Project 2 OpenFlow Based Stateless firewall

Student Name: Mohammed Ragab

Email: maragab@asu.edu

Submission Date: 8th November 2023

Class Name and Term: CSE548 Fall 2023

Video link: https://youtu.be/C9x4DCE4aEE

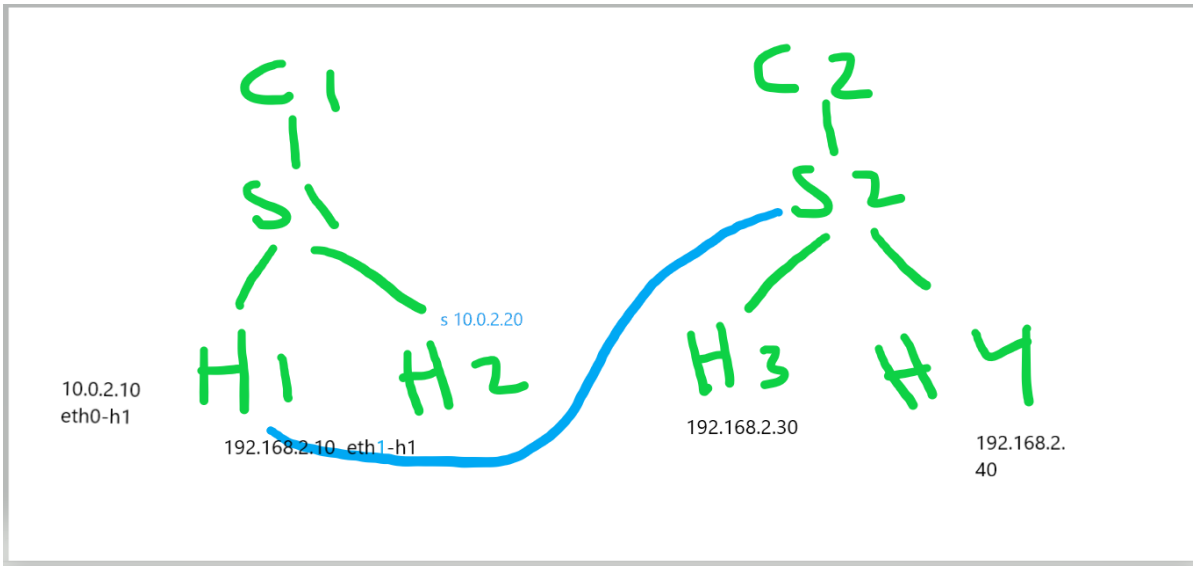
I. Project Overview

I have created a software-defined network firewall using OpenFlow protocol using POX controller and I created a virtual network using Mininet tool and I created a python script the uses the Mininet python APIs. The network has 4 hosts, 2 switches and 2 controllers. The links between these nodes are specified as in the assigned instructions and as shown in my video.

in the terminal window on Mininet (CLI), I added the linked between switches and the controllers as shown in my video

II. Network Setup

- Here is my layout of the network with the assigned IP addresses for each host,



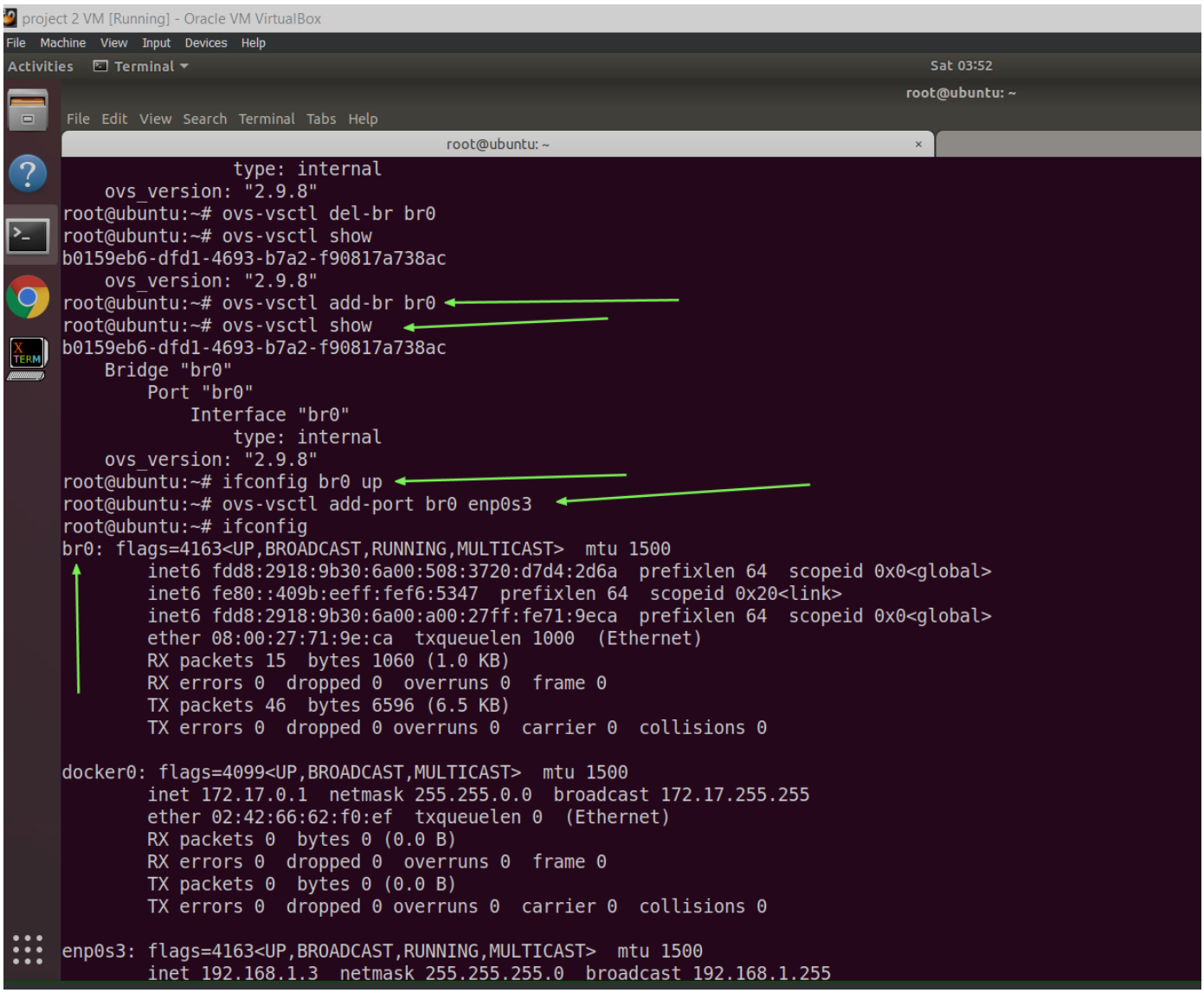
III. Software

- Mininet
- POX
- Containernet
- Ping
- Tcpdump
- Python3
- VScode as text editor
- Linux ubuntu
- Xterm terminal

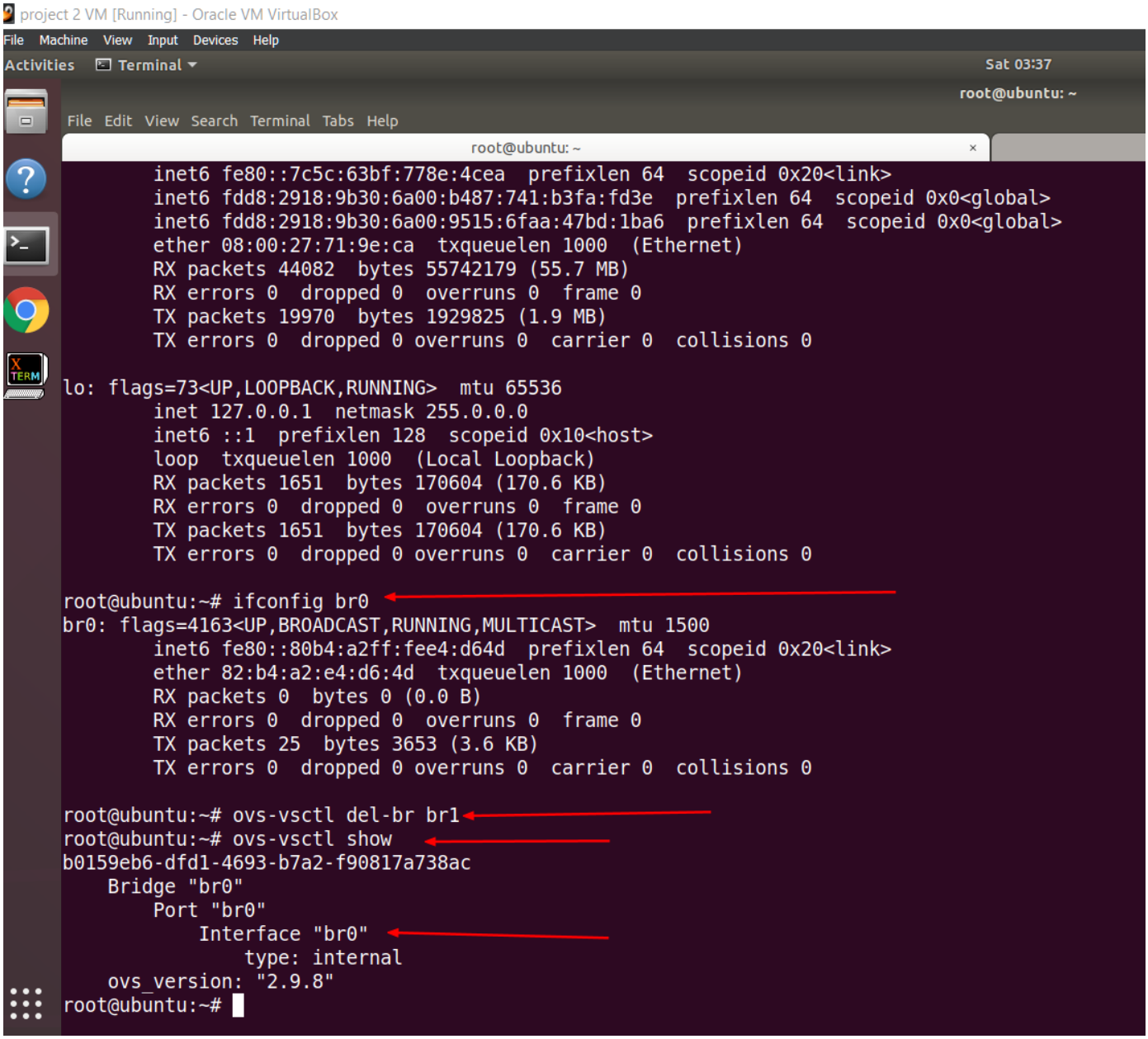
IV. Project Description

OVS lab

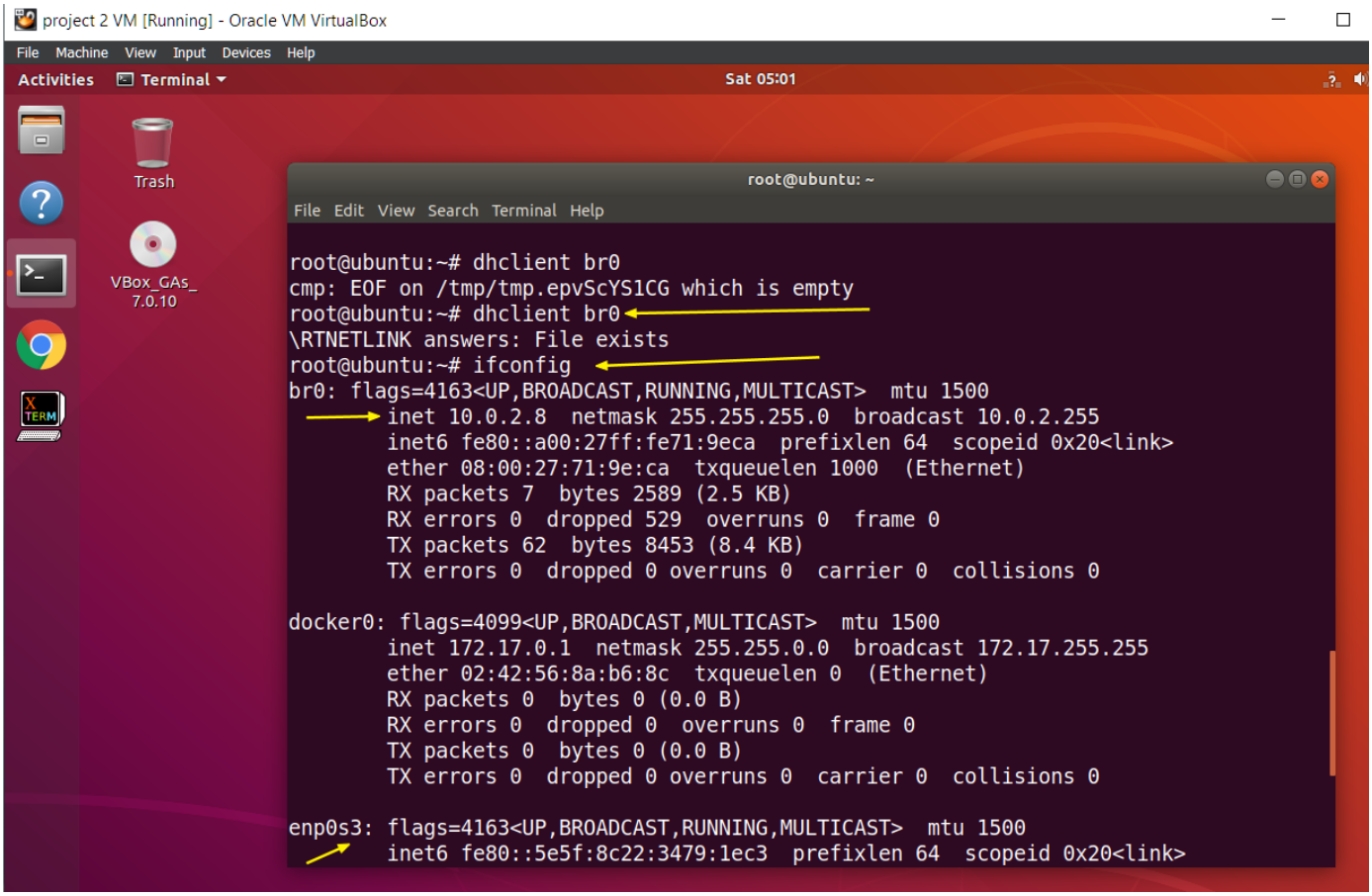
add internet gateway interface to bridge 0



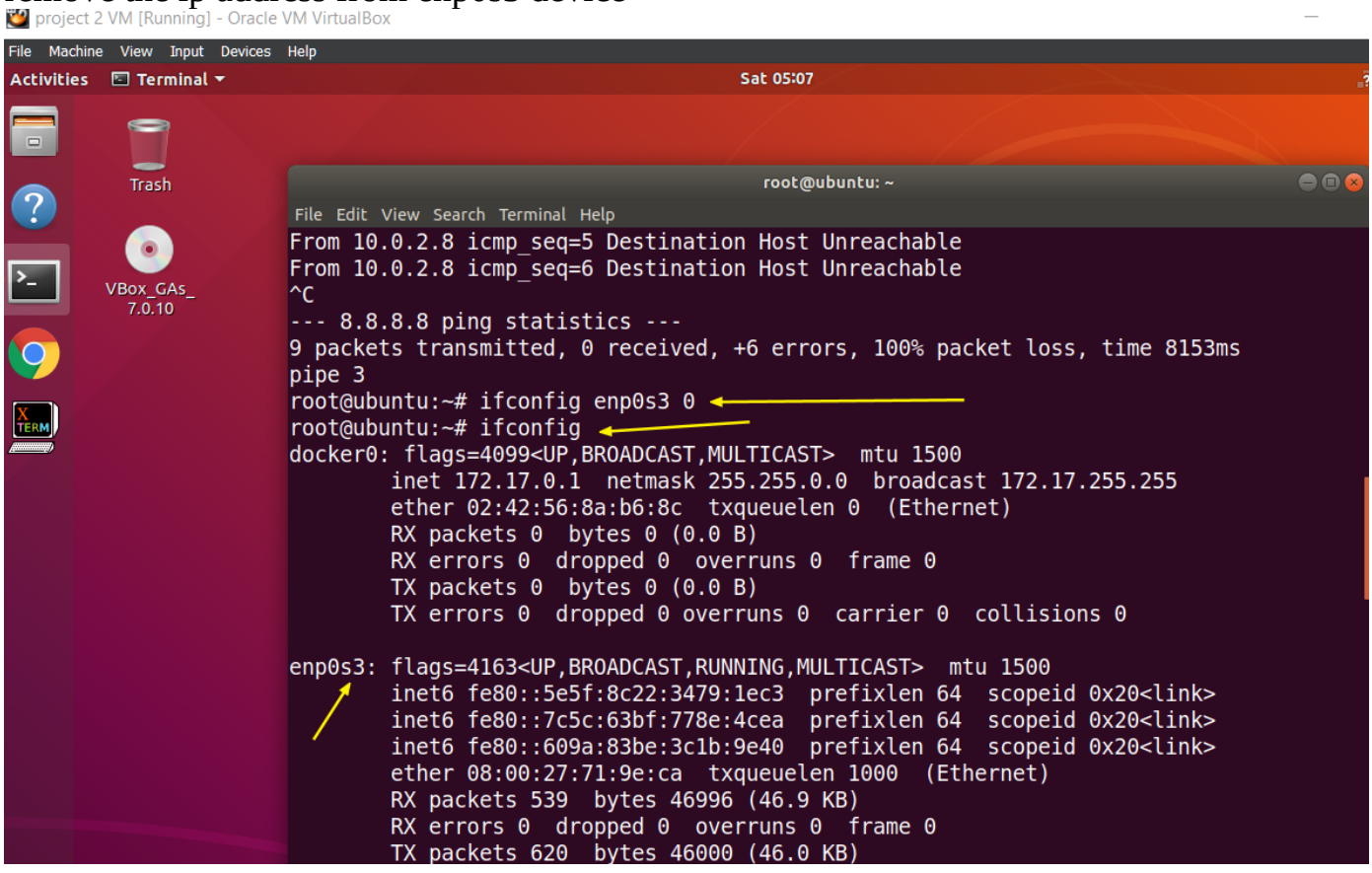
Check bridge setup



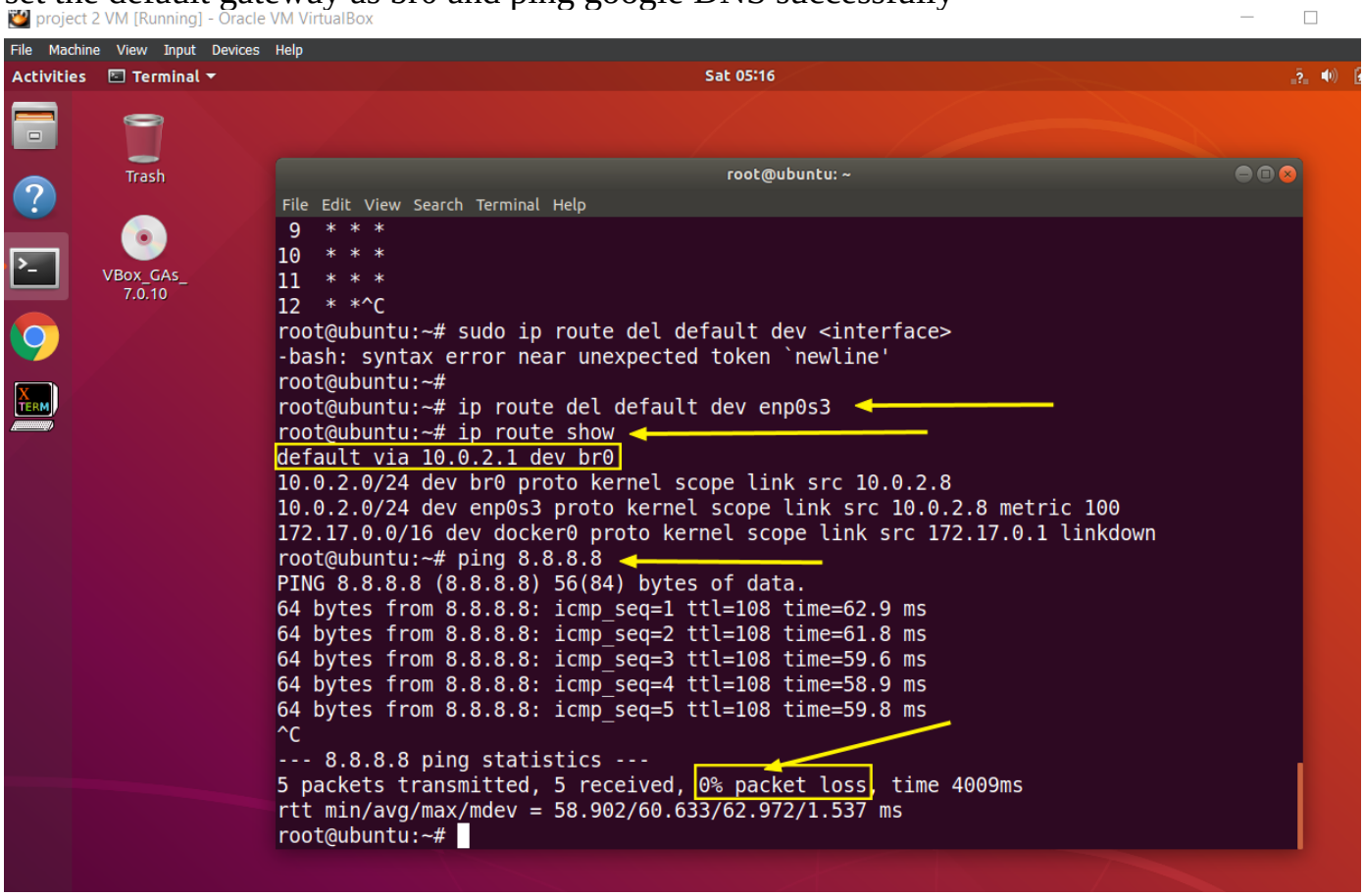
assign IP to br0 using dhclinet command



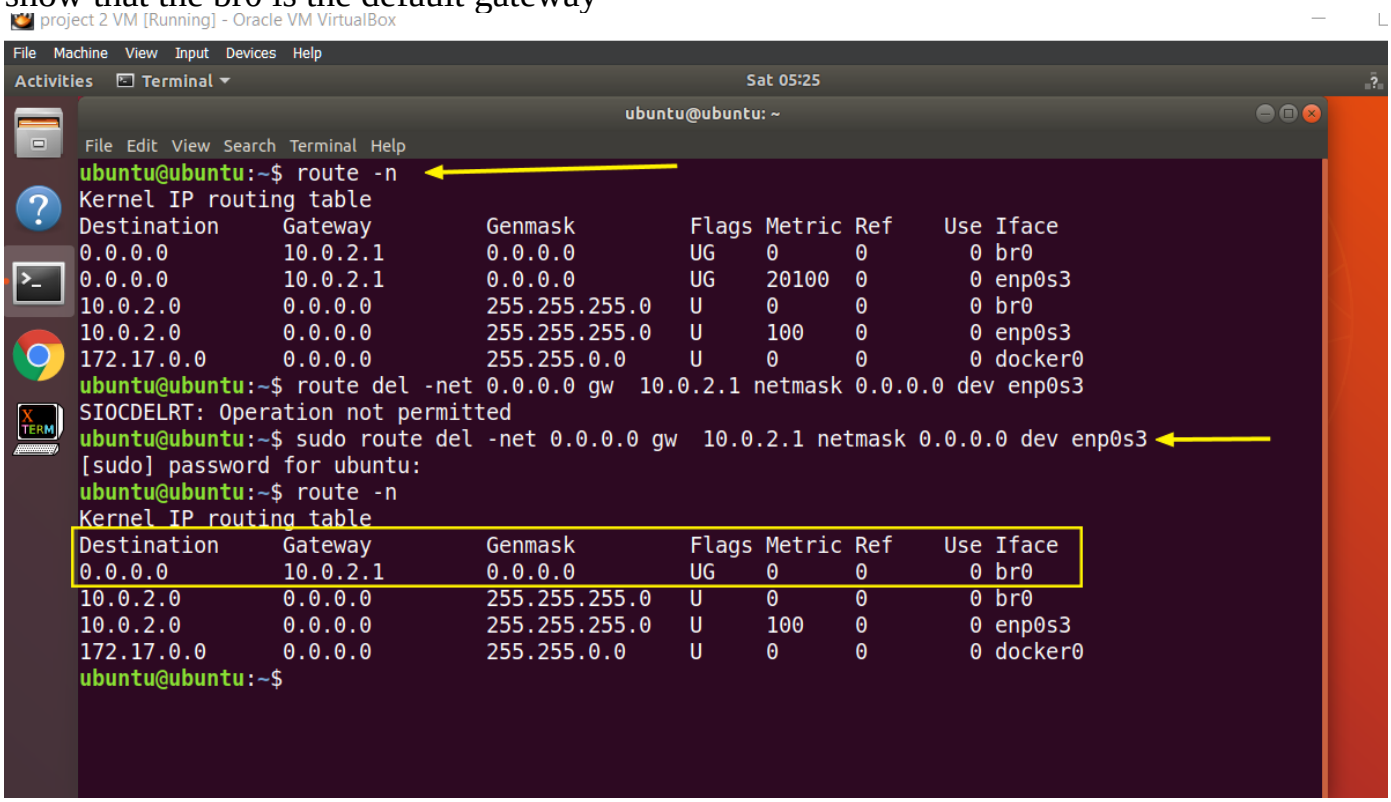
remove the ip address from enp0s3 device



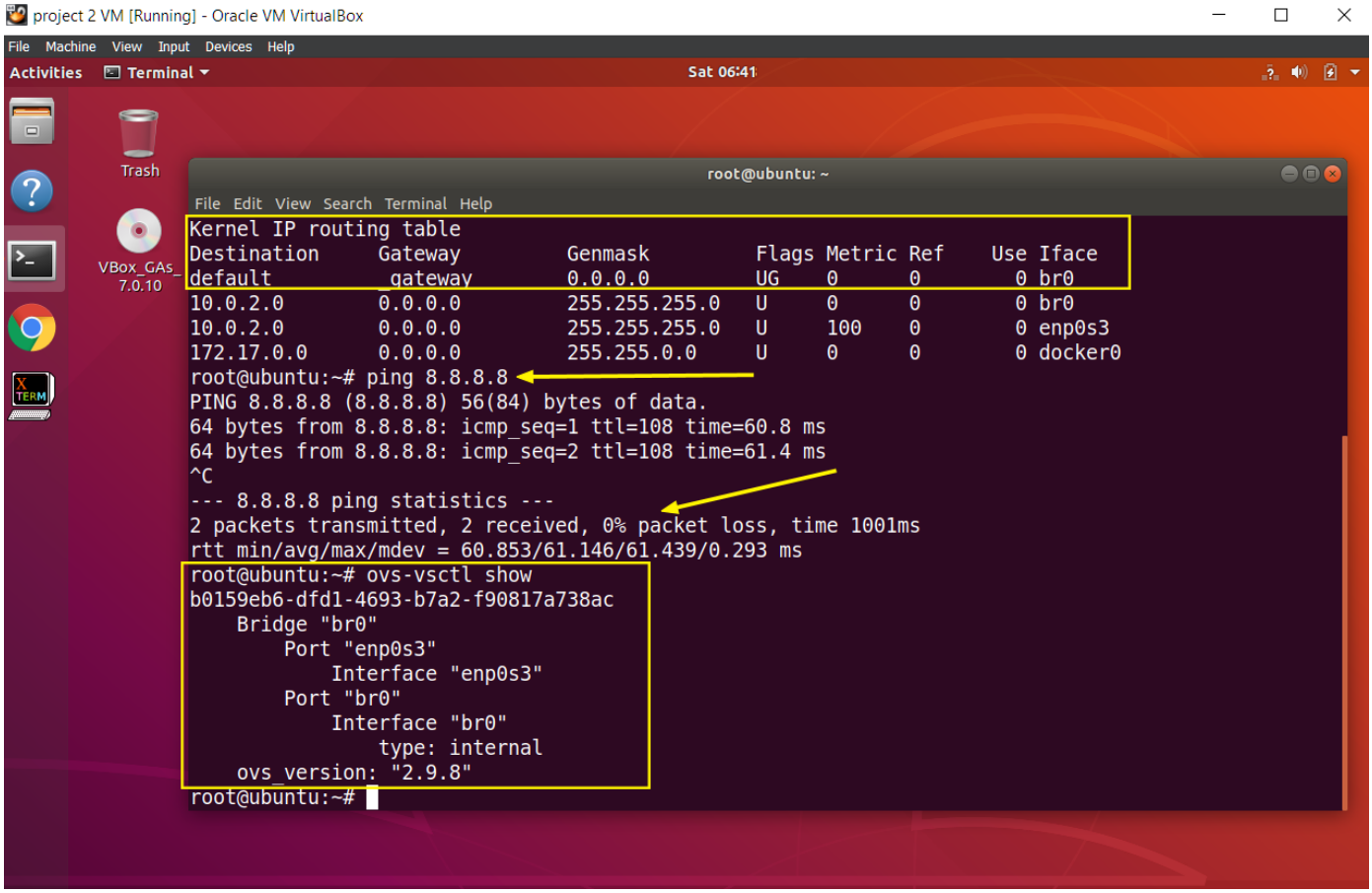
set the default gateway as br0 and ping google DNS successfully



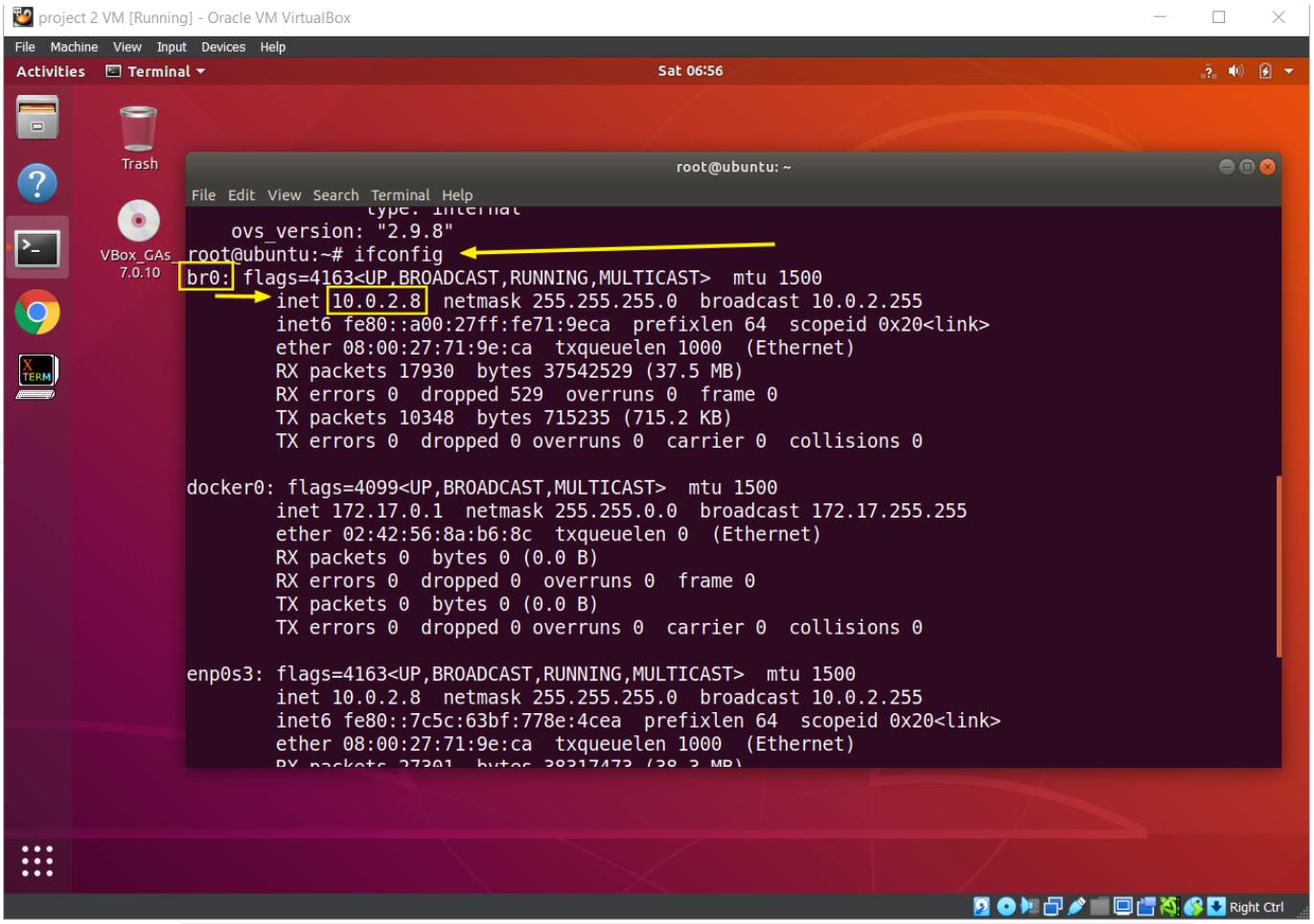
show that the br0 is the default gateway



show br0 and its interface and can ping google dns successfully and default route is the br0

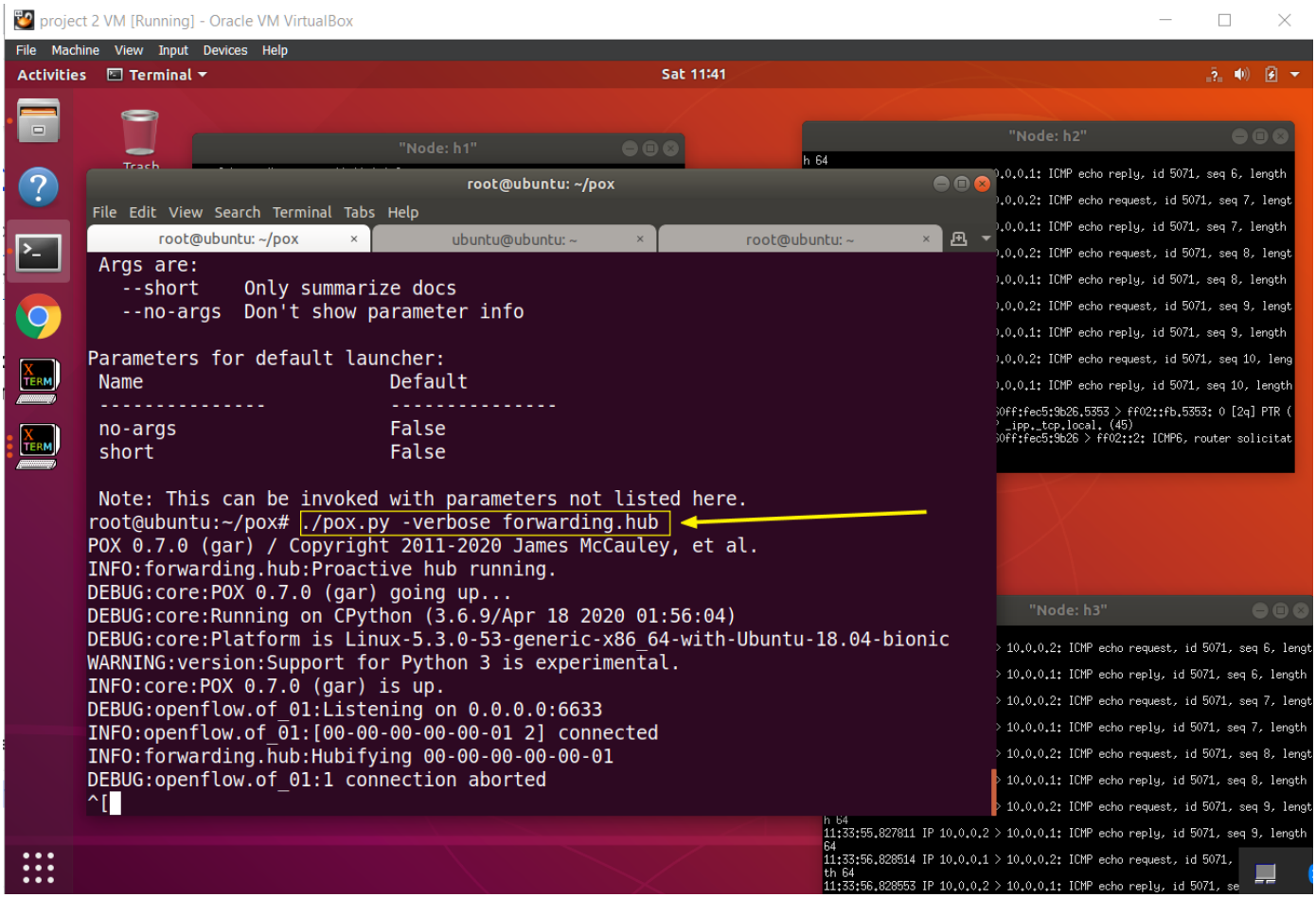


show the ip address of br0

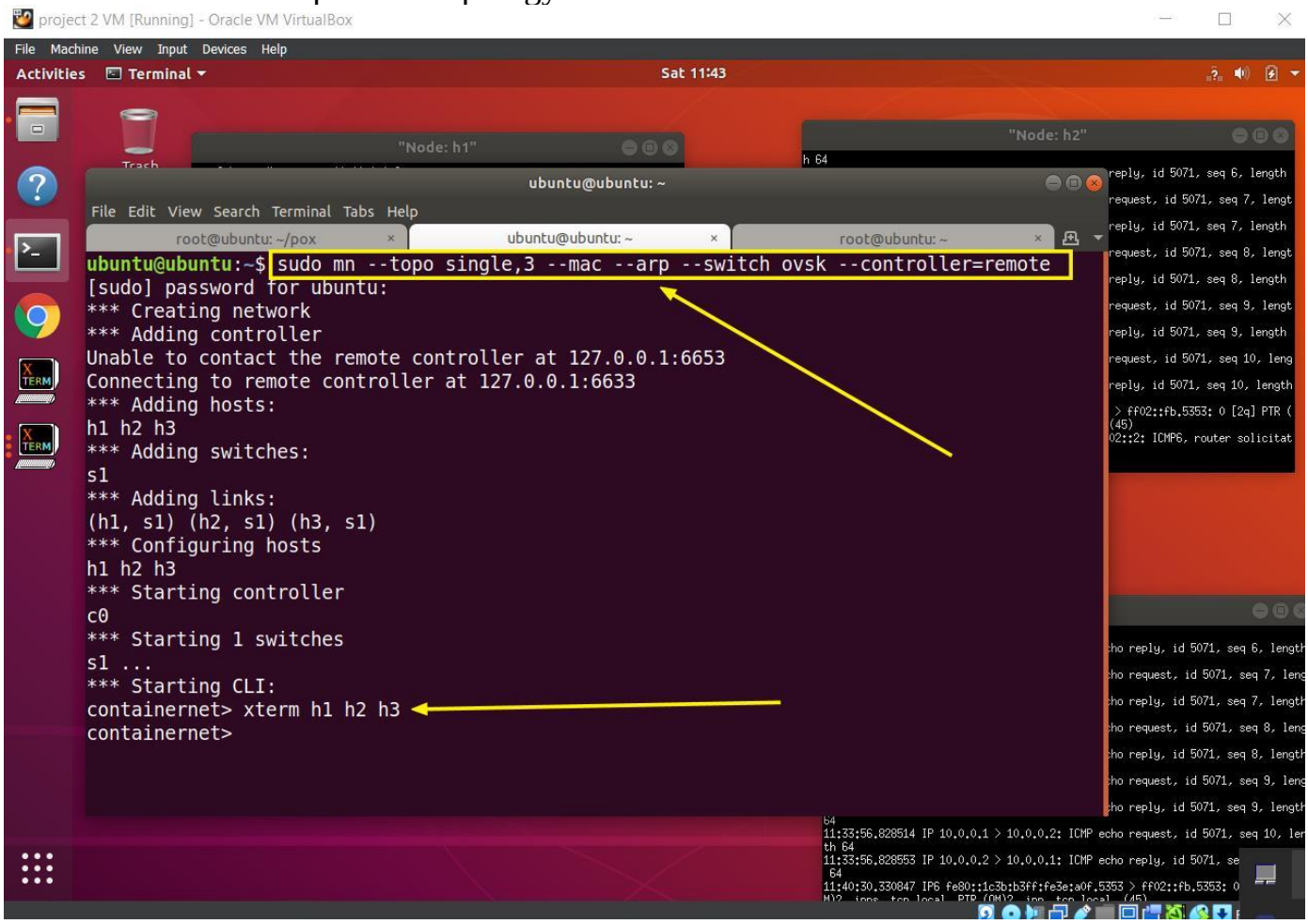


POX lab

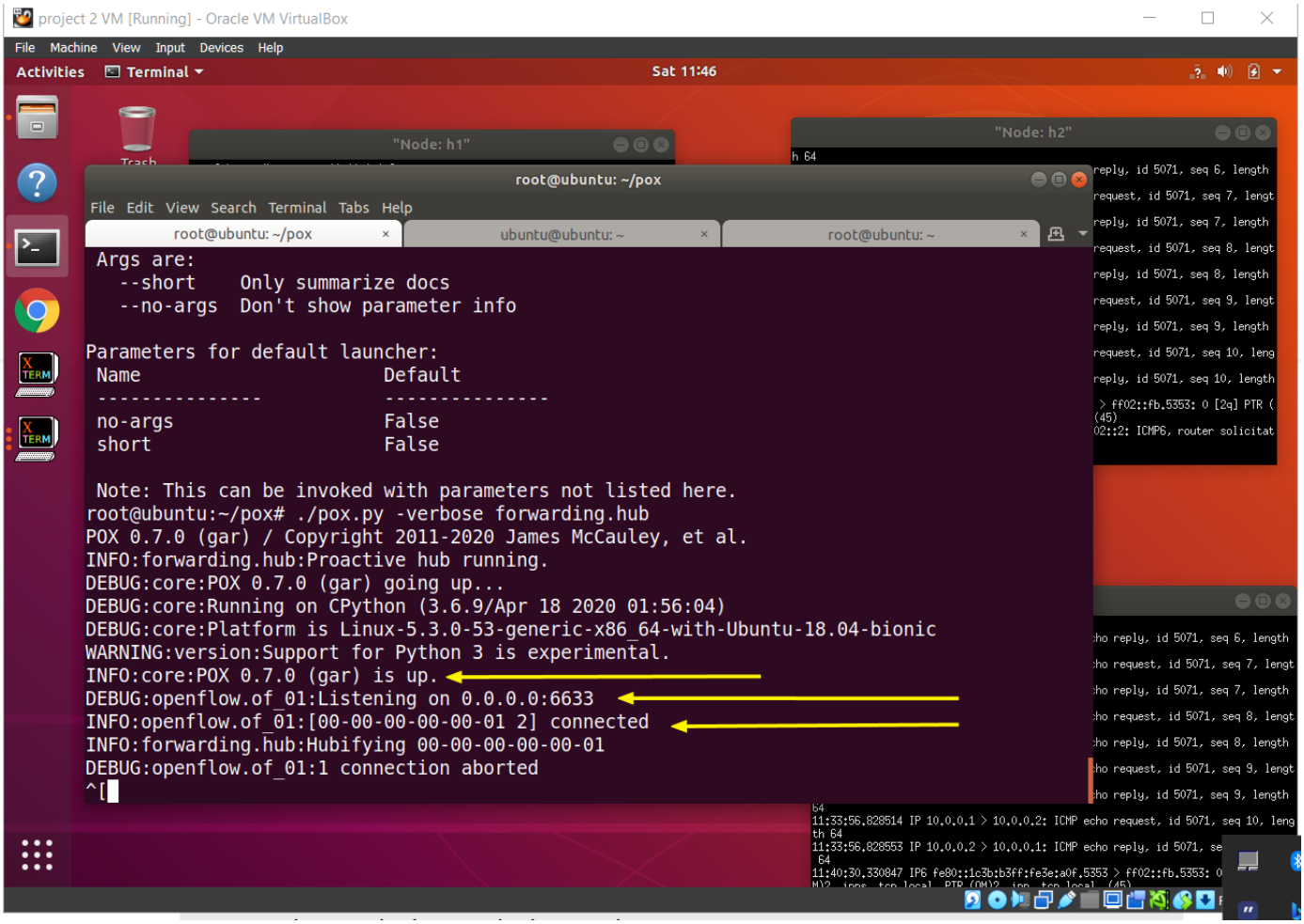
run POX



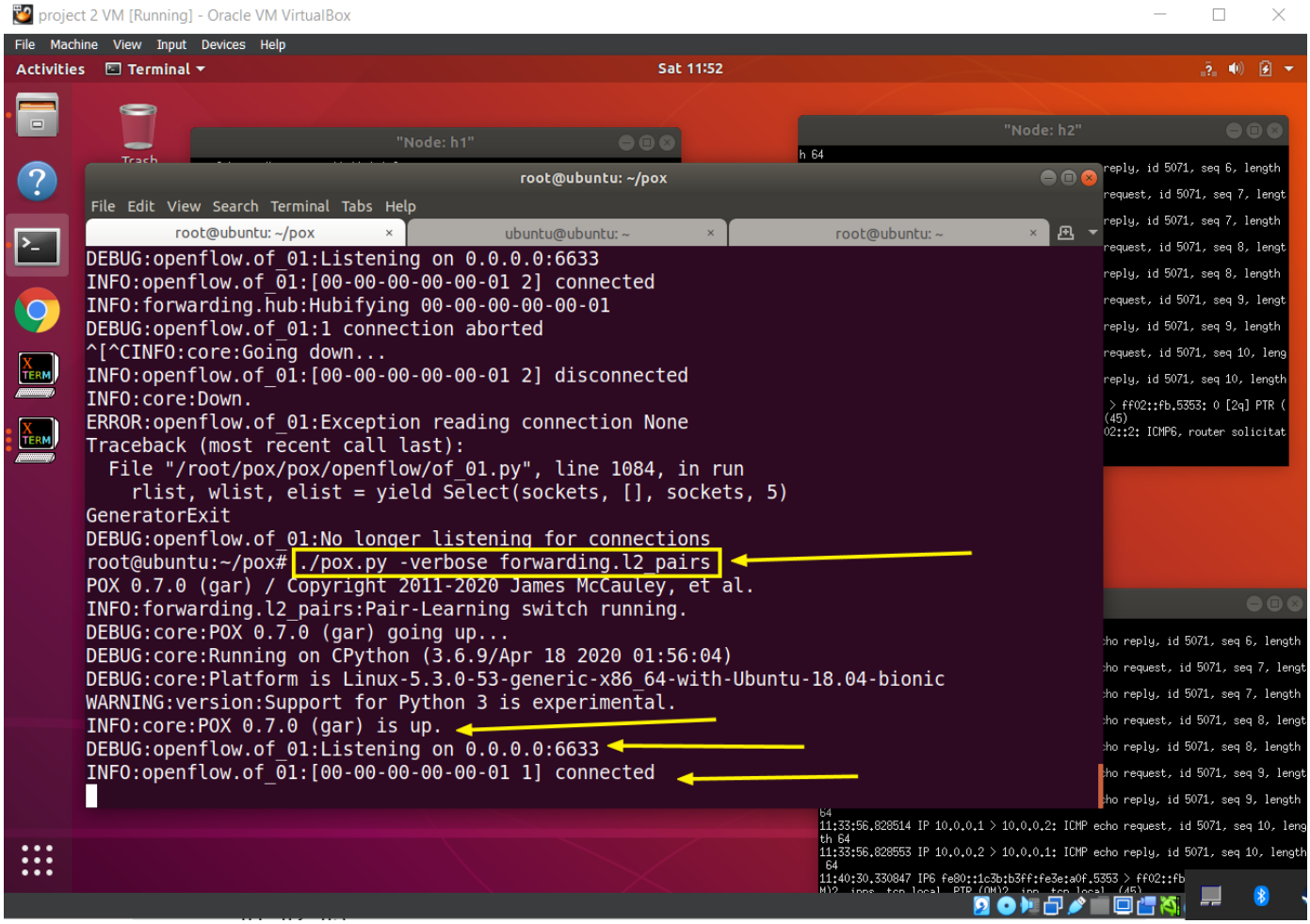
run mininet with the specified topology



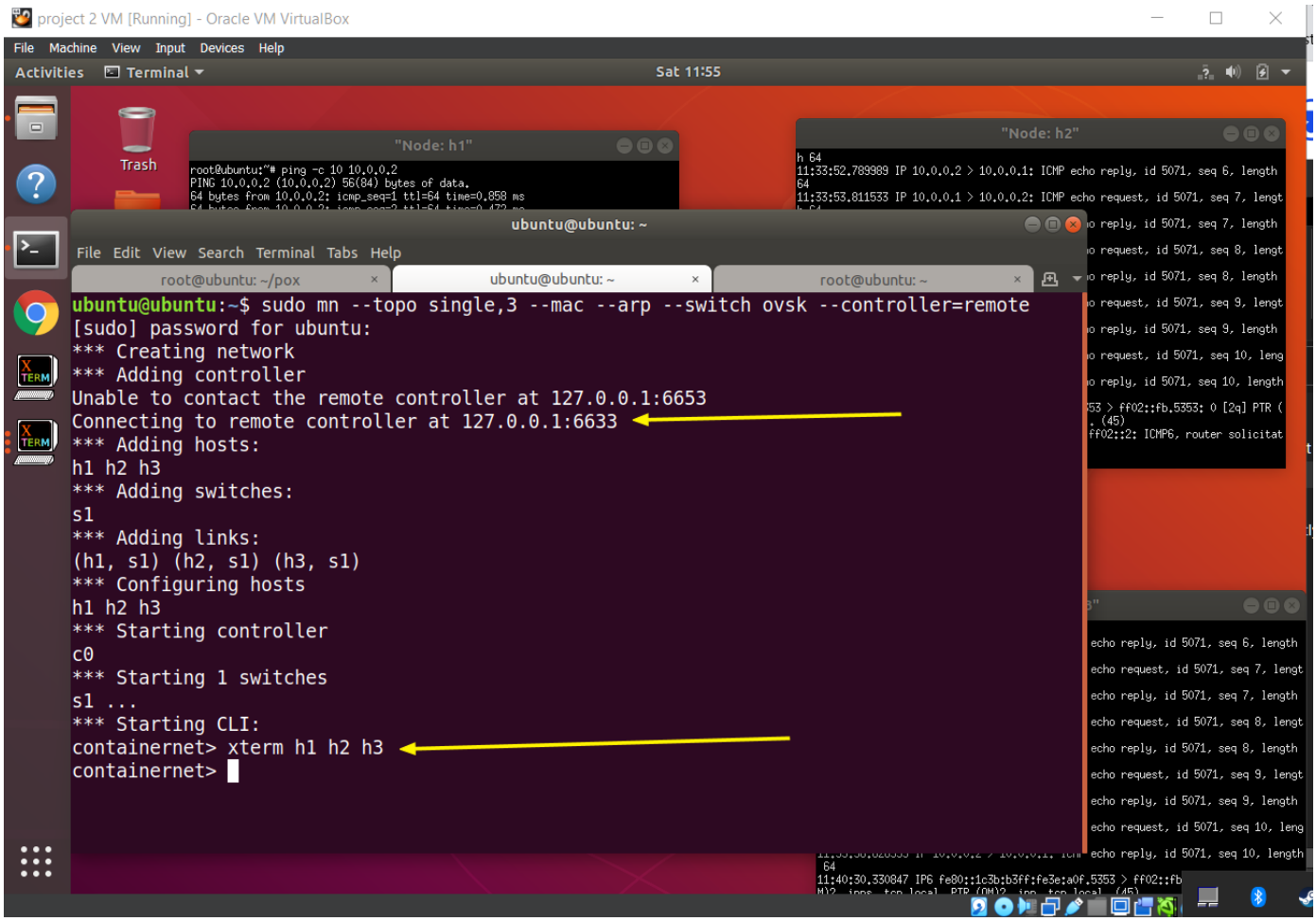
pox controller is up and running on the mininet network



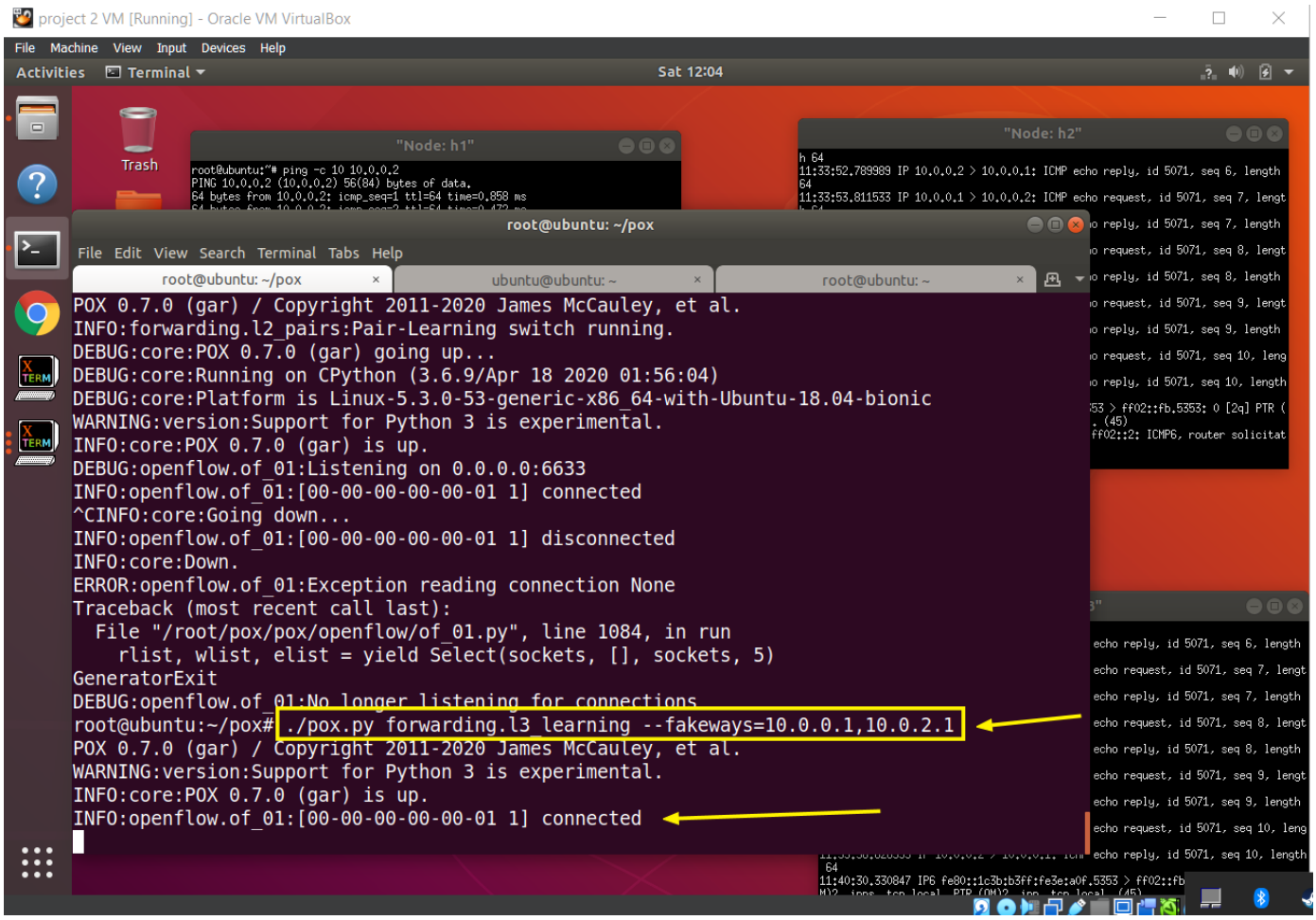
start the forwarding.l2 pairs component correctly



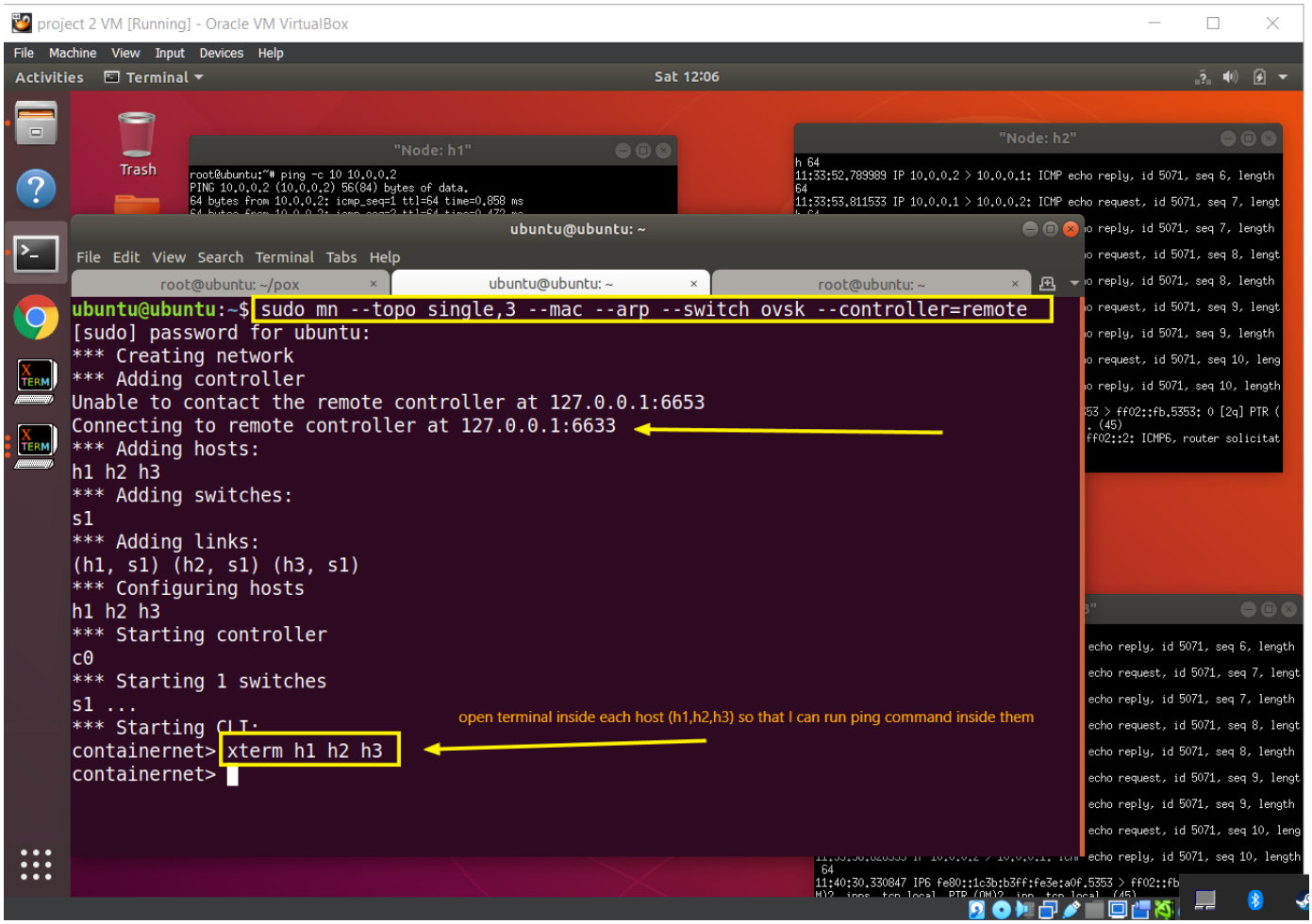
mininet topology after running POX controller with forwarding.l2 pairs component correctly



run POX with fake gateways

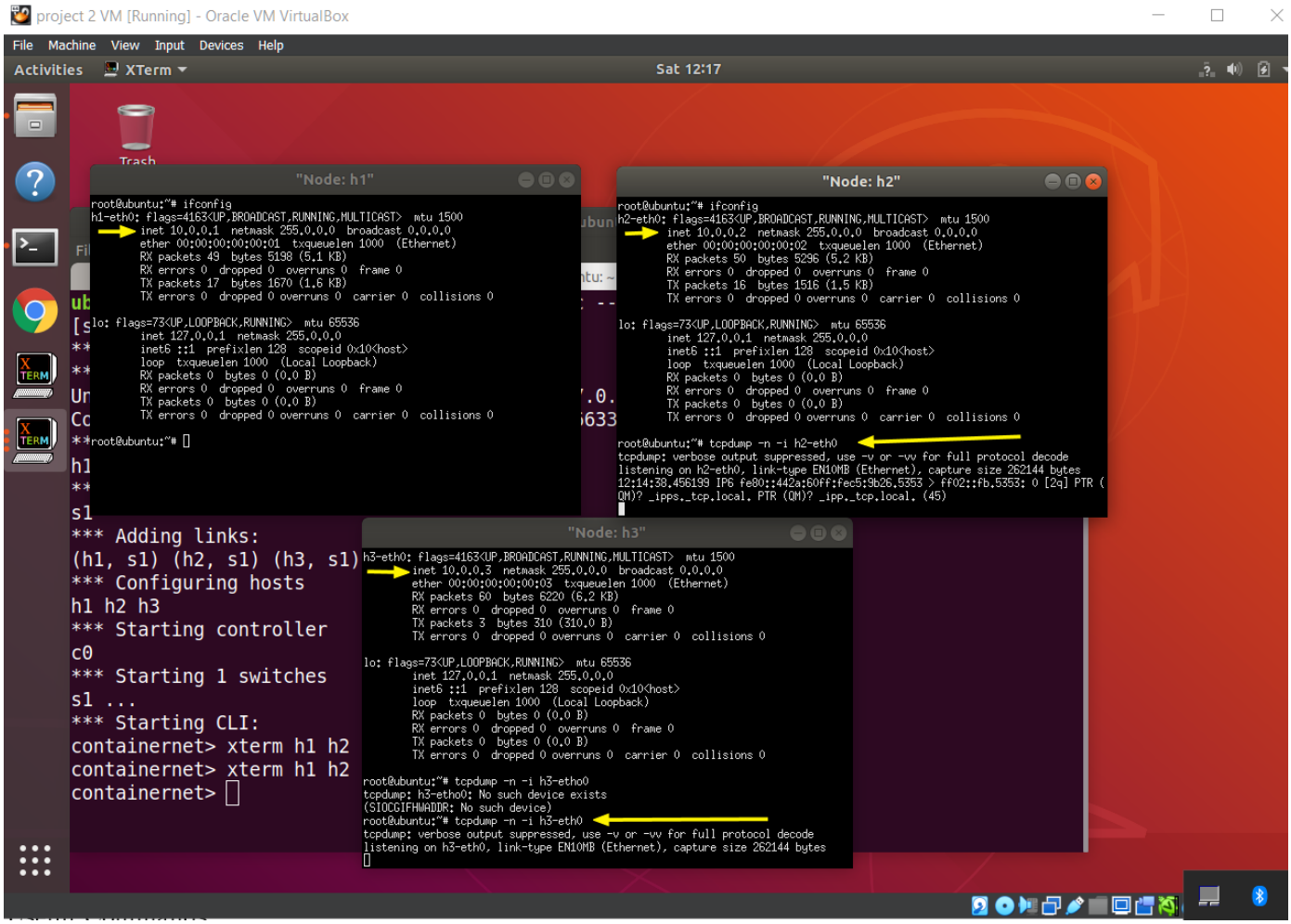


mininet topology after run POX with fake gateways

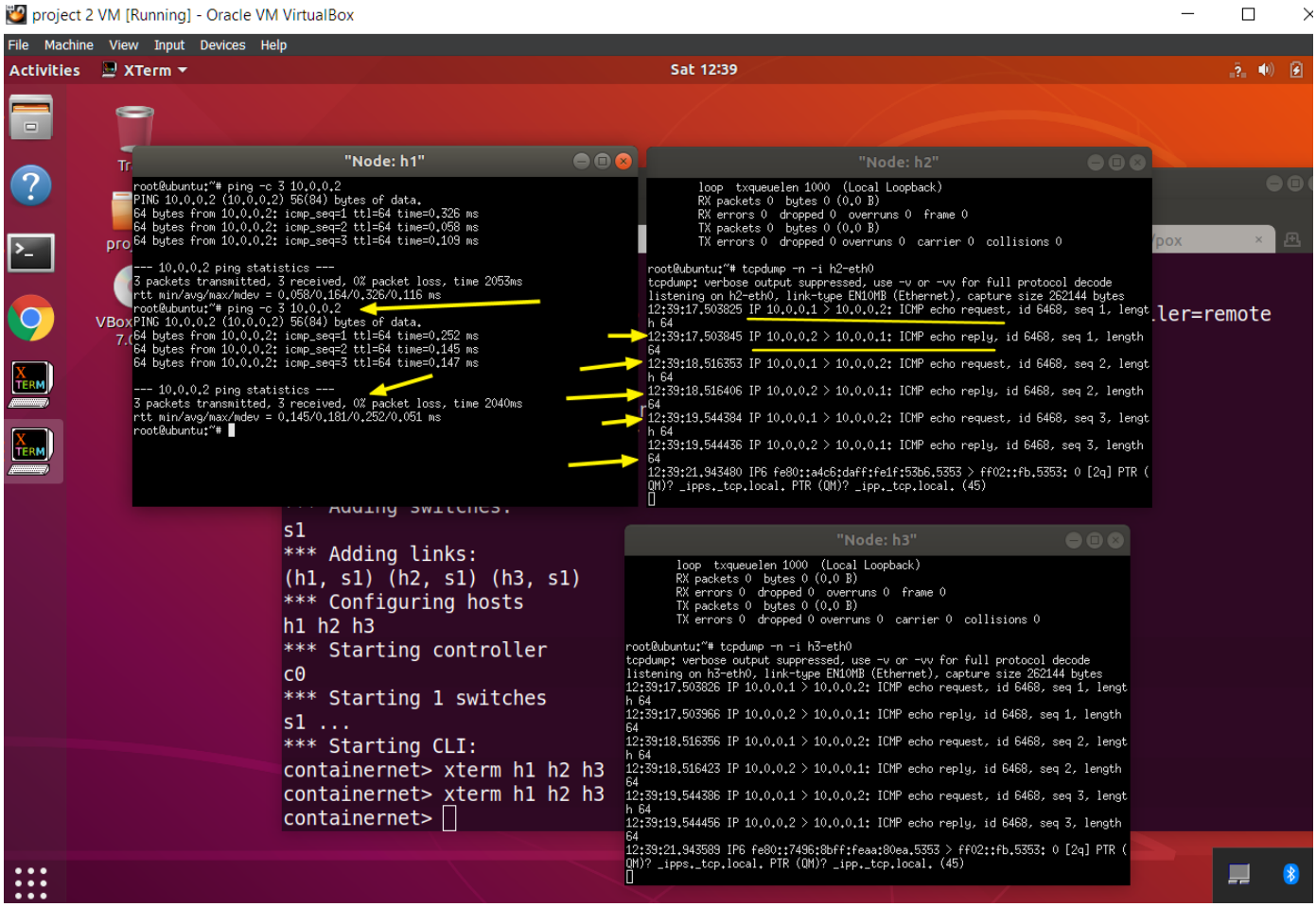


xterm of 3 hosts and their IP addresses and setting TCP dump on h2 and h3 to capture the ping request from h1 to h2

Please edit the highlighted portion.

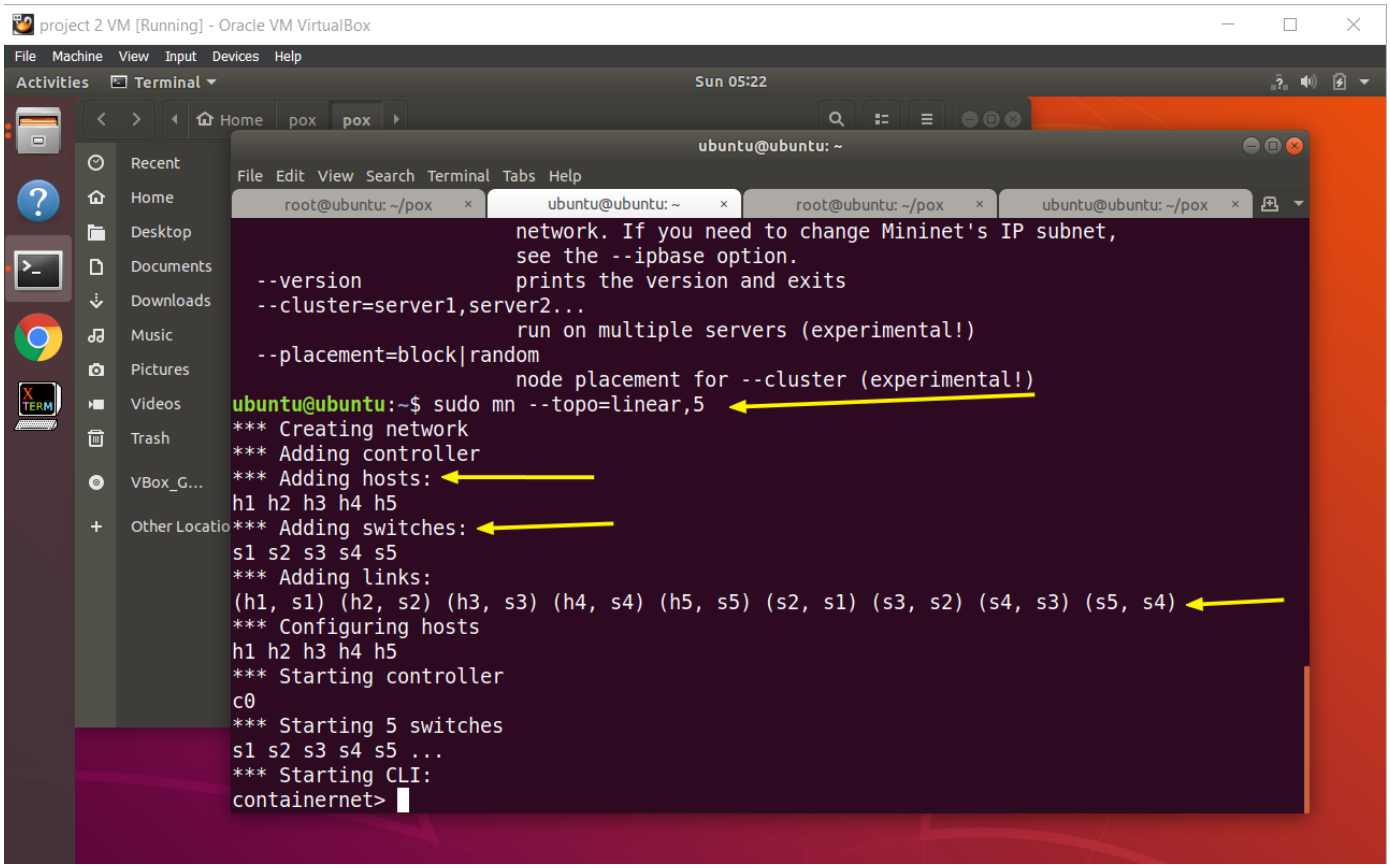


host 1 ping host 2 and TCPDump logs the ping request and reply showing the IP of host 1 and host 2

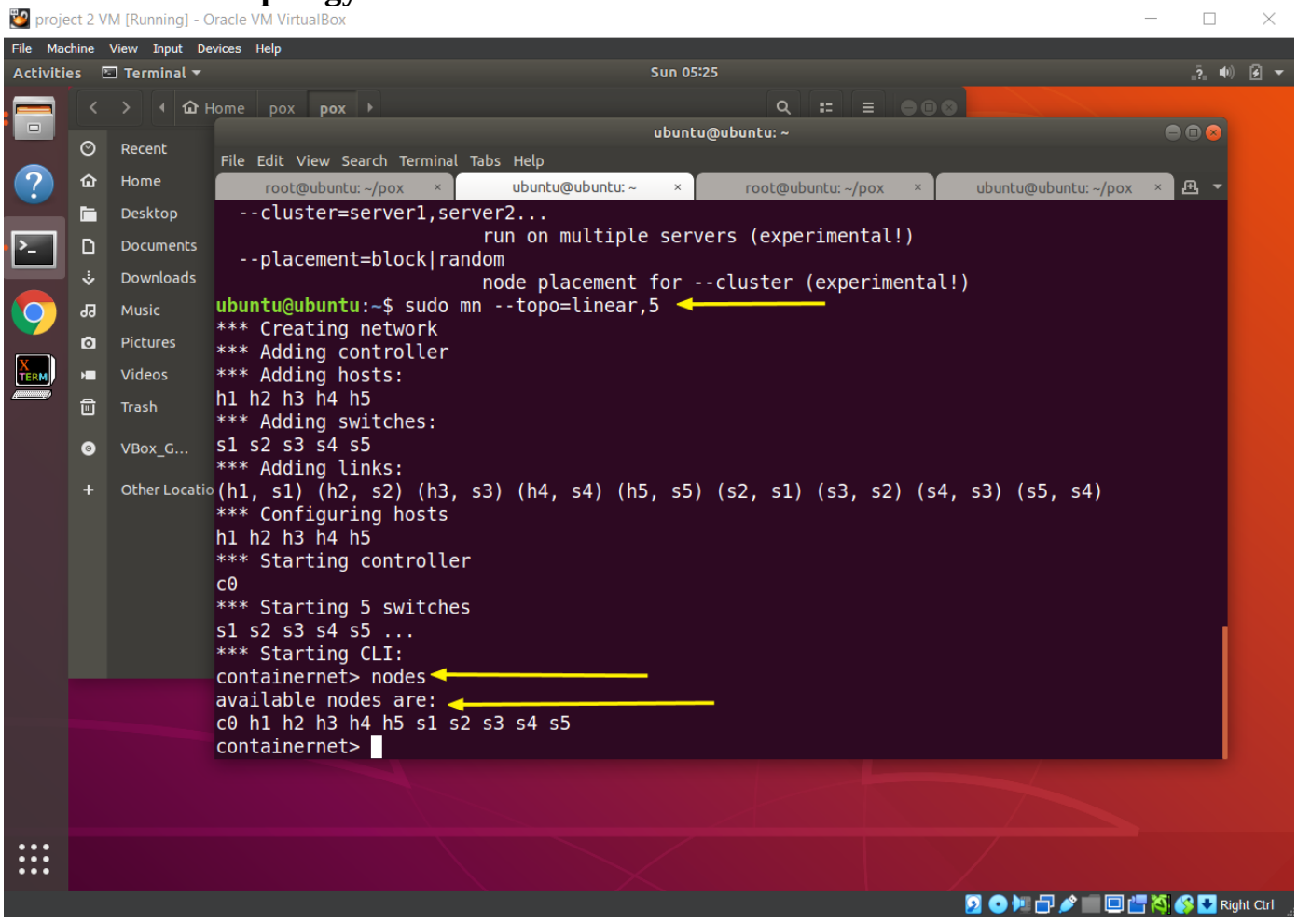


Mininet Lab

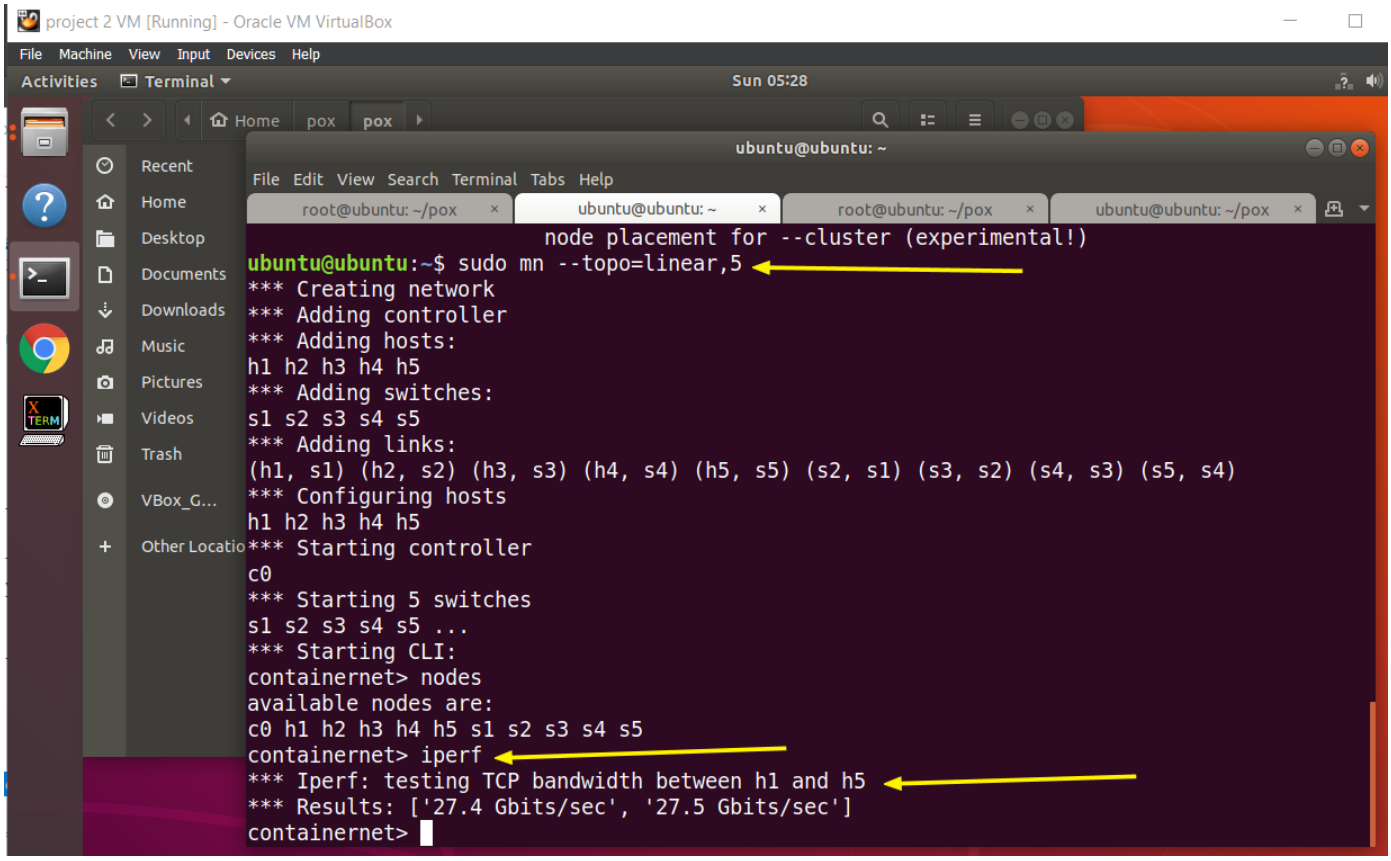
create linear topology with 5 switches and 1 host for each switch



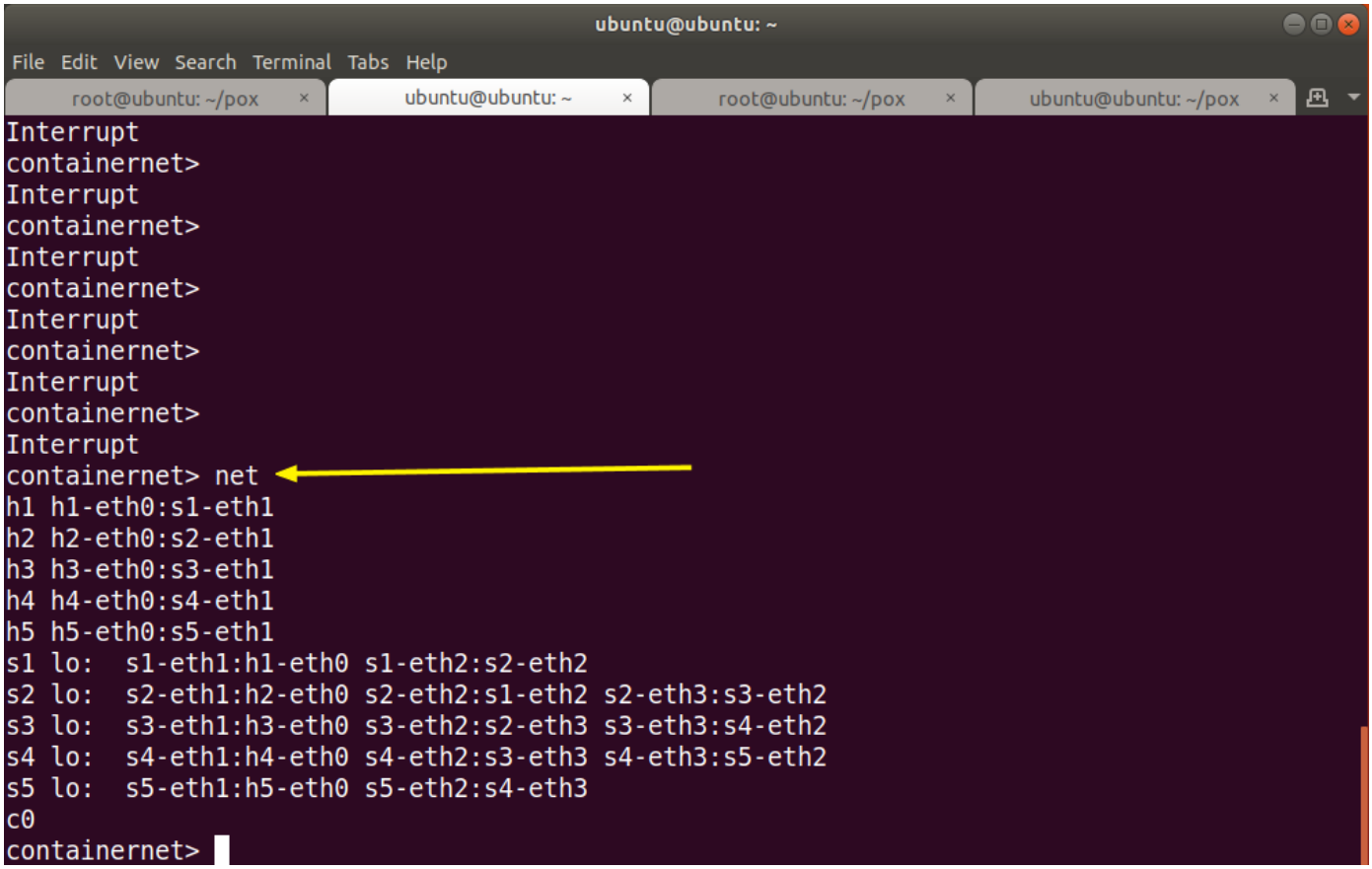
nodes of linear topology with 5 switches and 1 host for each switch



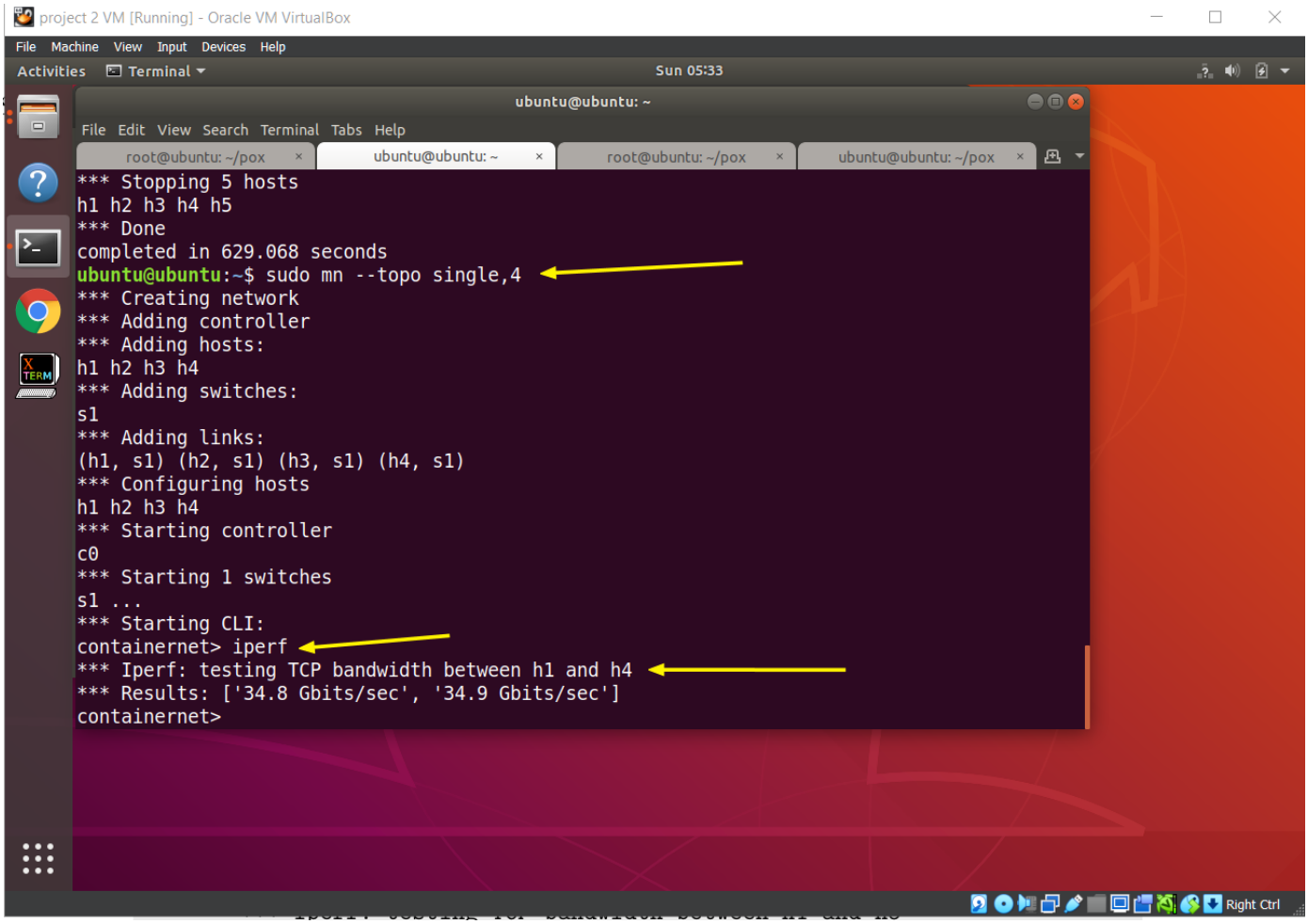
measure bandwidth of linear topology with 5 switches and 1 host for each switch



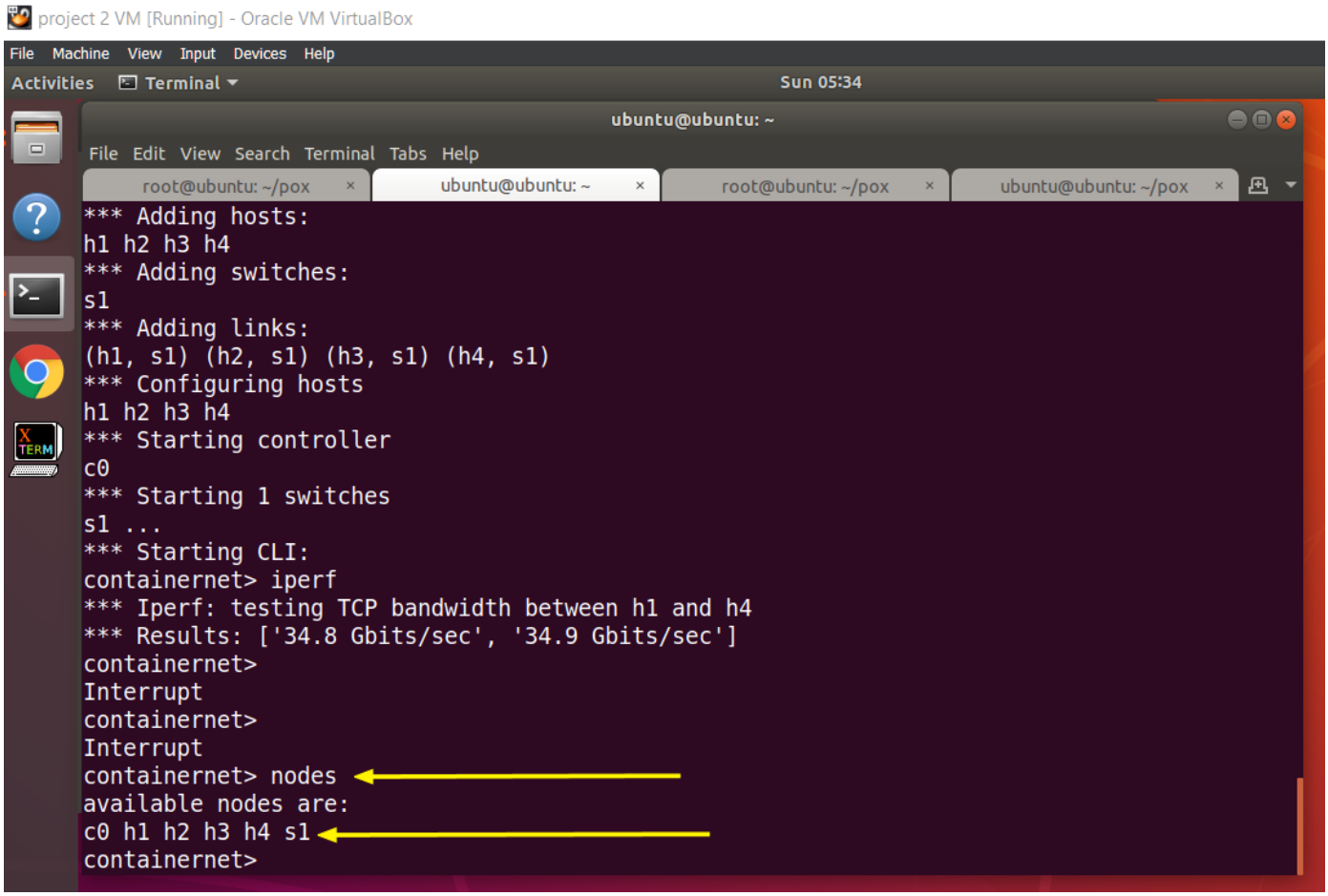
net links of linear topology with 5 switches and 1 host for each switch



measure bandwidth of single topology with 1 switch and 4 host

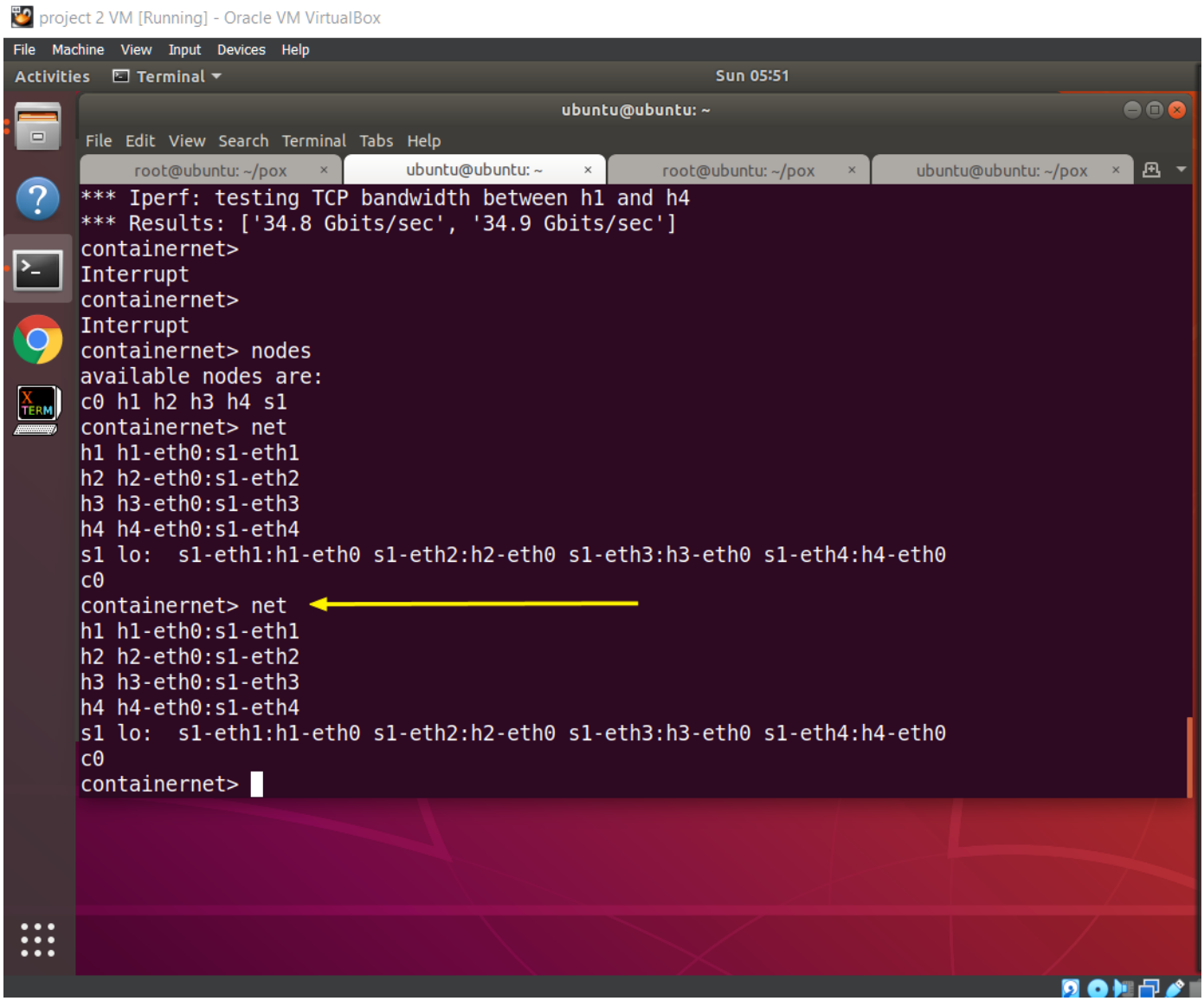


nodes of single topology with 1 switch and 4 host



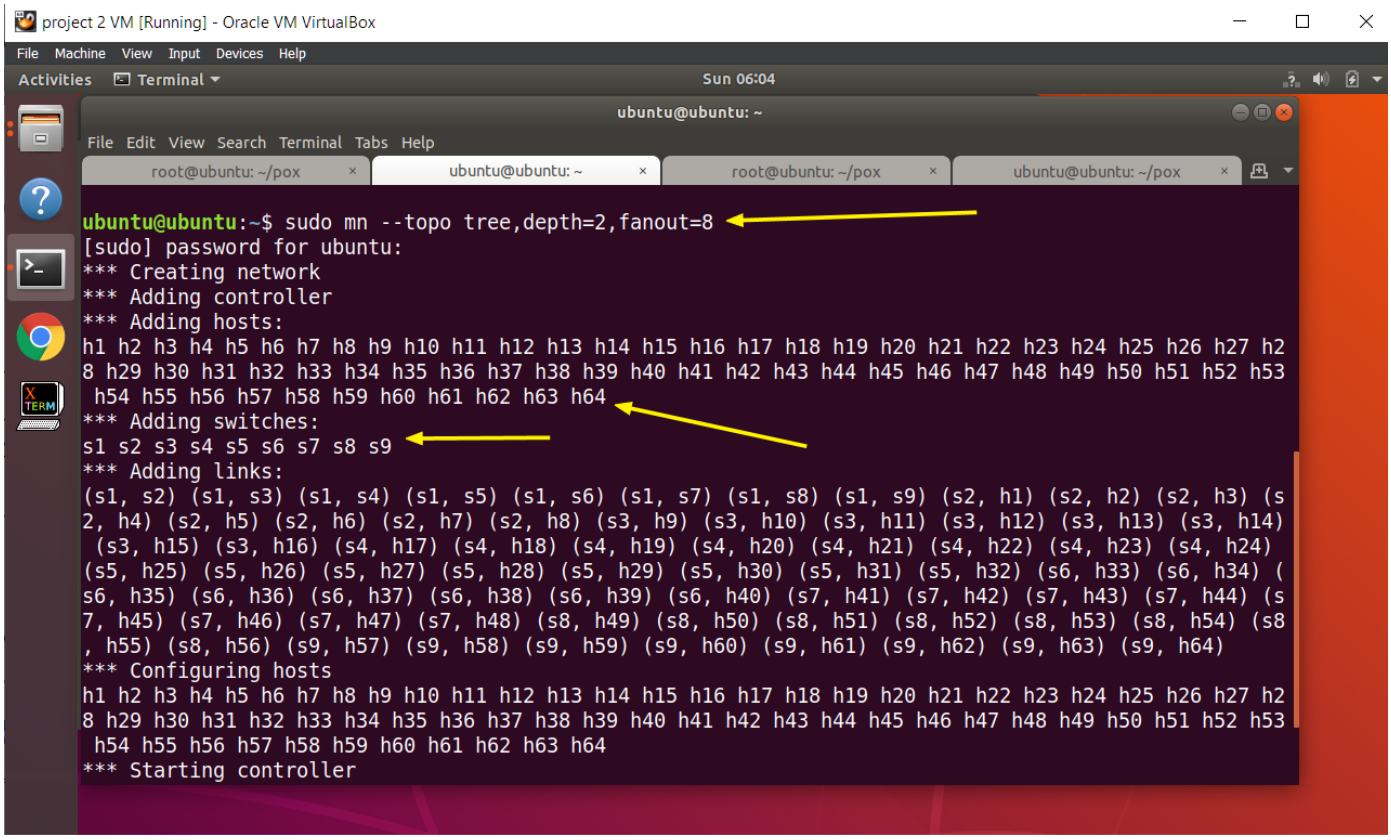
net links of single topology with 1 switch and 4 host

Please edit the highlighted portion.

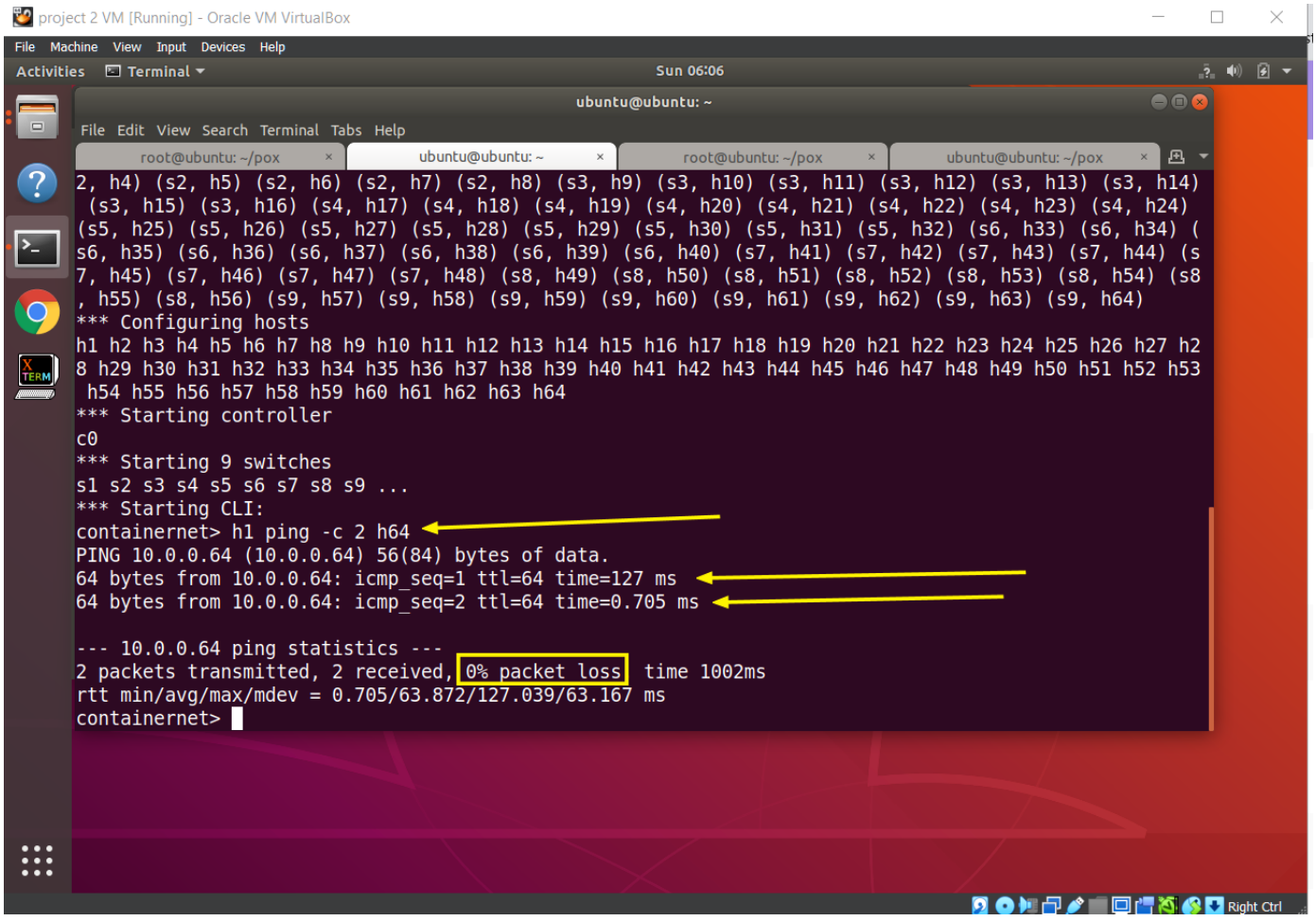


create a network topology with depth 2 and 8 fanout

Please edit the highlighted portion.

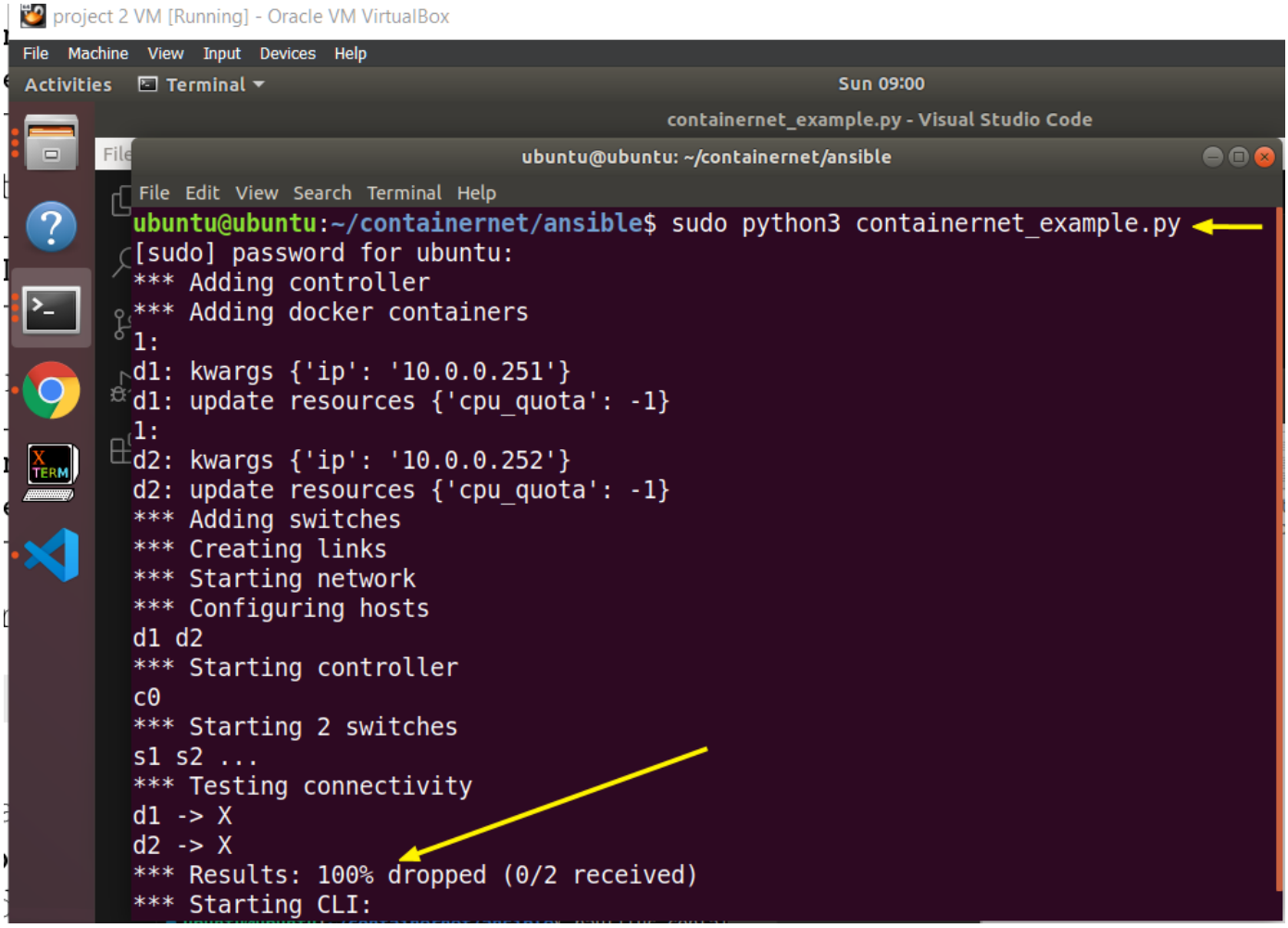


host 1 can ping host 64 successfully

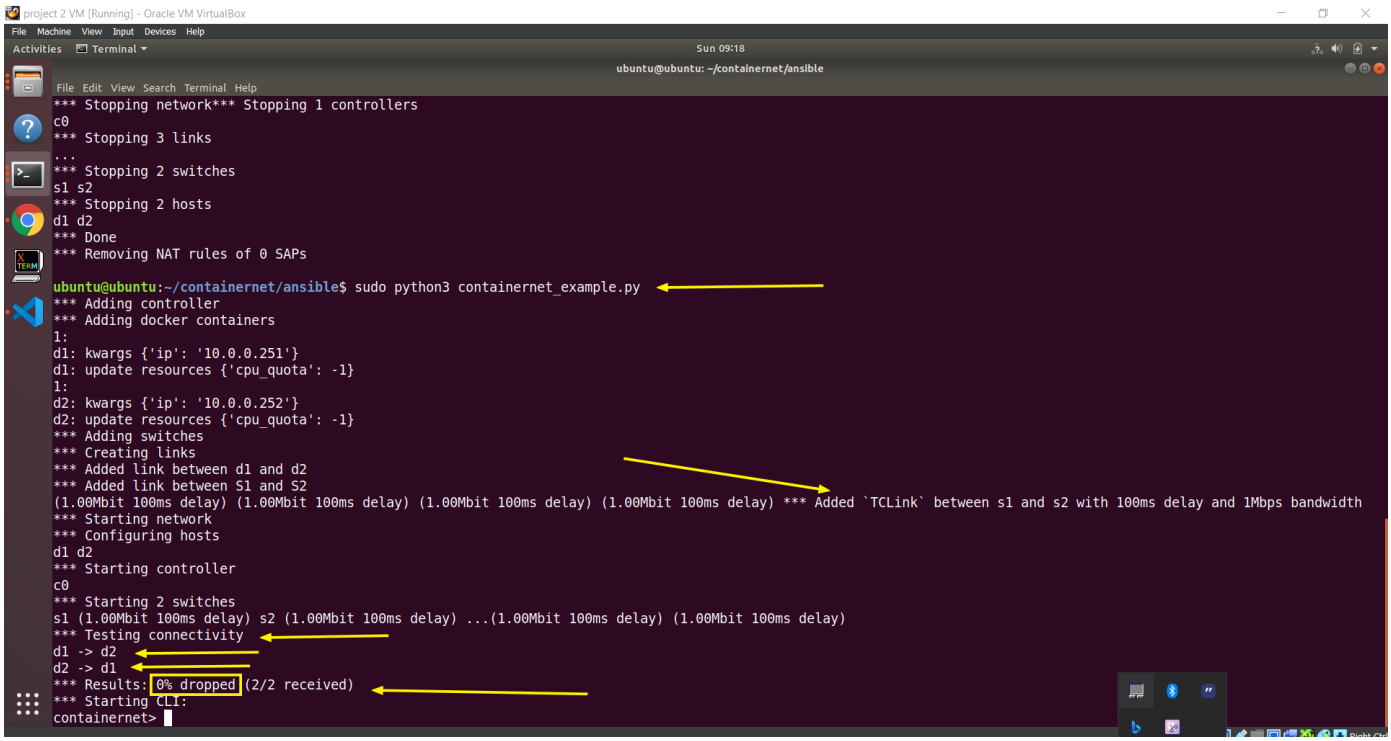


Containernet lab

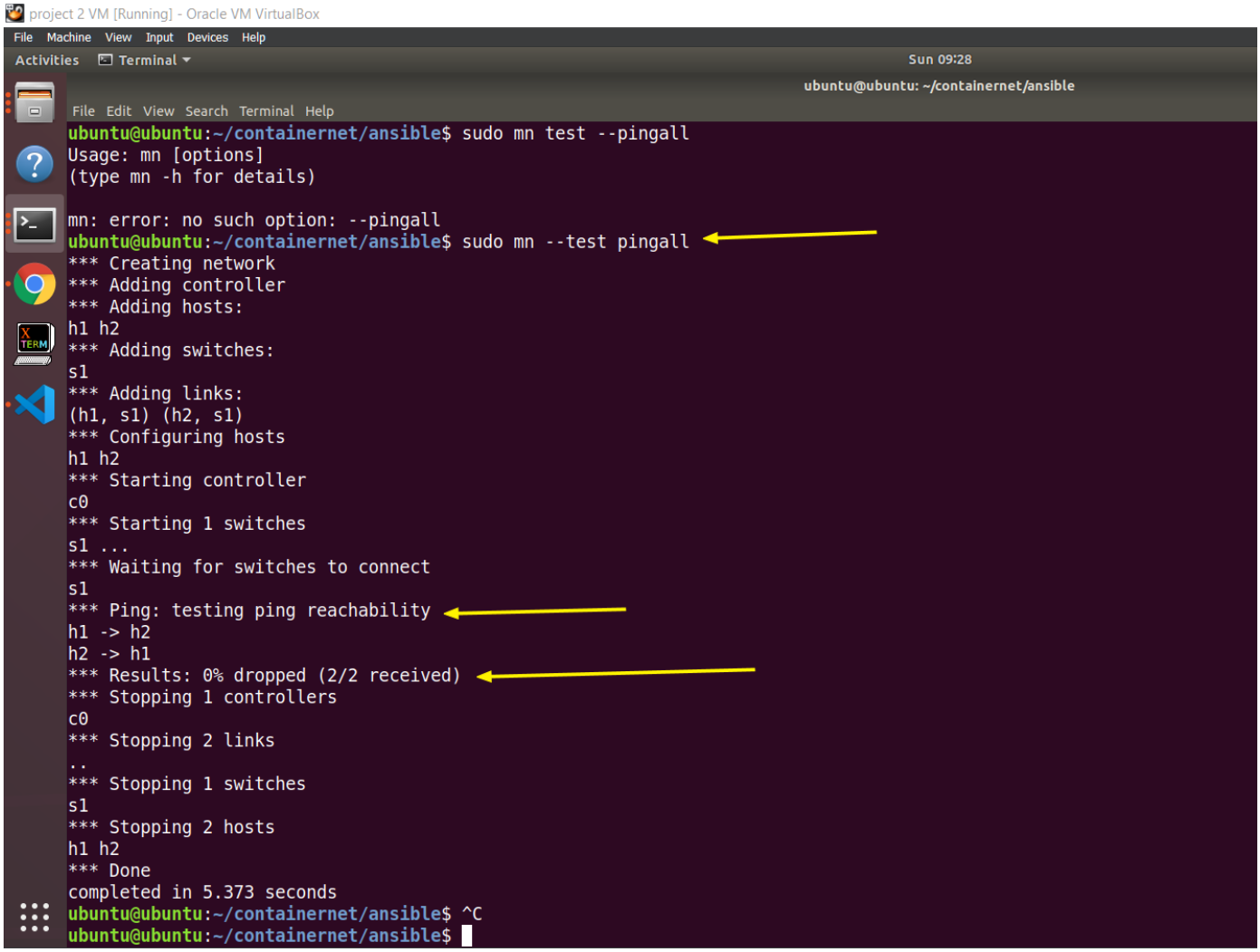
d1 (host 1 cannot ping d2 (host 2) because there is no link between them



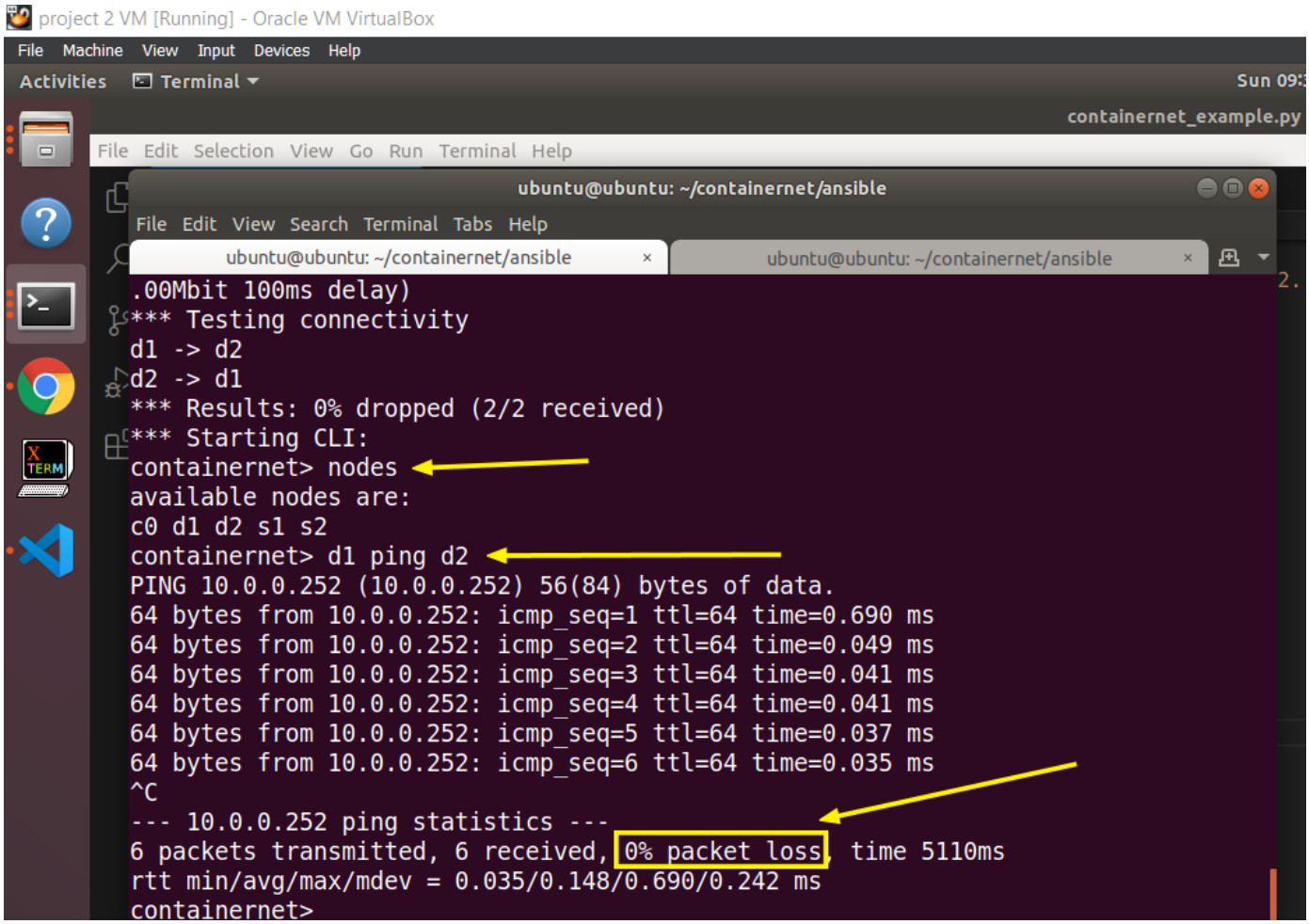
host 1 can ping host 2) after adding link between the 2 switches and set the bandwidth and TCLink between the 2 switches



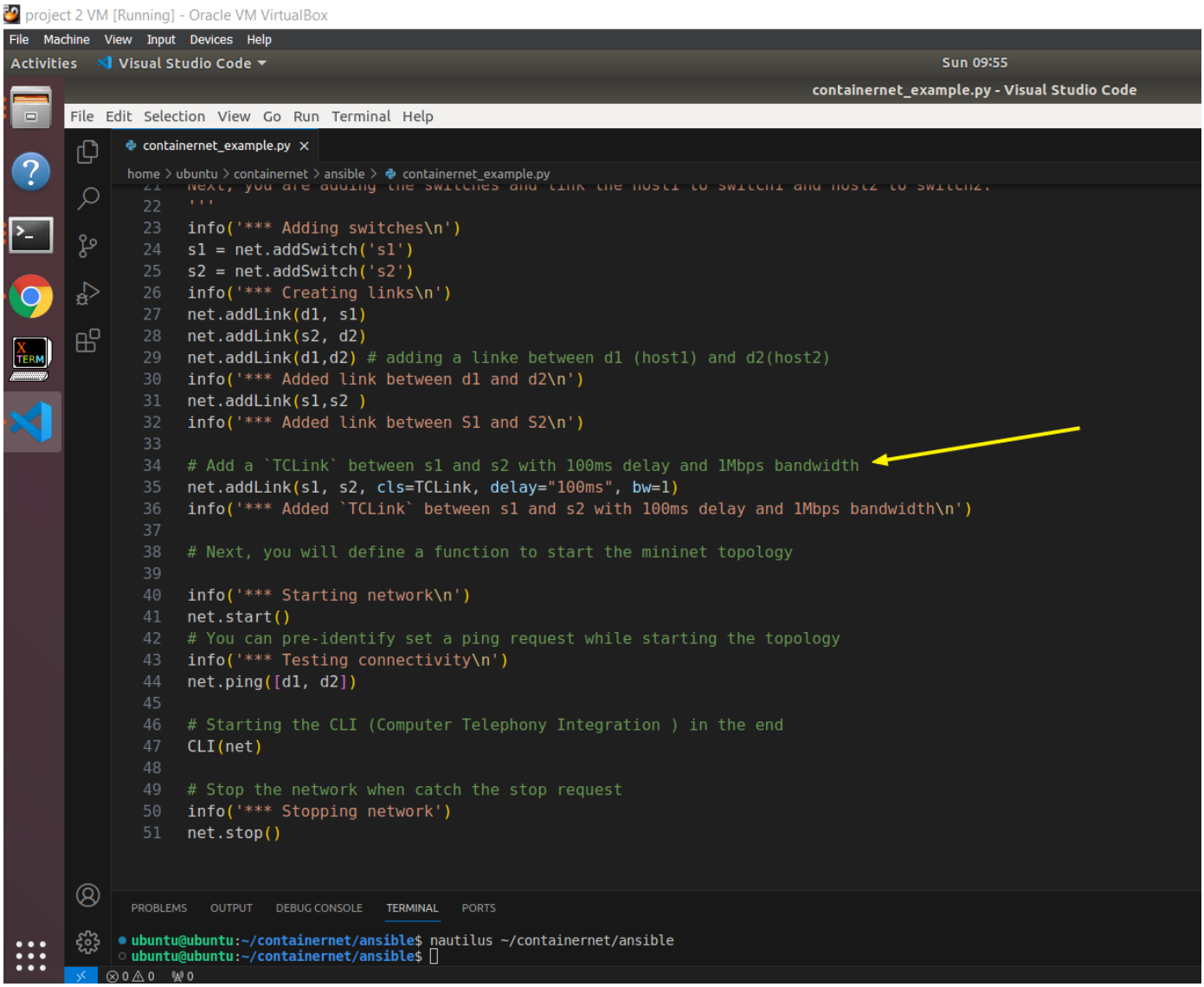
run sudo mn test –pingall



d1 ping successfully d2



added the TClink betweens1 and s2 python code



Main lab project 2

for more information , please watch my video that demonstrate the demo

https://www.youtube.com/watch?v=C9x4DCE4aEE

setup Mininet network and adding the links between cotinainers and check the links between nodes

project 2 VM [Running] - Oracle VM VirtualBox

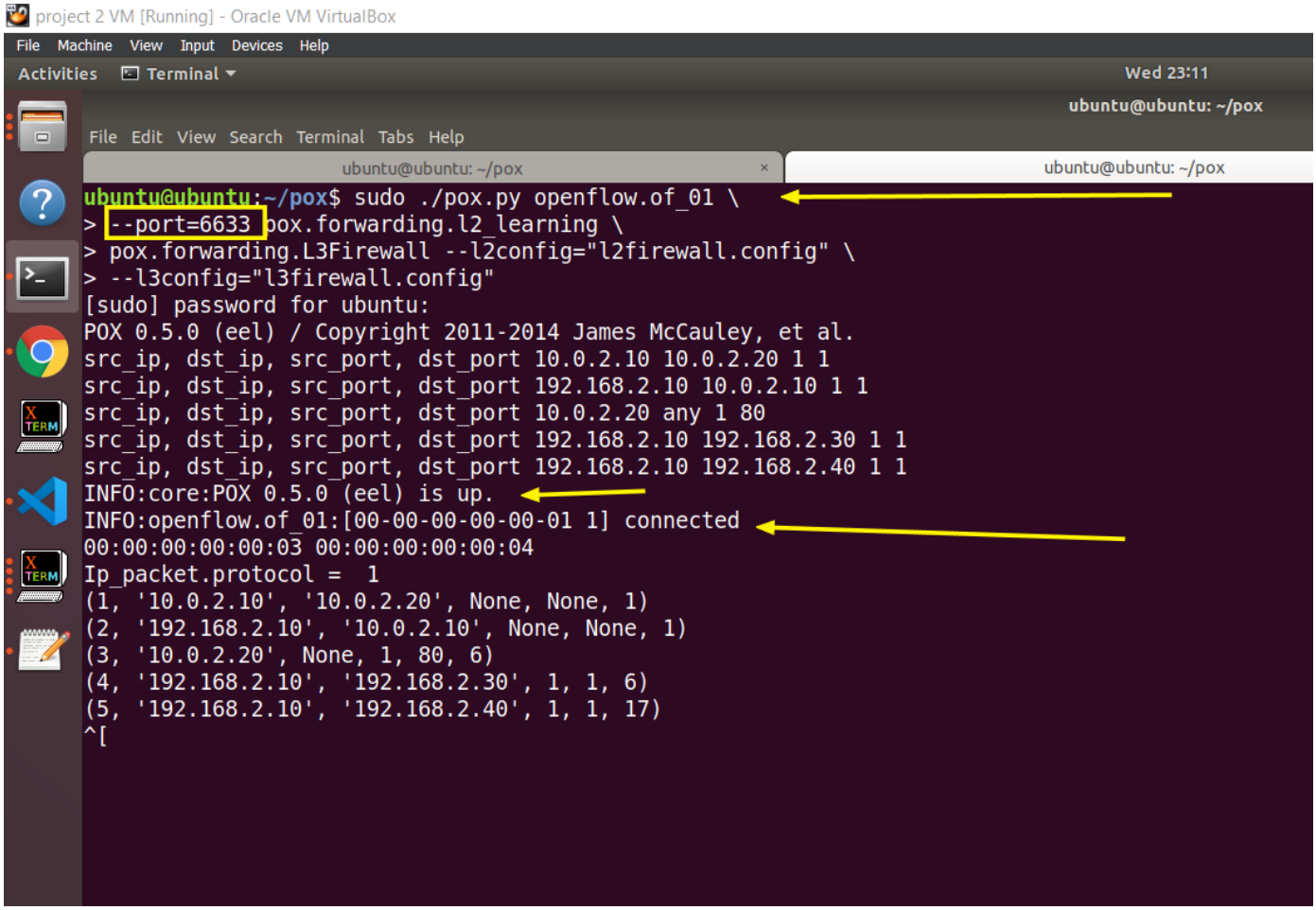
```
File Machine View Input Devices Help
Activities  Terminal                                   Wed 23:08
                                          ubuntu@ubuntu: ~/containernet/ansible
File Edit View Search Terminal Tabs Help
ubuntu@ubuntu: ~/pox                    ubuntu@ubuntu: ~/pox

ubuntu@ubuntu:~/containernet/ansible$ sudo python3 mininet_2switches_2controller_4hosts.py
*** Adding controllers
Unable to contact the remote controller at 127.0.0.1:6633
*** Adding switches
*** Adding hosts
*** Creating links
*** Added link between c1 and s1
*** Added link between c2 and s2
*** Starting network
*** Configuring hosts
h1 h2 h3 h4
*** Running CLI
*** Starting CLI:
containernet> py net.addLink(c1,s1)
<mininet.link.Link object at 0x7f9a4b16d470>
containernet> py net.addLink(c2,s2)
<mininet.link.Link object at 0x7f9a4b16d668>
containernet> h1 ifconfig h1-eth0 10.0.2.10
containernet> h2 ifconfig h2-eth0 10.0.2.20
containernet> h3 ifconfig h3-eth0 192.168.2.30
containernet> h4 ifconfig h4-eth0 192.168.2.40
containernet> h1 ip addr add 192.168.2.10/24 dev h1-eth1
containernet> net
h1 h1-eth0:s1-eth1 h1-eth1:s2-eth3
h2 h2-eth0:s1-eth2
h3 h3-eth0:s2-eth1
h4 h4-eth0:s2-eth2
s1 lo:  s1-eth1:h1-eth0 s1-eth2:h2-eth0 s1-eth3:c1-eth0
s2 lo:  s2-eth1:h3-eth0 s2-eth2:h4-eth0 s2-eth3:h1-eth1 s2-eth4:c2-eth0
c1 c1-eth0:s1-eth3
c2 c2-eth0:s2-eth4
containernet> links
s1-eth1<->h1-eth0 (OK OK)
s1-eth2<->h2-eth0 (OK OK)
s2-eth1<->h3-eth0 (OK OK)
s2-eth2<->h4-eth0 (OK OK)
```

assign IP address for host 1 and check it

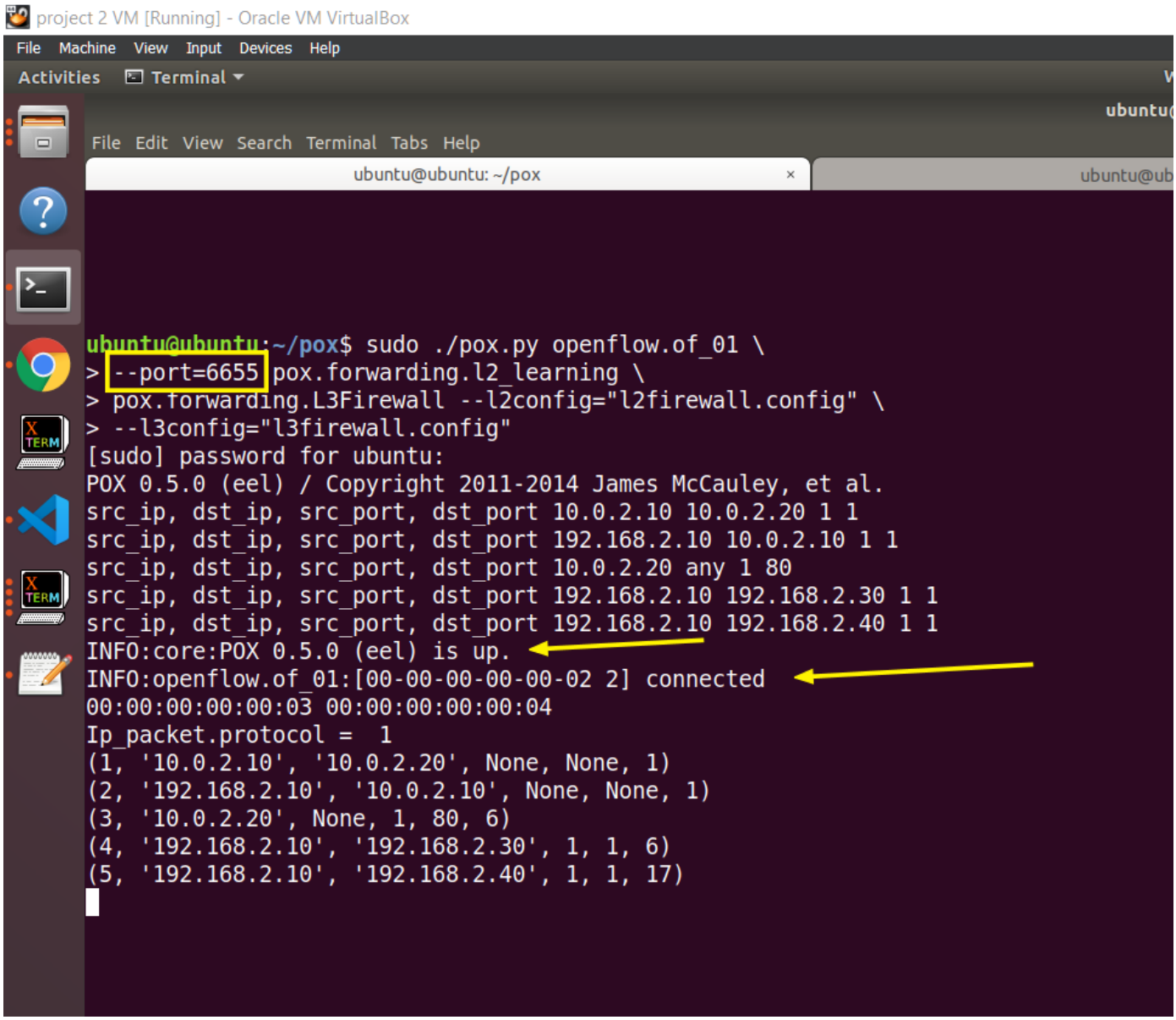
project 2 VM [Running] - Oracle VM VirtualBox

```
File Machine View Input Devices Help
Activities  Terminal                                   Wed 23:10
                                          ubuntu@ubuntu: ~/containernet/ansible
File Edit View Search Terminal Tabs Help
ubuntu@ubuntu: ~/pox    ubuntu@ubuntu: ~/pox    ubuntu@ubuntu: ~/containernet/ansible

^C^C^C
Interrupt
stopping h1
containernet> exterm h1 h2 h3 h4
*** Unknown command: exterm h1 h2 h3 h4
containernet> xterm h1 h2 h3 h4
containernet> py h1.IP
<bound method Node.IP of <Host h1: h1-eth0:10.0.0.1,h1-eth1:None pid=2746> >
containernet>
containernet> h1 ifconfig h1-eth1 192.168.2.10
containernet> h1 ifconfig
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.10  netmask 255.0.0.0  broadcast 10.255.255.255
        ether 62:c1:52:c3:de:32  txqueuelen 1000  (Ethernet)
        RX packets 67  bytes 9484 (9.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 31  bytes 2714 (2.7 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

h1-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.2.10  netmask 255.255.255.0  broadcast 0.0.0.0
        inet6 fe80::1cae:c2ff:fe66:5f3b  prefixlen 64  scopeid 0x20<link>
        ether 1e:ae:c2:66:5f:3b  txqueuelen 1000  (Ethernet)
        RX packets 79  bytes 10476 (10.4 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 25  bytes 1930 (1.9 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 15  bytes 1288 (1.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 15  bytes 1288 (1.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

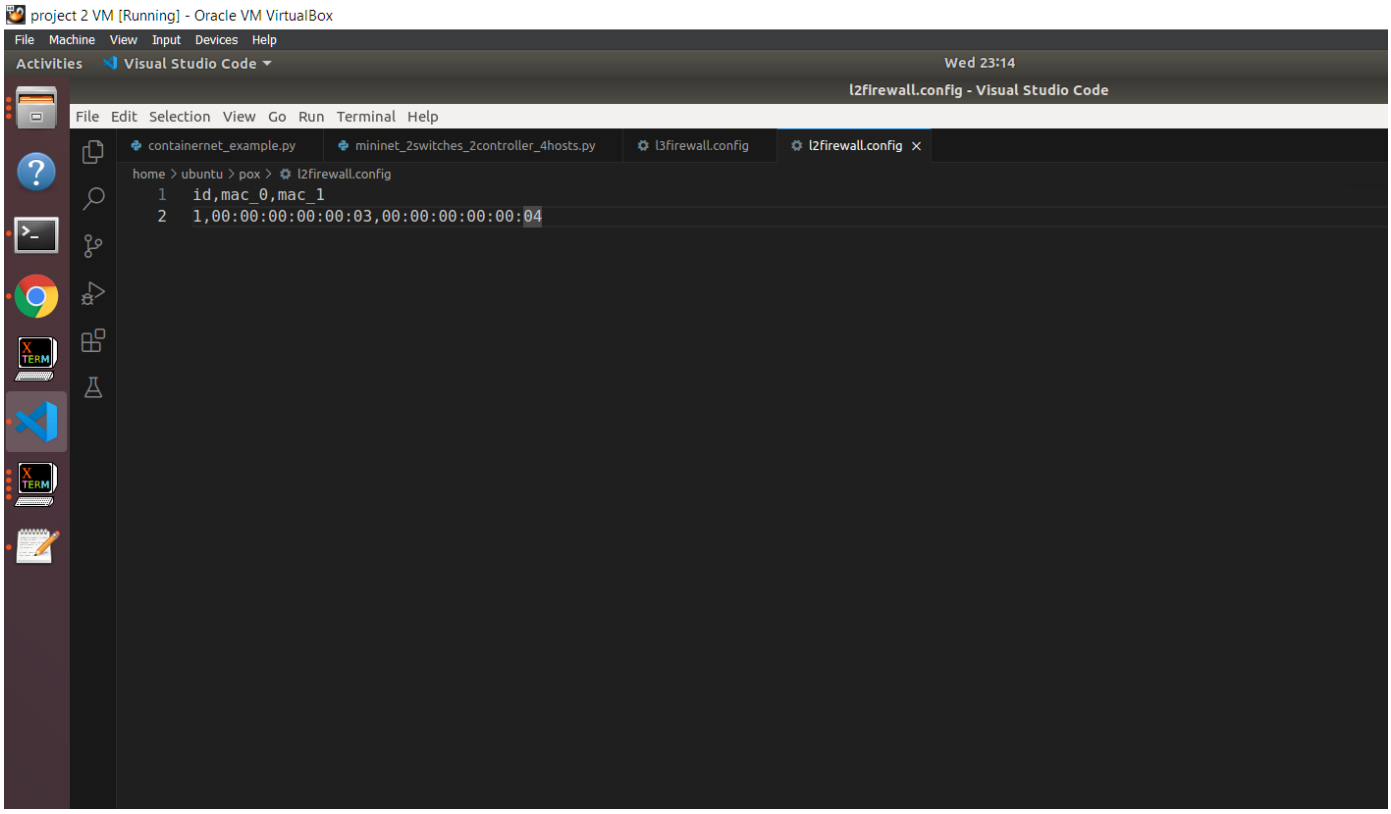
controller 1 port 6633 is up and running



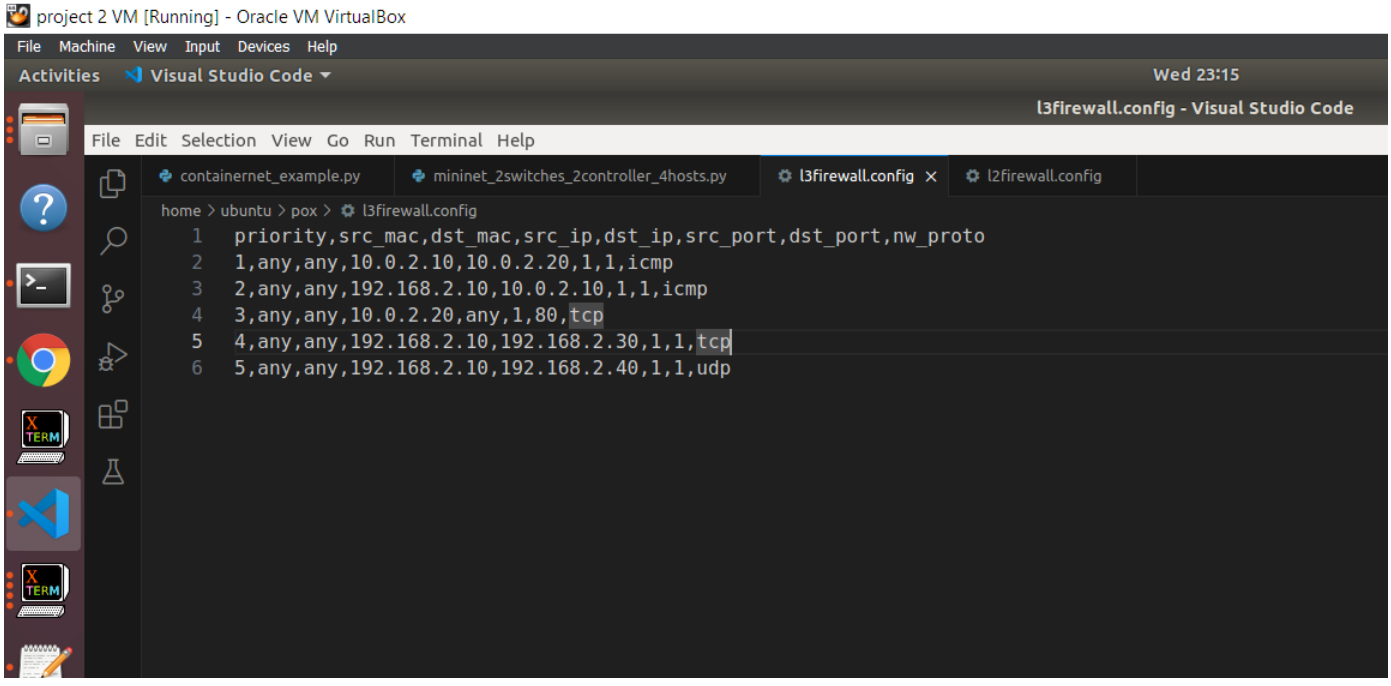
controller 2 port 6655 is up and running



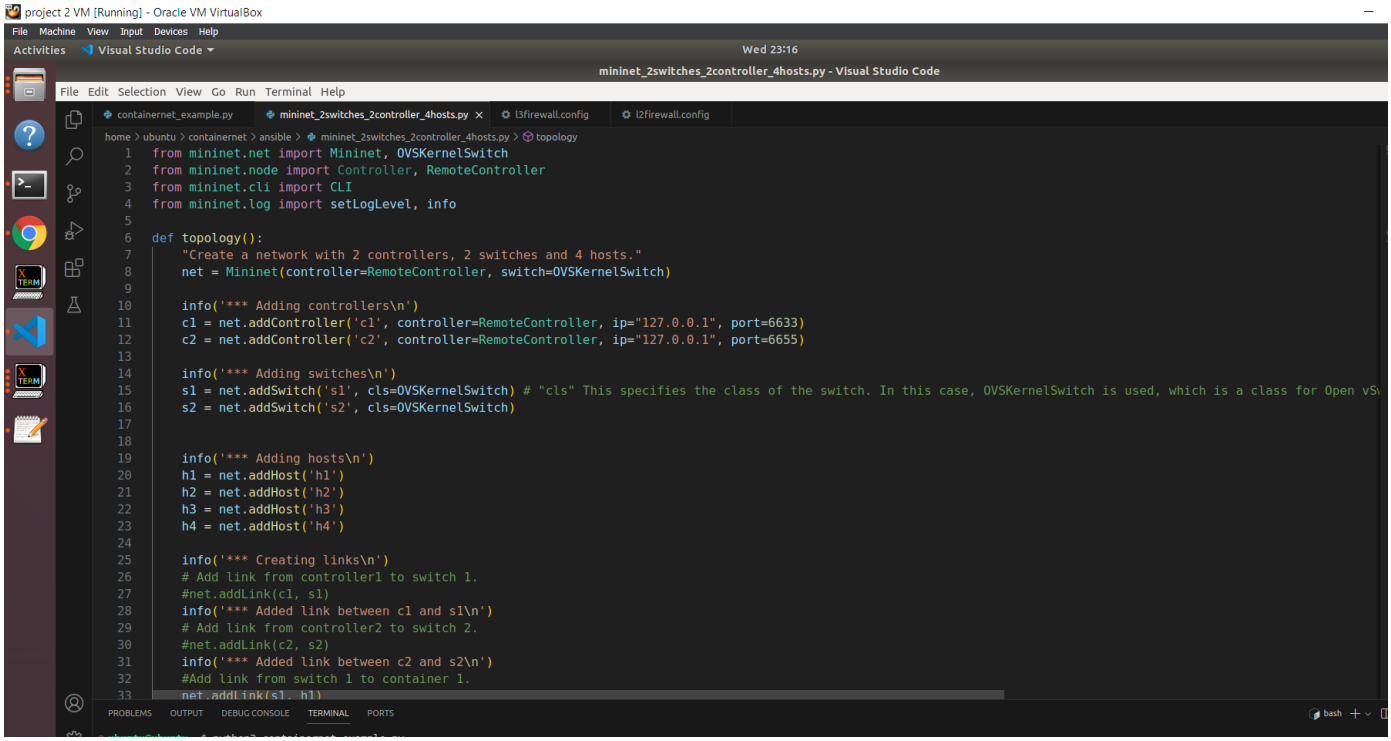
L2firewall.config file



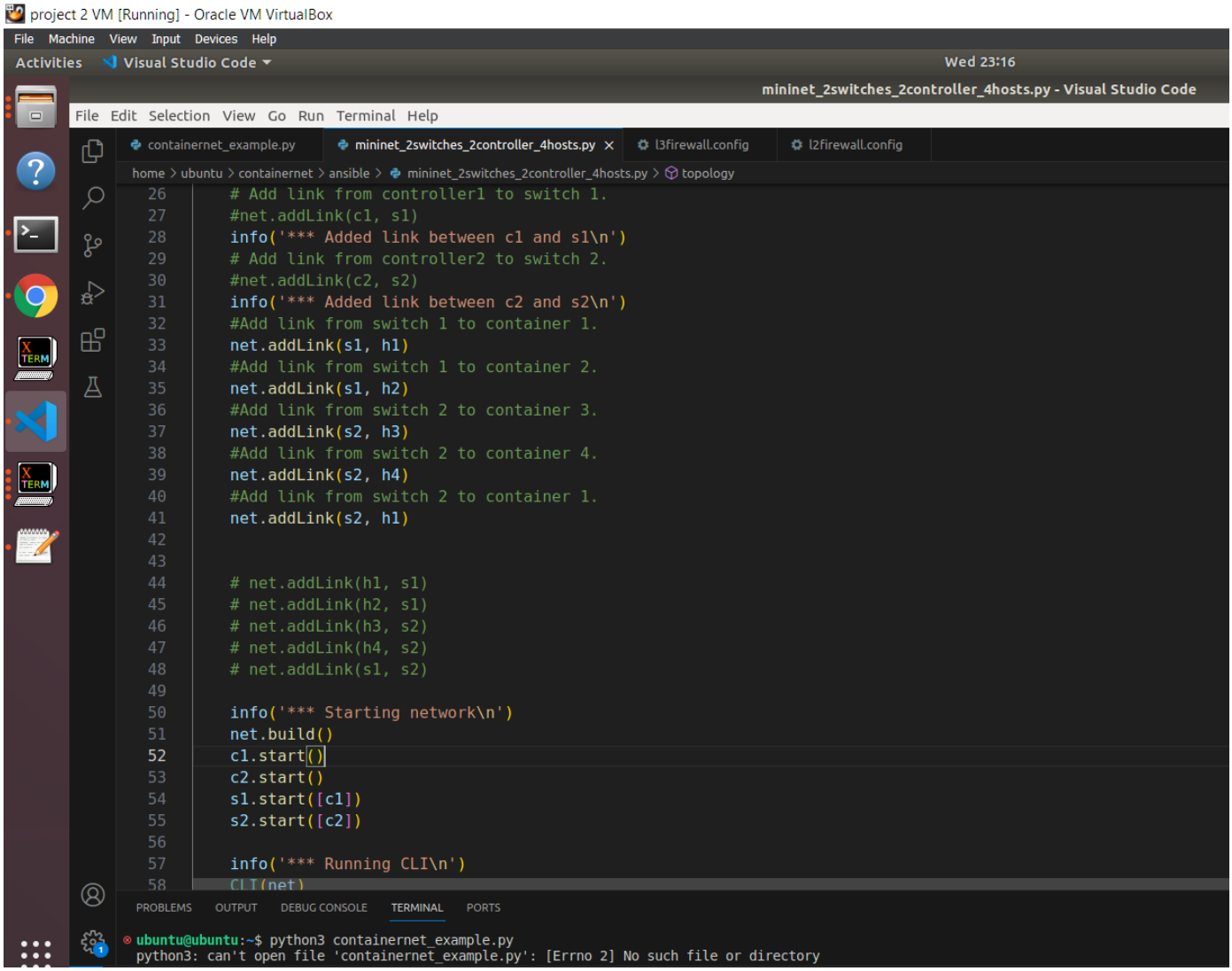
L3firewall.config file



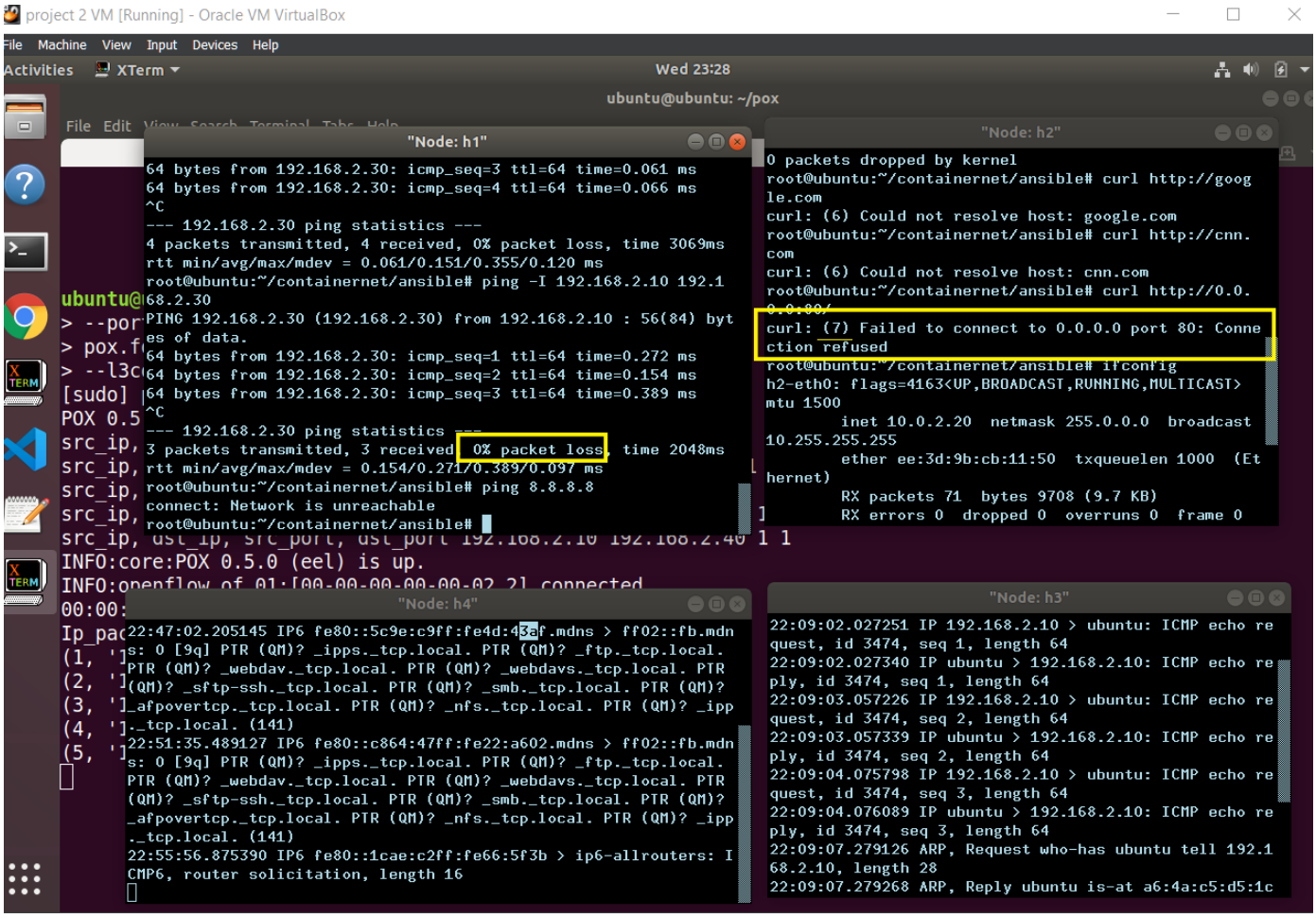
Python file for creating mininet network



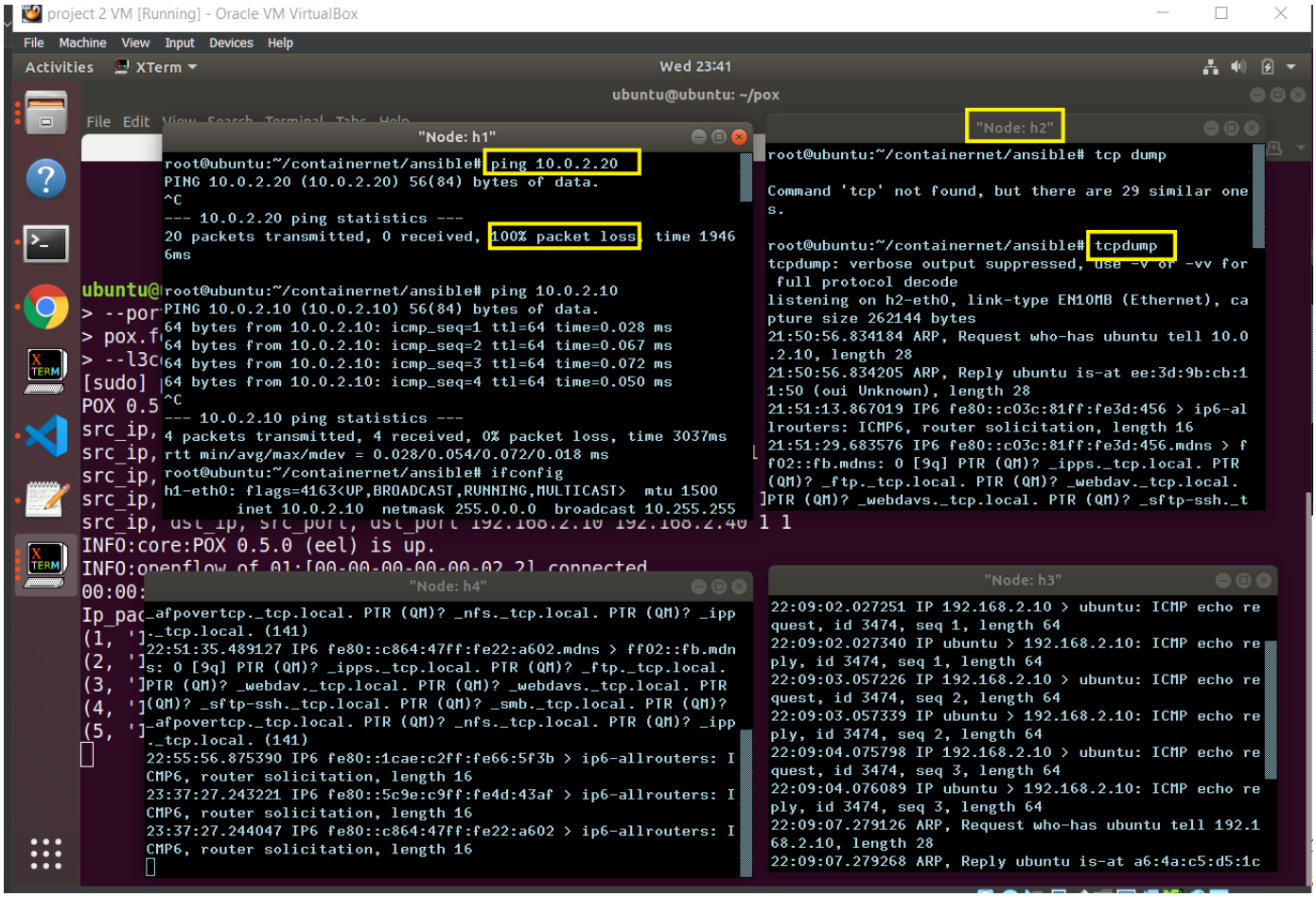
Please edit the highlighted portion.



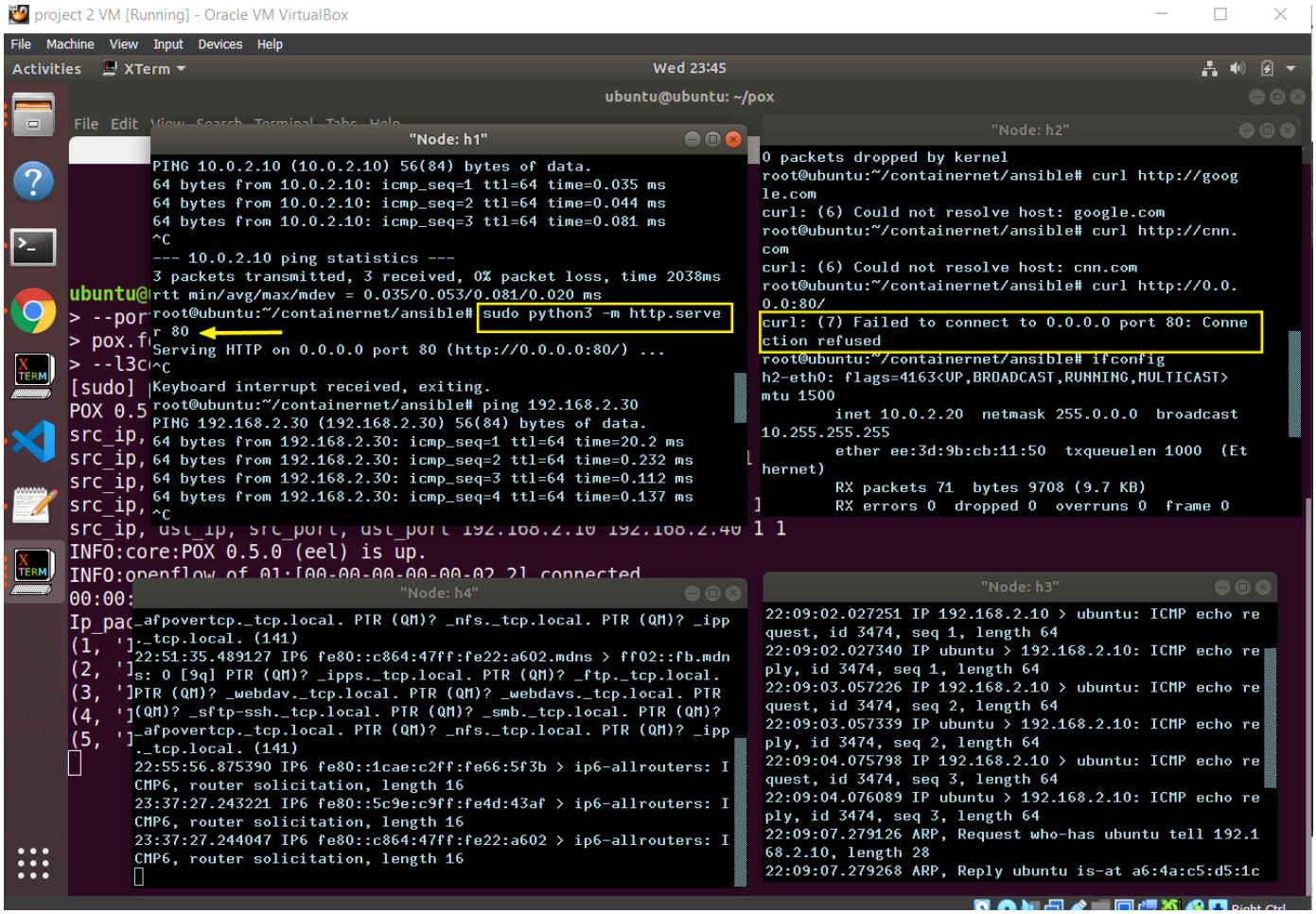
Xterm terminals of 4 hosts



Ping from h1 to h2 (it cannot ping)



h2 cannot send connection on port 80 (HTTP) as made a python server on h1



for more information , please watch my video that demonstrate the demo

https://www.youtube.com/watch?v=C9x4DCE4aEE

V. Conclusion

I have learned may amazing practical things for SDN concepts and I applied them using real tools such as Mininet and POX controller and python programming language, also it was really nice to use containers as hosts and simulate a complete virtual network in a linux environment

VI. Appendix B: Attached files

Find the files on my github

https://github.com/Mohammed-Ragab/CSE-548-advanced-network-security/tree/main/project%202%20files

VII. REFERENCES

- Mininet VM Setup Notes - Mininet