.

Server = 10.0.2.8

Client = 10.0.2.7

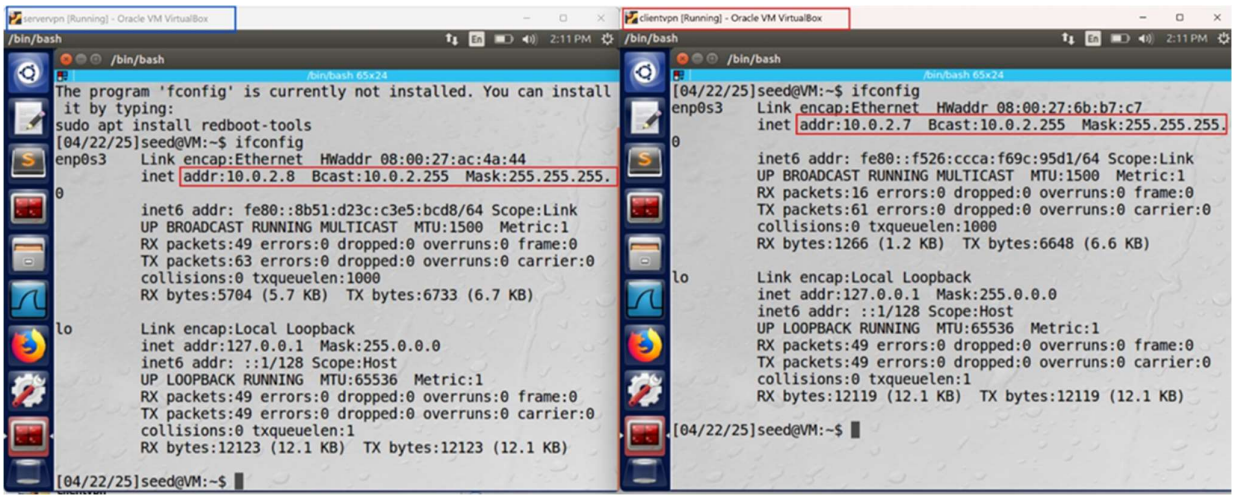

Figure 1: IP address of each VM

Task 2: Firewall Setup

Identifying Target Website and checking connectivity

In this assignment the target website is www.facebook.com, we first tested if we had connectivity and additionally got its IP address which we will use later on.

```
[04/22/25]seed@VM:~$ ping wwwfacebook.com
PING wwwfacebook.com (102.132.103.8) 56(84) bytes of data.
64 bytes from edge-star-shv-02-hbe1.facebook.com (102.132.103.8):
 icmp_seq=1 ttl=53 time=26.0 ms
64 bytes from edge-star-shv-02-hbe1.facebook.com (102.132.103.8):
 icmp_seq=2 ttl=53 time=30.9 ms
64 bytes from edge-star-shv-02-hbe1.facebook.com (102.132.103.8):
 icmp_seq=3 ttl=53 time=21.6 ms
64 bytes from edge-star-shv-02-hbe1.facebook.com (102.132.103.8):
 icmp_seq=4 ttl=53 time=22.1 ms
64 bytes from edge-star-shv-02-hbe1.facebook.com (102.132.103.8):
 icmp_seq=5 ttl=53 time=18.9 ms
64 bytes from edge-star-shv-02-hbe1.facebook.com (102.132.103.8):
 icmp_seq=6 ttl=53 time=22.6 ms
64 bytes from edge-star-shv-02-hbe1.facebook.com (102.132.103.8):
```

Figure 2: Identifying IP of Facebook and connectivity

Here we confirmed the connectivity and got the IP address which corresponds to 102.132.103.8

Firewall Configuration

Next we enabled the firewall using the command sudo ufw enable then used the command sudo ufw deny out on enp0s3 to 102.132.103.8 which blocks the traffic going from the client to this IP address (Facebook), and lastly used command sudo ufw status to ensure it is blocking the traffic.

```
8 packets transmitted, 8 received, 0% packet loss, time 7014ms
rtt min/avg/max/mdev = 18.985/22.959/30.990/3.663 ms
[04/22/25]seed@VM:~$ sudo ufw enable
Firewall is active and enabled on system startup
[04/22/25]seed@VM:~$ sudo ufw deny out on enp0s3 to 102.132.103.8
Rule added
[04/22/25]seed@VM:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
102.132.103.8              DENY OUT    Anywhere on enp0s3

[04/22/25]seed@VM:~$
```

Figure 3: Firewall Configurations

Verification

We then pinged the IP address again to check if the Firewall is actually blocking the traffic or not

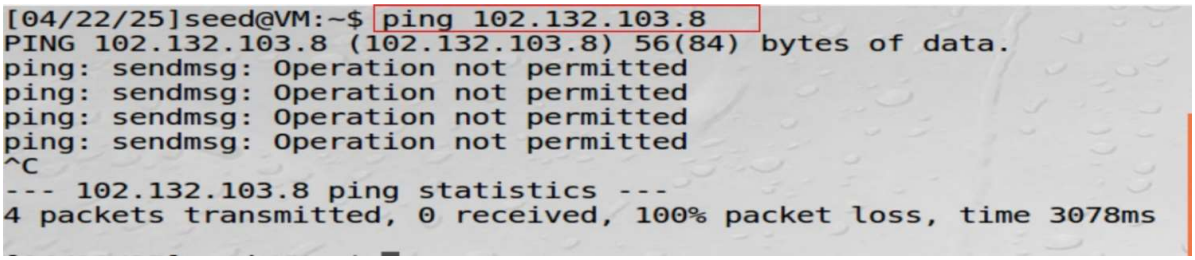

Figure 4: Verifying the firewall rule

Which turned out to be Successful

Task 3: Bypassing Firewall using VPN

Setting up a VPN server

First we started a VPN server using a C code which we got from SeedLabs and then we compiled the code and executed it

Figure 5: VPN server setup

We then configured a virtual tunnel

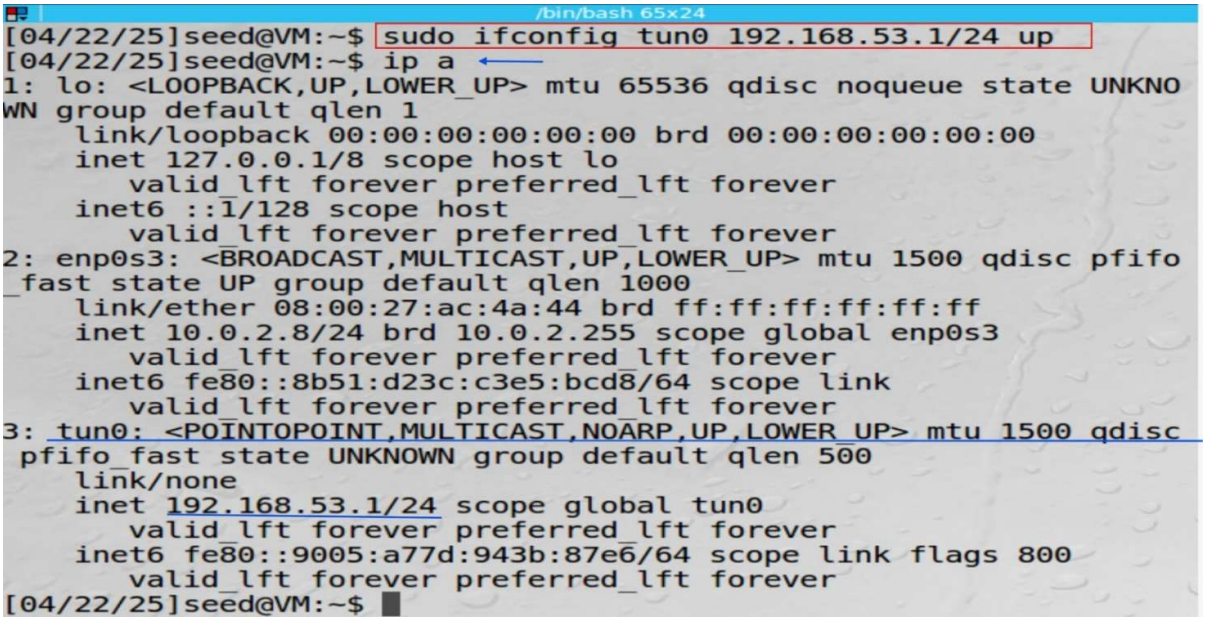

Figure 6: Virtual tunnel

For the server we gave it the tunnel IP address of 192.168.53.1/24 and for the client we gave the tunnel IP address of 192.168.53.5/24

Connecting VPN client

We used a C code which we got from SeedLabs and then we compiled the code and executed it, this code helped us in connecting with the VPN server

```
[04/22/25]seed@VM:~/.../vpn$ gcc vpnclient.c -o vpnclient
[04/22/25]seed@VM:~/.../vpn$ sudo ./vpnclient 10.0.2.8
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
```

Figure 7: Client connectivity with VPN server

We then returned to the VPN server to check if the connection was successful

```
[04/22/25]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 the tunnel
Got a packet from the tunnel
Got a packet from the tunnel
```

Figure 8: Confirming Connectivity

Which was confirmed to be a success with the message Connected with the client: Hello

Routing and NAT configurations

First we configured a route on the client which allowed traffic to move through the VPN

```
[04/22/25]seed@VM:~$ sudo route add -net 192.168.53.0/24 dev tun0

[04/22/25]seed@VM:~$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref
   Use Iface
default         10.0.2.1        0.0.0.0         UG    100    0
     0 enp0s3
10.0.2.0        *               255.255.255.0   U     100    0
     0 enp0s3
link-local      *               255.255.0.0     U     1000   0
     0 enp0s3
192.168.53.0    *               255.255.255.0   U     0      0
     0 tun0
192.168.53.0    *               255.255.255.0   U     0      0
     0 tun0
[04/22/25]seed@VM:~$
```

Figure 9: NAT configuration

Then we did the same to the server and added a command that enabled IP forwarding

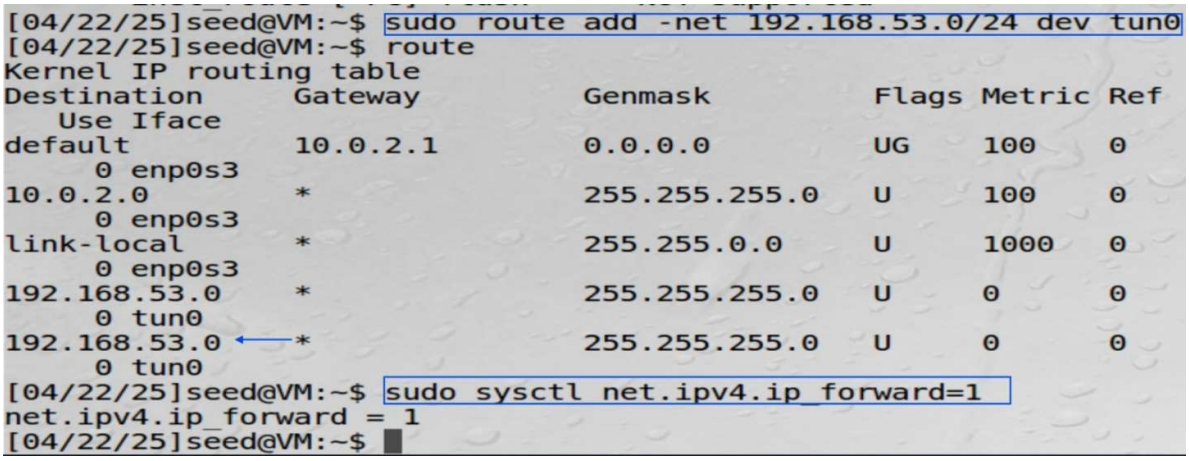

Figure 10: IP forwarding

Then we checked connectivity from client to server

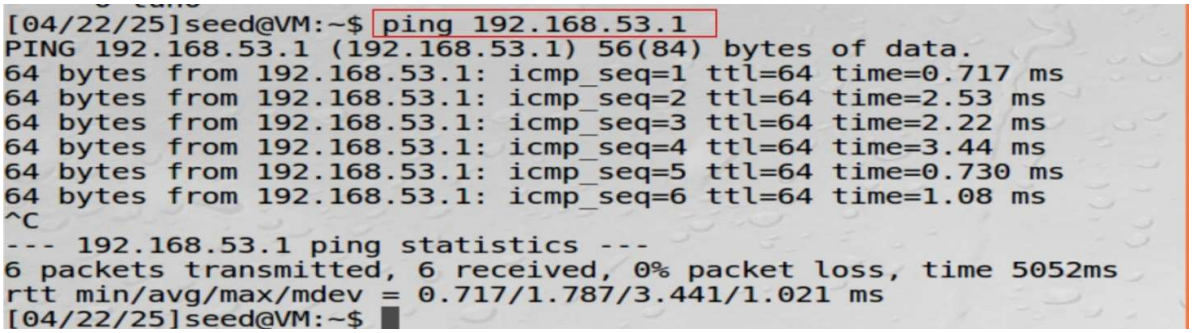

Figure 11: pinging Server

Which was a success

Telnet connection

Next we established a telnet connection which helped with the tunnel connection using the command telnet

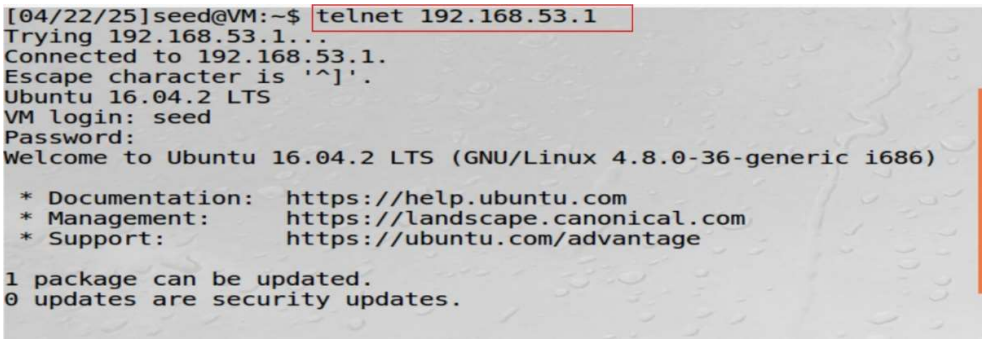

Figure 12: Telnet

NAT setup

To accomplish NAT connection on the VPN server we do the following commands

```
[04/22/25]seed@VM:~$ sudo iptables -F
[04/22/25]seed@VM:~$ sudo iptables -t nat -F
[04/22/25]seed@VM:~$ sudo iptables -t nat -A POSTROUTING -j MASQU
ERADE -o enp0s3
[04/22/25]seed@VM:~$
```

Figure 13: NAT setup

Sudo iptables -F: Cleared all firewall ruled
sudo iptables -t nat -F: Clear NAT table rules
sudo iptables -t nat -A POSTROUTING -j MASQUERADE -o enp0s3: Masks client traffics as if the traffic is forwarded from the server

Verify connectivity

Last we returned to the client to check if the connectivity to the blocked IP address is successful

```
[04/22/25]seed@VM:~$ ping 102.132.103.8
PING 102.132.103.8 (102.132.103.8) 56(84) bytes of data.
64 bytes from 102.132.103.8: icmp_seq=1 ttl=53 time=21.2 ms
64 bytes from 102.132.103.8: icmp_seq=2 ttl=53 time=21.0 ms
64 bytes from 102.132.103.8: icmp_seq=3 ttl=53 time=23.2 ms
64 bytes from 102.132.103.8: icmp_seq=4 ttl=53 time=31.8 ms
64 bytes from 102.132.103.8: icmp_seq=5 ttl=53 time=24.7 ms
64 bytes from 102.132.103.8: icmp_seq=6 ttl=53 time=19.6 ms
^C
--- 102.132.103.8 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5031ms
rtt min/avg/max/mdev = 19.613/23.635/31.846/4.029 ms
```

Figure 14: Successful ping to blocked IP

Which was successful
To fully check that the traffic is actually bring forwarded we used Wireshark to confirm

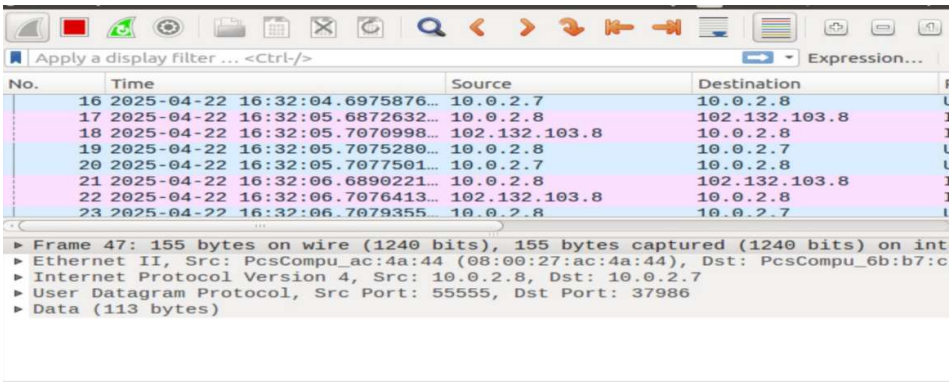

Figure 15: Wireshark

Conclusion

Using a virtual private network (VPN) to get around firewalls entails sending your internet traffic to a distant server via an encrypted tunnel, which conceals your IP address and your online behaviour. This enables users to bypass restrictions and access content that is blocked. Although efficient, it can be against local laws or terms of service, and not all VPNs can get past sophisticated firewalls like those that use deep packet inspection.