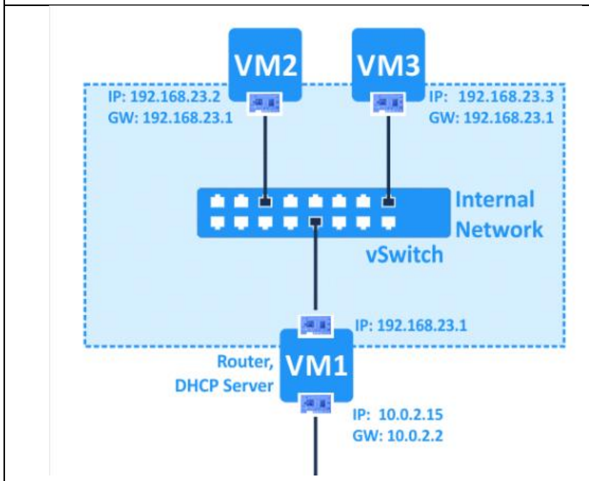
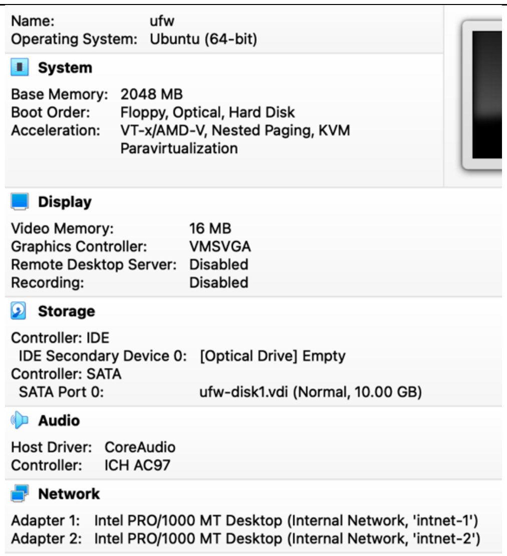
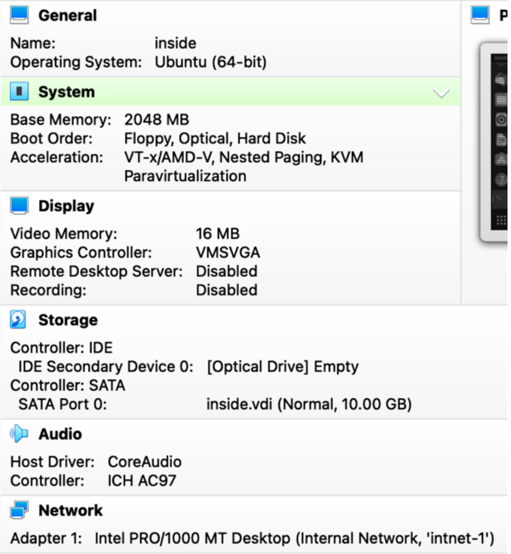
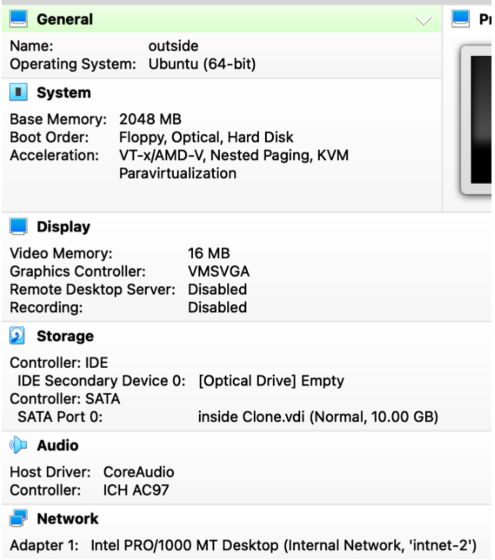
Introduction/Overview:

For this project, we will build a firewall with many attributes that I will discuss later in this report. Before I describe the nature of my firewall, I'd want to demonstrate and explain the procedure of configuring virtual machines for the project environment. I set up three virtual machines for the project environment. The first will be the inside client, the second the outer client, and the third the firewall. All three computers are running Ubuntu to run these various VMs, but instead of using pfSense for the firewall, I chose ufw, which I found to be quite friendly with Ubuntu. The reason for this adjustment was that while pfSense and the client VM were connected, I was unable to access the WebGui, which would allow me to make changes to the firewall. The 3 VMs were setup using an internal adapter which sets my project environment on an isolated virtual network similar to or just like this image (**image 1**). VM1 represents the firewall, VM2 represent outside client, and VM3 represent inside client. Because we have to simulate the outside client on a sever and the inside client already on the network, VM1 has to have two adapters (**image 2**). Adapter 1 with IP address 192.168.1.1 is named *intnet-1* is for VM3 with IP address 192.168.1.15 (**image 3**) and Adapter 2 is with IP address 192.168.3.1 is named *intnet-2* is for VM2 with IP address 192.168.3.15 (**image 4)**. To present my findings and demonstrate what happened during each rule, I had to reset the firewall each time so that I didn't accidentally lock myself out of my firewall or cause a mess.

Image 1	Image 2
 VM2 VM3 IP: 192.168.23.2 GW: 192.168.23.1 IP: 192.168.23.3 GW: 192.168.23.1 Internal Network vSwitch IP: 192.168.23.1 Router, DHCP Server VM1 IP: 10.0.2.15 GW: 10.0.2.2	Name: ufw Operating System: Ubuntu (64-bit) **System** Base Memory: 2048 MB Boot Order: Floppy, Optical, Hard Disk Acceleration: VT-x/AMD-V, Nested Paging, KVM Paravirtualization **Display** Video Memory: 16 MB Graphics Controller: VMSVGA Remote Desktop Server: Disabled Recording: Disabled **Storage** Controller: IDE IDE Secondary Device 0: [Optical Drive] Empty Controller: SATA SATA Port 0: ufw-disk1.vdi (Normal, 10.00 GB) **Audio** Host Driver: CoreAudio Controller: ICH AC97 **Network** Adapter 1: Intel PRO/1000 MT Desktop (Internal Network, 'intnet-1') Adapter 2: Intel PRO/1000 MT Desktop (Internal Network, 'intnet-2')
Image 3	Image 4



```
General
Name:               inside
Operating System:   Ubuntu (64-bit)
System
Base Memory:   2048 MB
Boot Order:    Floppy, Optical, Hard Disk
Acceleration:  VT-x/AMD-V, Nested Paging, KVM
               Paravirtualization
Display
Video Memory:            16 MB
Graphics Controller:     VMSVGA
Remote Desktop Server:   Disabled
Recording:               Disabled
Storage
Controller: IDE
  IDE Secondary Device 0:   [Optical Drive] Empty
Controller: SATA
  SATA Port 0:              inside.vdi (Normal, 10.00 GB)
Audio
Host Driver:   CoreAudio
Controller:    ICH AC97
Network
Adapter 1:   Intel PRO/1000 MT Desktop (Internal Network, 'intnet-1')
```

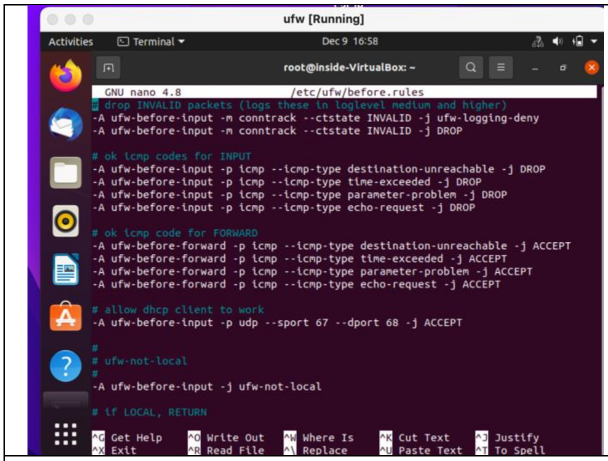


```
General
Name:               outside
Operating System:   Ubuntu (64-bit)
System
Base Memory:   2048 MB
Boot Order:    Floppy, Optical, Hard Disk
Acceleration:  VT-x/AMD-V, Nested Paging, KVM
               Paravirtualization
Display
Video Memory:            16 MB
Graphics Controller:     VMSVGA
Remote Desktop Server:   Disabled
Recording:               Disabled
Storage
Controller: IDE
  IDE Secondary Device 0:   [Optical Drive] Empty
Controller: SATA
  SATA Port 0:              inside Clone.vdi (Normal, 10.00 GB)
Audio
Host Driver:   CoreAudio
Controller:    ICH AC97
Network
Adapter 1:   Intel PRO/1000 MT Desktop (Internal Network, 'intnet-2')
```

Section 1: Block external ICMP messages (ping, tracerout, etc), but should allow these from interior clients

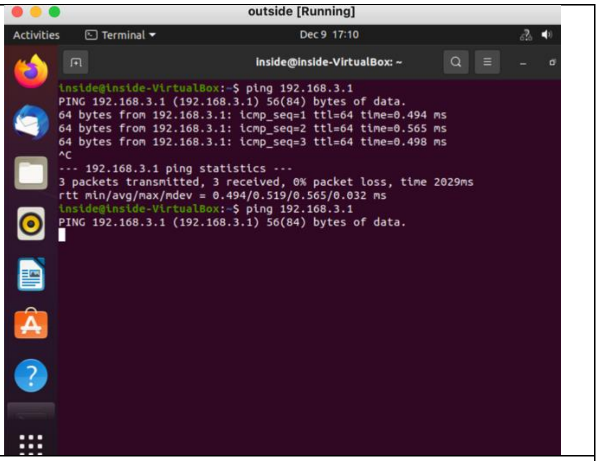
For this firewall rule, this will prevent an external client with IP address 192.168.3.15 from sending an ICMP message to the firewall while allowing an internal client with IP address 192.168.1.15 to ICMP the firewall. Unlike the other properties that will be done in this project, this rule will not be applied or changed using the command line. To activate this rule, I had to modify a file named *before.rules*, which contains the rights to ICMP the firewall. The only method to change this file was to log in as root on the ufw VM. To change the file, I used *nano /etc/ufw/before,rules* after gaining root access. I then had to go to the part under "*ok icmp codes for INPUT*" and modify those four lines of code from "*ACCEPT*" to "*DROP*" (**image 5**). After that, make sure to reboot ufw so that the modification takes effect. There is a before and after on the external client from pinging the firewall in **image 6**. Instead of notifying you that you can't access the host, Ubuntu would hang for a lengthy amount of time before exiting and letting you know that the host is unreachable. To enable the interior client to just ping the firewall VM, you would add a line of code to the *before.rules* file (**image 7**).The line is after the comment that says *"#ok icmp code for internal client*". [Because it cuts off here the full line is: *-A ufw-before-input -p icmp --icmp-type echo-request -s 192.168.1.15 -m state --state ESTABLISHED -j ACCEPT*] From there, the interior client will be the only one to ping the firewall (**image 8**).

Image 5	Image 6



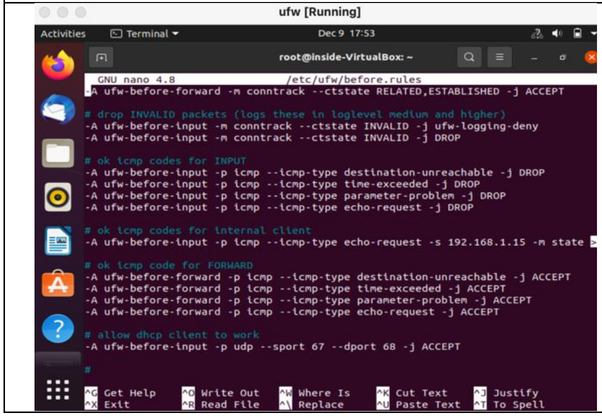
```
  GNU nano 4.8                    /etc/ufw/before.rules
# drop INVALID packets (logs these in loglevel medium and higher)
-A ufw-before-input -m conntrack --ctstate INVALID -j ufw-logging-deny
-A ufw-before-input -m conntrack --ctstate INVALID -j DROP

# ok icmp codes for INPUT
-A ufw-before-input -p icmp --icmp-type destination-unreachable -j DROP
-A ufw-before-input -p icmp --icmp-type time-exceeded -j DROP
-A ufw-before-input -p icmp --icmp-type parameter-problem -j DROP
-A ufw-before-input -p icmp --icmp-type echo-request -j DROP

# ok icmp code for FORWARD
-A ufw-before-forward -p icmp --icmp-type destination-unreachable -j ACCEPT
-A ufw-before-forward -p icmp --icmp-type time-exceeded -j ACCEPT
-A ufw-before-forward -p icmp --icmp-type parameter-problem -j ACCEPT
-A ufw-before-forward -p icmp --icmp-type echo-request -j ACCEPT

# allow dhcp client to work
-A ufw-before-input -p udp --sport 67 --dport 68 -j ACCEPT

#
# ufw-not-local
#
-A ufw-before-input -j ufw-not-local

# if LOCAL, RETURN
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify
^X Exit        ^R Read File   ^\ Replace     ^U Paste Text  ^T To Spell
```

Image 7

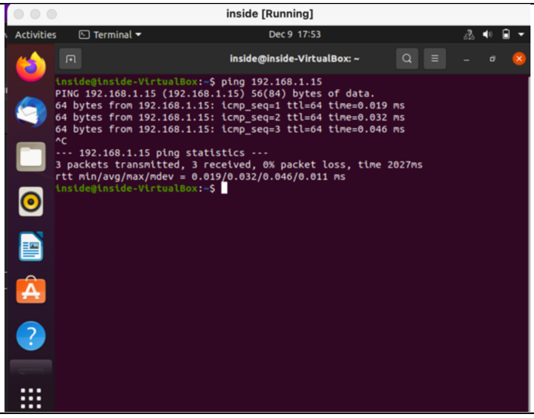


```
inside@inside-VirtualBox:~$ ping 192.168.3.1
PING 192.168.3.1 (192.168.3.1) 56(84) bytes of data.
64 bytes from 192.168.3.1: icmp_seq=1 ttl=64 time=0.494 ms
64 bytes from 192.168.3.1: icmp_seq=2 ttl=64 time=0.565 ms
64 bytes from 192.168.3.1: icmp_seq=3 ttl=64 time=0.498 ms
^C
--- 192.168.3.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2029ms
rtt min/avg/max/mdev = 0.494/0.519/0.565/0.032 ms
inside@inside-VirtualBox:~$ ping 192.168.3.1
PING 192.168.3.1 (192.168.3.1) 56(84) bytes of data.
```

Image 8



```
  GNU nano 4.8                    /etc/ufw/before.rules
-A ufw-before-forward -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT

# drop INVALID packets (logs these in loglevel medium and higher)
-A ufw-before-input -m conntrack --ctstate INVALID -j ufw-logging-deny
-A ufw-before-input -m conntrack --ctstate INVALID -j DROP

# ok icmp codes for INPUT
-A ufw-before-input -p icmp --icmp-type destination-unreachable -j DROP
-A ufw-before-input -p icmp --icmp-type time-exceeded -j DROP
-A ufw-before-input -p icmp --icmp-type parameter-problem -j DROP
-A ufw-before-input -p icmp --icmp-type echo-request -j DROP

# ok icmp codes for internal client
-A ufw-before-input -p icmp --icmp-type echo-request -s 192.168.1.15 -m state >

# ok icmp code for FORWARD
-A ufw-before-forward -p icmp --icmp-type destination-unreachable -j ACCEPT
-A ufw-before-forward -p icmp --icmp-type time-exceeded -j ACCEPT
-A ufw-before-forward -p icmp --icmp-type parameter-problem -j ACCEPT
-A ufw-before-forward -p icmp --icmp-type echo-request -j ACCEPT

# allow dhcp client to work
-A ufw-before-input -p udp --sport 67 --dport 68 -j ACCEPT

#
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify
^X Exit        ^R Read File   ^\ Replace     ^U Paste Text  ^T To Spell
```

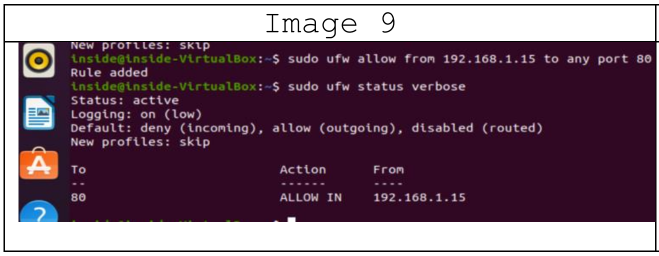


```
inside@inside-VirtualBox:~$ ping 192.168.1.15
PING 192.168.1.15 (192.168.1.15) 56(84) bytes of data.
64 bytes from 192.168.1.15: icmp_seq=1 ttl=64 time=0.019 ms
64 bytes from 192.168.1.15: icmp_seq=2 ttl=64 time=0.032 ms
64 bytes from 192.168.1.15: icmp_seq=3 ttl=64 time=0.046 ms
^C
--- 192.168.1.15 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2027ms
rtt min/avg/max/mdev = 0.019/0.032/0.046/0.011 ms
inside@inside-VirtualBox:~$
```

Section 2: Allow port 80 requests to the interior client

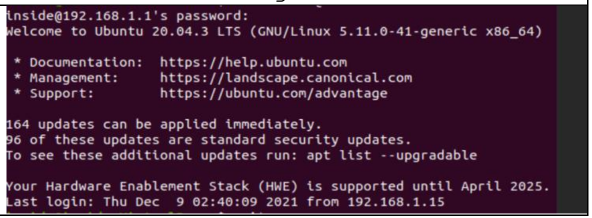
This firewall rule wasn't difficult to manage. This allows the internal client to connect to the firewall through port 80. In the ufw command line, type "*sudo ufw allow from 192.168.1.15 to any port 80*," as seen in **image 9**. To better display the firewall rule(s) in ufw, use "*sudo ufw status verbose*." After entering all of the credentials, your output for the interior client should look like **image 10**. (if you have that enabled). Also, in order for port 80 to operate, I had to create a rule to lets ports in so that the rule that I put in would work. This would be *sudo ufw allow 22/tcp*.

Image 9



```
New profiles: skip
inside@inside-VirtualBox:~$ sudo ufw allow from 192.168.1.15 to any port 80
Rule added
inside@inside-VirtualBox:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
80                         ALLOW IN    192.168.1.15
```

Image 10

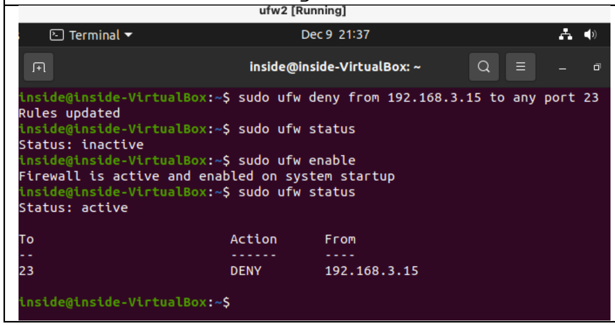


```
inside@192.168.1.1's password:
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.11.0-41-generic x86_64)

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

164 updates can be applied immediately.
96 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Thu Dec  9 02:40:09 2021 from 192.168.1.15
```

Section 3: Block external telnet, rlogin, and other similar requests

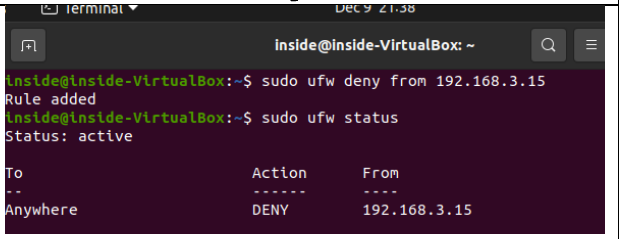
So for this particular firewall rule, the output for the exterior client would be similar to the first sections output. Where if you try to ping or ssh into that particular IP address, it would hang and after an extended amount of time, it would tell you that the host is unreachable. As so, in the ufw command line you would enter *sudo ufw deny from IP to any port 23*. Where IP represents the IP address you would want to deny and port 23 represents the port you would like to block them from **(image 11)**. I choose to add the port part to show you can either do that or another option like just *sudo ufw deny from IP* **(image 12).**

Image 11



```
inside@inside-VirtualBox:~$ sudo ufw deny from 192.168.3.15 to any port 23
Rules updated
inside@inside-VirtualBox:~$ sudo ufw status
Status: inactive
inside@inside-VirtualBox:~$ sudo ufw enable
Firewall is active and enabled on system startup
inside@inside-VirtualBox:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
23                         DENY        192.168.3.15

inside@inside-VirtualBox:~$
```

Image 12

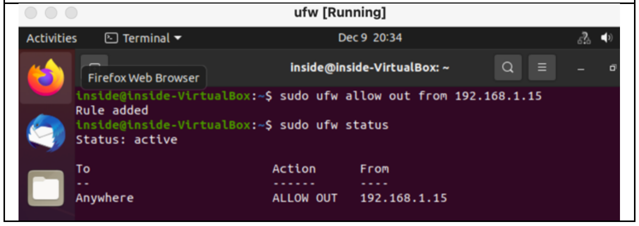


```
inside@inside-VirtualBox:~$ sudo ufw deny from 192.168.3.15
Rule added
inside@inside-VirtualBox:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
Anywhere                   DENY        192.168.3.15
```

Section 4: Allow internal messages using SMTP to be sent through the firewall

Finally, before departing, I'm attempting to allow my internal client to send SMTP messages via the firewall. To accomplish this, type *sudo ufw allow out from IP address* into the ufw command line. The IP address represents the specific user address or whichever IP address you want to apply this rule to. In this situation, the internal client IP address is 192.168.1.15 (as seen in **image 13**). Also if you wanted to instead of doing a particular user IP Address, you can specify the port number that you would allow the messages to leave from.

Image 13



```
inside@inside-VirtualBox:~$ sudo ufw allow out from 192.168.1.15
Rule added
inside@inside-VirtualBox:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
Anywhere                   ALLOW OUT   192.168.1.15
```

Conclusion:

Apart from learning about the various firewalls and the numerous types of rules that can be applied to them, this semester project broadened my understanding of how and in what ways the VMs may be utilized on my host system. According to numerous internet sources, users established and ran their own firewalls on their virtual machines (VMs) and used them to govern traffic that came in and out of their network. Although this endeavor was a fun and instructive experience, I did found myself experimenting with several firewalls to get a sense of how they were organized. Despite certain parallels in what they perform, they demonstrated alternative ways in which some rules may be implemented. Overall, it was a fantastic experience, and this project greatly aided my understanding of network connectivity.