

OLD DOMINION UNIVERSITY

CYSE 270 LINUX SYSTEM FOR CYBERSECURITY

Assignment #13 Advanced Network Configurations

John Wilson

01179411

CYSE 270 Assignment #13 Advanced Network Configurations

Scenario: You, as a network admin, are going to set up your Ubuntu VM as a gateway to provide Internet access to another client Ubuntu VM. The client VM needs to be in the same internal network as the gateway (as shown in Figure 1). Once the connection is ready, you need to configure the firewall to secure the network properly. The following requirements need to be satisfied to receive full credits.

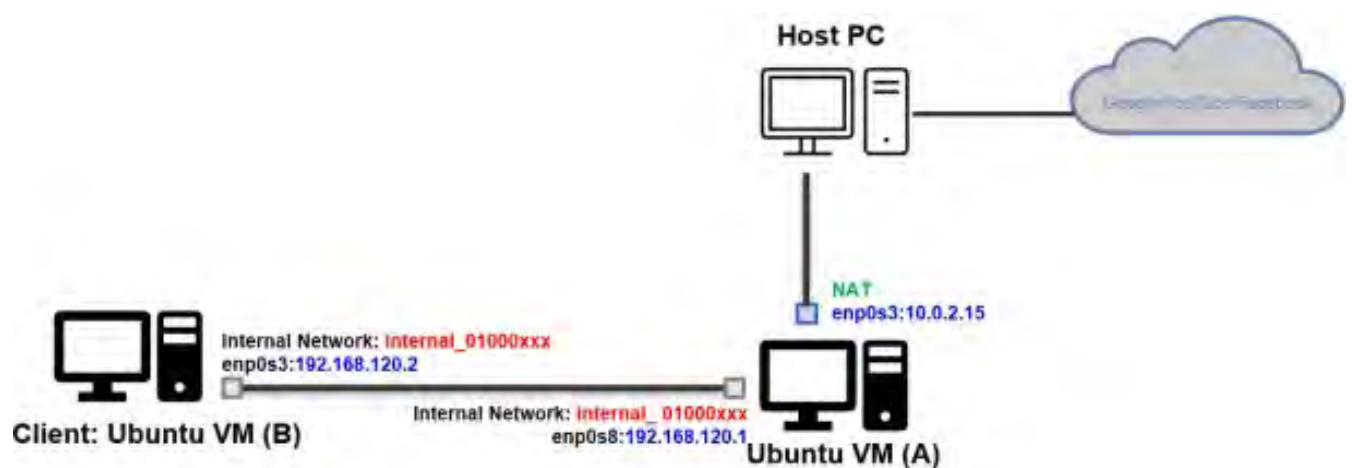


Figure 1 Desired Network Topology

Please note that you need to customize the value in the fields marked in RED above.

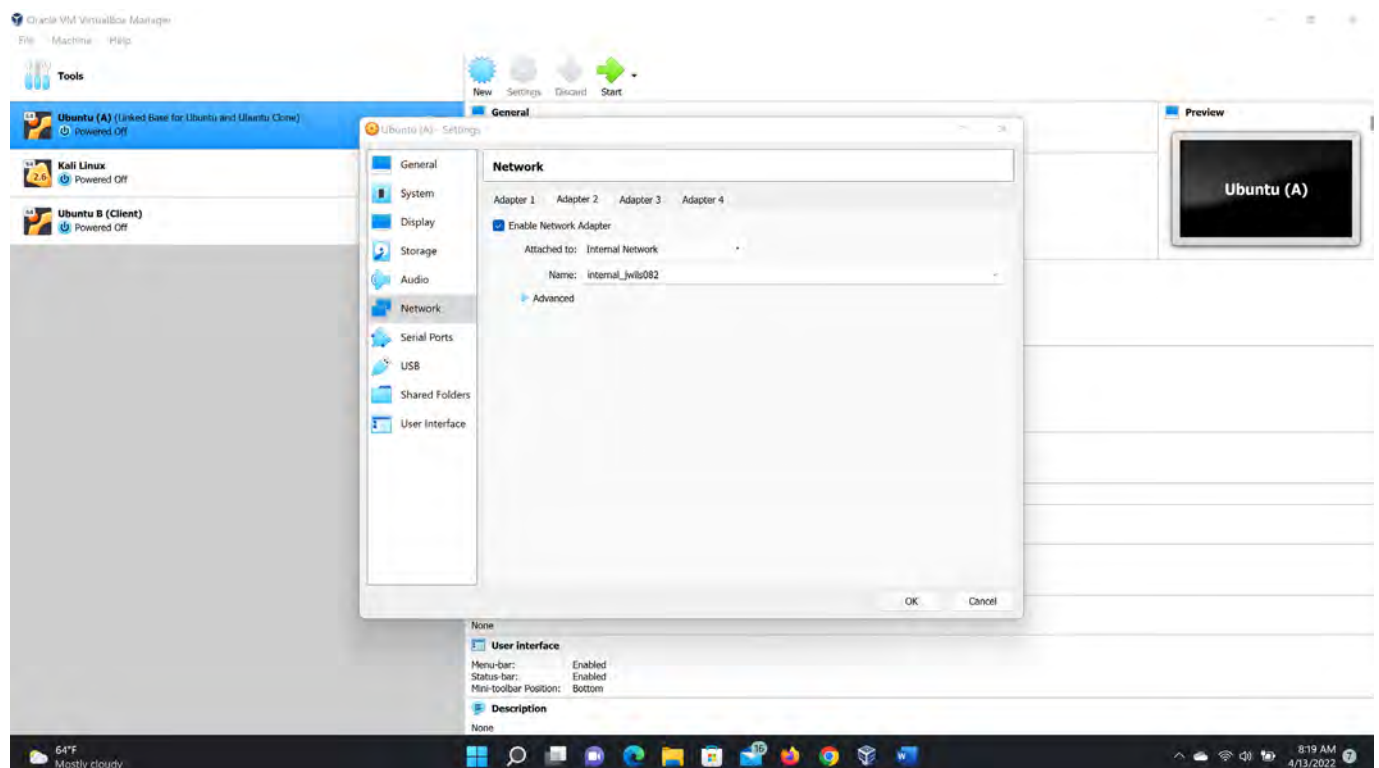
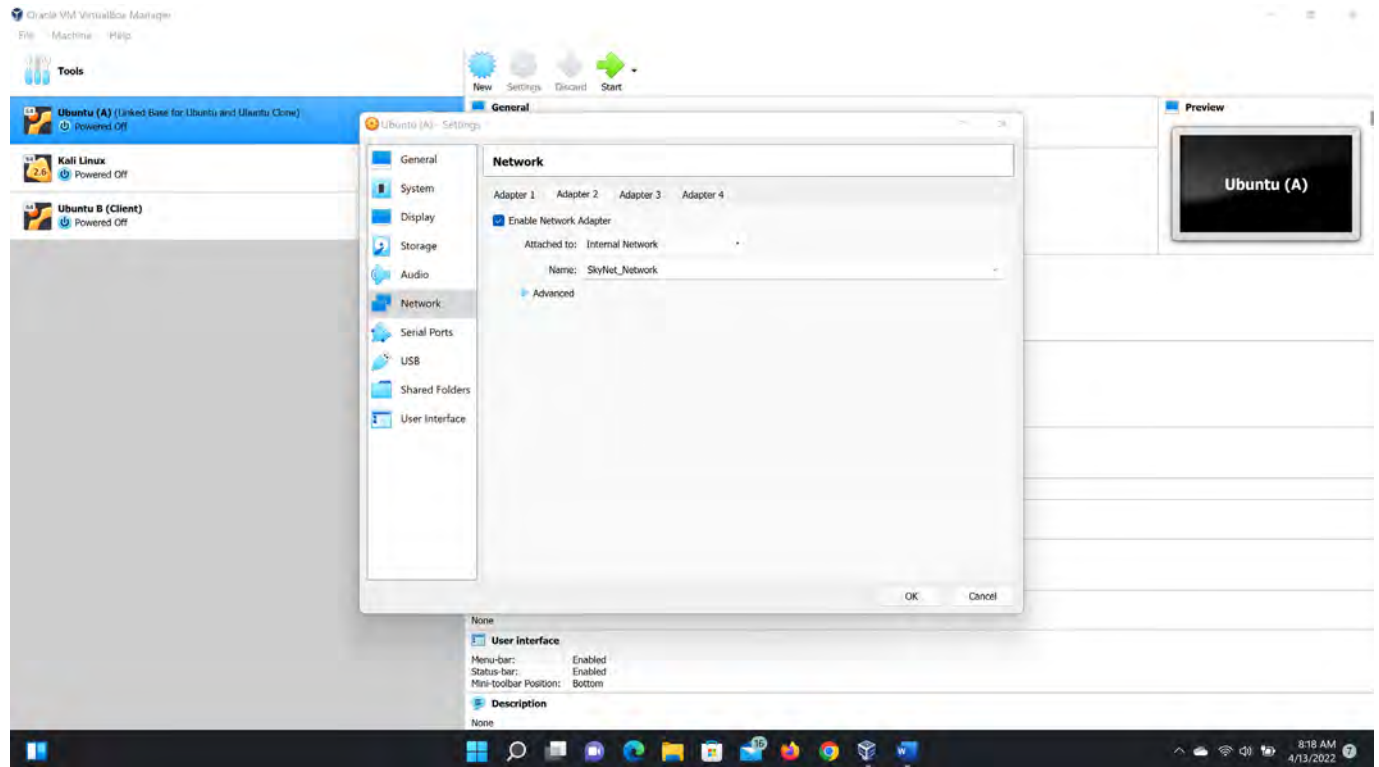
Please configure the network with the following requirement:

CYSE 270 Assignment #13 Advanced Network Configurations

Task A –Network Configuration (60 points)

1. In the virtual box setting, connect two VMs in the same internal network, “internal_{UIN}”.

Replace {UIN} with your real UIN.



CYSE 270 Assignment #13 Advanced Network Configurations

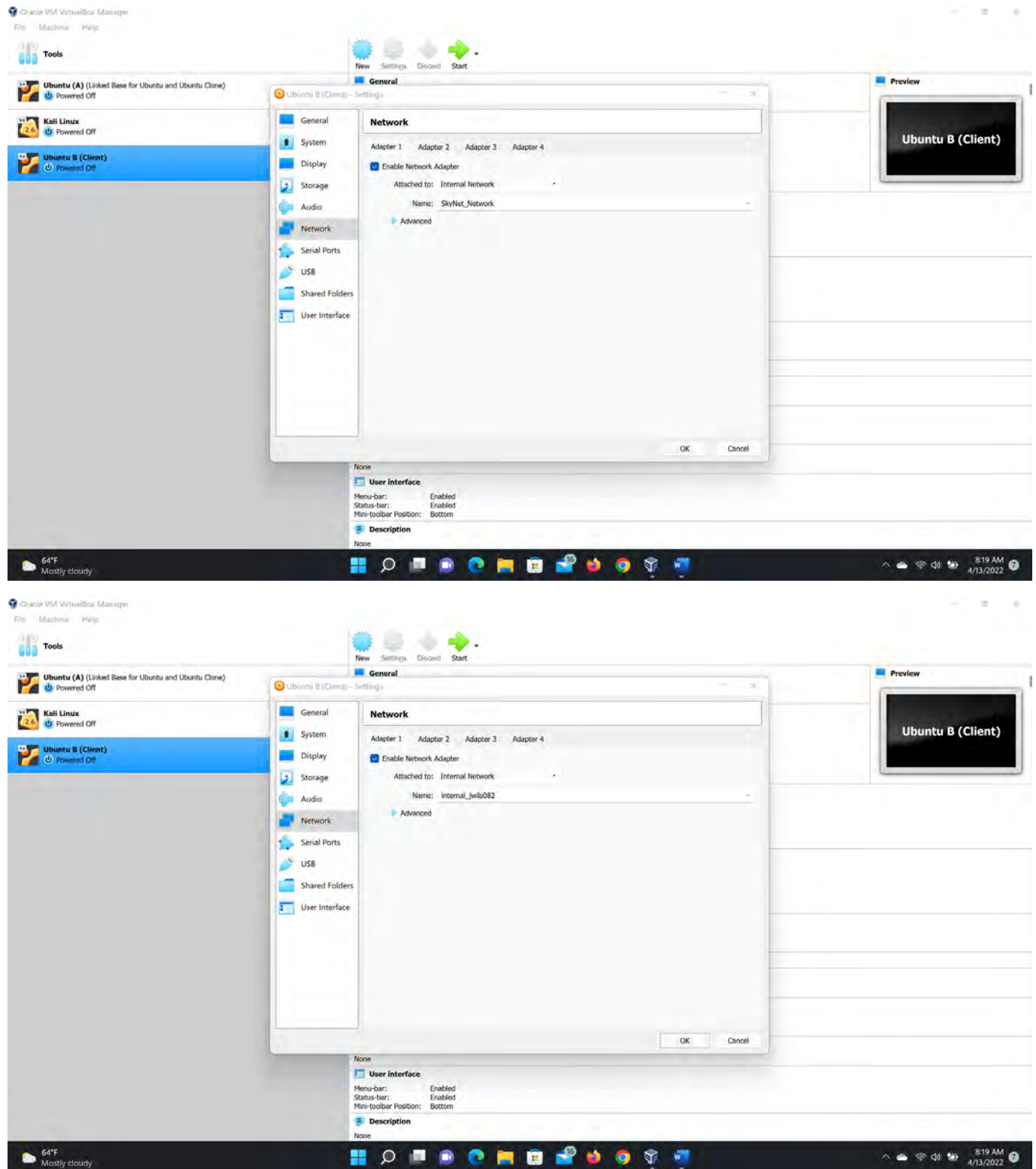
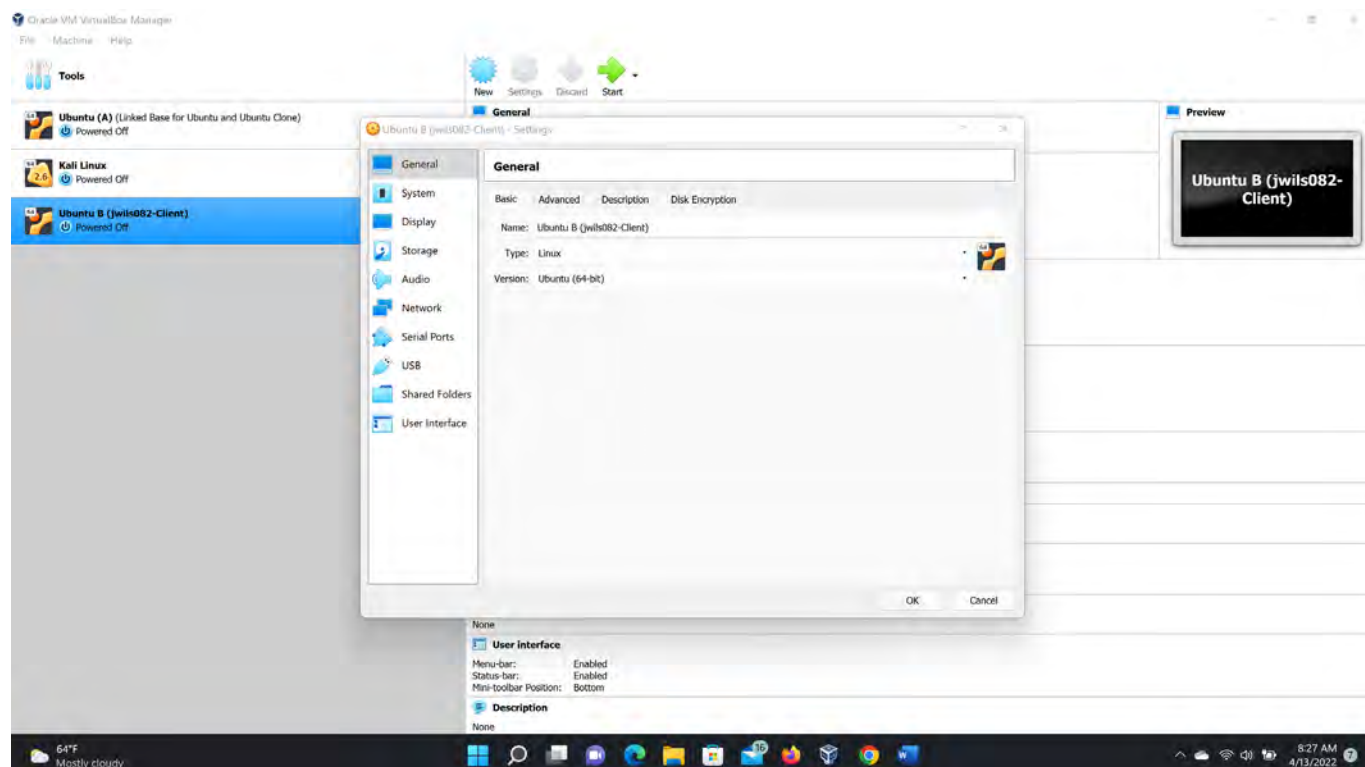
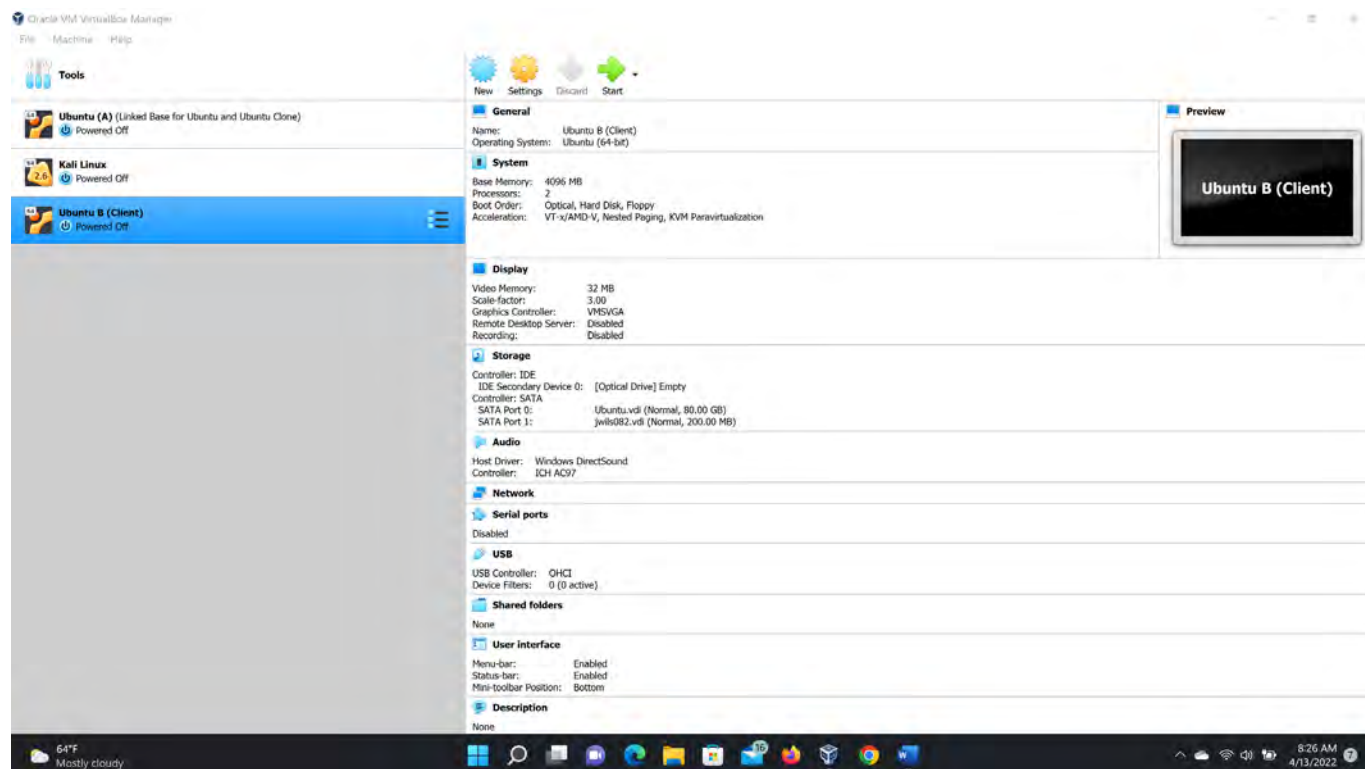


Figure 1 Screenshots of JWILS082 Computer to Step 1.

Above are the screen shots changing both Ubuntu systems (A) and (B) internal network name from “SkyNet_Network” to “internal_jwils082”.

CYSE 270 Assignment #13 Advanced Network Configurations

2. Change the hostname of the Client VM to “{MIDAS}-Client.” Replace {MIDAS} with your real MIDAS.

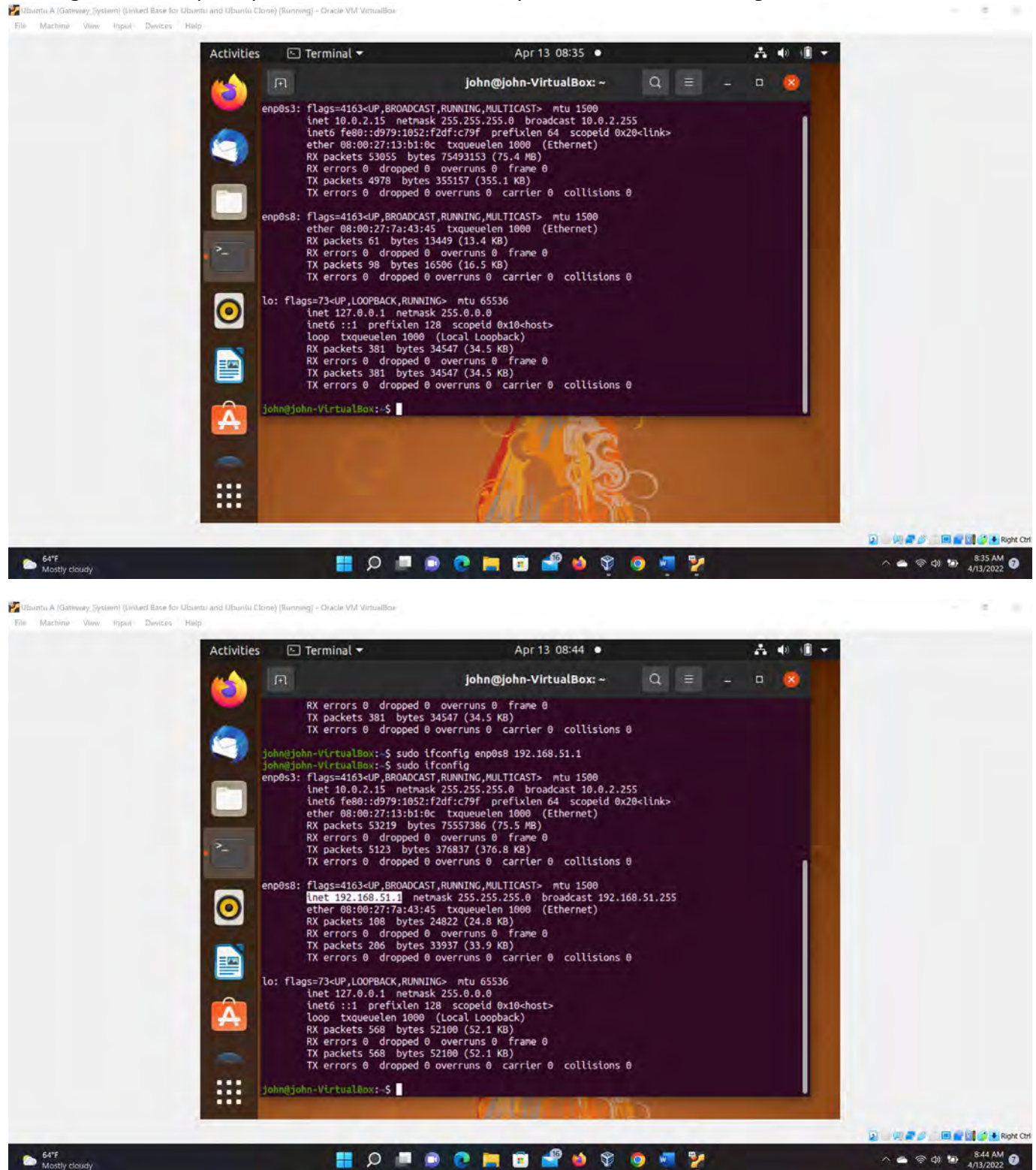


CYSE 270 Assignment #13 Advanced Network Configurations

Figure 2 Screenshots of JWILS082 Computer to Step 2.

Above are the screenshots changing Ubuntu B (Client) machine name to Ubuntu B (jwils082-Client) through the settings menu.

3. Configure the temporary IP address on the Gateway Ubuntu, as shown in Figure 1.



CYSE 270 Assignment #13 Advanced Network Configurations

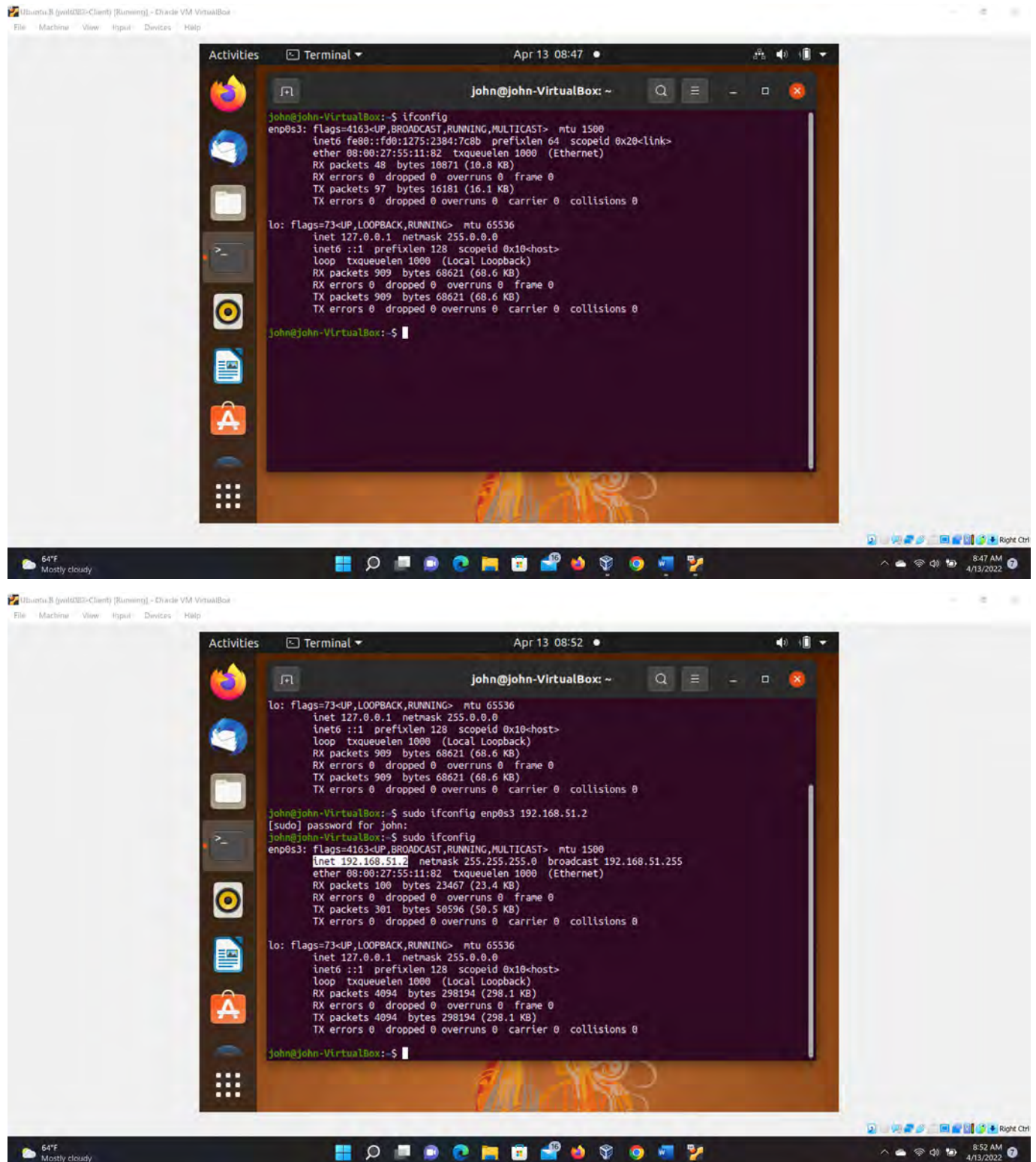
Figure 3 Screenshots of JWILS082 Computer to Step 3.

Above is the screen shot of using the command “ifconfig” that displays the information on current network configuration information, ip address, netmask, or broadcast address to a network interface, creating an alias for the network interface, setting up hardware address, and enable or disable network interfaces. This is to make sure I have an internet connection and that I have the two networks running enp0s3 (is my gateway to the web) and enp0s8 (which is the internal network). You can see that en0s8 has no IP address listed.

In the second screen shot is using the command “sudo ifconfig enp0s8 192.168.51.1” which provides an IP address (highlighted) for the internal network to talk. Additionally, I used the “sudo ifconfig” to affirm the change to network enp0s8 had an IP address added.

CYSE 270 Assignment #13 Advanced Network Configurations

4. Configure the temporary IP address, routing table, and DNS server on Client VM as shown in Figure 1.



CYSE 270 Assignment #13 Advanced Network Configurations

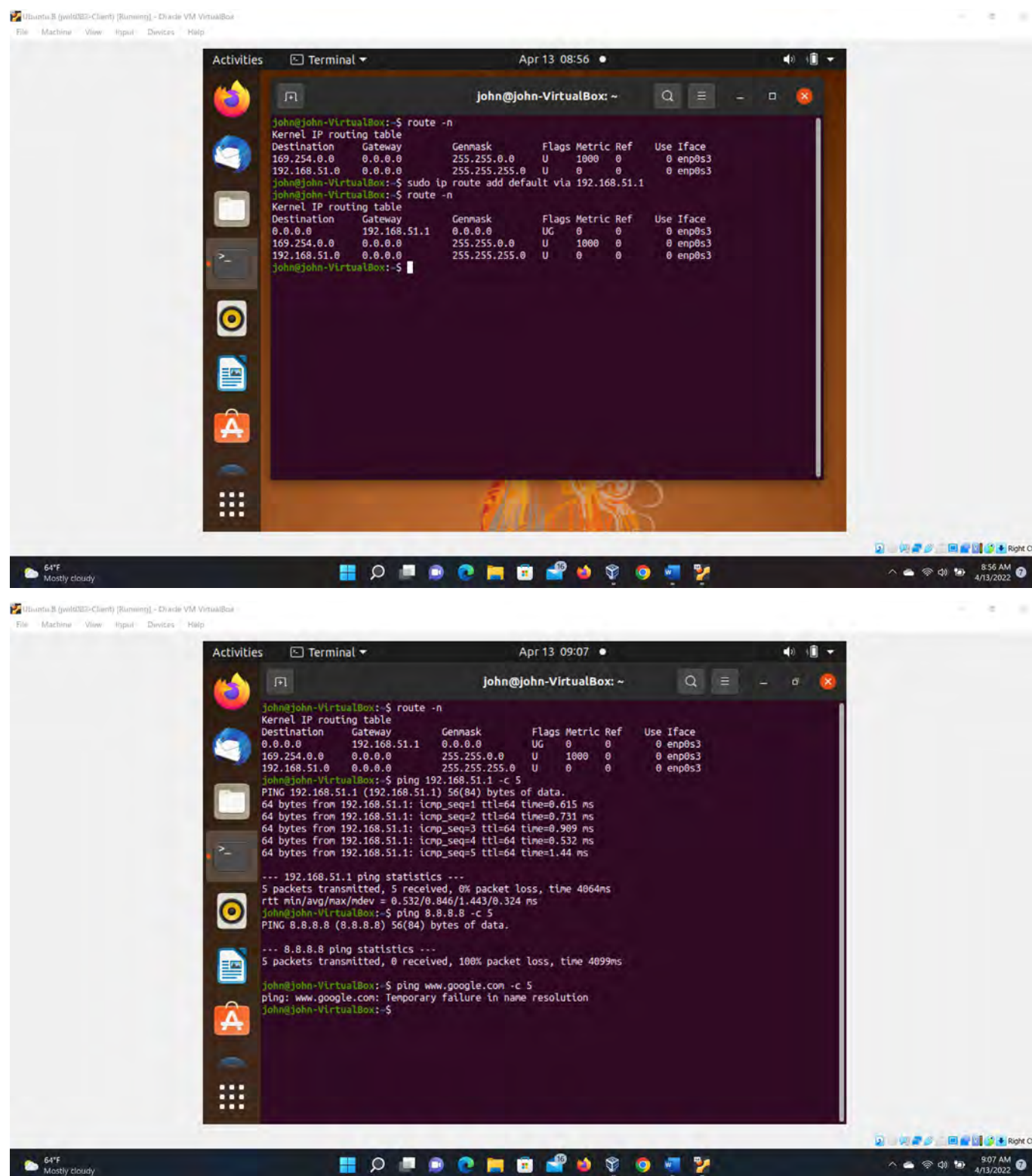


Figure 4 Screenshots of JWILS082 Computer to Step 4.

Above is the screen shot of using the command “ifconfig” that displays the information on current network configuration information, ip address, netmask, or broadcast address to a network interface, creating an alias for the network interface, setting up hardware address, and enable or disable network interfaces. This is to illustrate that I

CYSE 270 Assignment #13 Advanced Network Configurations

have only one network on the client system and that it has no IP address assigned. As you can see the enp0s3 is the only one and it has no network IP address.

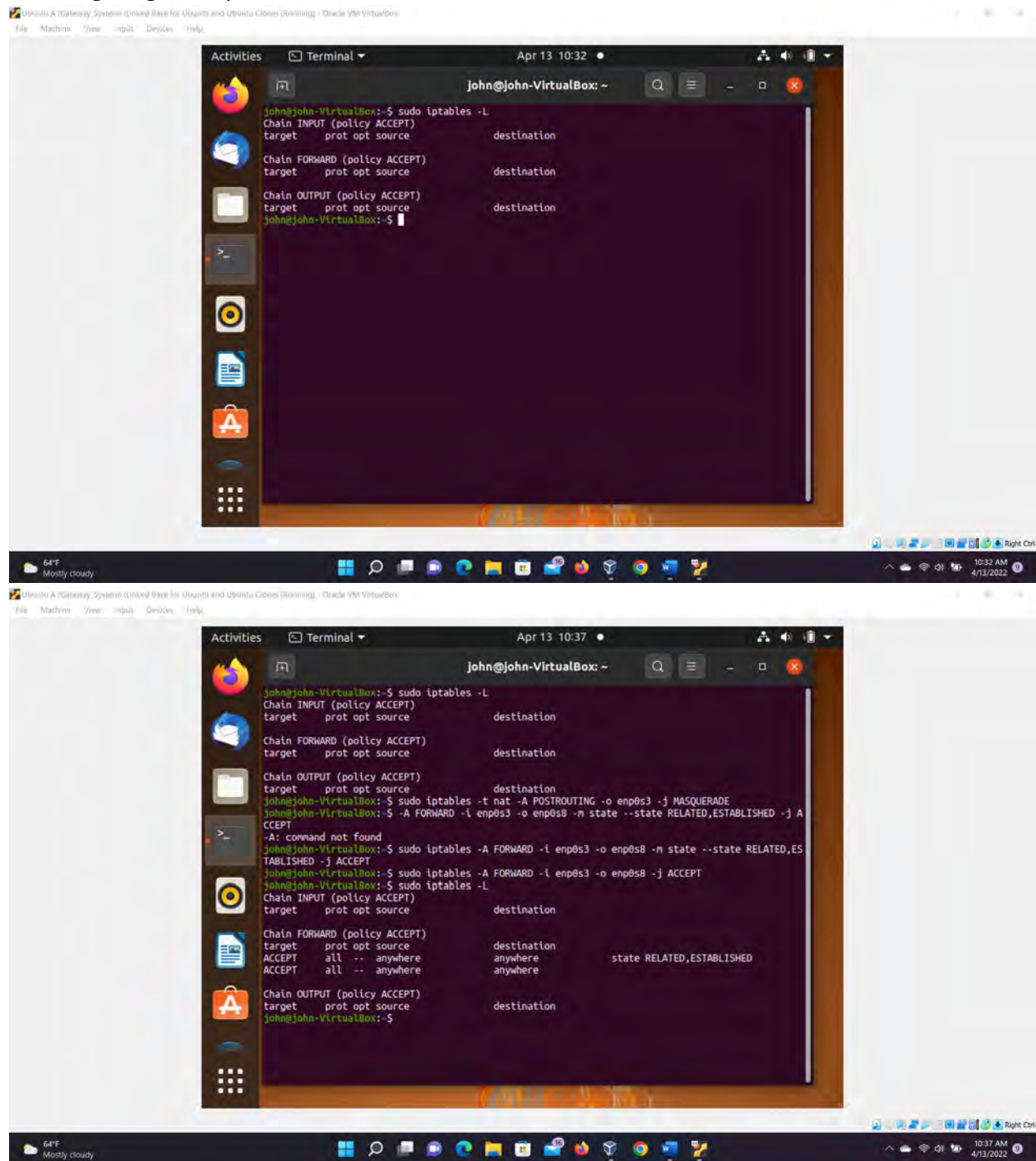
In the second screen shot is using the command “sudo ifconfig enp0s3 192.168.51.2” which provides an IP address (highlighted) for the internal network to talk. Additionally, I used the “sudo ifconfig” to affirm the change to network enp0s8 had an IP address added.

In the third screenshot I used the command “sudo ip route add default via 192.168.51.1” to configure the routing table. In other words I added the default IP address for the client machine to talk to the gateway machine.

On the fourth screen shot I used the command “ping 192.168.51.1 -c 5” to test the connection between the client and the gateway machine. As you can see the client machine can talk to the gateway machine. After this I tried to test the client machine to gain access to the internet and as you can see it failed. At this point I can talk between the client and gateway machines within a local network; however, the client machine still cannot talk outside the local network.

CYSE 270 Assignment #13 Advanced Network Configurations

5. Configure gateway Ubuntu to forward the traffic from the Client to the Internet.



The image shows two screenshots of a terminal window in a VirtualBox environment, demonstrating the configuration of iptables for traffic forwarding on a gateway Ubuntu system.

Top Screenshot: The terminal shows the user running `sudo iptables -L` to list the current iptables rules. The output displays three chains: INPUT, FORWARD, and OUTPUT, all with a policy of ACCEPT. The INPUT and OUTPUT chains have a target of 'destination', while the FORWARD chain has a target of 'destination' and a protocol of 'opt'.

```
john@john-VirtualBox:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source                destination

Chain FORWARD (policy ACCEPT)
target     prot opt source                destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source                destination
john@john-VirtualBox:~$
```

Bottom Screenshot: The terminal shows the user running `sudo iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE` to enable NAT. The output shows the rule being added to the POSTROUTING chain. The user then runs `sudo iptables -A FORWARD -i enp0s3 -o enp0s8 -m state --state RELATED,ESTABLISHED -j ACCEPT` to allow traffic from the client to the internet. The output shows the rule being added to the FORWARD chain.

```
john@john-VirtualBox:~$ sudo iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE
john@john-VirtualBox:~$ sudo iptables -A FORWARD -i enp0s3 -o enp0s8 -m state --state RELATED,ESTABLISHED -j ACCEPT
-A: command not found
john@john-VirtualBox:~$ sudo iptables -A FORWARD -i enp0s3 -o enp0s8 -m state --state RELATED,ESTABLISHED -j ACCEPT
john@john-VirtualBox:~$ sudo iptables -A FORWARD -i enp0s3 -o enp0s8 -j ACCEPT
john@john-VirtualBox:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source                destination

Chain FORWARD (policy ACCEPT)
target     prot opt source                destination
ACCEPT     all  --  anywhere             anywhere             state RELATED,ESTABLISHED
ACCEPT     all  --  anywhere             anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source                destination
john@john-VirtualBox:~$
```

CYSE 270 Assignment #13 Advanced Network Configurations

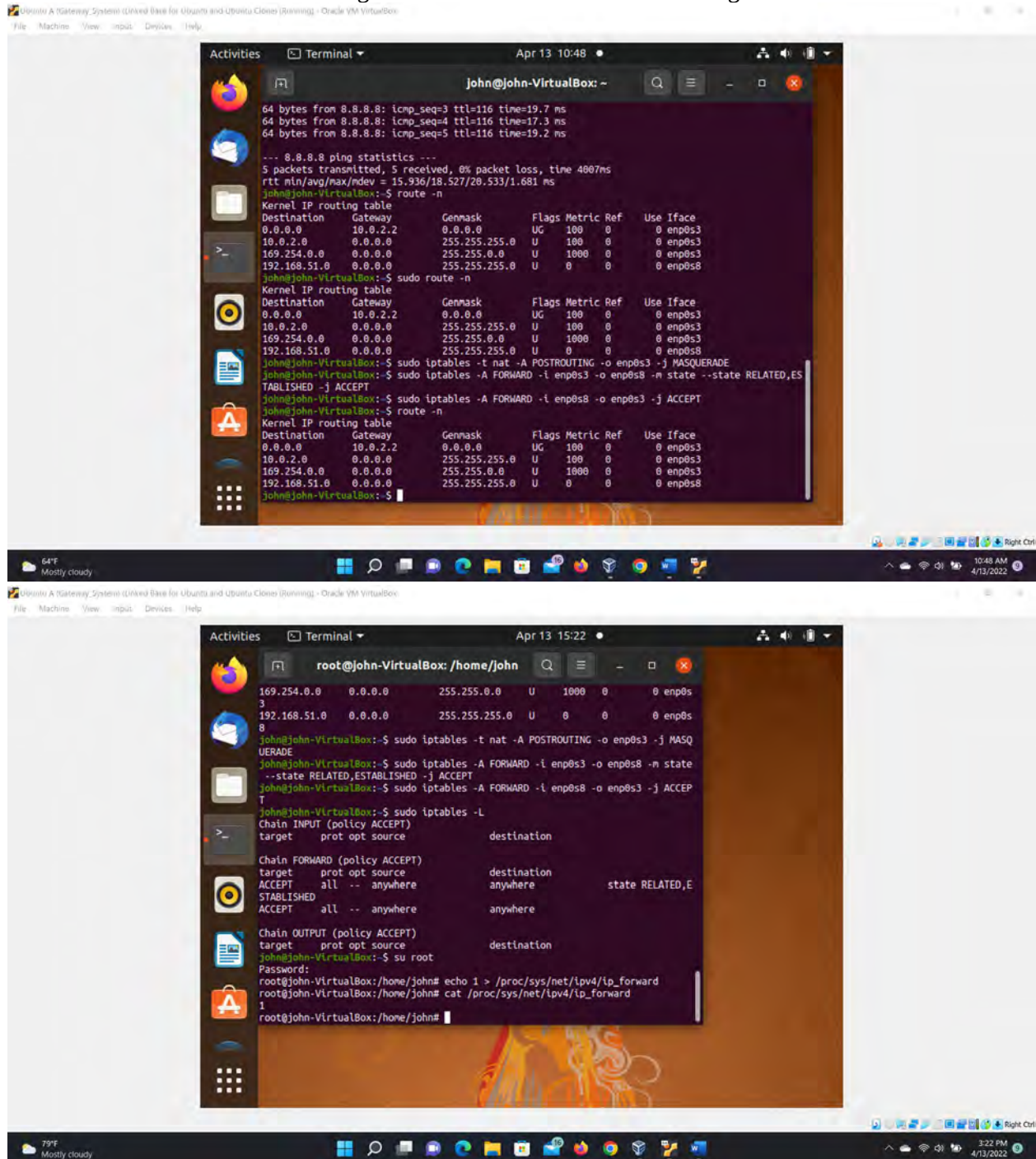


Figure 5 Screenshots of JWILS082 Computer to Step 5.

CYSE 270 Assignment #13 Advanced Network Configurations

Above is the screen shot using the command “sudo iptables -L” to show how the iptables look prior to the forwarding commands.

Additionally, I used the command “sudo iptables -t nat -A POSTROUTING -o enp0s3 -j MASQUERADE” which manipulates the IP table. Basically this sets the rules for the WAN interface. “sudo” is the command that allows you to change things. “iptables -t nat -A” is the command that will alter packets that create a connection used for the Network Address Translation (NAT). “POSTROUTING” is the altering the IP packet after the routing is complete. “-o enp0s3” is the output name and in this case the output name is enp0s3. “-j” is the target. “MASQUERADE” is used when you have an internet network with dynamically assigned private IP addresses; this funnels the traffic access to a live ip address through a single system (i.e. the gateway machine).

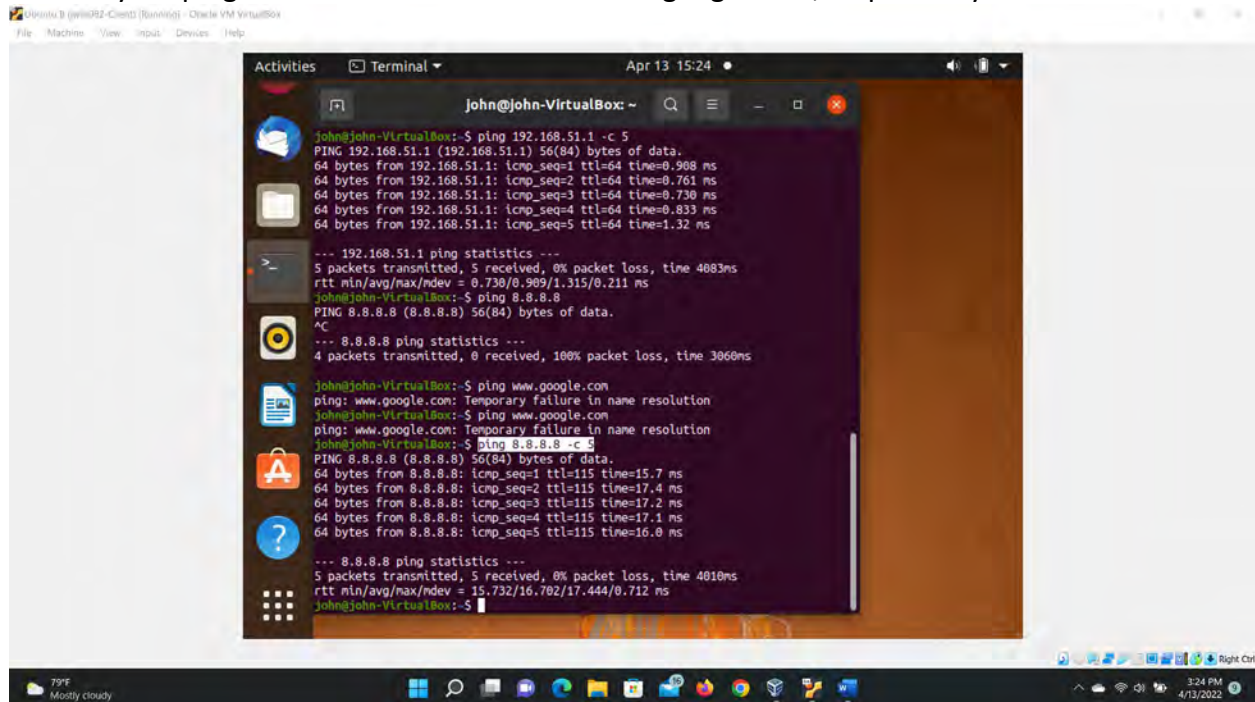
In the second screenshot I used the commands “sudo iptables -A FORWARD -i enp0s3 -o enp0s8 -m state --state RELATED,ESTABLISHED -j ACCEPT”

I also used the command “sudo iptables -A FORWARD -i enp0s8 -o enp0s3 -j ACCEPT”. I made a typing mistake in the second screenshot and corrected this in the third screenshot. I had incorrectly input the network names (enp0s3 and enp0s8). This was fixed.

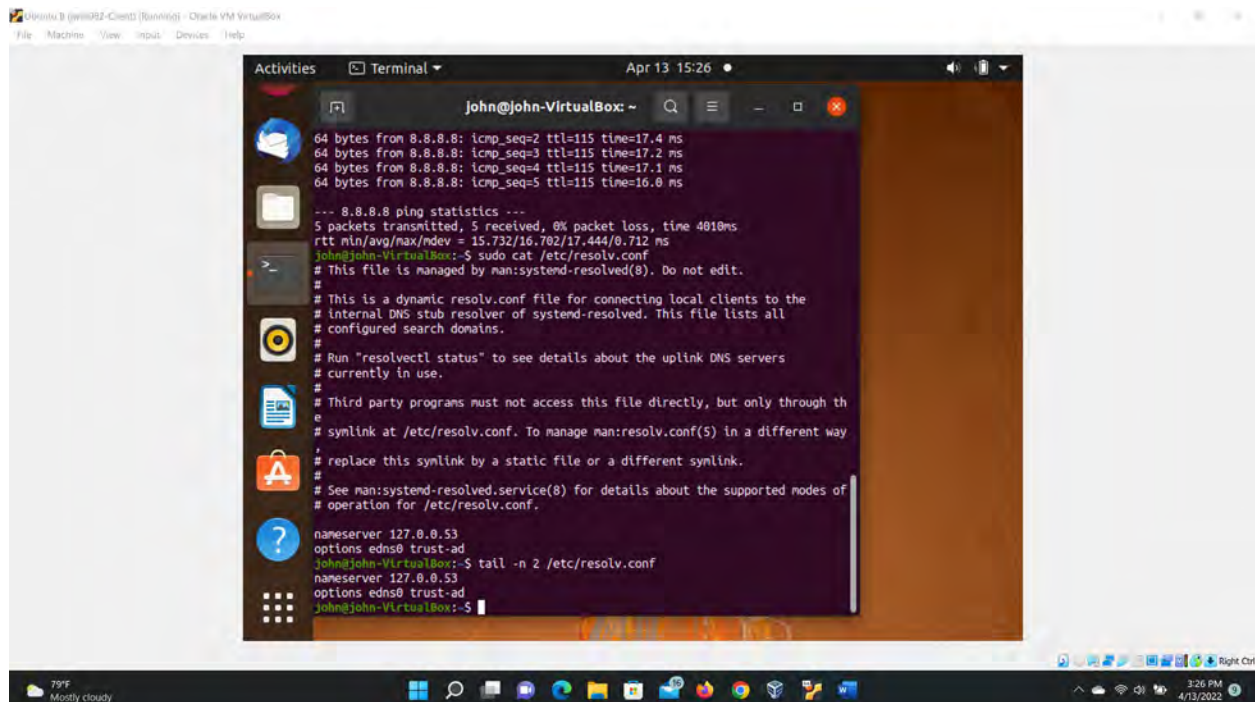
I then used the command “sudo iptables -L” to affirm the changes that were made. Additionally, I have to forward the traffic through the gateway by chaining to the root account “su root” then using the commands “echo 1 > /proc/sys/net/ipv4/ip_forward” and then pressing enter. I then used the commands “cat /proc/sys/net/ipv4/ip_forward” to show that it opened up. The number 1 means everything is good to go (the ip is forwarding).

CYSE 270 Assignment #13 Advanced Network Configurations

6. Test your ping connection to 8.8.8.8 and www.google.com, respectively.

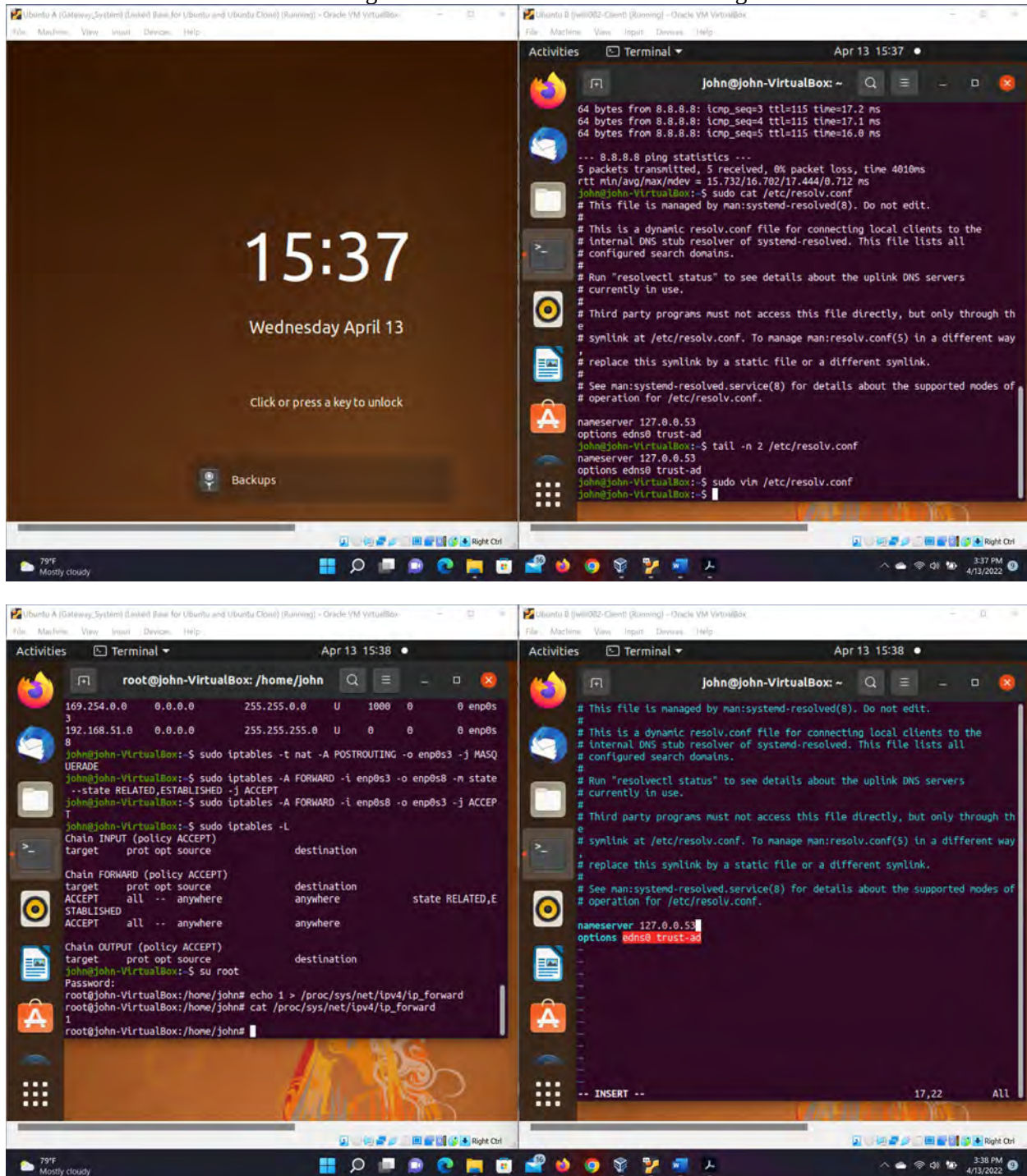


```
John@John-VirtualBox: ~  
John@John-VirtualBox:~$ ping 192.168.51.1 -c 5  
PING 192.168.51.1 (192.168.51.1) 56(84) bytes of data:  
64 bytes from 192.168.51.1: icmp_seq=1 ttl=64 time=0.908 ms  
64 bytes from 192.168.51.1: icmp_seq=2 ttl=64 time=0.761 ms  
64 bytes from 192.168.51.1: icmp_seq=3 ttl=64 time=0.730 ms  
64 bytes from 192.168.51.1: icmp_seq=4 ttl=64 time=0.833 ms  
64 bytes from 192.168.51.1: icmp_seq=5 ttl=64 time=1.32 ms  
  
--- 192.168.51.1 ping statistics ---  
5 packets transmitted, 5 received, 0% packet loss, time 4083ms  
rtt min/avg/max/mdev = 0.730/0.909/1.315/0.211 ms  
John@John-VirtualBox:~$ ping 8.8.8.8  
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data:  
^C  
--- 8.8.8.8 ping statistics ---  
4 packets transmitted, 0 received, 100% packet loss, time 3060ms  
  
John@John-VirtualBox:~$ ping www.google.com  
ping: www.google.com: Temporary failure in name resolution  
John@John-VirtualBox:~$ ping www.google.com  
ping: www.google.com: Temporary failure in name resolution  
John@John-VirtualBox:~$ ping 8.8.8.8 -c 5  
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data:  
64 bytes from 8.8.8.8: icmp_seq=1 ttl=115 time=15.7 ms  
64 bytes from 8.8.8.8: icmp_seq=2 ttl=115 time=17.4 ms  
64 bytes from 8.8.8.8: icmp_seq=3 ttl=115 time=17.2 ms  
64 bytes from 8.8.8.8: icmp_seq=4 ttl=115 time=17.1 ms  
64 bytes from 8.8.8.8: icmp_seq=5 ttl=115 time=16.0 ms  
  
--- 8.8.8.8 ping statistics ---  
5 packets transmitted, 5 received, 0% packet loss, time 4010ms  
rtt min/avg/max/mdev = 15.732/16.702/17.444/0.712 ms  
John@John-VirtualBox:~$
```



```
John@John-VirtualBox: ~  
John@John-VirtualBox:~$ ping 8.8.8.8 -c 5  
64 bytes from 8.8.8.8: icmp_seq=2 ttl=115 time=17.4 ms  
64 bytes from 8.8.8.8: icmp_seq=3 ttl=115 time=17.2 ms  
64 bytes from 8.8.8.8: icmp_seq=4 ttl=115 time=17.1 ms  
64 bytes from 8.8.8.8: icmp_seq=5 ttl=115 time=16.0 ms  
  
--- 8.8.8.8 ping statistics ---  
5 packets transmitted, 5 received, 0% packet loss, time 4010ms  
rtt min/avg/max/mdev = 15.732/16.702/17.444/0.712 ms  
John@John-VirtualBox:~$ sudo cat /etc/resolv.conf  
# This file is managed by man:systemd-resolved(8). Do not edit.  
#  
# This is a dynamic resolv.conf file for connecting local clients to the  
# internal DNS stub resolver of systemd-resolved. This file lists all  
# configured search domains.  
#  
# Run "resolvectl status" to see details about the uplink DNS servers  
# currently in use.  
#  
# Third party programs must not access this file directly, but only through th  
# symlink at /etc/resolv.conf. To manage man:resolv.conf(5) in a different way  
# replace this symlink by a static file or a different symlink.  
#  
# See man:systemd-resolved.service(8) for details about the supported modes of  
# operation for /etc/resolv.conf.  
nameserver 127.0.0.53  
options edns0 trust-ad  
John@John-VirtualBox:~$ tail -n 2 /etc/resolv.conf  
nameserver 127.0.0.53  
options edns0 trust-ad  
John@John-VirtualBox:~$
```

CYSE 270 Assignment #13 Advanced Network Configurations



CYSE 270 Assignment #13 Advanced Network Configurations

The image displays four terminal windows arranged in a 2x2 grid, showing the configuration of network settings in two Ubuntu VMs. The top-left window shows the configuration of iptables rules for NAT and forwarding. The top-right window shows the configuration of the /etc/resolv.conf file. The bottom-left window shows the configuration of the /etc/passwd file. The bottom-right window shows the configuration of the /etc/passwd file.

```
root@john-VirtualBox: /home/john
169.254.0.0 0.0.0.0 255.255.0.0 U 1000 0 0 enp0s3
192.168.51.0 0.0.0.0 255.255.255.0 U 0 0 0 enp0s8
john@john-VirtualBox:~$ sudo iptables -t nat -A POSTROUTING -o enp0s3 -j MASQ
john@john-VirtualBox:~$ sudo iptables -A FORWARD -i enp0s3 -o enp0s8 -n state
--state RELATED,ESTABLISHED -j ACCEPT
john@john-VirtualBox:~$ sudo iptables -A FORWARD -i enp0s8 -o enp0s3 -j ACCEPT
john@john-VirtualBox:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target prot opt source destination
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
john@john-VirtualBox:~$ su root
root@john-VirtualBox: /home/john# echo 1 > /proc/sys/net/ipv4/ip_forward
root@john-VirtualBox: /home/john# cat /proc/sys/net/ipv4/ip_forward
1
root@john-VirtualBox: /home/john#
```

```
John@john-VirtualBox: ~
# This file is managed by man:systemd-resolved(8). Do not edit.
# This is a dynamic resolv.conf file for connecting local clients to the
# internal DNS stub resolver of systemd-resolved. This file lists all
# configured search domains.
# Run "resolvectl status" to see details about the uplink DNS servers
# currently in use.
# Third party programs must not access this file directly, but only through the
# symlink at /etc/resolv.conf. To manage man:resolv.conf(5) in a different way
# replace this symlink by a static file or a different symlink.
# See man:systemd-resolved.service(8) for details about the supported modes of
# operation for /etc/resolv.conf.
nameserver 8.8.8.8
options edns0 trust-ad
```

```
root@john-VirtualBox: /home/john
169.254.0.0 0.0.0.0 255.255.0.0 U 1000 0 0 enp0s3
192.168.51.0 0.0.0.0 255.255.255.0 U 0 0 0 enp0s8
john@john-VirtualBox:~$ sudo iptables -t nat -A POSTROUTING -o enp0s3 -j MASQ
john@john-VirtualBox:~$ sudo iptables -A FORWARD -i enp0s3 -o enp0s8 -n state
--state RELATED,ESTABLISHED -j ACCEPT
john@john-VirtualBox:~$ sudo iptables -A FORWARD -i enp0s8 -o enp0s3 -j ACCEPT
john@john-VirtualBox:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target prot opt source destination
Chain FORWARD (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
john@john-VirtualBox:~$ su root
root@john-VirtualBox: /home/john# echo 1 > /proc/sys/net/ipv4/ip_forward
root@john-VirtualBox: /home/john# cat /proc/sys/net/ipv4/ip_forward
1
root@john-VirtualBox: /home/john#
```

```
John@john-VirtualBox: ~
--- 8.8.8.8 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4010ms
rtt min/avg/max/mdev = 15.732/16.702/17.444/0.712 ms
john@john-VirtualBox:~$ sudo cat /etc/resolv.conf
# This file is managed by man:systemd-resolved(8). Do not edit.
# This is a dynamic resolv.conf file for connecting local clients to the
# internal DNS stub resolver of systemd-resolved. This file lists all
# configured search domains.
# Run "resolvectl status" to see details about the uplink DNS servers
# currently in use.
# Third party programs must not access this file directly, but only through the
# symlink at /etc/resolv.conf. To manage man:resolv.conf(5) in a different way
# replace this symlink by a static file or a different symlink.
# See man:systemd-resolved.service(8) for details about the supported modes of
# operation for /etc/resolv.conf.
nameserver 127.0.0.53
options edns0 trust-ad
john@john-VirtualBox:~$ tail -n 2 /etc/resolv.conf
nameserver 127.0.0.53
options edns0 trust-ad
john@john-VirtualBox:~$ sudo vi /etc/resolv.conf
john@john-VirtualBox:~$ sudo vi /etc/resolv.conf
john@john-VirtualBox:~$ tail -n 2 /etc/resolv.conf
nameserver 8.8.8.8
options edns0 trust-ad
john@john-VirtualBox:~$
```

CYSE 270 Assignment #13 Advanced Network Configurations

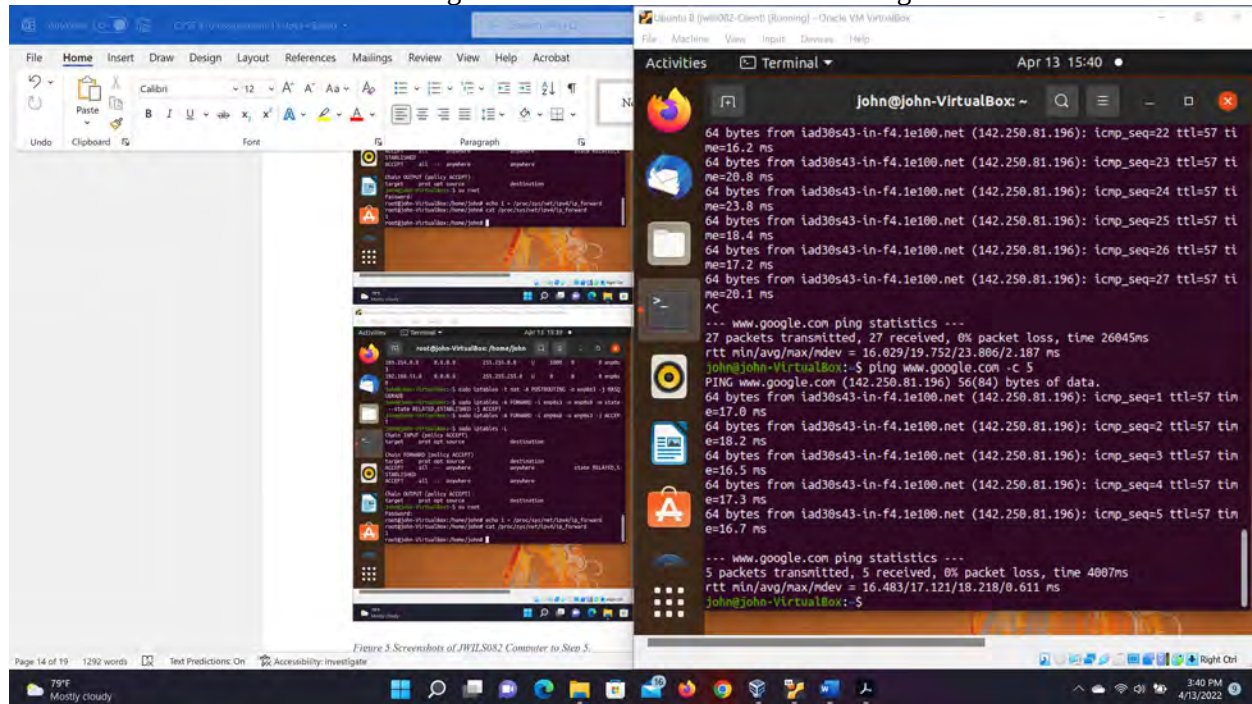


Figure 6 Screenshots of JWILS082 Computer to Step 6.

Above are the screen shots pinging a ip address and a URL. As you can see the Ip address worked fine; however the URL did not so I am going to have to configure the /etc/resolv.conf file through VIM to make this work.

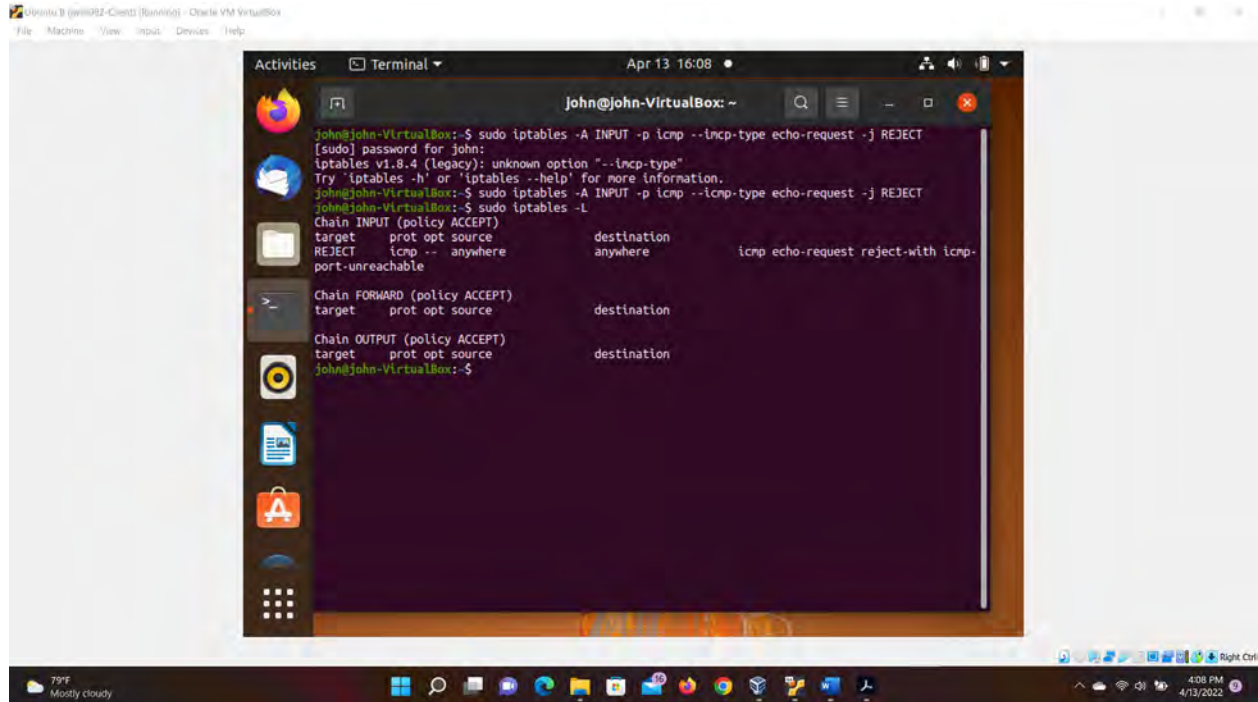
And as you can see the url now works with no problem after editing the /etc/resolv.conf file through VIM.

I also wanted to apologise for the last couple of screenshots as I was double screening the VM machines so that I could quickly observe things to trouble shoot. It looks messy but the work is there.

CYSE 270 Assignment #13 Advanced Network Configurations

Task B –Firewall Configuration (40 points)

1. Configure the iptables on the gateway Ubuntu to block all the inbound ICMP packets from the Client VM.



```
John@John-VirtualBox: ~  
John@John-VirtualBox:~$ sudo iptables -A INPUT -p icmp --icmp-type echo-request -j REJECT  
[sudo] password for john:  
iptables v1.8.4 (legacy): unknown option "--icmp-type"  
Try 'iptables -h' or 'iptables --help' for more information.  
John@John-VirtualBox:~$ sudo iptables -A INPUT -p icmp --icmp-type echo-request -j REJECT  
John@John-VirtualBox:~$ sudo iptables -L  
Chain INPUT (policy ACCEPT)  
target prot opt source destination  
REJECT icmp -- anywhere anywhere icmp echo-request reject-with icmp-  
port-unreachable  
  
Chain FORWARD (policy ACCEPT)  
target prot opt source destination  
  
Chain OUTPUT (policy ACCEPT)  
target prot opt source destination  
John@John-VirtualBox:~$
```

Figure 1 Screenshots of JWILS082 Computer to Step 1.

Above is the screen shot using the command “sudo iptables -A INPUT -p icmp --icmp-type echo-request -j REJECT” that will block all the inbound ICMP packets from the client machine.

CYSE 270 Assignment #13 Advanced Network Configurations

2. Configure the iptables on the gateway Ubuntu to block all the outbound ICMP packets that originated from the gateway Ubuntu itself.

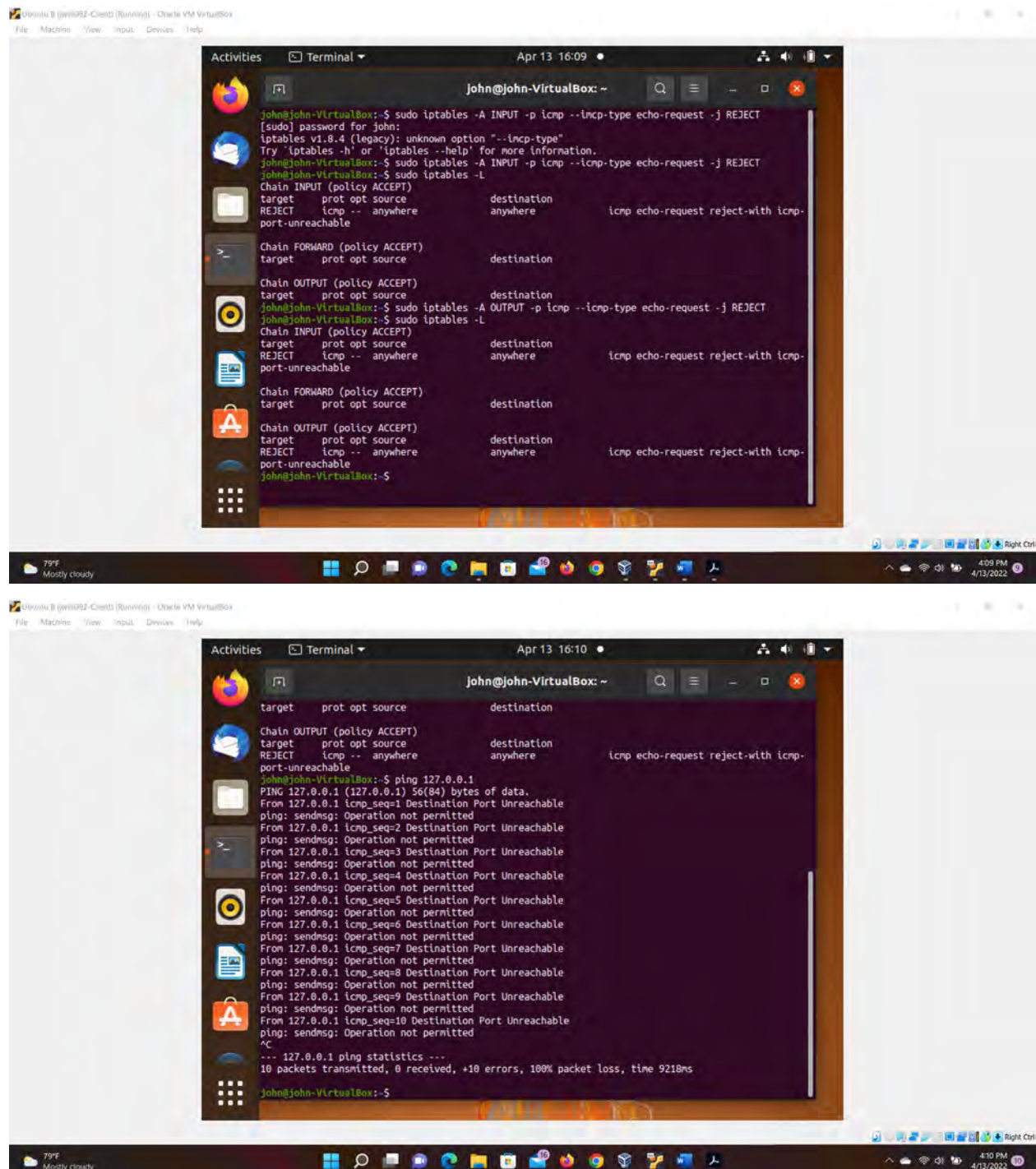


Figure 2 Screenshots of JWILS082 Computer to Step 2.

Above is the screen shot using the command “sudo iptables -A OUTPUT -p icmp --icmp-type echo-request -j REJECT” that will block all the outbound ICMP packets from the client machine.

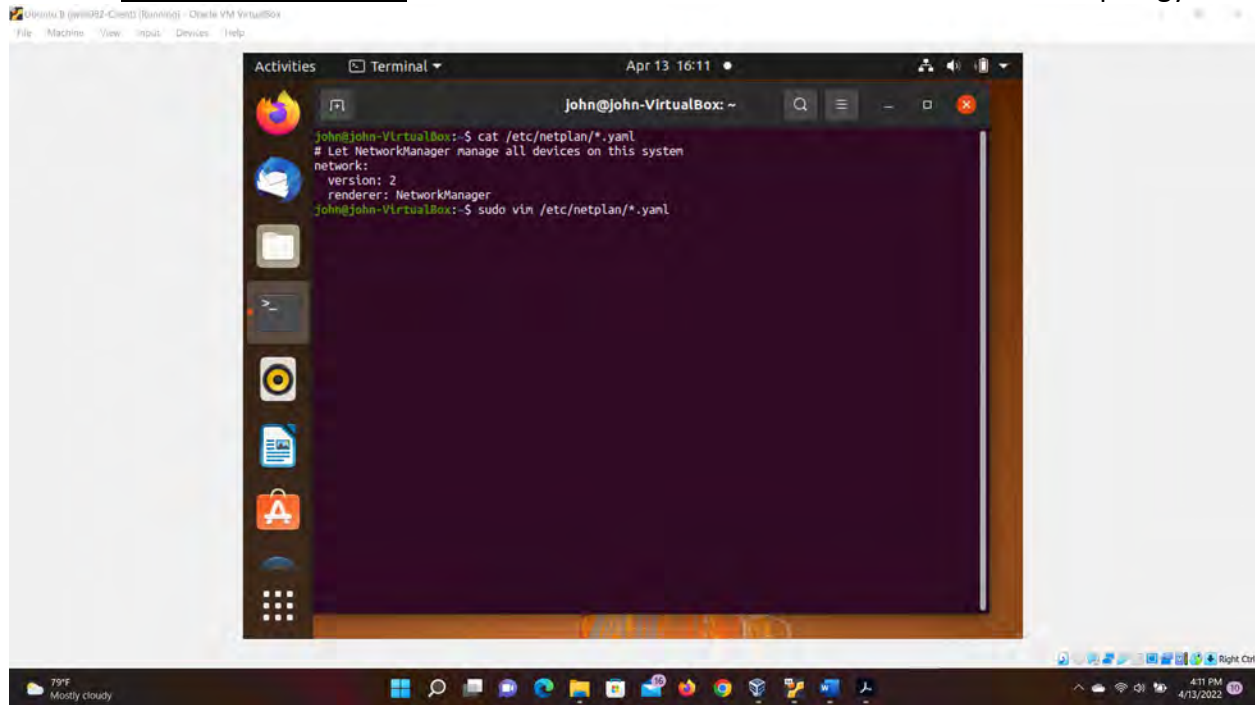
CYSE 270 Assignment #13 Advanced Network Configurations

I also wanted to make sure that you could not ping a icmp address so checked it by sending a ping to address 127.0.0.1 and it was denied. So it seems the changes to the iptables worked.

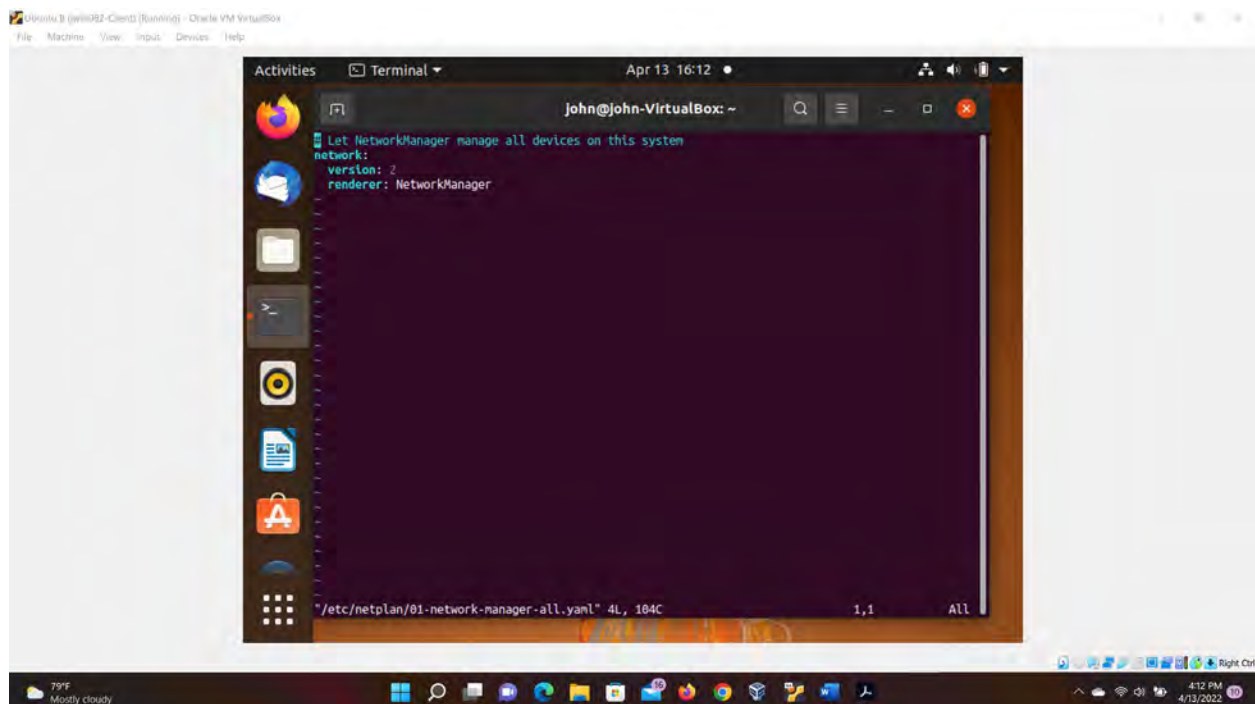
CYSE 270 Assignment #13 Advanced Network Configurations

Extra credit:

Set the permanent IP address on the Client Ubuntu based on the above network topology.

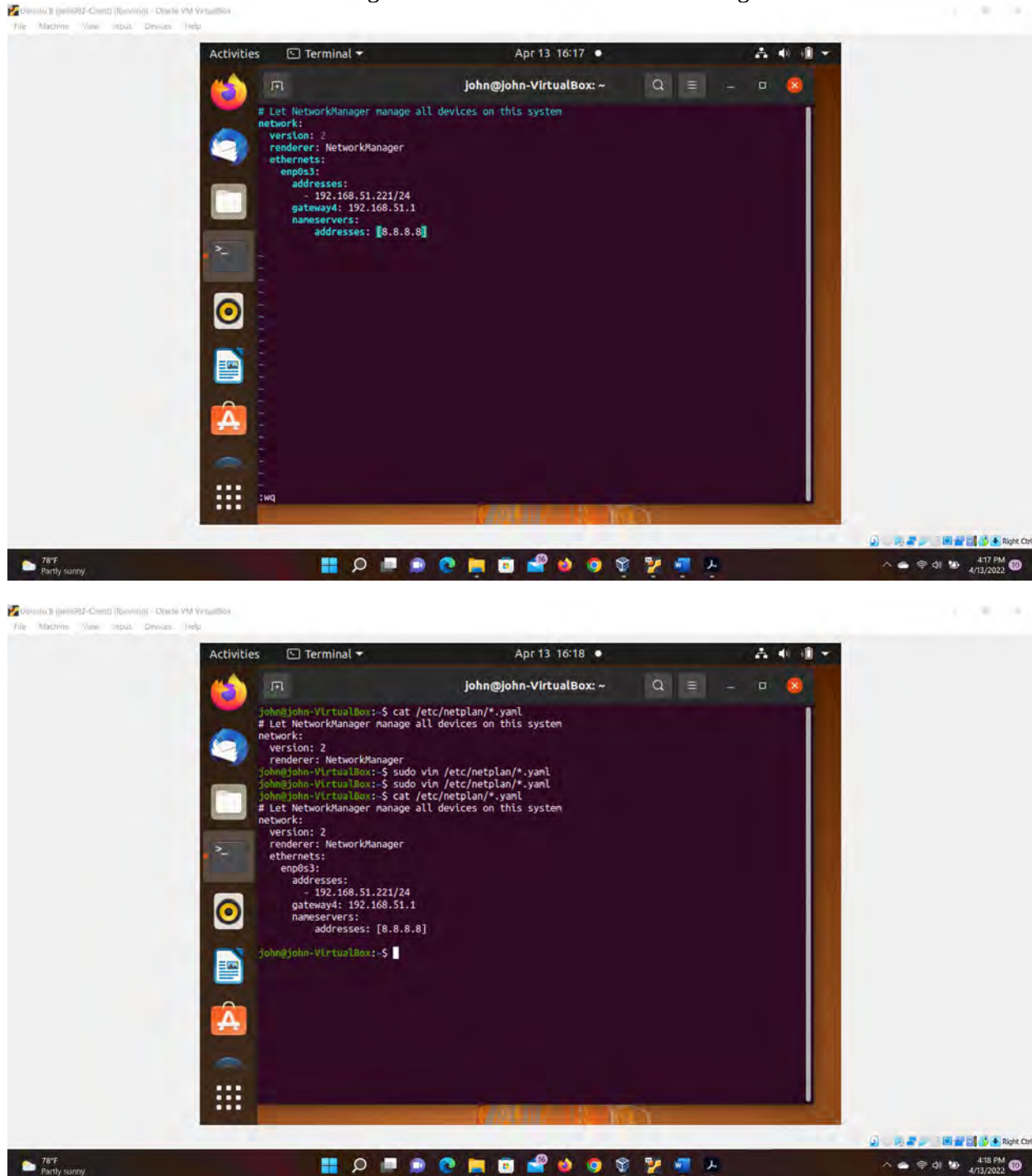


A screenshot of a terminal window within a Virtual Machine (Ubuntu 20.04 LTS). The terminal shows the user 'john' at the prompt 'john@john-VirtualBox: ~'. The user has executed the command 'cat /etc/netplan/*.yaml', which displays the following content: '# Let NetworkManager manage all devices on this system', 'network:', 'version: 2', and 'renderer: NetworkManager'. The user then enters the command 'sudo vim /etc/netplan/*.yaml'. The terminal window is titled 'john@john-VirtualBox: ~' and has a search bar and window controls. The background shows the Ubuntu desktop environment with a dock on the left and a system tray on the bottom right.



A screenshot of a terminal window within a Virtual Machine (Ubuntu 20.04 LTS). The terminal shows the user 'john' at the prompt 'john@john-VirtualBox: ~'. The user has executed the command 'cat /etc/netplan/*.yaml', which displays the following content: '# Let NetworkManager manage all devices on this system', 'network:', 'version: 2', and 'renderer: NetworkManager'. The user then enters the command 'sudo vim /etc/netplan/*.yaml'. The terminal window is titled 'john@john-VirtualBox: ~' and has a search bar and window controls. The background shows the Ubuntu desktop environment with a dock on the left and a system tray on the bottom right.

CYSE 270 Assignment #13 Advanced Network Configurations



The image consists of two screenshots of a terminal window within a virtual machine environment. The terminal is titled 'john@john-VirtualBox: ~' and shows the following commands and output:

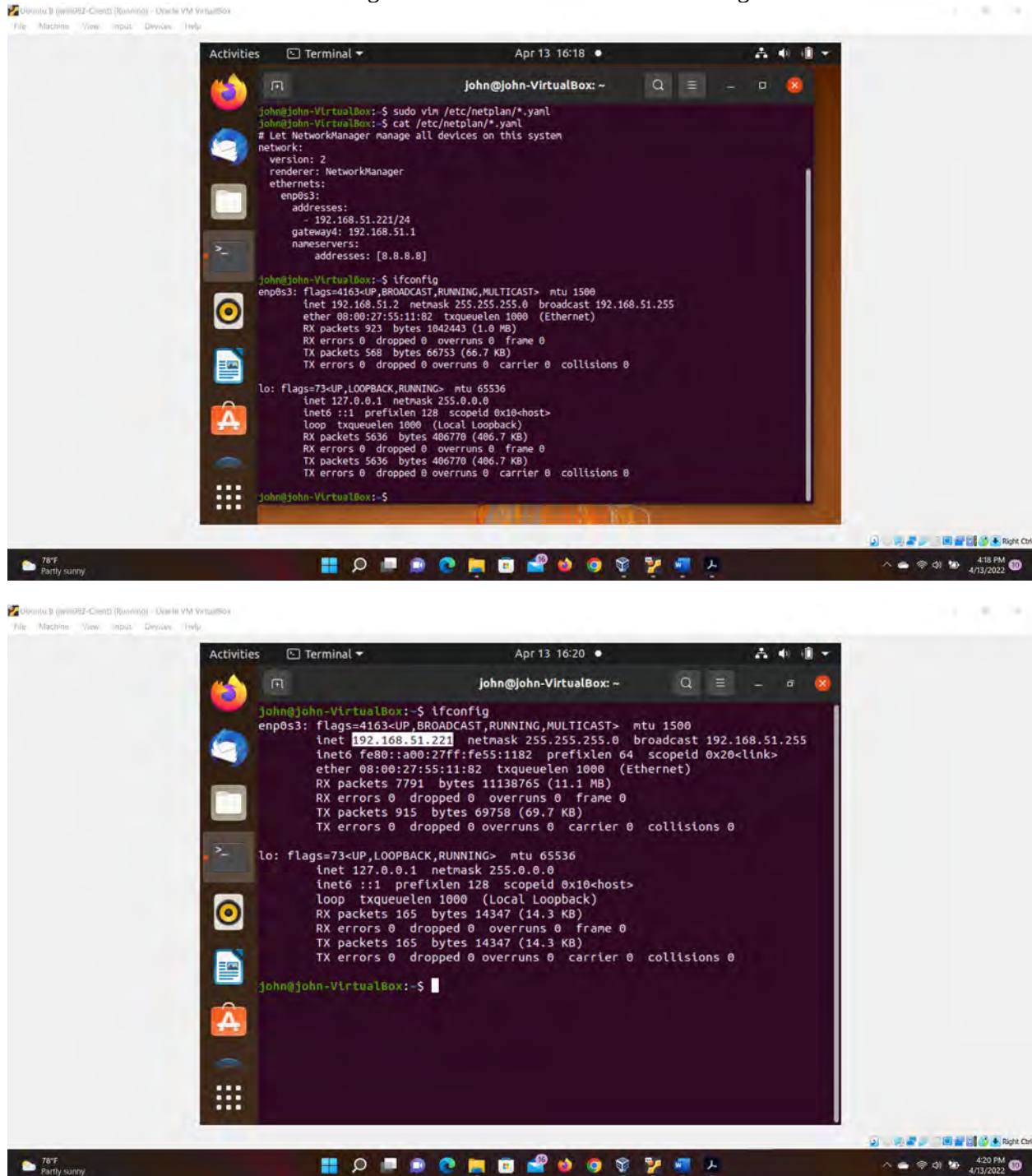
```
# Let NetworkManager manage all devices on this system
network:
  version: 2
  renderer: NetworkManager
  ethernet:
    addresses:
      - 192.168.51.221/24
    gateway4: 192.168.51.1
    nameservers:
      addresses: [8.8.8.8]
```

The first screenshot shows the initial configuration. The second screenshot shows the same configuration after several commands have been executed to verify and apply the settings:

```
john@john-VirtualBox:~$ cat /etc/netplan/*.yaml
# Let NetworkManager manage all devices on this system
network:
  version: 2
  renderer: NetworkManager
john@john-VirtualBox:~$ sudo vim /etc/netplan/*.yaml
john@john-VirtualBox:~$ sudo vim /etc/netplan/*.yaml
john@john-VirtualBox:~$ cat /etc/netplan/*.yaml
# Let NetworkManager manage all devices on this system
network:
  version: 2
  renderer: NetworkManager
  ethernet:
    addresses:
      - 192.168.51.221/24
    gateway4: 192.168.51.1
    nameservers:
      addresses: [8.8.8.8]
```

The terminal output in the second screenshot shows the configuration file content after the commands. The virtual machine's desktop environment is visible in the background, showing a weather widget (78°F, Partly sunny) and system clock (4:17 PM, 4/13/2022).

CYSE 270 Assignment #13 Advanced Network Configurations



The image displays two screenshots of a terminal window within a virtual machine environment, showing network configuration steps.

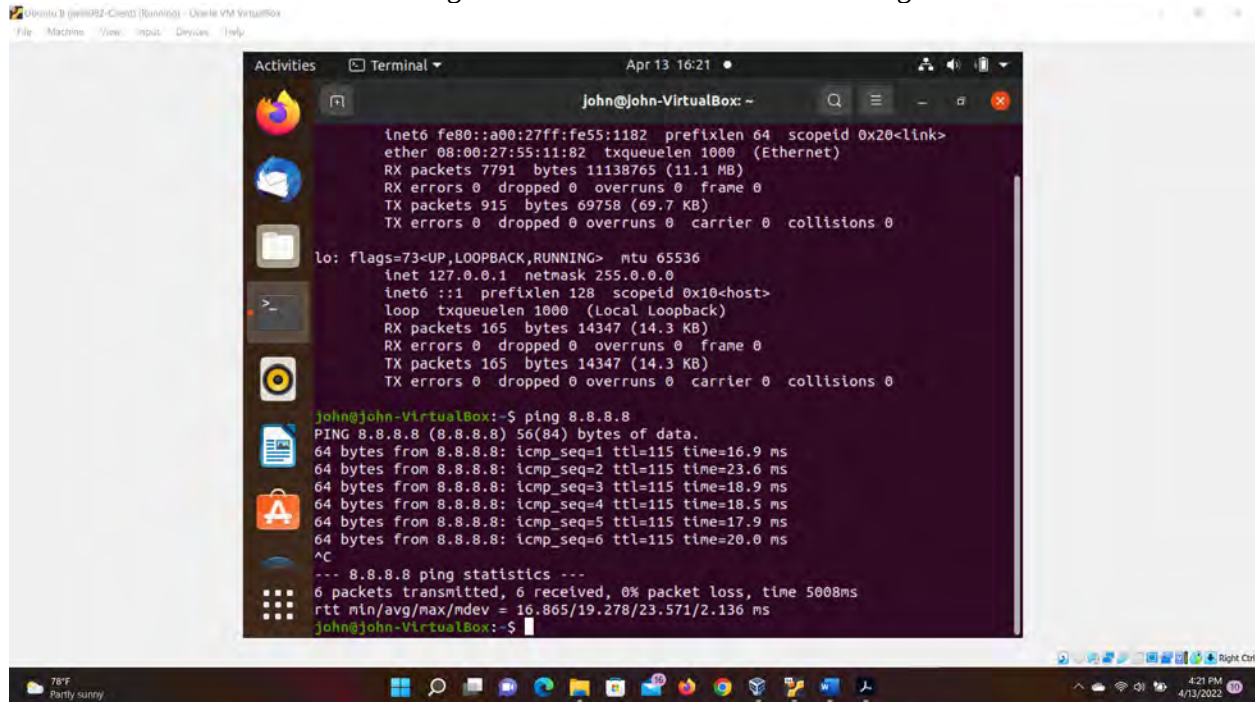
Top Screenshot (Apr 13 16:18):

```
John@John-VirtualBox: ~  
John@John-VirtualBox:~$ sudo vim /etc/netplan/*.yaml  
John@John-VirtualBox:~$ cat /etc/netplan/*.yaml  
# Let NetworkManager manage all devices on this system  
network:  
  version: 2  
  renderer: NetworkManager  
  ethernet:  
    enp0s3:  
      addresses:  
        - 192.168.51.221/24  
      gateway4: 192.168.51.1  
      nameservers:  
        addresses: [8.8.8.8]  
John@John-VirtualBox:~$ ifconfig  
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
        inet 192.168.51.2 netmask 255.255.255.0 broadcast 192.168.51.255  
        ether 08:00:27:55:11:82 txqueuelen 1000 (Ethernet)  
        RX packets 923 bytes 1042443 (1.0 MB)  
        RX errors 0 dropped 0 overruns 0 frame 0  
        TX packets 568 bytes 66753 (66.7 KB)  
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
    inet 127.0.0.1 netmask 255.0.0.0  
    inet6 ::1 prefixlen 128 scopeid 0x10<host>  
    loop txqueuelen 1000 (Local Loopback)  
    RX packets 5636 bytes 486770 (486.7 KB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 5636 bytes 486770 (486.7 KB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
John@John-VirtualBox:~$
```

Bottom Screenshot (Apr 13 16:20):

```
John@John-VirtualBox: ~  
John@John-VirtualBox:~$ ifconfig  
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
        inet 192.168.51.221 netmask 255.255.255.0 broadcast 192.168.51.255  
        inet6 fe80::a00:27ff:fe55:1182 prefixlen 64 scopeid 0x20<link>  
        ether 08:00:27:55:11:82 txqueuelen 1000 (Ethernet)  
        RX packets 7791 bytes 11138765 (11.1 MB)  
        RX errors 0 dropped 0 overruns 0 frame 0  
        TX packets 915 bytes 69758 (69.7 KB)  
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
    inet 127.0.0.1 netmask 255.0.0.0  
    inet6 ::1 prefixlen 128 scopeid 0x10<host>  
    loop txqueuelen 1000 (Local Loopback)  
    RX packets 165 bytes 14347 (14.3 KB)  
    RX errors 0 dropped 0 overruns 0 frame 0  
    TX packets 165 bytes 14347 (14.3 KB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
John@John-VirtualBox:~$
```

CYSE 270 Assignment #13 Advanced Network Configurations



```
john@john-VirtualBox: ~  
inet6 fe80::a00:27ff:fe55:1182 prefixlen 64 scopeid 0x20<link>  
ether 08:00:27:55:11:82 txqueuelen 1000 (Ethernet)  
RX packets 7791 bytes 11138765 (11.1 MB)  
RX errors 0 dropped 0 overruns 0 frame 0  
TX packets 915 bytes 69758 (69.7 KB)  
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
inet 127.0.0.1 netmask 255.0.0.0  
inet6 ::1 prefixlen 128 scopeid 0x10<host>  
loop txqueuelen 1000 (Local Loopback)  
RX packets 165 bytes 14347 (14.3 KB)  
RX errors 0 dropped 0 overruns 0 frame 0  
TX packets 165 bytes 14347 (14.3 KB)  
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
john@john-VirtualBox:~$ ping 8.8.8.8  
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.  
64 bytes from 8.8.8.8: icmp_seq=1 ttl=115 time=16.9 ms  
64 bytes from 8.8.8.8: icmp_seq=2 ttl=115 time=23.6 ms  
64 bytes from 8.8.8.8: icmp_seq=3 ttl=115 time=18.9 ms  
64 bytes from 8.8.8.8: icmp_seq=4 ttl=115 time=18.5 ms  
64 bytes from 8.8.8.8: icmp_seq=5 ttl=115 time=17.9 ms  
64 bytes from 8.8.8.8: icmp_seq=6 ttl=115 time=20.0 ms  
^C  
--- 8.8.8.8 ping statistics ---  
6 packets transmitted, 6 received, 0% packet loss, time 5008ms  
rtt min/avg/max/mdev = 16.865/19.278/23.571/2.136 ms  
john@john-VirtualBox:~$
```

Figure 1 Screenshots of JWILS082 Computer to Step Extra Credit.

Above are the screenshots of changing the ip address from temporary to permanent. What I did was look at the file using the command “cat /etc/netplan/*.yaml” to see what was listed as commands or directions. There was a general commands listed with no directions to change the ip address permanently.

Next I needed to edit the file in VIM by using the commands “sudo vim /etc/netplan/*.yaml”. I added the necessary information saved and exited VIM. The information I entered in to the .yaml file is as follows:

```
network:  
  version: 2  
  renderer: NetworkManager  
  ethernets:  
    enp0s3:  
      dhcp4: no  
      addresses:  
        - 192.168.51.221/24  
      gateway4: 192.168.151.1  
      nameservers:  
        addresses: [8.8.8.8]
```

To check that the changes were made I ran ifconfig and found that the former ip address of 192.168.51.2 was still attached to the network. I decided to restart the machine to see if the new ip address would take effect and low-and-behold it did as you can see from the screenshot.

Just to see if I still had connectivity I sent a ping to 8.8.8.8 and it seems to work. (well it will until I shut down the gateway machine because I have not set that ip address permanently).

CYSE 270 Assignment #13 Advanced Network Configurations

References

15 Useful “ifconfig” Commands to Configure Network in Linux. (n.d.). Wwww.tecmint.com.

<https://www.tecmint.com/ifconfig-command-examples/>

How to Configure Static IP Address on Ubuntu 20.04. (2020, September 15). Linuxize.com.

<https://linuxize.com/post/how-to-configure-static-ip-address-on-ubuntu-20-04/>

How to block/allow ping using iptables in Ubuntu. (2019, March 13). VITUX.

<https://vitux.com/how-to-block-allow-ping-using-iptables-in-ubuntu/>