Student Name: Roey Ben Hayun  
Email: [rroeybh@gmail.com](mailto:rroeybh@gmail.com), rbenhayu@asu.edu  
Submission Date: 22.10.2022  
Class Name and Term: CSE548 Summer 2021

Packet Filter Firewall (Iptables)

# Project Overview

The main objective of this project was to learn about iptables and the role they play when setting NAT network. This was

done by understanding routing rules on different network flows by using open-source networking tools.

In this project I set up two NAT networks and built different communication flows between those networks and the open

Public network.

# Network Setup

# Software

For this project I have used the following tools

* ifconfig – to check the network configuration
* ping – check host connectivity
* iptables – set different routing rules on the INPUT, OUTPUT and FORWARD chains
* route -n : to check network routing info
* iptables –L –n –v : get the status and rules of the routing tables (add –t nat to see the same for NAT tables)
* Wireshark – to sniff packet traffic between the client and the server

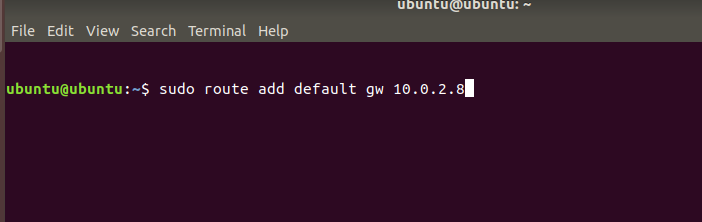
# Project Description

Creating the client and Server/Gateway VM is explain in the supporting docs.

**Configure Routes**

For the Client to access the public network via the Gateway, I had to setup the default gateway on the client side using the following command:

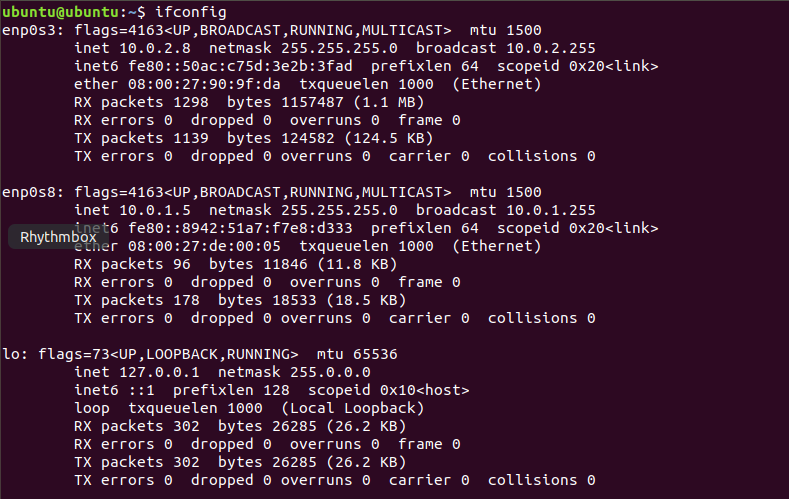
Client:



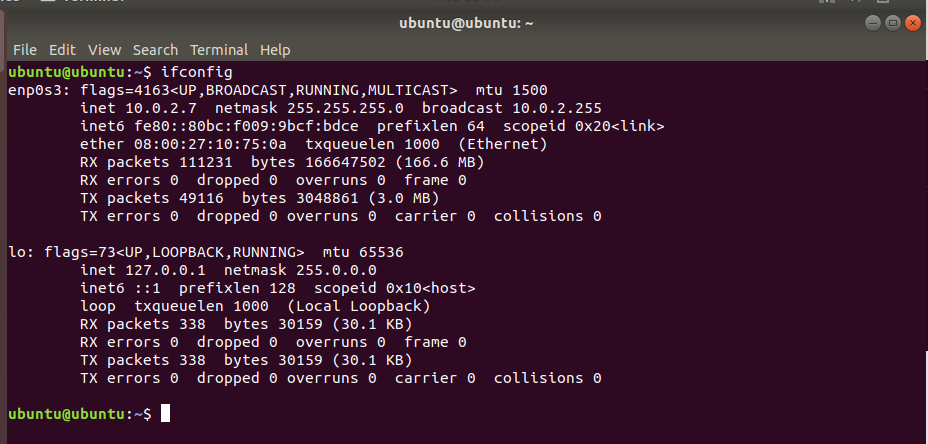
To get the Default Gateway IP address and the client network interface you should run ifconfig on both the client and

the Gateway

Gateway:

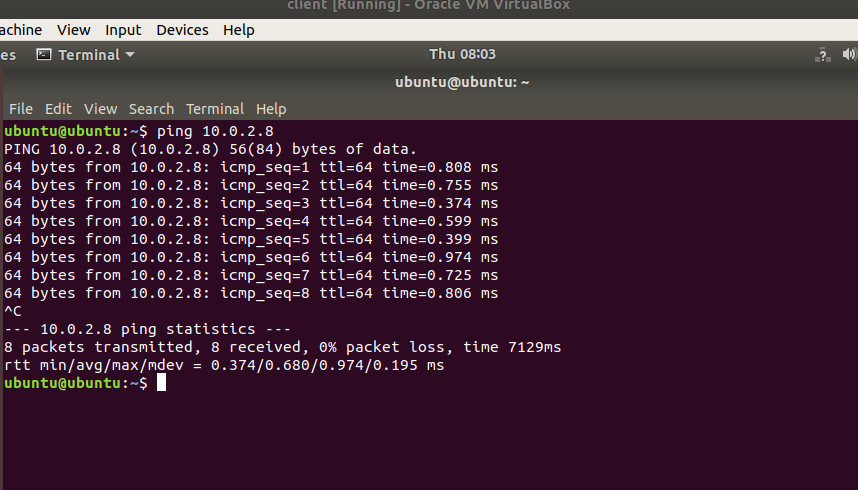


Client:

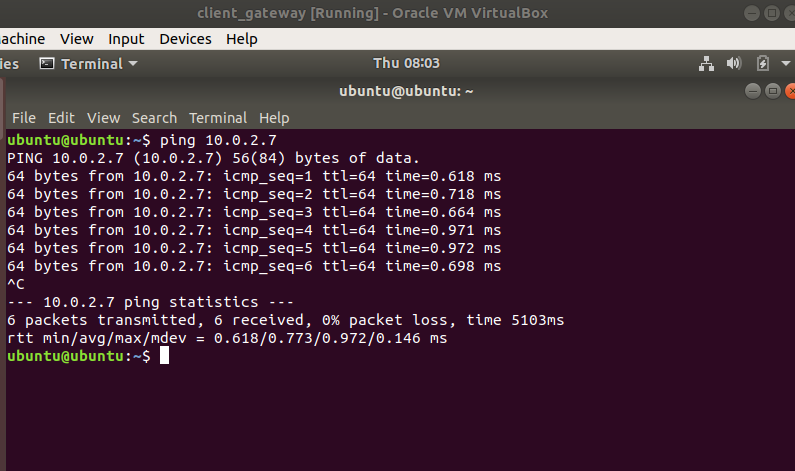


**Check Client – Gateway connectivity**

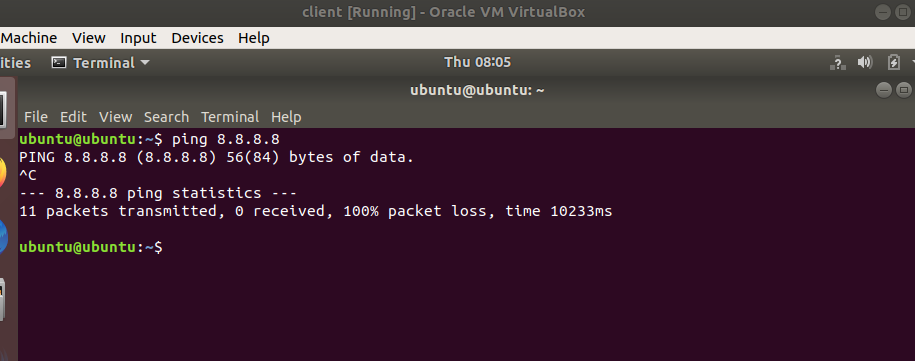
For that I run ping from client to Gateway(10.0.2.8)



From Gateway to Client (10.0.2.7)

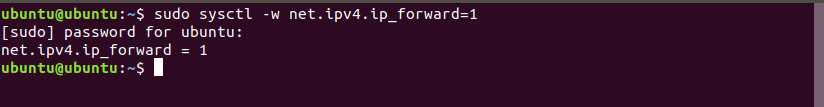


From Client to internet (Google DNS) ping should not work. To enable that we should allow packet FW on the Gateway



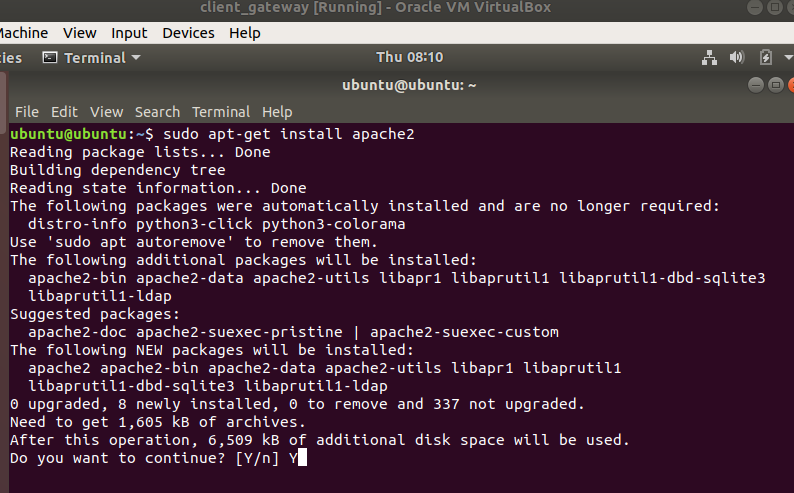
**Enable packet forwarding on Gateway**

In order the client would be able to send/receive packet from the internet via the gateway, I had to setup the gateway machine Linux kernel driver to enable packet forwarding.

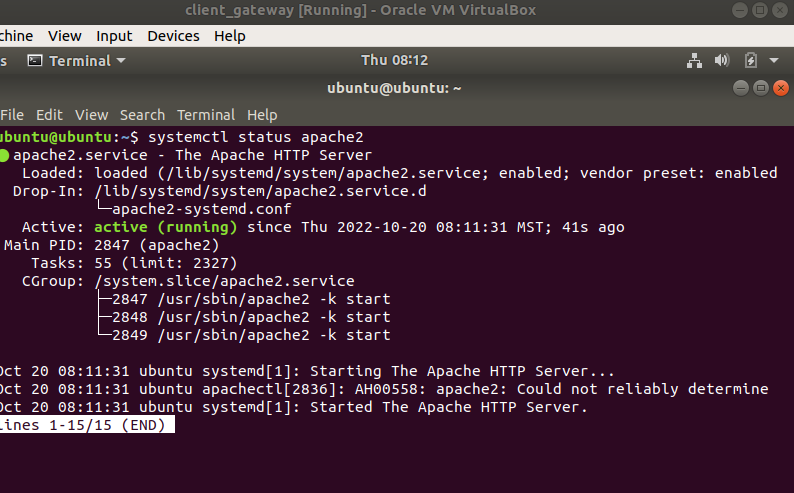


**Connectivity check**

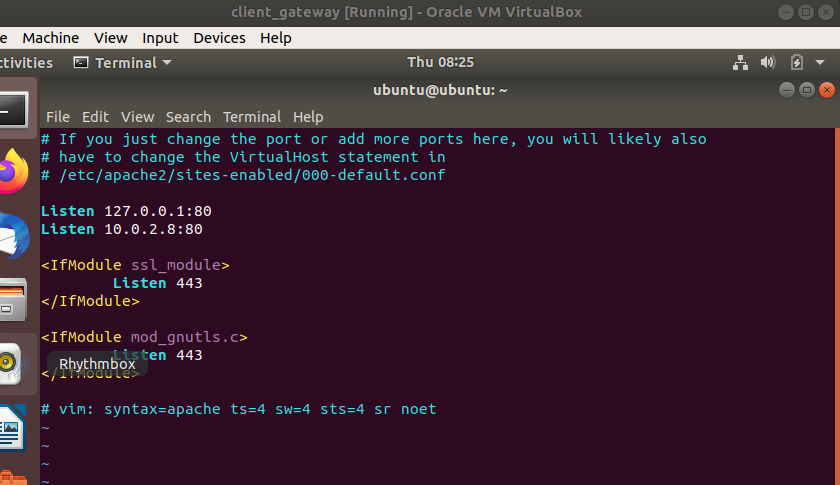
Install Apache web server in the Gateway



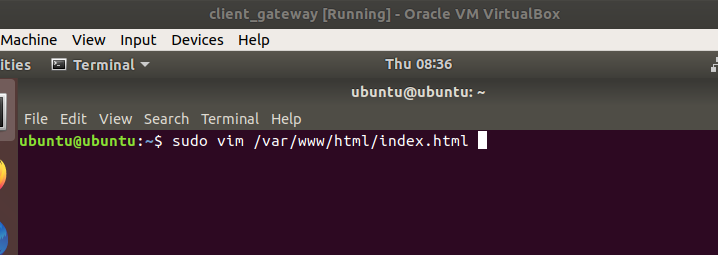
Make sure Apache server is running on the Gateway



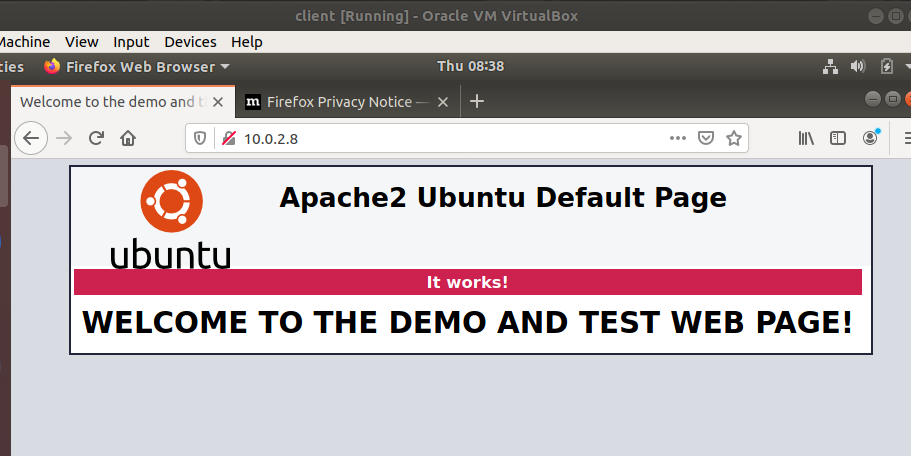
Set Apache web server to Listen to client requests on port80 (you can see here, gateway local ip address and default port are set in /etc/apache2/ports.con



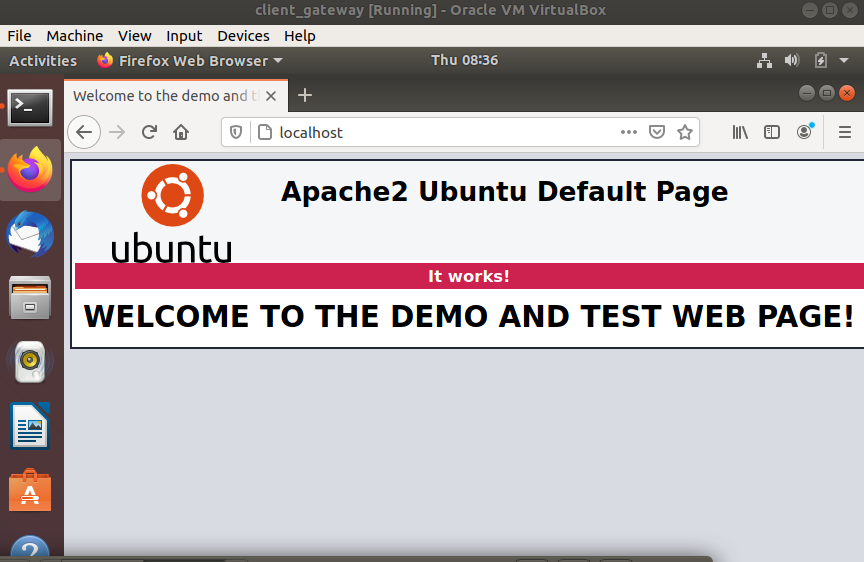
First edit index.html with the custom message



Open the client browser and set the Gateway IP address (10.0.2.8)

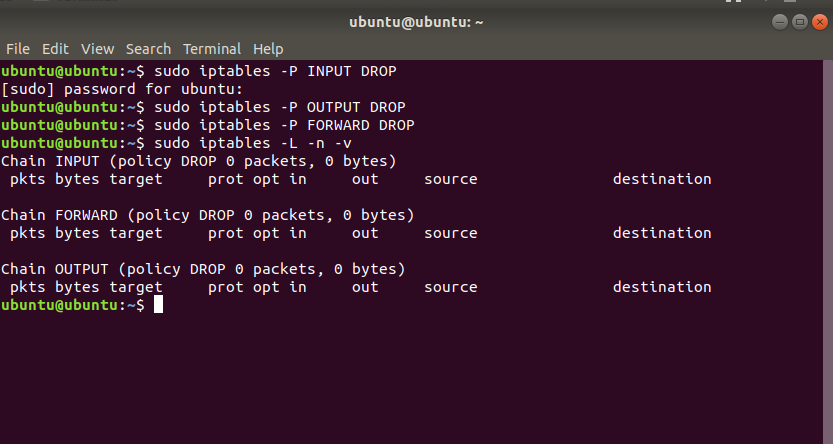


Open the Gateway browser and type ‘localhost’



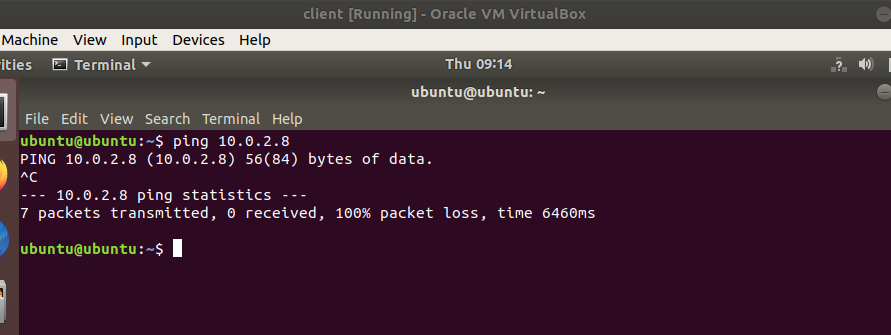
**Stateless Packet Filter firewall**

First, switch from ‘blacklist’ to ‘whitelist’ type filtering on the Gateway. In order to that, set iptables policies to DROP for the INPUT, OUTPUT and FORWARD chains. Below you can see the state of iptables after whitelisting the chains

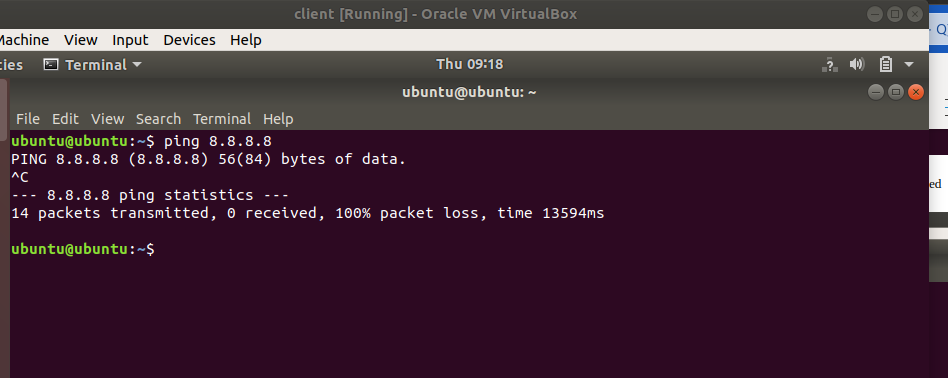


After this step, traffic from client to gateway and gateway to client is blocked

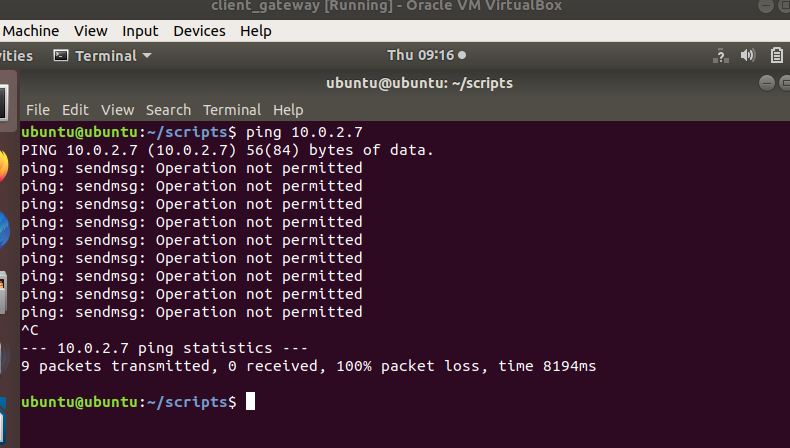
Ping client to Gateway



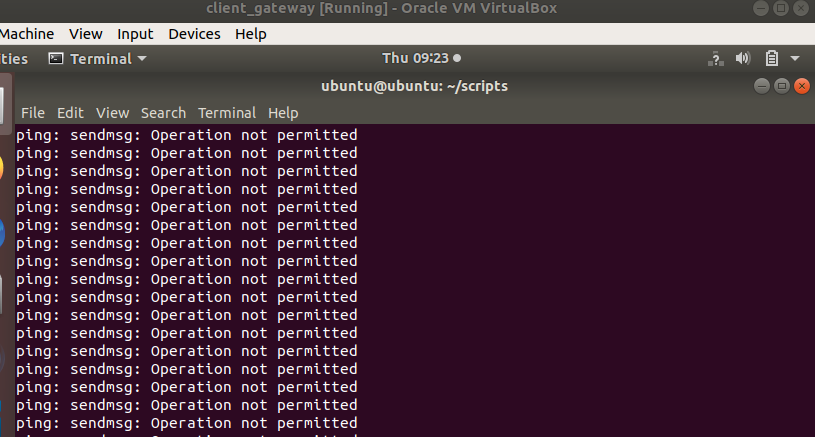
Also, Client is not able to ping to the network (Google DNS 8.8.8.8)



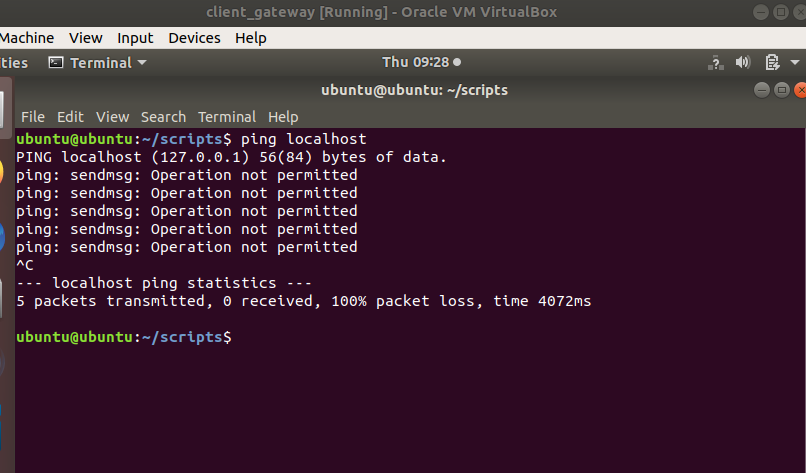
The same for the gateway, ping to the client is blocked



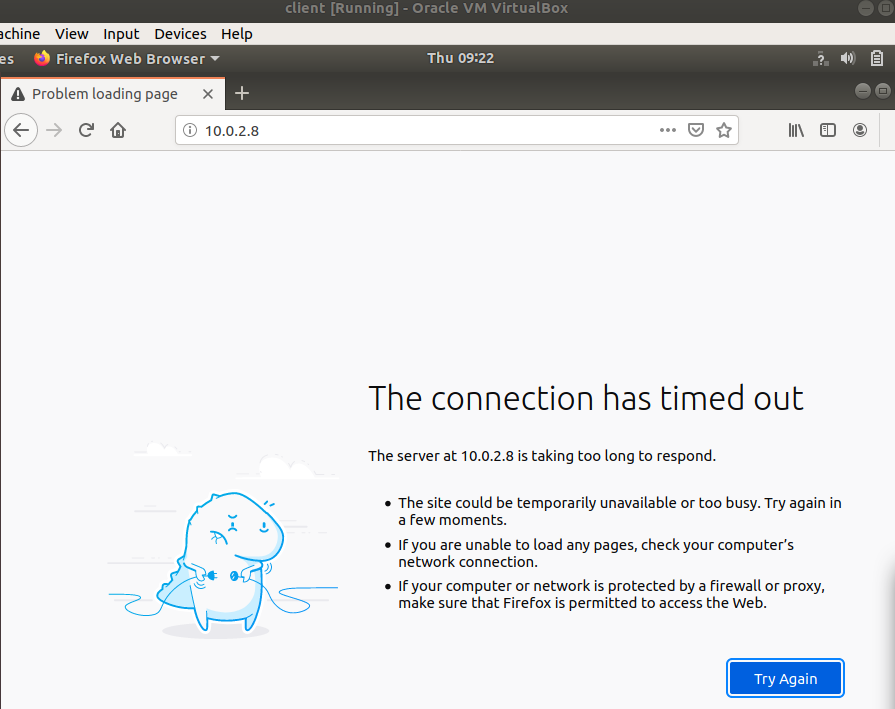
And also ping to Googles 8.8.8.8 from the Gateway is blocked



Ping from gateway to local host is also blocked



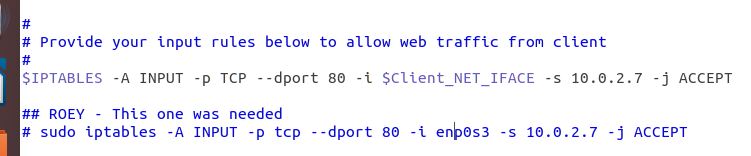
The client can not access the web page hosted by the web server running on the gateway host



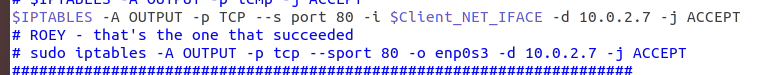
**Client Web Page access**

For that I added the following iptables rules

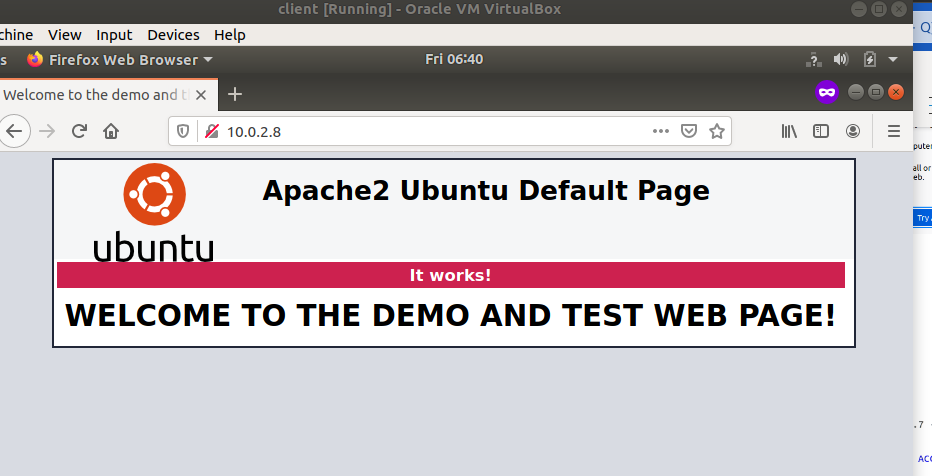
This one is to accept connections http connection from client ip address



Allow http response back to the client

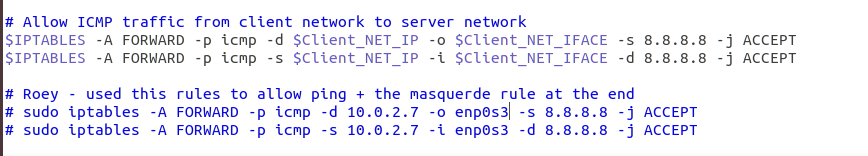


After those configurations I was able to launch the web pages hosted on the server

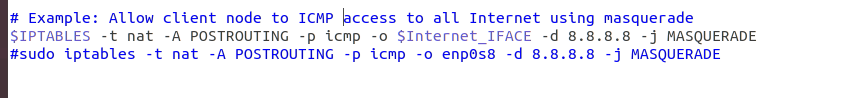


**Enable client to ping 8.8.8.8**

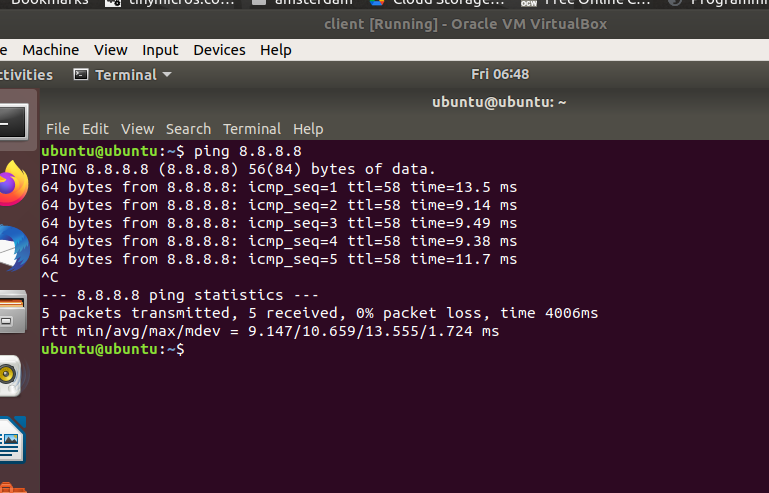
Symmetric rule to allow inbound and outbound icmp from/to the client



Use this rule to set the source ip to the gateway to access the public network

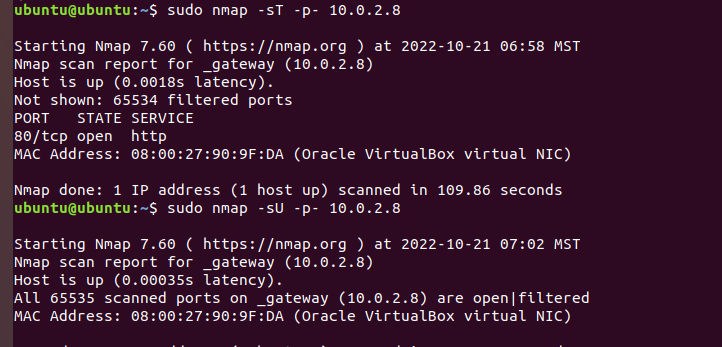


After that, ping from client to 8.8.8.8 was enabled



**Port scan using nmap**

We were asked to run scan open UDP and TCP ports



And received the expected output

# Conclusion

In this project I learned how iptables controls packets flow under different use cases. When working on this project I learned how to use Wireshark to inspect and understand packets flow coming from the client side.

Besides Wireshark, the best way to debug and understand how to set up correctly the iptables was to use the following command in another terminal tab:

sudo watch –d iptable –L –n –v

By using this command, I was able to see counter change for each of the tables. This helped to understand if the filter rules are correct or not.

For me, I have invested a lot of time with setting the virtual machine. First, I tried running the VMs on my OSx based machine. There were many performance issues (or maybe others) which cause the virtual machine to crush and not working properly. I tried to look for fix and followed different blogs however, none helped.

After few days, I have decided to the virtual machine on a native ubuntu machine. This try was not smooth as well. Luckly, the guides and explanations I have found in the web, helped with fixing the issues and start working on the project.

# Appendix B: Attached files

<https://github.com/roeybenhayun/cse548-advanced_computer_network_security>

# References