FIREWALL EVASION LAB

Bypassing Firewalls using VPN

Joseph Tsai
CSS 537 – Winter 2021

Table of Contents

Introduction

The usage of firewalls to prevent users from accessing certain services has become common practice. Through the usage of firewalls, nation states, companies, and internet service providers (ISP's) have found ways to prevent network traffic on their respective networks from reaching certain internet services.

While the reasons for prevention may differ from entity to entity, the end-goal is the same: prevent users from accessing certain services. One common way to achieve this goal is to use egress filtering, where packets that are leaving a given network are analyzed for appropriateness against the entity's firewall policies. In the case that the packet's intended destination is allowed, the packet is allowed to proceed, and vice versa.

With such preventative measures, there have been technologies created to allow users to bypass such firewall rules. This report explores one such technology: Virtual Private Networks (VPN's). Through the usage of VPN's, users are able to bypass the firewall settings previously imposed upon them by the corresponding entity network administrators.

How VPN's are constructed and used are explored in greater depth in the following lab report sections.

Lab Environment Setup and Lab Tasks

Lab Environment Setup

The following steps articulate how the lab environment was established:

1. Two hosts, Machine A and Machine B, were placed on the same local area network.
   a. Machine A represents the VPN server.
   b. Machine B represents the client that is attempting to reach a blocked website.

2. To simulate a firewall, network filtering rules were placed on Machine B to drop any connection to a specified website's IP address (128.119.245.12).

Lab Tasks

The following tasks articulate a high-level overview of how the lab was conducted:

Task 1 – VM Setup: The Virtual Machines were configured to simulate a VPN server and client on the same network

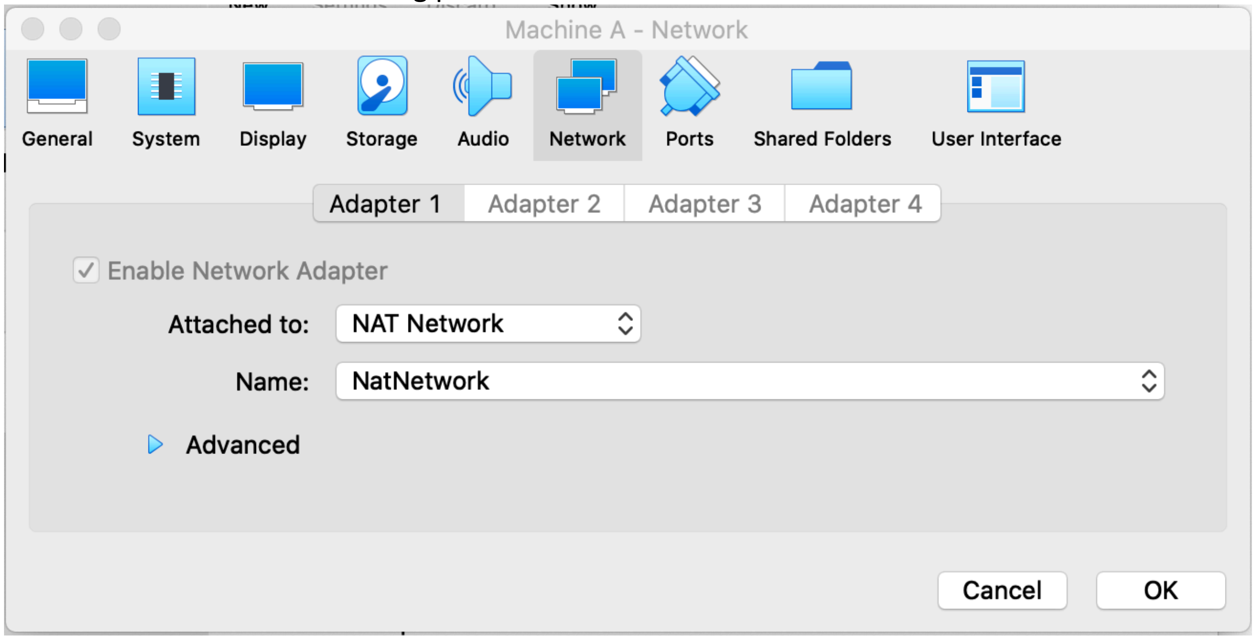
Task 2 – Set Up Firewall: The egress traffic filtering rules were applied on Machine B.

Task 3 – Bypass Firewall Using VPN: The firewall on Machine B is bypassed through the usage of a VPN established between Machine A and Machine B.

Task 1: VM Setup

In order for Machine A and Machine B to connect, Machine A (the VPN server) and Machine B (VPN client) are placed on the same local area network. This is displayed in the following screenshots through the configuration of the Virtual Machines running each respective host.

Exhibit 1.1: Machine A being placed on the NAT network which is named "NatNetwork".



Machine A - Network

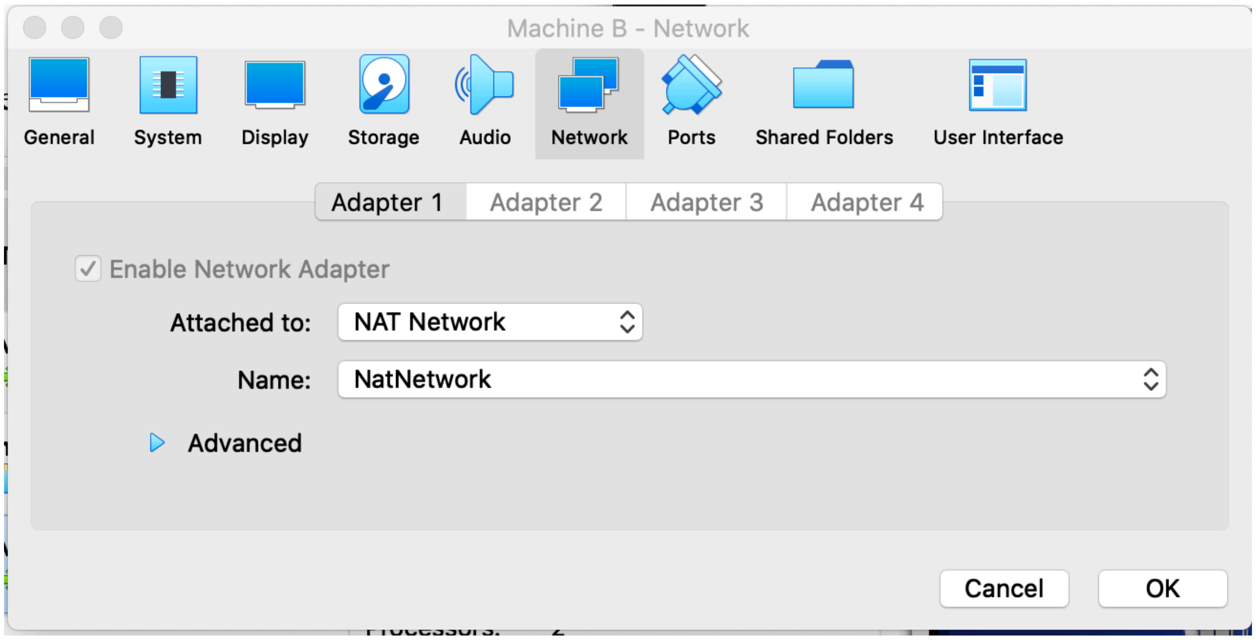
Adapter 1

Enable Network Adapter ✓

Attached to: NAT Network

Name: NatNetwork

Exhibit 1.2: Machine B being placed on the same NAT network as Machine A which is named "NatNetwork".



Machine B - Network

Adapter 1

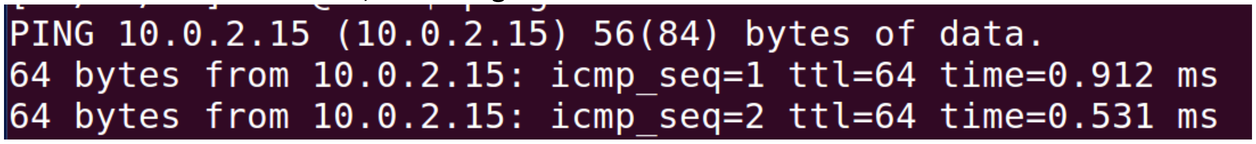
Enable Network Adapter ✓

Attached to: NAT Network

Name: NatNetwork

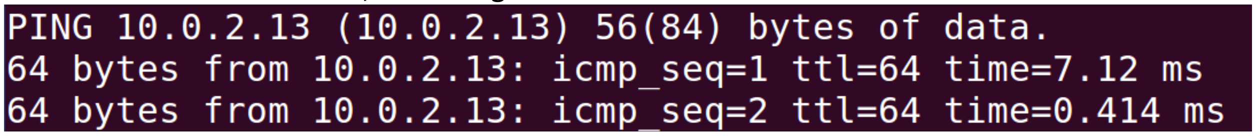
To check that this network configuration is working properly, one can perform a *ping* from each host to the other. This is displayed in the following screenshots.

Exhibit 1.3: Machine A pinging Machine B (10.0.2.15). Bytes are returned via the *ping* command, indicating that Machine B is alive on the network.



```
PING 10.0.2.15 (10.0.2.15) 56(84) bytes of data.
64 bytes from 10.0.2.15: icmp_seq=1 ttl=64 time=0.912 ms
64 bytes from 10.0.2.15: icmp_seq=2 ttl=64 time=0.531 ms
```

Exhibit 1.4: Machine B pinging Machine A (10.0.2.13). Bytes are returned via the *ping* command, indicating that Machine A is alive on the network.



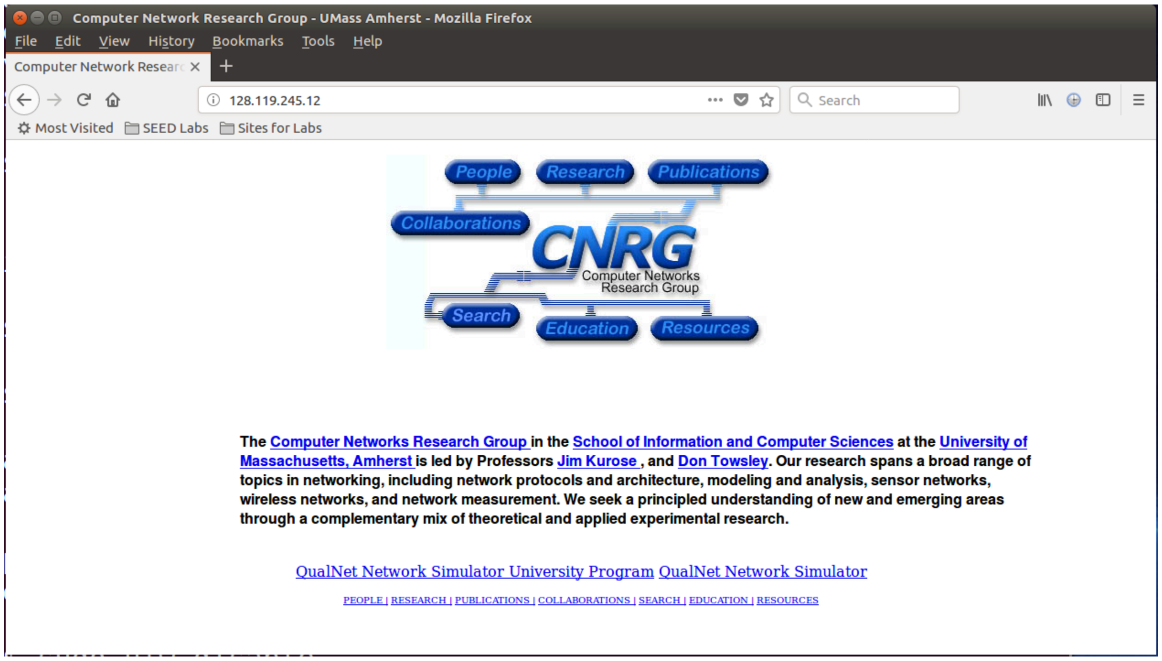
```
PING 10.0.2.13 (10.0.2.13) 56(84) bytes of data.
64 bytes from 10.0.2.13: icmp_seq=1 ttl=64 time=7.12 ms
64 bytes from 10.0.2.13: icmp_seq=2 ttl=64 time=0.414 ms
```

Now that the two hosts have been placed on the same network, the next step is to configure the firewall on Machine B to simulate an environment where there is a firewall filtering egress traffic. This is displayed in the following section.

Task 2: Set Up Firewall

The chosen website to block has the IP address of 128.119.245.12, as shown in the URL address of Exhibit 2.1. A screenshot has been provided to display that before the firewall rules are enabled, that Machine B is indeed able to access the given website without any issues.

Exhibit 2.1: Displaying that Machine B is able to access the website with IP address 128.119.245.12 before the firewall rules are enabled.



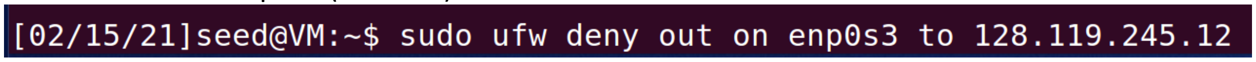
The Computer Networks Research Group in the School of Information and Computer Sciences at the University of Massachusetts, Amherst is led by Professors Jim Kurose , and Don Towsley. Our research spans a broad range of topics in networking, including network protocols and architecture, modeling and analysis, sensor networks, wireless networks, and network measurement. We seek a principled understanding of new and emerging areas through a complementary mix of theoretical and applied experimental research.

QualNet Network Simulator University Program QualNet Network Simulator

PEOPLE | RESEARCH | PUBLICATIONS | COLLABORATIONS | SEARCH | EDUCATION | RESOURCES

To prevent Machine B from accessing the website, firewall rules are added on Machine B. These are displayed in Exhibit 2.2.

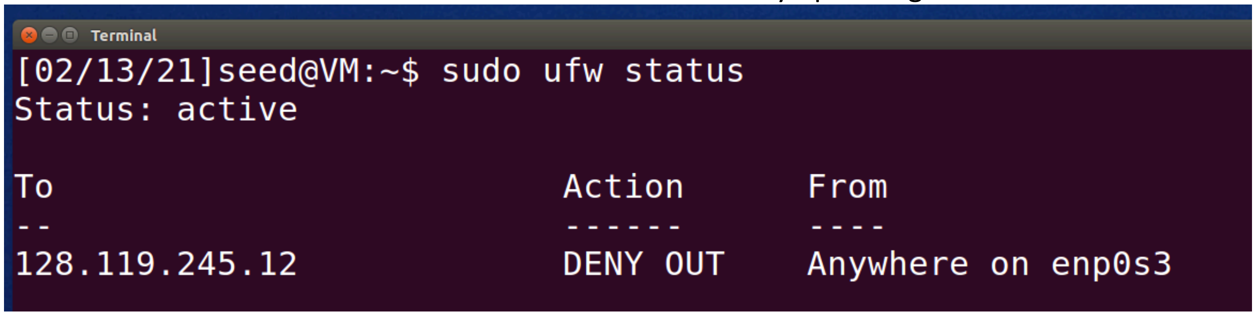
Exhibit 2.2: Firewall rule which was added onto Machine B to deny any outbound traffic from Machine B's enp0s3 (internet) interface to the website with IP address 128.119.245.12.



```
[02/15/21]seed@VM:~$ sudo ufw deny out on enp0s3 to 128.119.245.12
```

To verify that the relevant firewall rule has indeed been put into place, one can run the *sudo ufw status* command, which will display the status of the firewall itself, as well as any firewall rules that are configured on Machine B.

Exhibit 2.3: Running the *sudo ufw status* command, which displays that the firewall rule displayed in Exhibit 2.2 has indeed been put into effect. The status of the firewall is also "active" which indicates that the rules are actively operating on Machine B.



```
[02/13/21]seed@VM:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
128.119.245.12             DENY OUT    Anywhere on enp0s3
```

Once the firewall rules have been put into place, the expected behavior is that Machine B will no longer be able to access 128.119.245.12. This is observed in Exhibit 2.4 and Exhibit 2.5, below.

Exhibit 2.4: Machine B's browser is unable to access the website and is stuck attempting to load the page.

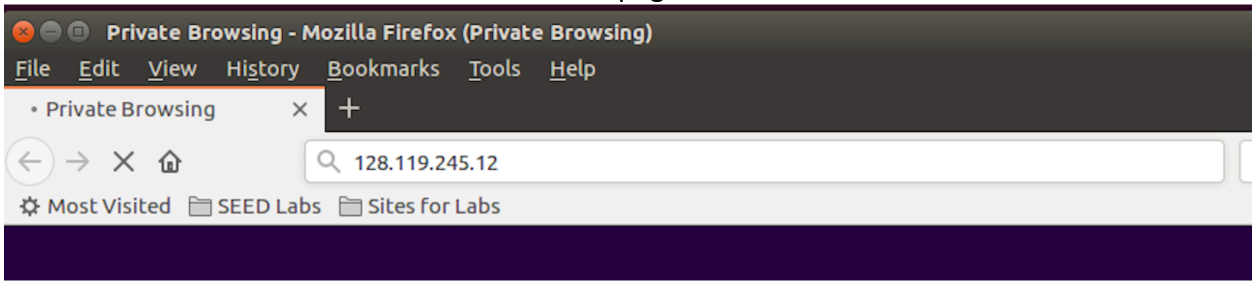
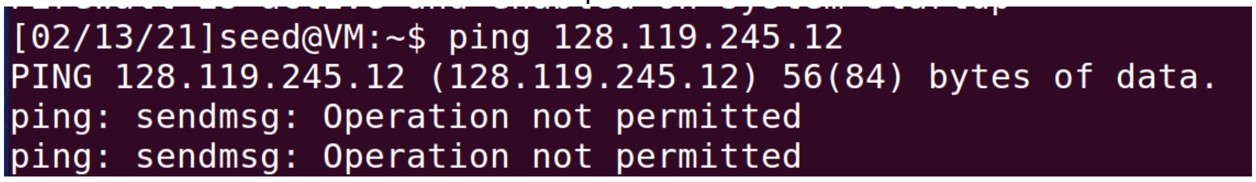

Private Browsing - Mozilla Firefox (Private Browsing)
File Edit View History Bookmarks Tools Help
Private Browsing
128.119.245.12
Most Visited
SEED Labs
Sites for Labs

Exhibit 2.5: Machine B is also unable to ping the given website and is told that the operation is not permitted.



```
[02/13/21]seed@VM:~$ ping 128.119.245.12
PING 128.119.245.12 (128.119.245.12) 56(84) bytes of data.
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
```

Now that the relevant firewall rules have been established on Machine B, one can explore how the usage of a VPN would allow a user on Machine B to bypass the given firewall rules. This is explored in the following section.

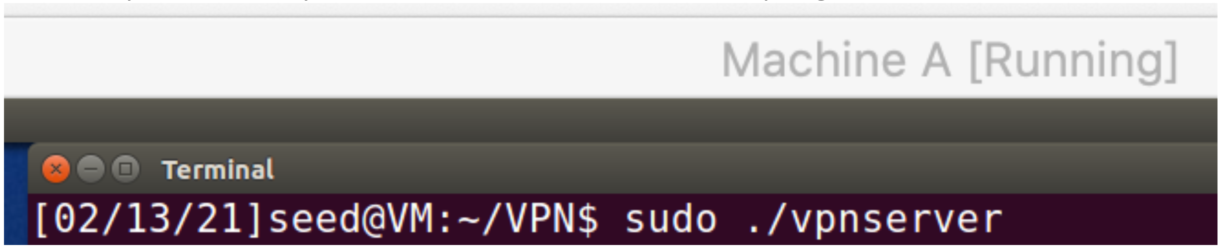
Task 3: Bypassing Firewall Using VPN

There are two main components for a VPN: tunneling, and encryption. Tunneling refers to the connection that is established between the VPN server and the VPN client, while encryption refers to the encryption of the data that is transferred between the VPN server and the VPN client.

For this lab report, the tunneling aspect of VPN's was explored to display how usage of a VPN would allow Machine B to bypass the given firewall rules. The corresponding code for the VPN server and VPN client have been included as supplemental files, and are not explored in detail in the following exhibits, as only a single line of code was changed in the vpnclient.c file. This change is displayed in the appendix of this report. However, the functionality provided by the VPN code *is* shown and explained in the following steps.

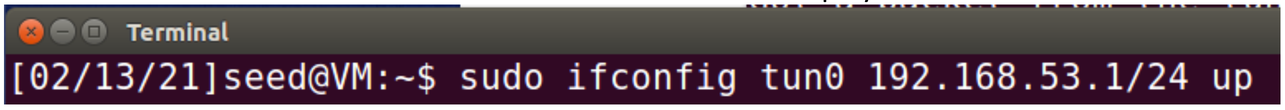
Step 1: Run VPN Server

Exhibit 3.1: Firstly, the compiled VPN server code is run on Machine A. This allows Machine A to actively listen for any VPN connections which are attempting to connect to its VPN service.



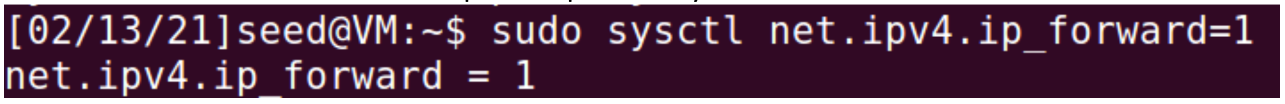
```
[02/13/21]seed@VM:~/VPN$ sudo ./vpnserver
```

Exhibit 3.2: As the enp0s3 interface will be used on Machine A to retrieve requested web traffic, another network interface, tun0, must be created to allow for tunneling between Machine A and Machine B. Hence, the enp0s3 interface on Machine A can be used to retrieve requested traffic from Machine B, which is then forwarded into the tunnel that is established between Machine A and Machine B so that Machine B can receive its requested traffic. The command to create the tun0 interface is displayed below.



```
[02/13/21]seed@VM:~$ sudo ifconfig tun0 192.168.53.1/24 up
```

Exhibit 3.3: Most hosts are not configured to forward packets by default. Hence, the following command is run on Machine A in order to configure packet forwarding, which will allow Machine A to forward requested packets by Machine B into the VPN tunnel.

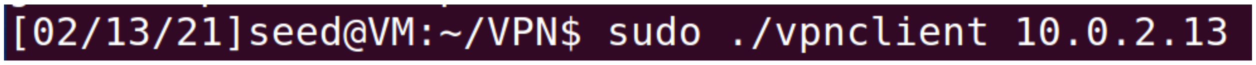


```
[02/13/21]seed@VM:~$ sudo sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
```

Step 2: Run VPN Client

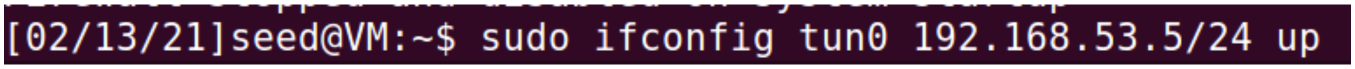
Now that the VPN server has been configured, the VPN client can be configured. This is shown in the following exhibits.

Exhibit 3.4: Firstly, the client runs the compiled VPN client code, and indicates the IP address of the VPN server on the network. As the VPN server is Machine A, the IP address of 10.0.2.13 is used.



```
[02/13/21]seed@VM:~/VPN$ sudo ./vpnclient 10.0.2.13
```

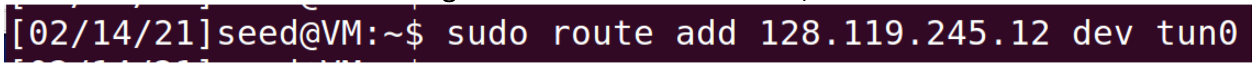
Exhibit 3.5: Similar to the VPN server, a new interface must be created to establish the VPN tunnel on Machine B. This is because the enp0s3 interface on Machine B currently has its network traffic filtered by the firewall. By creating a new network interface, Machine B can send network packets to Machine A and receive requested traffic through the tunnel instead of having its packets blocked by the firewall. The redirection of this traffic on Machine B is shown in step 3. The following command establishes the VPN tunnel, similar to that of Exhibit 3.2 on the VPN server.



```
[02/13/21]seed@VM:~$ sudo ifconfig tun0 192.168.53.5/24 up
```

Step 3: Set Up Routing on Client VM

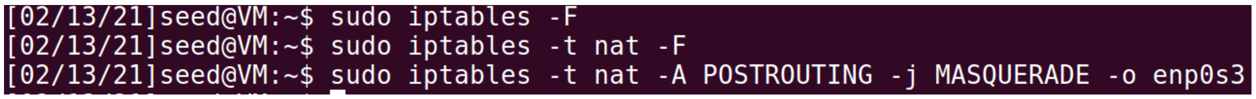
Exhibit 3.6: In order to bypass the firewall filter, a new route must be added on Machine B to redirect traffic to the blocked website through the VPN tunnel instead of Machine B's standard enp0s3 interface, where it is currently having its traffic blocked. The following command essentially tells Machine B to route all traffic with the destination of the blocked IP address through the VPN tunnel interface, tun0.



```
[02/14/21]seed@VM:~$ sudo route add 128.119.245.12 dev tun0
```

Step 4: Set up NAT on Server VM

Exhibit 3.7: The final step in configuring the VPN is to set up the NAT router on the VPN Server. This will essentially forward packets which it receives on its own enp0s3 interface through the VPN tunnel. The following commands display how this can be configured on Machine A, the VPN server.



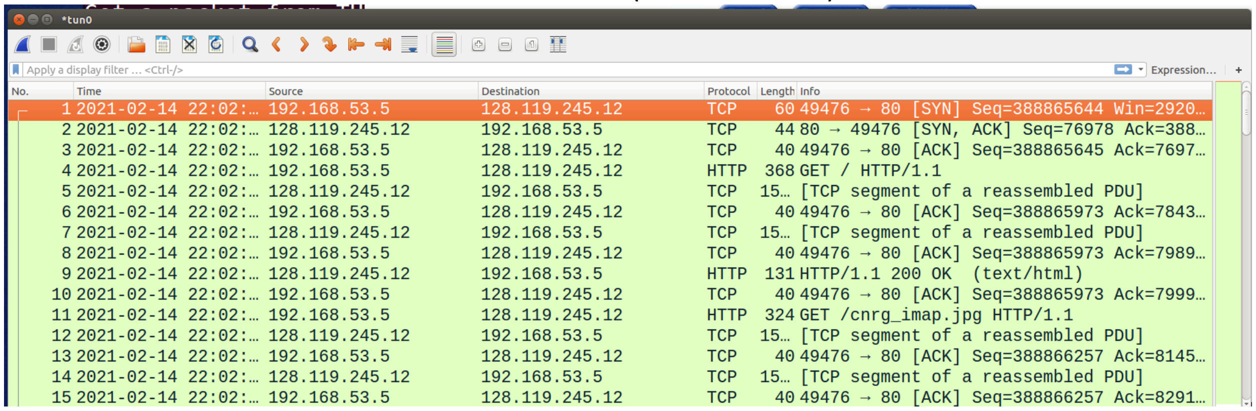
```
[02/13/21]seed@VM:~$ sudo iptables -F
[02/13/21]seed@VM:~$ sudo iptables -t nat -F
[02/13/21]seed@VM:~$ sudo iptables -t nat -A POSTROUTING -j MASQUERADE -o enp0s3
```

At a high level, the flow of network traffic can be described in the following steps:

1. Machine B (VPN client) creates a request to access the blocked website.
2. Because Machine B has specified that traffic with the destination of the blocked website should use the VPN interface (tun0) the traffic will be sent to Machine A (the VPN server).
3. Machine A will then use its own enp0s3 interface to retrieve the requested information and forward the information back to Machine B via the VPN tunnel. This essentially allows Machine B to receive traffic from the originally blocked website.

To explore the flow of network traffic in greater detail, one can look at the network traffic which is captured upon attempted access of the blocked website on both the tun0 on Machine B and the enp0s3 interface on Machine A. This is shown in the following exhibits.

Exhibit 3.8: Firstly, one can look at the traffic that is captured on the tun0 interface on Machine B when the initial request to the website is made. Once Machine B attempts to access the given website, one observes traffic displayed within the tun0 interface with the destination of the blocked website (128.119.245.0).



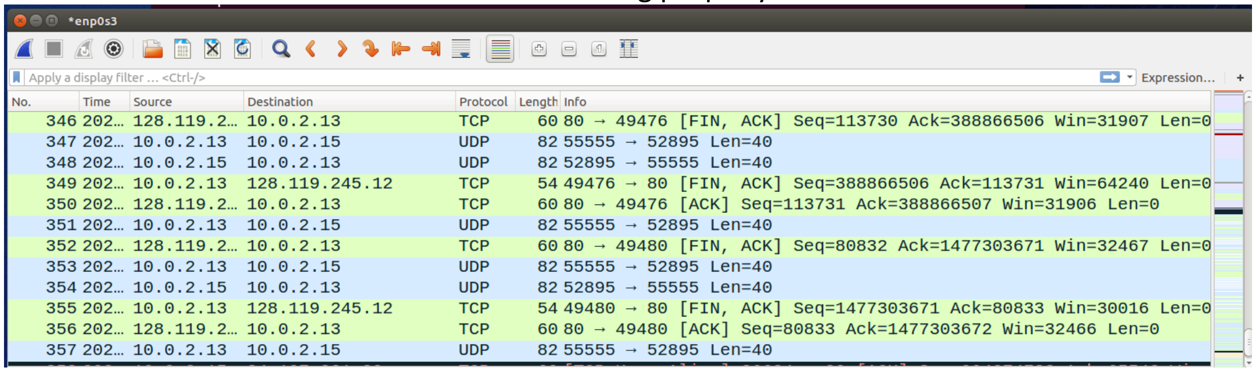
*tun0

Apply a display filter … <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	2021-02-14 22:02:…	192.168.53.5	128.119.245.12	TCP	60	49476 → 80 [SYN] Seq=388865644 Win=2920…
2	2021-02-14 22:02:…	128.119.245.12	192.168.53.5	TCP	44	80 → 49476 [SYN, ACK] Seq=76978 Ack=388…
3	2021-02-14 22:02:…	192.168.53.5	128.119.245.12	TCP	40	49476 → 80 [ACK] Seq=388865645 Ack=7697…
4	2021-02-14 22:02:…	192.168.53.5	128.119.245.12	HTTP	368	GET / HTTP/1.1
5	2021-02-14 22:02:…	128.119.245.12	192.168.53.5	TCP	15…	[TCP segment of a reassembled PDU]
6	2021-02-14 22:02:…	192.168.53.5	128.119.245.12	TCP	40	49476 → 80 [ACK] Seq=388865973 Ack=7843…
7	2021-02-14 22:02:…	128.119.245.12	192.168.53.5	TCP	15…	[TCP segment of a reassembled PDU]
8	2021-02-14 22:02:…	192.168.53.5	128.119.245.12	TCP	40	49476 → 80 [ACK] Seq=388865973 Ack=7989…
9	2021-02-14 22:02:…	128.119.245.12	192.168.53.5	HTTP	131	HTTP/1.1 200 OK (text/html)
10	2021-02-14 22:02:…	192.168.53.5	128.119.245.12	TCP	40	49476 → 80 [ACK] Seq=388865973 Ack=7999…
11	2021-02-14 22:02:…	192.168.53.5	128.119.245.12	HTTP	324	GET /cnrg_imap.jpg HTTP/1.1
12	2021-02-14 22:02:…	128.119.245.12	192.168.53.5	TCP	15…	[TCP segment of a reassembled PDU]
13	2021-02-14 22:02:…	192.168.53.5	128.119.245.12	TCP	40	49476 → 80 [ACK] Seq=388866257 Ack=8145…
14	2021-02-14 22:02:…	128.119.245.12	192.168.53.5	TCP	15…	[TCP segment of a reassembled PDU]
15	2021-02-14 22:02:…	192.168.53.5	128.119.245.12	TCP	40	49476 → 80 [ACK] Seq=388866257 Ack=8291…

On the other hand, traffic should also be seen on the enp0s3 interface of Machine A, as it is forwarding the packets to the blocked website for Machine B and returning those packets through the VPN tunnel. This is shown in the following exhibit.

Exhibit 3.9: On Machine A's enp0s3 interface, one can see TCP traffic on the enp0s3 interface to the desired website, as well as the UDP the traffic between Machine B (client) and Machine A (server) displaying the VPN. This screenshot displays that the packet forwarding between Machine A and Machine B is working properly to the blocked website.



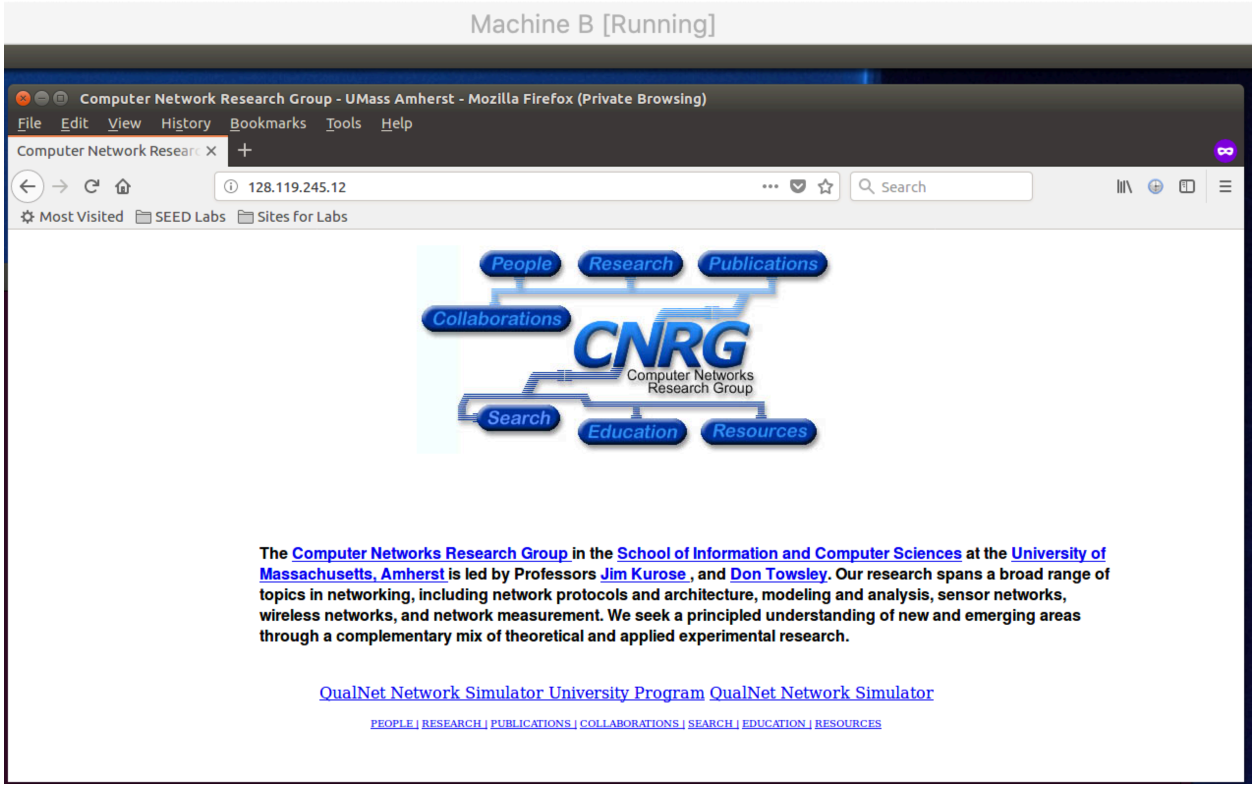
*enp0s3

Apply a display filter … <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
346	202…	128.119.2…	10.0.2.13	TCP	60	80 → 49476 [FIN, ACK] Seq=113730 Ack=388866506 Win=31907 Len=0
347	202…	10.0.2.13	10.0.2.15	UDP	82	55555 → 52895 Len=40
348	202…	10.0.2.15	10.0.2.13	UDP	82	52895 → 55555 Len=40
349	202…	10.0.2.13	128.119.245.12	TCP	54	49476 → 80 [FIN, ACK] Seq=388866506 Ack=113731 Win=64240 Len=0
350	202…	128.119.2…	10.0.2.13	TCP	60	80 → 49476 [ACK] Seq=113731 Ack=388866507 Win=31906 Len=0
351	202…	10.0.2.13	10.0.2.15	UDP	82	55555 → 52895 Len=40
352	202…	128.119.2…	10.0.2.13	TCP	60	80 → 49480 [FIN, ACK] Seq=80832 Ack=1477303671 Win=32467 Len=0
353	202…	10.0.2.13	10.0.2.15	UDP	82	55555 → 52895 Len=40
354	202…	10.0.2.15	10.0.2.13	UDP	82	52895 → 55555 Len=40
355	202…	10.0.2.13	128.119.245.12	TCP	54	49480 → 80 [FIN, ACK] Seq=1477303671 Ack=80833 Win=30016 Len=0
356	202…	128.119.2…	10.0.2.13	TCP	60	80 → 49480 [ACK] Seq=80833 Ack=1477303672 Win=32466 Len=0
357	202…	10.0.2.13	10.0.2.15	UDP	82	55555 → 52895 Len=40

The final piece of expected behavior is that ultimately, Machine B would be able to access the blocked website. This is indeed what is observed in the following exhibit.

Exhibit 3.10: Displaying that Machine B is now able to access the blocked website, even with the firewall filtering rules enabled (Exhibit 2.3).


Machine B [Running]
Computer Network Research Group - UMass Amherst - Mozilla Firefox (Private Browsing)
File Edit View History Bookmarks Tools Help
128.119.245.12
Search
Most Visited
SEED Labs
Sites for Labs
People
Research
Publications
Collaborations
CNRG
Computer Networks Research Group
Search
Education
Resources

The Computer Networks Research Group in the School of Information and Computer Sciences at the University of Massachusetts, Amherst is led by Professors Jim Kurose , and Don Towsley. Our research spans a broad range of topics in networking, including network protocols and architecture, modeling and analysis, sensor networks, wireless networks, and network measurement. We seek a principled understanding of new and emerging areas through a complementary mix of theoretical and applied experimental research.

QualNet Network Simulator University Program QualNet Network Simulator

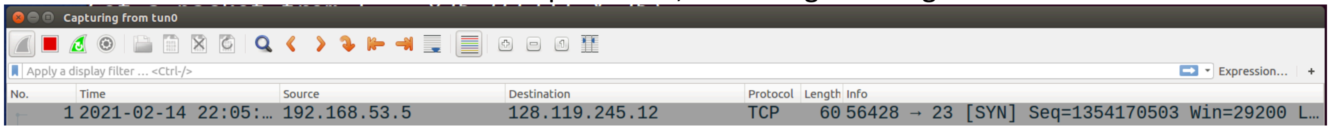
PEOPLE | RESEARCH | PUBLICATIONS | COLLABORATIONS | SEARCH | EDUCATION | RESOURCES

It is important to note that with how the VPN configured, *any* sort of traffic to the blocked website should be properly forwarded through the VPN. That is, not only standard web traffic should be forwarded through the VPN tunnel, but other protocols as well, such as *telnet*. This is shown in the final exhibits of this report, below.

To prove that the VPN would also work for *telnet* traffic, the Wireshark capture of the network traffic was also obtained between Machine B and the blocked website.

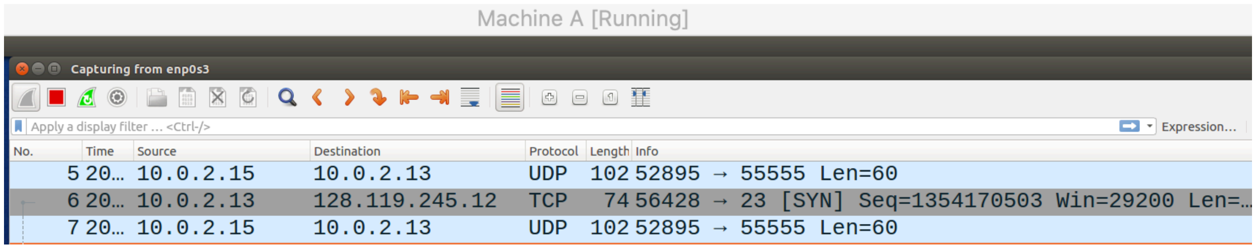
It should be noted that the website blocked the attempted *telnet* request, as expected, but the traffic was still properly funneled through the VPN.

Exhibit 3.11: Network traffic capture on the tun0 interface on Machine B, displaying the forwarded telnet attempt to the blocked website, as displayed through the destination IP address and the destination port of 23, indicating the usage of *telnet*.



```
No.  Time                  Source          Destination      Protocol Length Info
   1 2021-02-14 22:05:…  192.168.53.5    128.119.245.12   TCP        60 56428 → 23 [SYN] Seq=1354170503 Win=29200 L…
```

Exhibit 3.12: Similar to Exhibit 3.9, viewing the captured network traffic on enp0s3 shows the UDP traffic between Machine A and Machine B for the VPN traffic, as well as the attempted *telnet* connection performed by Machine A (10.0.2.13), on behalf of the request by Machine B to do so.



```
No. Time  Source     Destination     Protocol Length Info
  5 20… 10.0.2.15   10.0.2.13       UDP      102 52895 → 55555 Len=60
  6 20… 10.0.2.13   128.119.245.12  TCP       74 56428 → 23 [SYN] Seq=1354170503 Win=29200 Len=…
  7 20… 10.0.2.15   10.0.2.13       UDP      102 52895 → 55555 Len=60
```

Conclusion

In summary, the usage of VPN's allows hosts to bypass firewalls which reside on their networks. In this lab, the usage of a VPN allowed Machine B to access a website which was previously blocked by the firewall via a VPN which was configured between Machine B and Machine A.

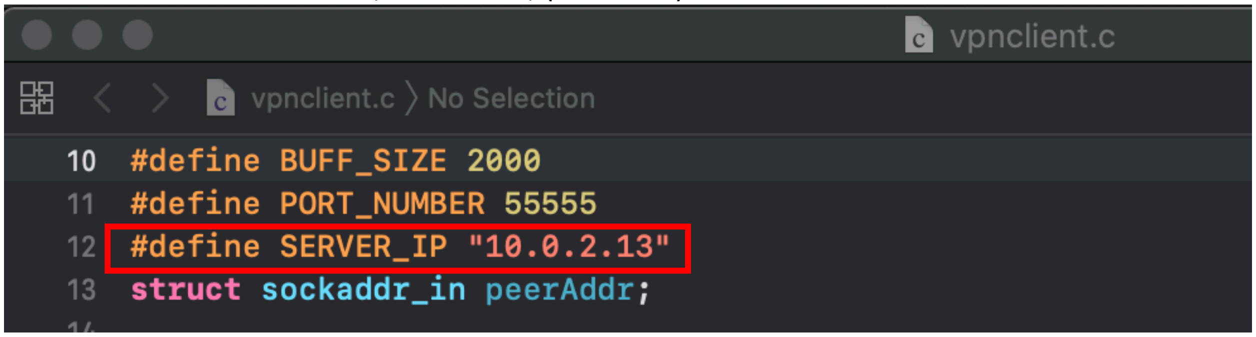
The key component which allows the VPN to function properly is the usage of Machine B as a packet forwarding device that sends the desired packets to Machine A. Using this technique, the desired traffic to the blocked website is provided to Machine A through the VPN tunnel, rather than using Machine A's standard internet interface (enp0s3).

Ultimately, the firewall on Machine B does not filter the incoming traffic to Machine B using the VPN tunneling mechanism, allowing Machine B to access the previously blocked website.

This lab report was completed in relation to SEED Lab's "Firewall Evasion Lab: Bypassing Firewalls using VPN".

Appendix

Exhibit A.1: Single configuration which was changed in the vpnclient.c file, which points the VPN client to the VPN server, Machine A, (10.0.2.13) on the network.



```
#define BUFF_SIZE 2000
#define PORT_NUMBER 55555
#define SERVER_IP "10.0.2.13"
struct sockaddr_in peerAddr;
```