

CS 491

Assignment 5

665336275

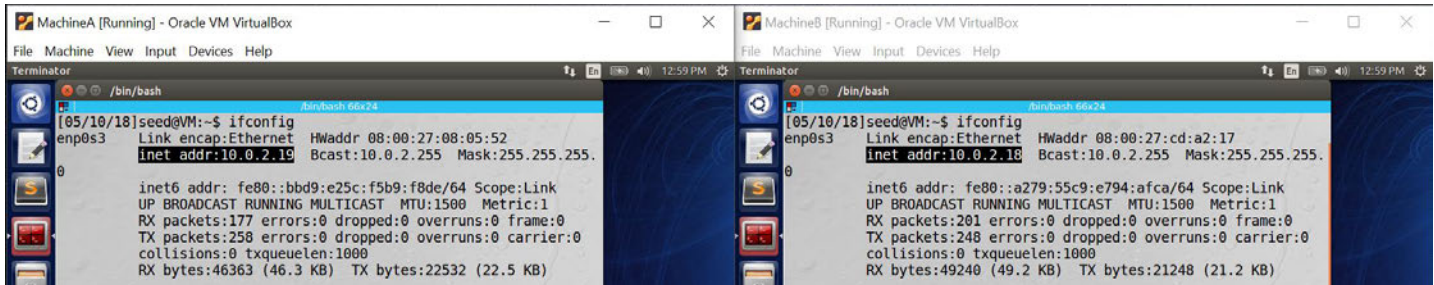
Akshay Kataria

Task 1: VM Setup

VM (Virtual Machine) IP address won't change for the entire lab experiment. We will implement the UFW (Uncomplicated Firewall) on MachineA / Client machine.

IP address: 10.0.2.19
Client (MachineA)

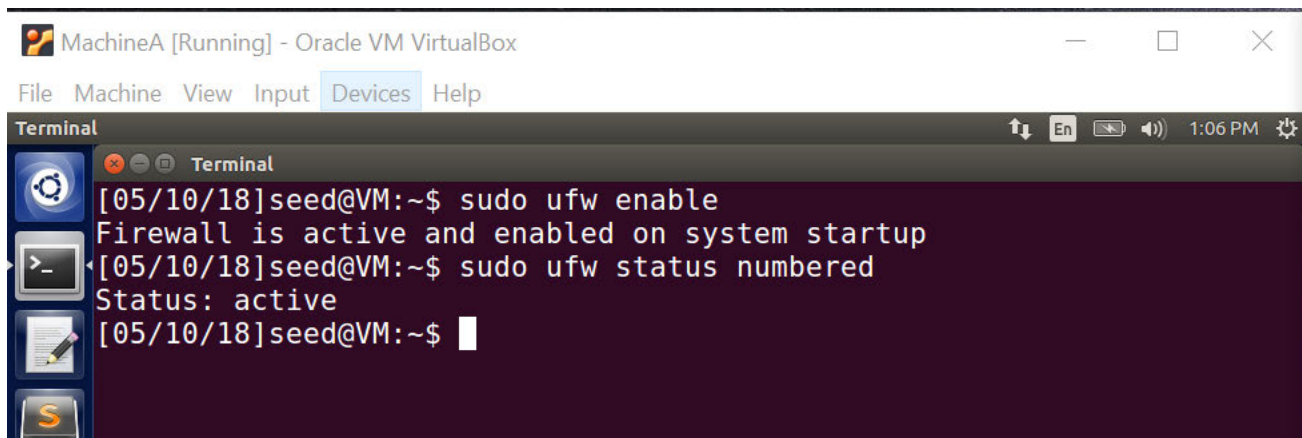
IP address: 10.0.2.18
Server (MachineB)



Obs: We can observe that the VM's are up and running and the 2 machines have been assigned the IP address.

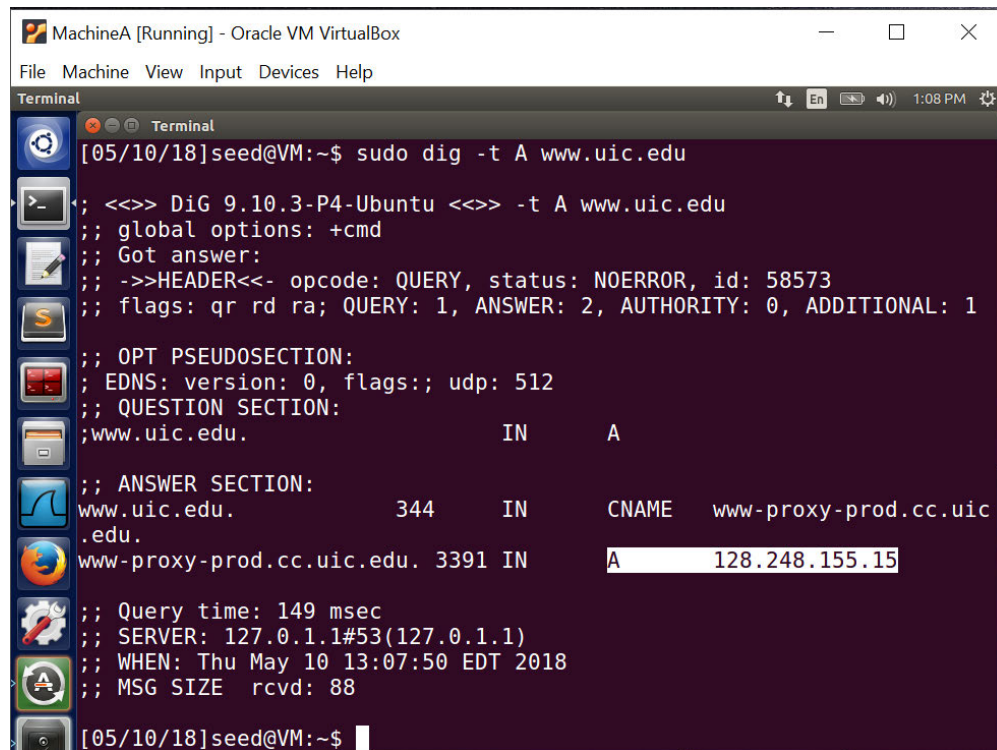
Task 2: Set up the Firewall

Organizations often block the internal users from accessing certain external sites (known as Egress Filtering). For this task, we will be working on the UFW (Uncomplicated Firewall) to block MachineA from reaching out to a particular website.



Obs: We can observe that UFW (Uncomplicated Firewall) have been installed and enabled on machineA and is working fine. There is no rule appearing after "sudo ufw status numbered" command because none has been mentioned yet.

For this Lab experiment, we will try to search the ip address of UIC and will try to block the traffic from machineA to UIC website.



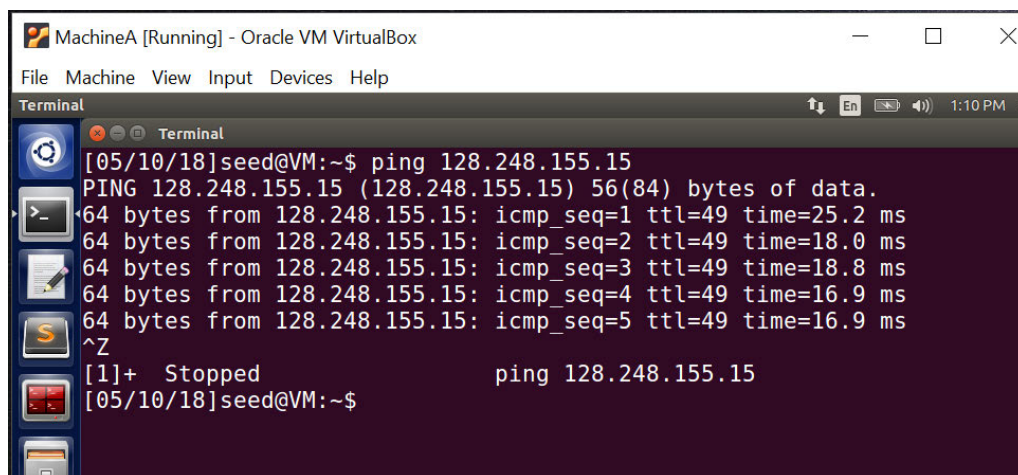
```
MachineA [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
[05/10/18]seed@VM:~$ sudo dig -t A www.uic.edu

;; <<>> DiG 9.10.3-P4-Ubuntu <<>> -t A www.uic.edu
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 58573
;; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1
;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;www.uic.edu.                IN      A
;; ANSWER SECTION:
www.uic.edu.                 344     IN      CNAME   www-proxy-prod.cc.uic
.edu.
www-proxy-prod.cc.uic.edu. 3391 IN      A       128.248.155.15

;; Query time: 149 msec
;; SERVER: 127.0.1.1#53(127.0.1.1)
;; WHEN: Thu May 10 13:07:50 EDT 2018
;; MSG SIZE rcvd: 88

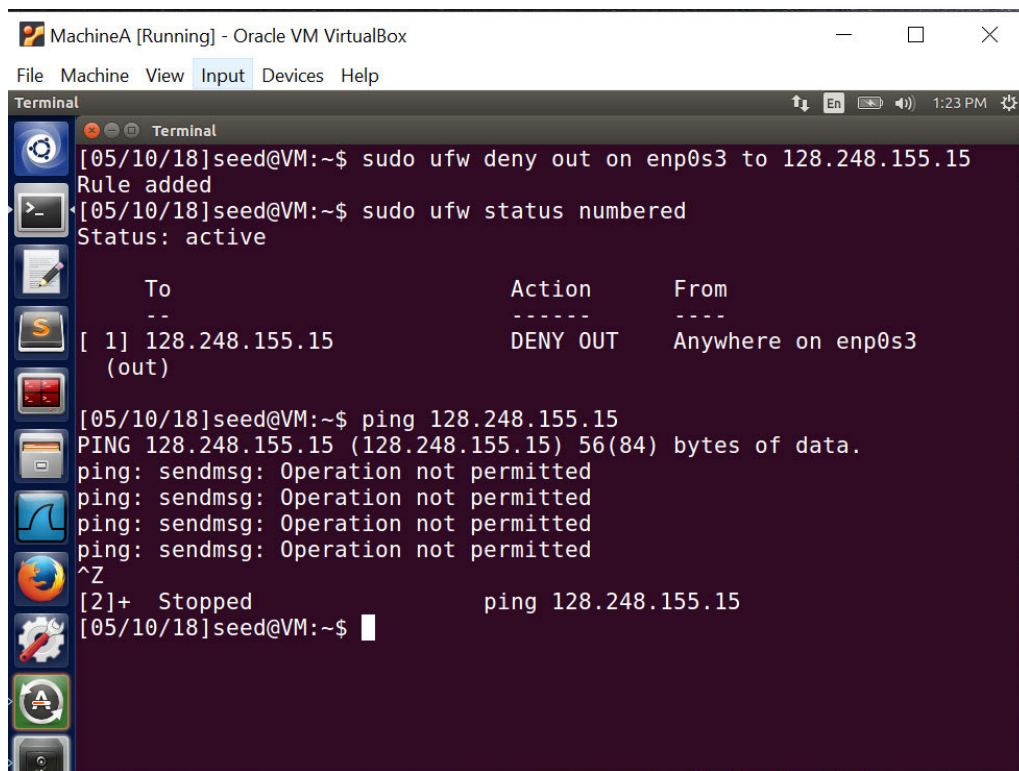
[05/10/18]seed@VM:~$
```

Obs: We are now trying to block access to a particular website from machine A. For our lab we are blocking “www.uic.edu”. We found out it’s address with the help of ‘sudo dig -t A www.uic.edu’ command. Address of website to be blocked is: 128.248.155.15.



```
MachineA [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
[05/10/18]seed@VM:~$ ping 128.248.155.15
PING 128.248.155.15 (128.248.155.15) 56(84) bytes of data.
64 bytes from 128.248.155.15: icmp_seq=1 ttl=49 time=25.2 ms
64 bytes from 128.248.155.15: icmp_seq=2 ttl=49 time=18.0 ms
64 bytes from 128.248.155.15: icmp_seq=3 ttl=49 time=18.8 ms
64 bytes from 128.248.155.15: icmp_seq=4 ttl=49 time=16.9 ms
64 bytes from 128.248.155.15: icmp_seq=5 ttl=49 time=16.9 ms
^Z
[1]+  Stopped                  ping 128.248.155.15
[05/10/18]seed@VM:~$
```

Obs: We can observe that before any rule in UFW is initialized to block the traffic from machineA, we are able to ping “www.uic.edu” (128.248.155.15) with the help of the ping command. It shows that we able to access this website.

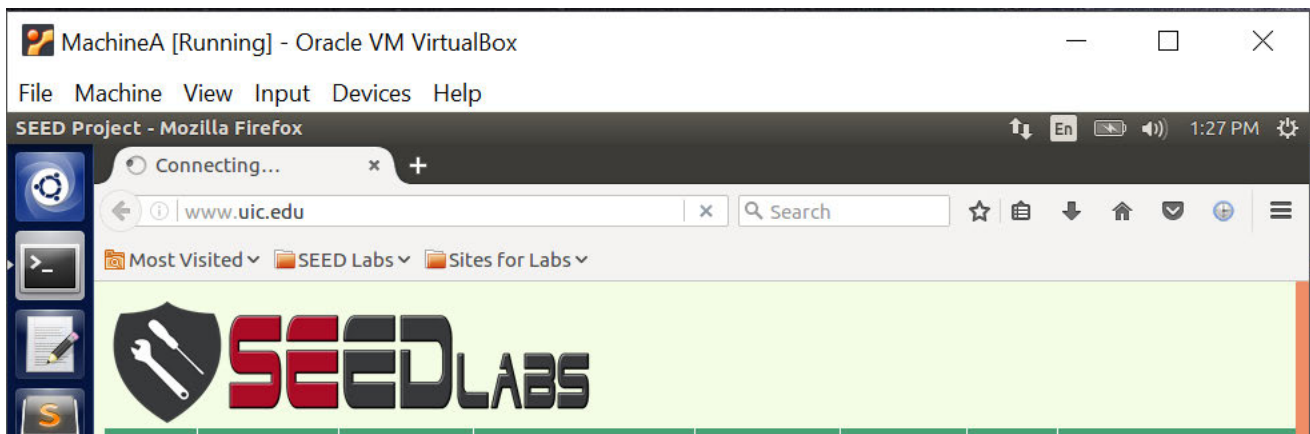


```
MachineA [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
[05/10/18]seed@VM:~$ sudo ufw deny out on enp0s3 to 128.248.155.15
Rule added
[05/10/18]seed@VM:~$ sudo ufw status numbered
Status: active

      To      Action      From
      --      -
[ 1] 128.248.155.15 DENY OUT    Anywhere on enp0s3
      (out)

[05/10/18]seed@VM:~$ ping 128.248.155.15
PING 128.248.155.15 (128.248.155.15) 56(84) bytes of data.
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
^Z
[2]+  Stopped                  ping 128.248.155.15
[05/10/18]seed@VM:~$
```

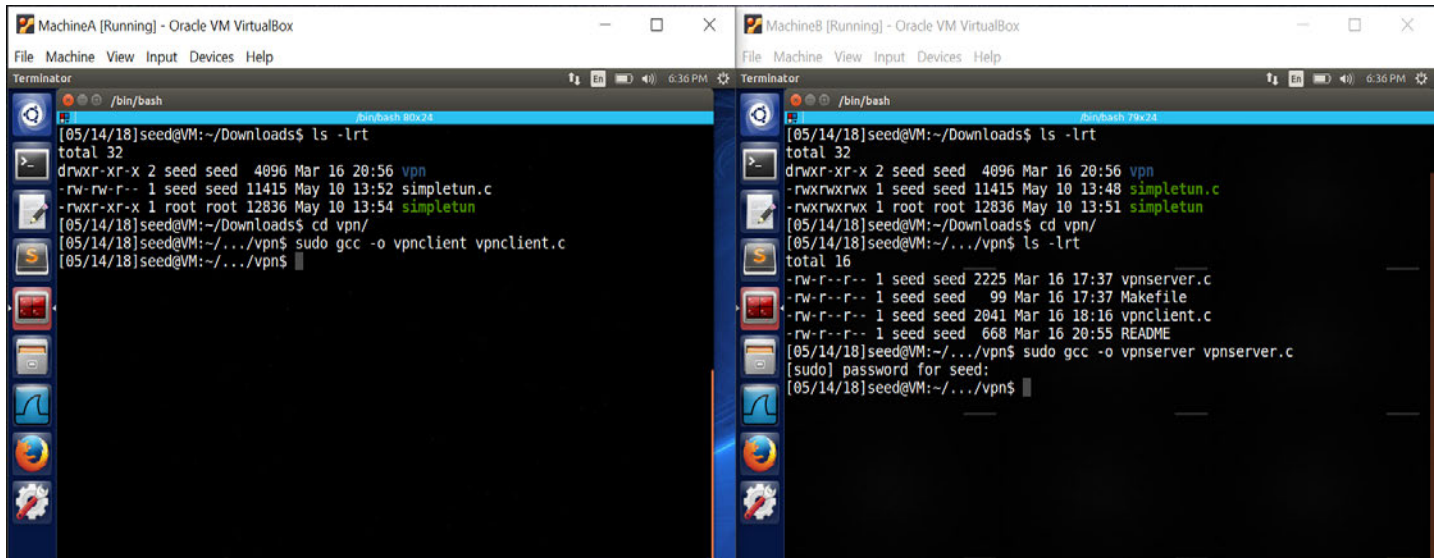
Obs: We can now observe that we are not able to ping “www.uic.edu” (128.248.155.15). This is because we have implemented a rule on ufw (sudo ufw deny out on enp0s3 to 128.248.155.15) that prevents machineA from reaching that particular website. This can be seen as rule 1 in the ufw rule table. You can also observe “Operation not permitted”.



Obs: We can also observe from the web browser that we are not able to reach out to “www.uic.edu” website.

Task 3: Bypassing Firewall using VPN

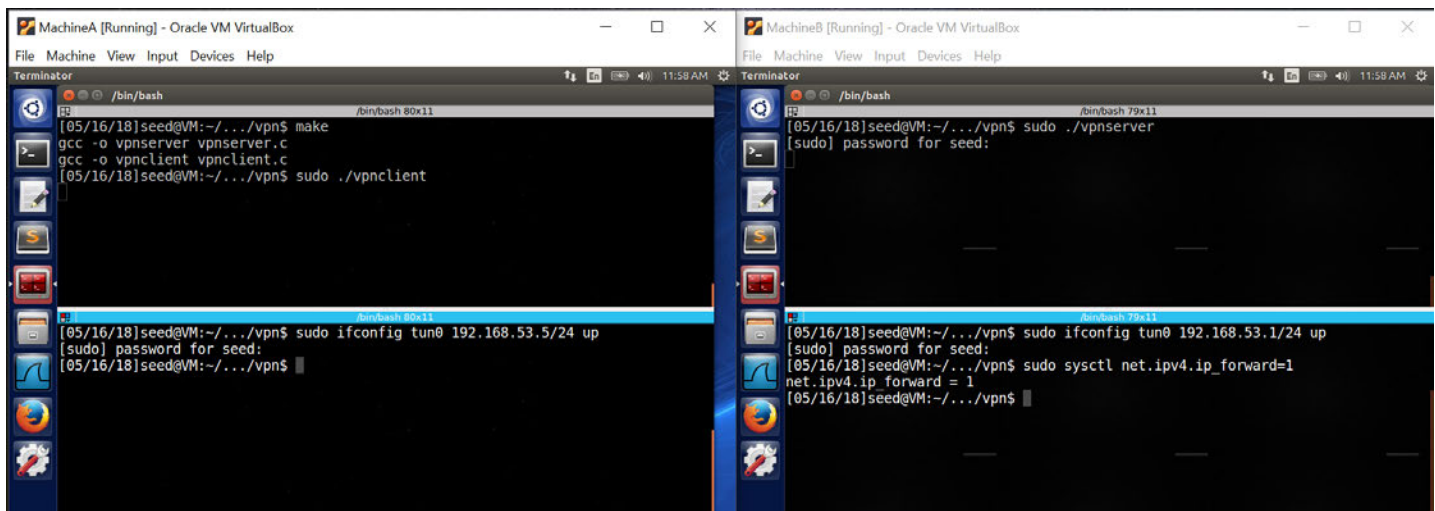
In this task, we will try to bypass the UFW firewall rules by creating a VPN tunnel between machineA (client) and the machineB (server).



```
MachineA [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminator
/bin/bash
[05/14/18]seed@VM:~/Downloads$ ls -lrt
total 32
drwxr-xr-x 2 seed seed 4096 Mar 16 20:56 vpn
-rw-r--r-- 1 seed seed 11415 May 10 13:52 simpletun.c
-rwxr-xr-x 1 root root 12836 May 10 13:54 simpletun
[05/14/18]seed@VM:~/Downloads$ cd vpn/
[05/14/18]seed@VM:~/vpn$ sudo gcc -o vpnclient vpnclient.c
[05/14/18]seed@VM:~/vpn$

MachineB [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminator
/bin/bash
[05/14/18]seed@VM:~/Downloads$ ls -lrt
total 32
drwxr-xr-x 2 seed seed 4096 Mar 16 20:56 vpn
-rwxr-xr-x 1 seed seed 11415 May 10 13:48 simpletun.c
-rwxr-xr-x 1 root root 12836 May 10 13:51 simpletun
[05/14/18]seed@VM:~/Downloads$ cd vpn/
[05/14/18]seed@VM:~/vpn$ ls -lrt
total 16
-rw-r--r-- 1 seed seed 2225 Mar 16 17:37 vpnsrvr.c
-rw-r--r-- 1 seed seed 99 Mar 16 17:37 Makefile
-rw-r--r-- 1 seed seed 2041 Mar 16 18:16 vpnclient.c
-rw-r--r-- 1 seed seed 668 Mar 16 20:55 README
[05/14/18]seed@VM:~/vpn$ sudo gcc -o vpnsrvr vpnsrvr.c
[sudo] password for seed:
[05/14/18]seed@VM:~/vpn$
```

Obs: To implement a VPN tunnel in the virtual machines, “vpnclient.c” was downloaded and compiled with root privileges with the help of “sudo gcc -o vpnclient vpnclient.c” on the vpn client/machineA. Similarly, vpnsrvr.c was downloaded and compiled on vpn server/machineB.



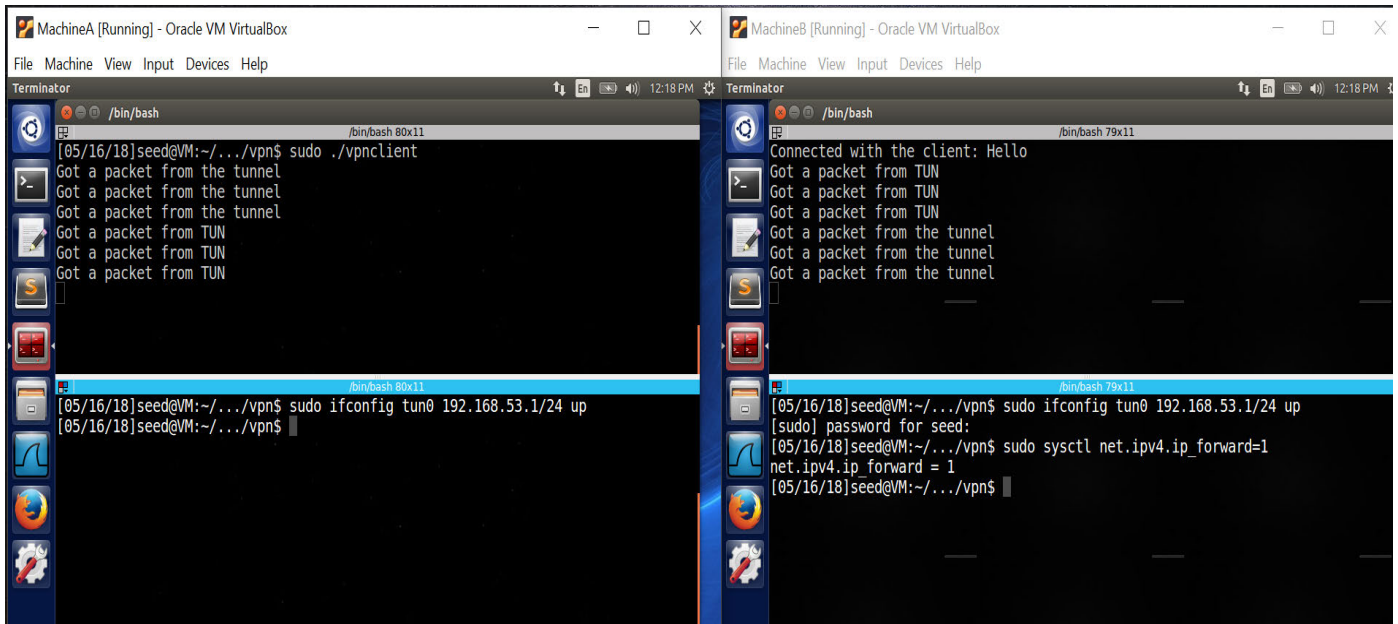
```
MachineA [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminator
/bin/bash
[05/16/18]seed@VM:~/vpn$ make
gcc -o vpnsrvr vpnsrvr.c
gcc -o vpnclient vpnclient.c
[05/16/18]seed@VM:~/vpn$ sudo ./vpnclient

MachineB [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminator
/bin/bash
[05/16/18]seed@VM:~/vpn$ sudo ./vpnsrvr
[sudo] password for seed:

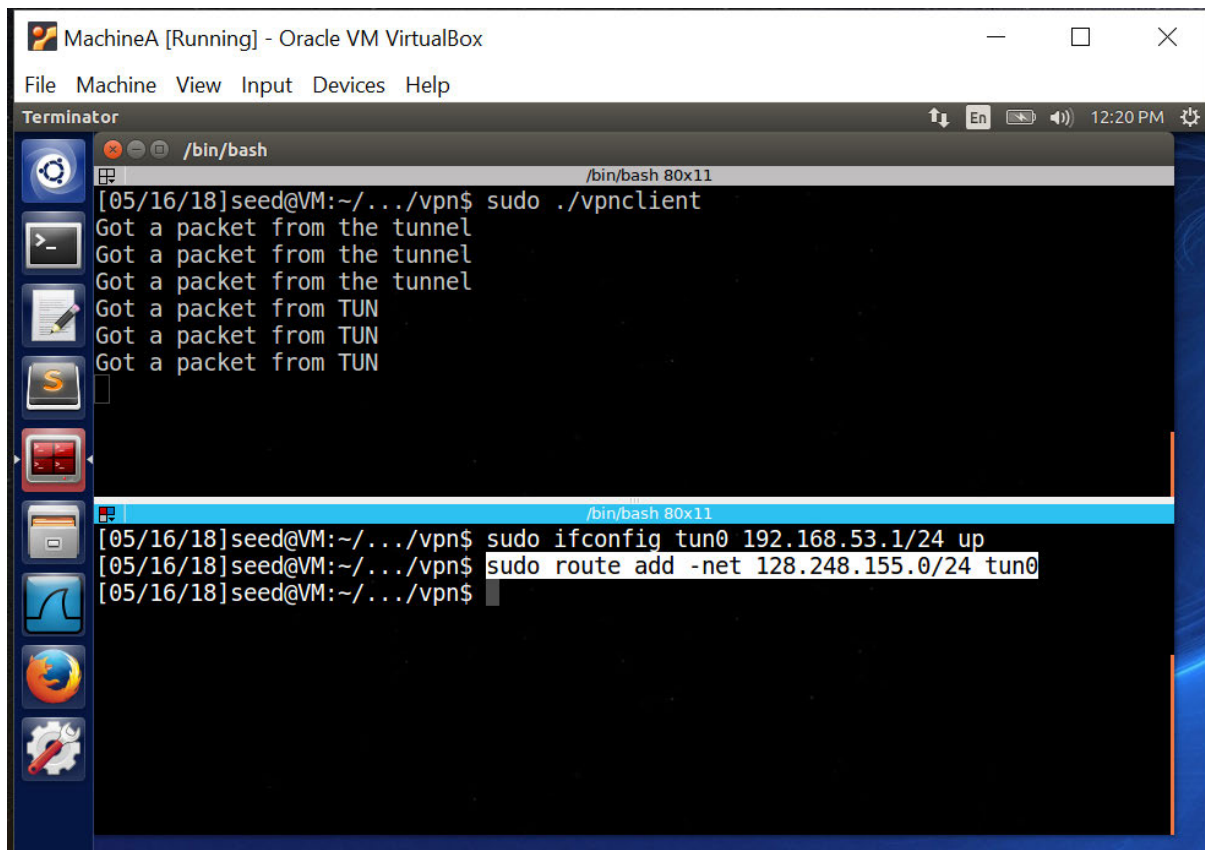
[05/16/18]seed@VM:~/vpn$ sudo ifconfig tun0 192.168.53.5/24 up
[05/16/18]seed@VM:~/vpn$

[05/16/18]seed@VM:~/vpn$ sudo ifconfig tun0 192.168.53.1/24 up
[sudo] password for seed:
[05/16/18]seed@VM:~/vpn$ sudo sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
[05/16/18]seed@VM:~/vpn$
```

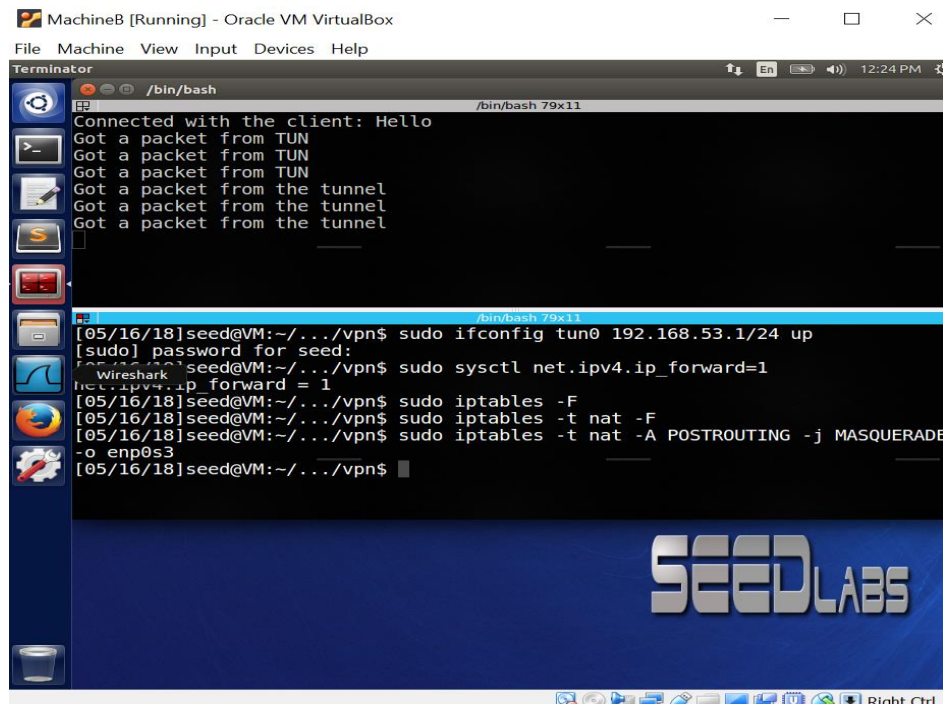
Obs: It can be observed that we first open up the vpnsrvr by “sudo ./vpnsrvr” and then assigned that interface an ip address by “sudo ifconfig tun0 192.168.53.1/24 up” in a different terminal window. Similarly, we open the vpnclient machine interface for tunneling by “sudo ./vpnclient” and then assigned that interface an ip address by “sudo ifconfig tun0 192.168.53.5/24 up”. After this setup, we want to make sure that we have enabled the IP forwarding on the VPN server using command “sudo sysctl net.ipv4.ip_forward=1”.



Obs: It can be observed that as soon as the tunnel was up, client and server started sending packets to each other. Confirms that the tunnel has been successfully established.



Obs: Now, we can configure a route from the VPN client to the VPN server via VPN tunnel interface “tun0” with the help of the “sudo route add -net 128.248.155.0/24 tun0”.



Obs: It can be observed that after the addition of route, we need to add the NAT on VPN Server. This can be done with the help of following commands:

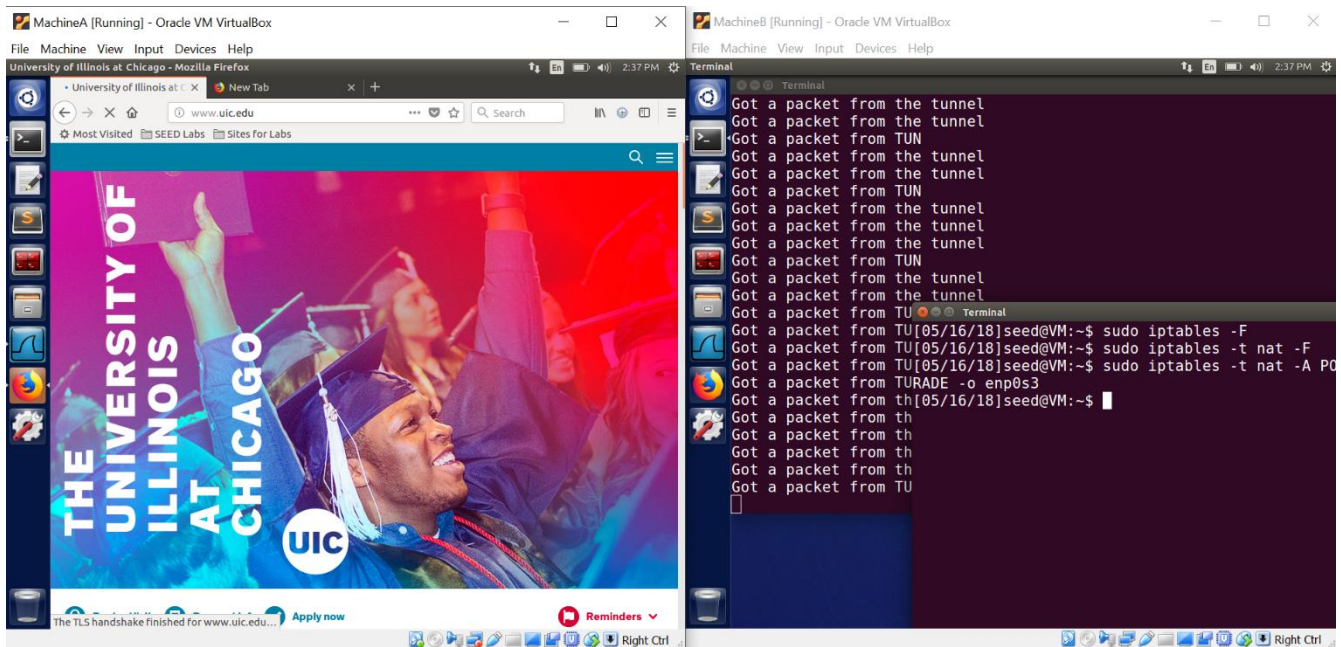
- `sudo iptables -F`
- `sudo iptables -t nat -F`
- `sudo iptables -t nat -A POSTROUTING -j MASQUERADE -o enp0s3`

After the setup is complete, you can ping the external website that has been blocked on the VPN Client and but can be reached out through VPN Server.

ping instance.pcapng

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	10.0.2.19	10.0.2.18	UDP	126	36652 → 55555 Len=84
2	0.004278153	10.0.2.18	128.248.155.15	ICMP	98	Echo (ping) request id=0x09df, seq=1/256, ttl=63 (reply in 3)
3	0.052700879	128.248.155.15	10.0.2.18	ICMP	98	Echo (ping) reply id=0x09df, seq=1/256, ttl=50 (request in 2)
4	0.054636512	10.0.2.18	10.0.2.19	UDP	126	55555 → 36652 Len=84
5	1.003382078	10.0.2.19	10.0.2.18	UDP	126	36652 → 55555 Len=84
6	1.007462420	10.0.2.18	128.248.155.15	ICMP	98	Echo (ping) request id=0x09df, seq=2/512, ttl=63 (reply in 7)
7	1.048454587	128.248.155.15	10.0.2.18	ICMP	98	Echo (ping) reply id=0x09df, seq=2/512, ttl=50 (request in 6)
8	1.051767489	10.0.2.18	10.0.2.19	UDP	126	55555 → 36652 Len=84
9	2.008843409	10.0.2.19	10.0.2.18	UDP	126	36652 → 55555 Len=84
10	2.022213888	10.0.2.18	128.248.155.15	ICMP	98	Echo (ping) request id=0x09df, seq=3/768, ttl=63 (reply in 11)
11	2.077577247	128.248.155.15	10.0.2.18	ICMP	98	Echo (ping) reply id=0x09df, seq=3/768, ttl=50 (request in 10)
12	2.084165702	10.0.2.18	10.0.2.19	UDP	126	55555 → 36652 Len=84
13	3.010319162	10.0.2.19	10.0.2.18	UDP	126	36652 → 55555 Len=84
14	3.023496367	10.0.2.18	128.248.155.15	ICMP	98	Echo (ping) request id=0x09df, seq=4/1024, ttl=63 (reply in 15)
15	3.067328158	128.248.155.15	10.0.2.18	ICMP	98	Echo (ping) reply id=0x09df, seq=4/1024, ttl=50 (request in 14)
16	3.070649332	10.0.2.18	10.0.2.19	UDP	126	55555 → 36652 Len=84

Obs: It can be clearly observed from the packets captured from wireshark that whenever VPN client (10.0.2.19) tries to ping 128.248.155.15, the ICMP ECHO Request traffic goes through the VPN server (10.0.2.18) and ICMP ECHO Reply reaches VPN Client via VPN Server only. This shows that the tunnel is working perfectly and we are able to bypass the UFW firewall as is doesn't account for the DPI (Deep Packet Inspection).



Obs: I have also tried to reach out to the blocked website through the browser and as we can observe, we are able to resolve the domain name for that website. It can be safely presumed that the VPN tunnel is working just fine.

451	13.698086182	10.0.2.19	192.168.1.254	DNS	71 Standard query 0x5cf1 A www.uic.edu
452	13.700316804	10.0.2.19	192.168.1.254	DNS	71 Standard query 0xb31e AAAA www.uic.edu
453	13.750581446	192.168.1.254	10.0.2.19	DNS	119 Standard query response 0x5cf1 A www.uic.edu CNAME www-proxy-prod.cc.uic.edu A 128.2
454	13.750602165	192.168.1.254	10.0.2.19	DNS	156 Standard query response 0xb31e AAAA www.uic.edu CNAME www-proxy-prod.cc.uic.edu SOA
455	13.790796389	10.0.2.19	72.21.91.29	TCP	54 46106 → 80 [FIN, ACK] Seq=1 Ack=1 Win=29200 Len=0
456	13.792782734	10.0.2.19	52.10.151.19	TLSv1...	85 Encrypted Alert

Obs: From the captured wireshark packets, you can observe that the query about www.uic.edu (128.248.155.15) that was initiated by VPN Client (10.0.2.19) is getting resolved via VPN Server. It can be deduced that the VPN tunnel is working fine.