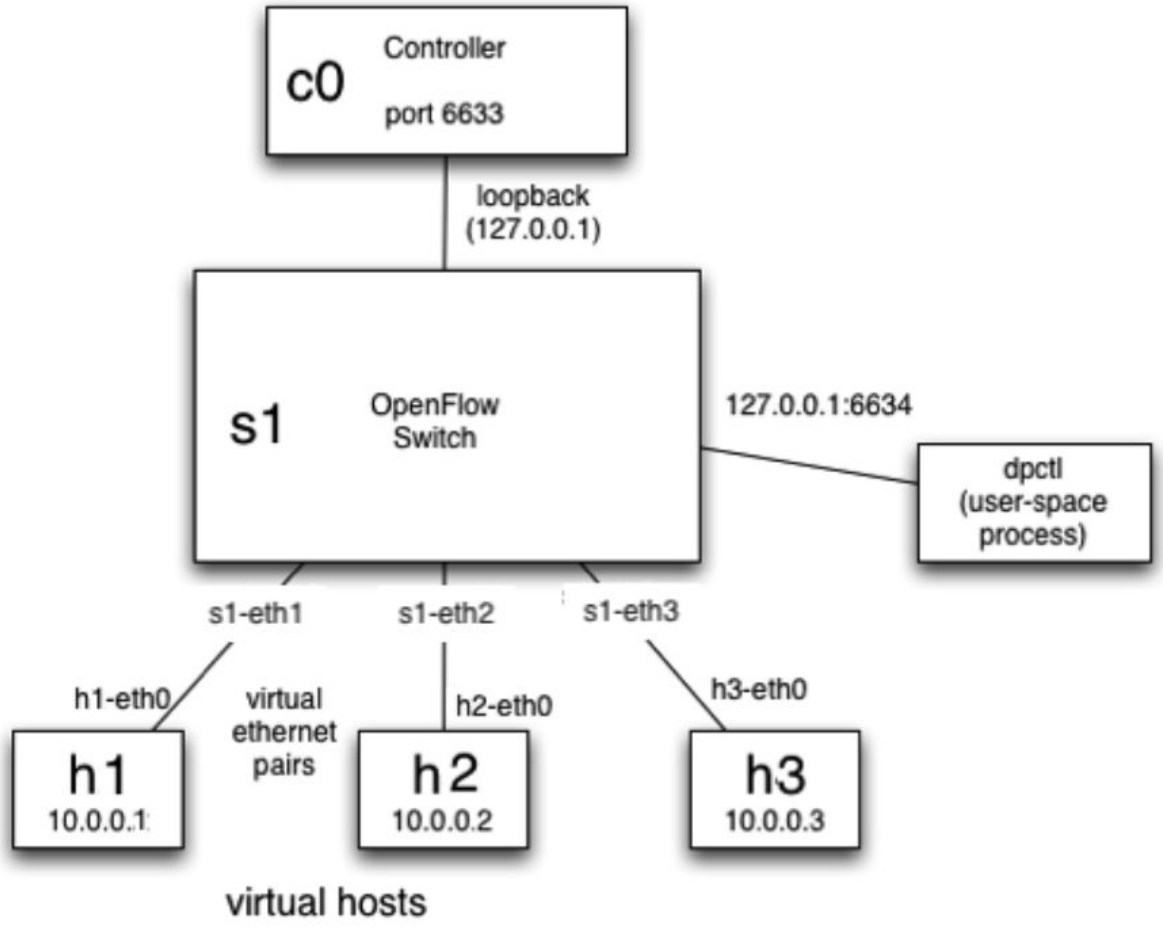
Computer Network Design - Lab 3

Network topology

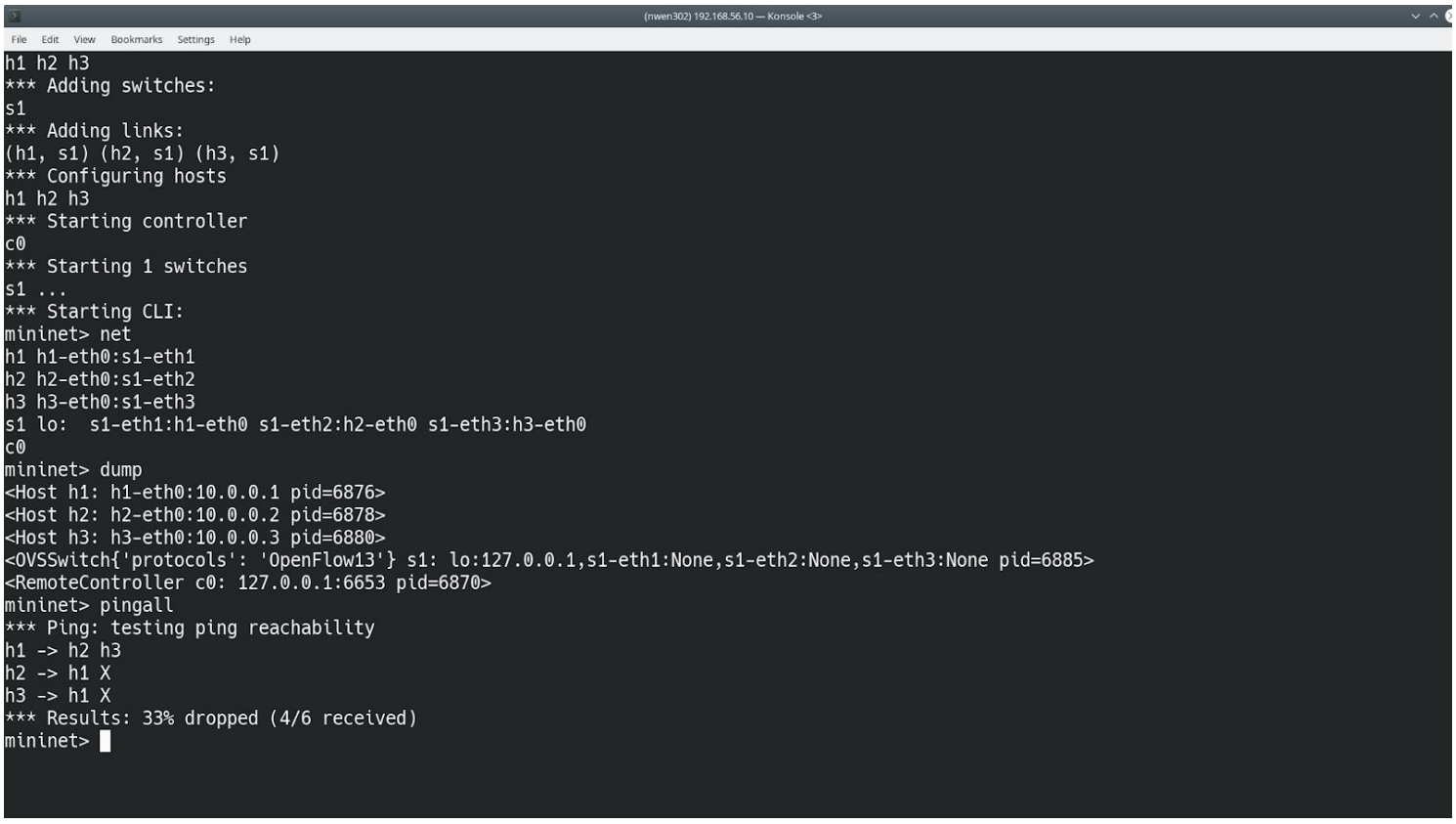
The topology has three hosts named h1, h2 and h3 respectively. Each host has an Ethernet interface called h1-eth0, h2-eth0 and h3-eth0 respectively. The three hosts are connected through a switch named s1.
The switch s1 has three ports named s1-eth1, s1-eth2 and s1-eth3. The controller is connected on the loopback interface (in real life this may or may not be the case, it means the switch and controller are built in a single box).
The controller is identified as c0 and connected through port 6653.



Key Task 1:

To block traffic between host 2 and host 3. I added the following code to the Simple_Switch_13.py:

```
------------------------------------------------------------
actions = []
match = parser.OFPMatch(eth_src = "00:00:00:00:00:02" ,
eth_dst = "00:00:00:00:00:03")
self.add_flow(datapath, 1, match, actions)
match = parser.OFPMatch(eth_src = "00:00:00:00:00:03" ,
eth_dst = "00:00:00:00:00:02")
self.add_flow(datapath, 1, match, actions)
------------------------------------------------------------
```



Then ,I run the ryu-manager as shown in the figure below just to see what is actually happening.

```
gui_topology          rest_router.py       simple_switch_13.pyc    simple_switch.py          ws_topology.py
__init__.py           rest_topology.py     simple_switch_14.py     simple_switch_rest_13.py
ofctl                 rest_vtep.py         simple_switch_15.py     simple_switch_snort.py
nwen302@nwen302-core:~/lab3/ryu/app$ which ryu-manager
/home/nwen302/.local/bin/ryu-manager
nwen302@nwen302-core:~/lab3/ryu/app$ sudo /home/nwen302/.local/bin/ryu-manager
[sudo] password for nwen302:
loading app ryu.controller.ofp_handler
instantiating app ryu.controller.ofp_handler of OFPHandler
^Z
[1]+  Stopped                 sudo /home/nwen302/.local/bin/ryu-manager
nwen302@nwen302-core:~/lab3/ryu/app$ sudo /home/nwen302/.local/bin/ryu-manager simple_switch_13.py
loading app simple_switch_13.py
loading app ryu.controller.ofp_handler
instantiating app simple_switch_13.py of SimpleSwitch13
instantiating app ryu.controller.ofp_handler of OFPHandler
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
packet in 1 b2:bb:10:38:f1:de 33:33:00:00:00:02 2
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
packet in 1 2a:80:81:b2:ba:5b 33:33:00:00:00:02 3
packet in 1 16:46:09:51:45:6c 33:33:00:00:00:02 1
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
```

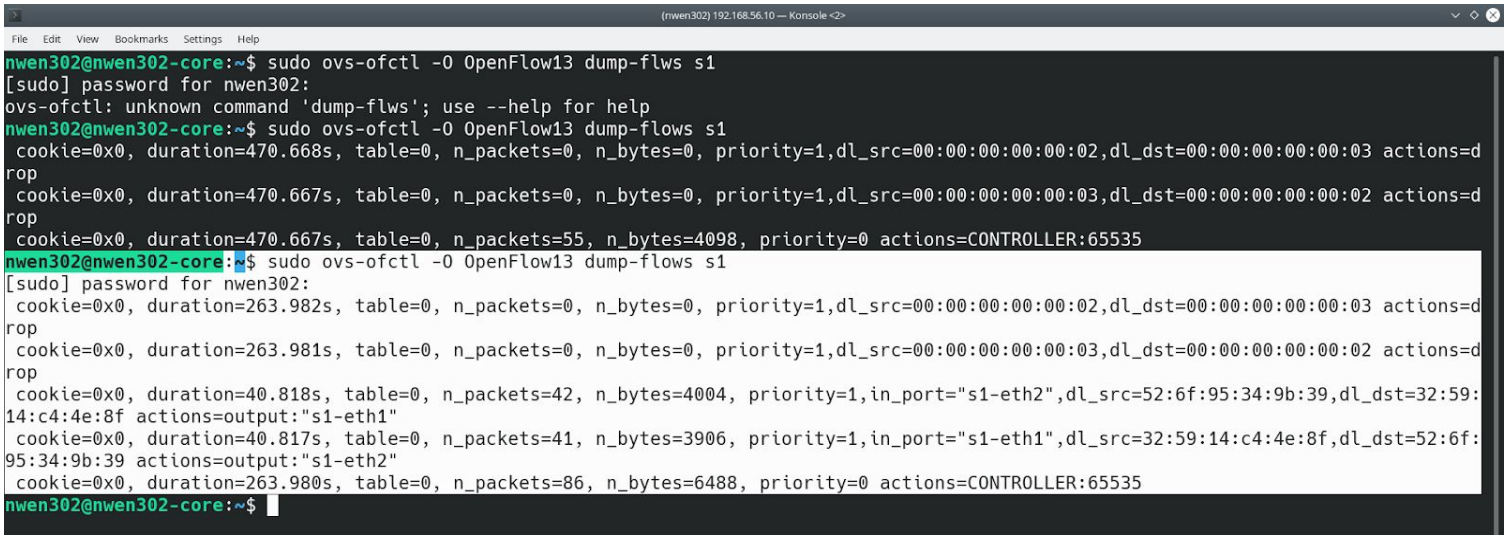
Next, I checked the flows using this command below:

```
% sudo ovs-ofctl -O OpenFlow13 dump-flws s1
```

```
nwen302@nwen302-core:~$ sudo ovs-ofctl -O OpenFlow13 dump-flws s1
[sudo] password for nwen302:
ovs-ofctl: unknown command 'dump-flws'; use --help for help
nwen302@nwen302-core:~$ sudo ovs-ofctl -O OpenFlow13 dump-flows s1
 cookie=0x0, duration=470.668s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:02,dl_dst=00:00:00:00:00:03 actions=d
rop
 cookie=0x0, duration=470.667s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:03,dl_dst=00:00:00:00:00:02 actions=d
rop
 cookie=0x0, duration=470.667s, table=0, n_packets=55, n_bytes=4098, priority=0 actions=CONTROLLER:65535
nwen302@nwen302-core:~$
```

From the source (dl_src=00:00:00:00:00:02) we have a packet of ‘0’(zero).
From the source (dl_src=00:00:00:00:00:03) we have a packet of ‘0’(zero)as well.

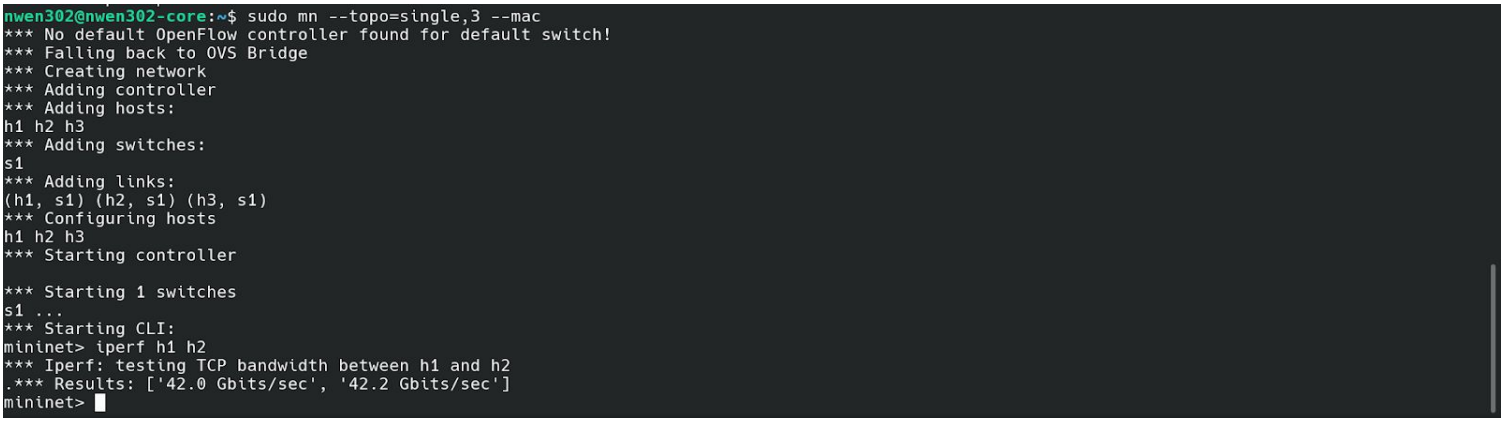
To have a clear idea. I start a ping from h1 to h2. The number of packets will keep increasing because I used a continuous ping.
From the source (dl_src=52:6f:95:34:9b:39) we have a packet of 42.
From the source (dl_src=32:59:14:c4:4e:8f) we have a packet of 41.



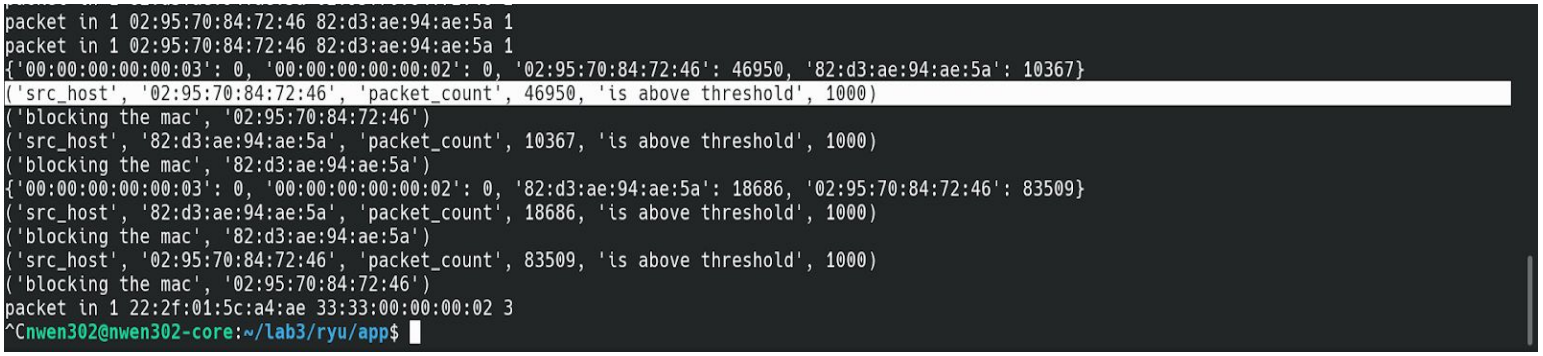
If I start a ping from h1 to h3 I start a ping from h1 to h2. The number of packets will get aggregated. The number of packets will keep increasing(added).

Now, let us generate a lot of packets to see what is happening here. To do that I used the command below in Mininet:

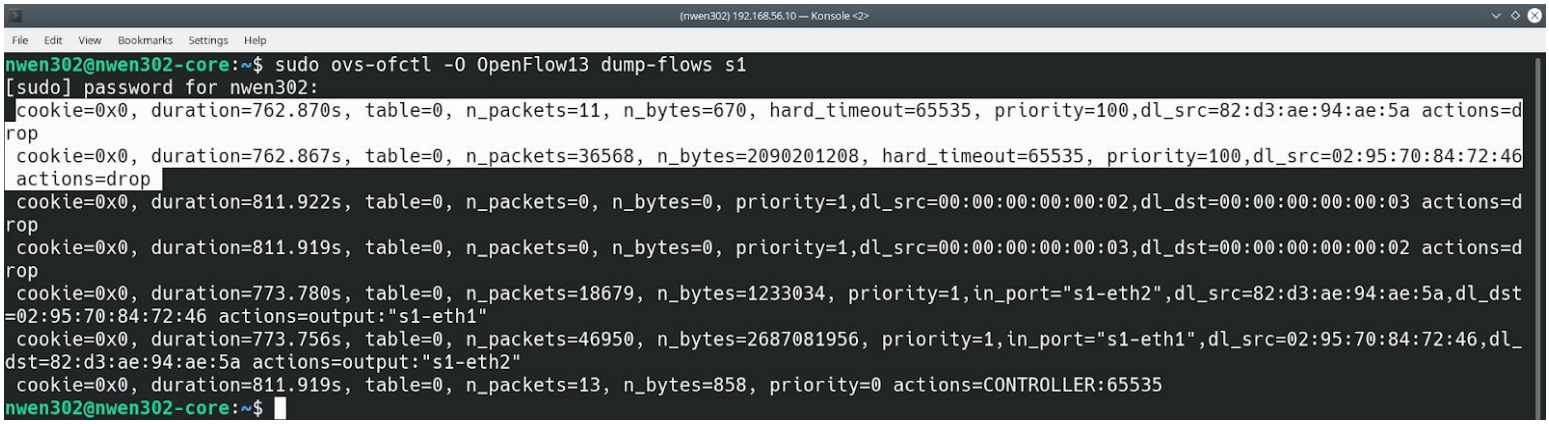
```
----------------------
mininet> iperf h1 h2
----------------------
```



If we look closely at the highlighted line, it shows that ‘packet_count’ number goes above the threshold(1000).



Now, let us check the Flows again. We need to look at these two arguments, hard_timeout=65535 and priority=100. These two arguments help to block the sources only. I set the priority = 100 to block the source. If I set the priority = 1 we will not be able to see which flow gets executed. In task 3, we need to block all the traffic originating from this host for 24 hours. 24 hours will be equal to 86400 seconds, but we can not block the traffic for one day(24h). Therefore, I had to set the hard_timeout to the maximum. For the hard_timeout=65535, the ‘65535’ basically represents the number of seconds, The hard and idle timeout range is from 0 through 65535. Traffic will be blocked for 65535 sec, after that, the flow will be removed automatically. In other words, the flow will be persistent during the hard_timeout=65535 and exactly after 65535 sec the flow will be removed.



All the Tasks Have been achieved. However, task 3 is a bit difficult.

Task 1:

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3
h2 -> h1 X
h3 -> h1 X
*** Results: 33% dropped (4/6 received)
mininet>
```

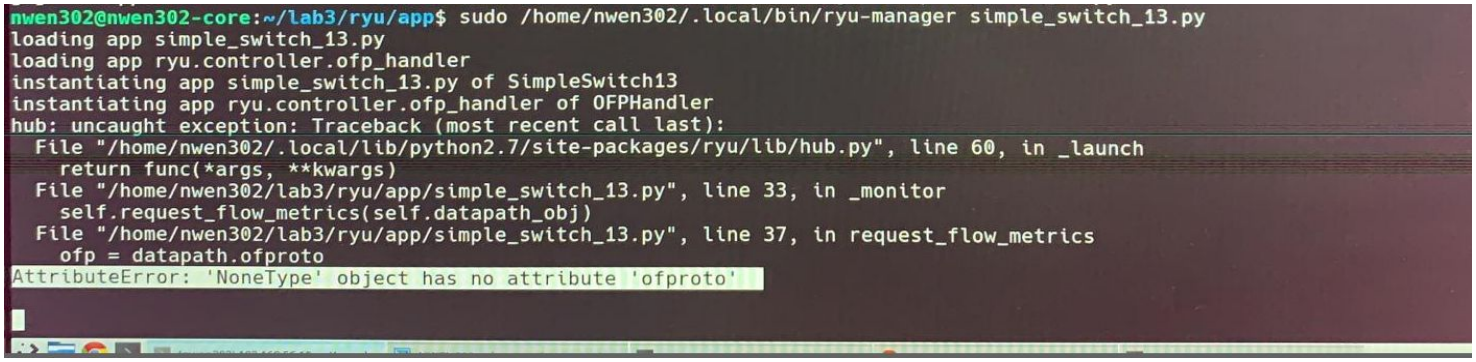
Task 2:

```
nwen302@nwen302-core:~/lab3/ryu/app$ sudo /home/nwen302/.local/bin/ryu-manager simple_switch_13.py
loading app simple_switch_13.py
loading app ryu.controller.ofp_handler
instantiating app simple_switch_13.py of SimpleSwitch13
instantiating app ryu.controller.ofp_handler of OFPHandler
packet in 1 02:95:70:84:72:46 33:33:00:00:00:02 1
packet in 1 22:2f:01:5c:a4:ae 33:33:00:00:00:02 3
packet in 1 82:d3:ae:94:ae:5a 33:33:00:00:00:02 2
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
packet in 1 02:95:70:84:72:46 33:33:00:00:00:02 1
packet in 1 22:2f:01:5c:a4:ae 33:33:00:00:00:02 3
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
packet in 1 82:d3:ae:94:ae:5a 33:33:00:00:00:02 2
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0}
packet in 1 02:95:70:84:72:46 ff:ff:ff:ff:ff:ff 1
packet in 1 82:d3:ae:94:ae:5a 02:95:70:84:72:46 2
packet in 1 02:95:70:84:72:46 82:d3:ae:94:ae:5a 1
packet in 1 02:95:70:84:72:46 82:d3:ae:94:ae:5a 1
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0, '02:95:70:84:72:46': 46950, '82:d3:ae:94:ae:5a': 10367}
('src_host', '02:95:70:84:72:46', 'packet_count', 46950, 'is above threshold', 1000)
('blocking the mac', '02:95:70:84:72:46')
('src_host', '82:d3:ae:94:ae:5a', 'packet_count', 10367, 'is above threshold', 1000)
('blocking the mac', '82:d3:ae:94:ae:5a')
{'00:00:00:00:00:03': 0, '00:00:00:00:00:02': 0, '82:d3:ae:94:ae:5a': 18686, '02:95:70:84:72:46': 83509}
('src_host', '82:d3:ae:94:ae:5a', 'packet_count', 18686, 'is above threshold', 1000)
('blocking the mac', '82:d3:ae:94:ae:5a')
('src_host', '02:95:70:84:72:46', 'packet_count', 83509, 'is above threshold', 1000)
('blocking the mac', '02:95:70:84:72:46')
packet in 1 22:2f:01:5c:a4:ae 33:33:00:00:00:02 3
^Cnwen302@nwen302-core:~/lab3/ryu/app$
```

Task 3:

```
(nwen302) 192.168.56.10 — Konsole <2>
File  Edit  View  Bookmarks  Settings  Help
nwen302@nwen302-core:~$ sudo ovs-ofctl -O OpenFlow13 dump-flows s1
[sudo] password for nwen302:
 cookie=0x0, duration=762.870s, table=0, n_packets=11, n_bytes=670, hard_timeout=65535, priority=100,dl_src=82:d3:ae:94:ae:5a actions=d
rop
 cookie=0x0, duration=762.867s, table=0, n_packets=36568, n_bytes=2090201208, hard_timeout=65535, priority=100,dl_src=02:95:70:84:72:46
 actions=drop
 cookie=0x0, duration=811.922s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:02,dl_dst=00:00:00:00:00:03 actions=d
rop
 cookie=0x0, duration=811.919s, table=0, n_packets=0, n_bytes=0, priority=1,dl_src=00:00:00:00:00:03,dl_dst=00:00:00:00:00:02 actions=d
rop
 cookie=0x0, duration=773.780s, table=0, n_packets=18679, n_bytes=1233034, priority=1,in_port="s1-eth2",dl_src=82:d3:ae:94:ae:5a,dl_dst
=02:95:70:84:72:46 actions=output:"s1-eth1"
 cookie=0x0, duration=773.756s, table=0, n_packets=46950, n_bytes=2687081956, priority=1,in_port="s1-eth1",dl_src=02:95:70:84:72:46,dl_
dst=82:d3:ae:94:ae:5a actions=output:"s1-eth2"
 cookie=0x0, duration=811.919s, table=0, n_packets=13, n_bytes=858, priority=0 actions=CONTROLLER:65535
nwen302@nwen302-core:~$
```

Things That I spend very long time to fix are so many but here are the toughest ones:
The first I faced I could not fix for while was this error below.

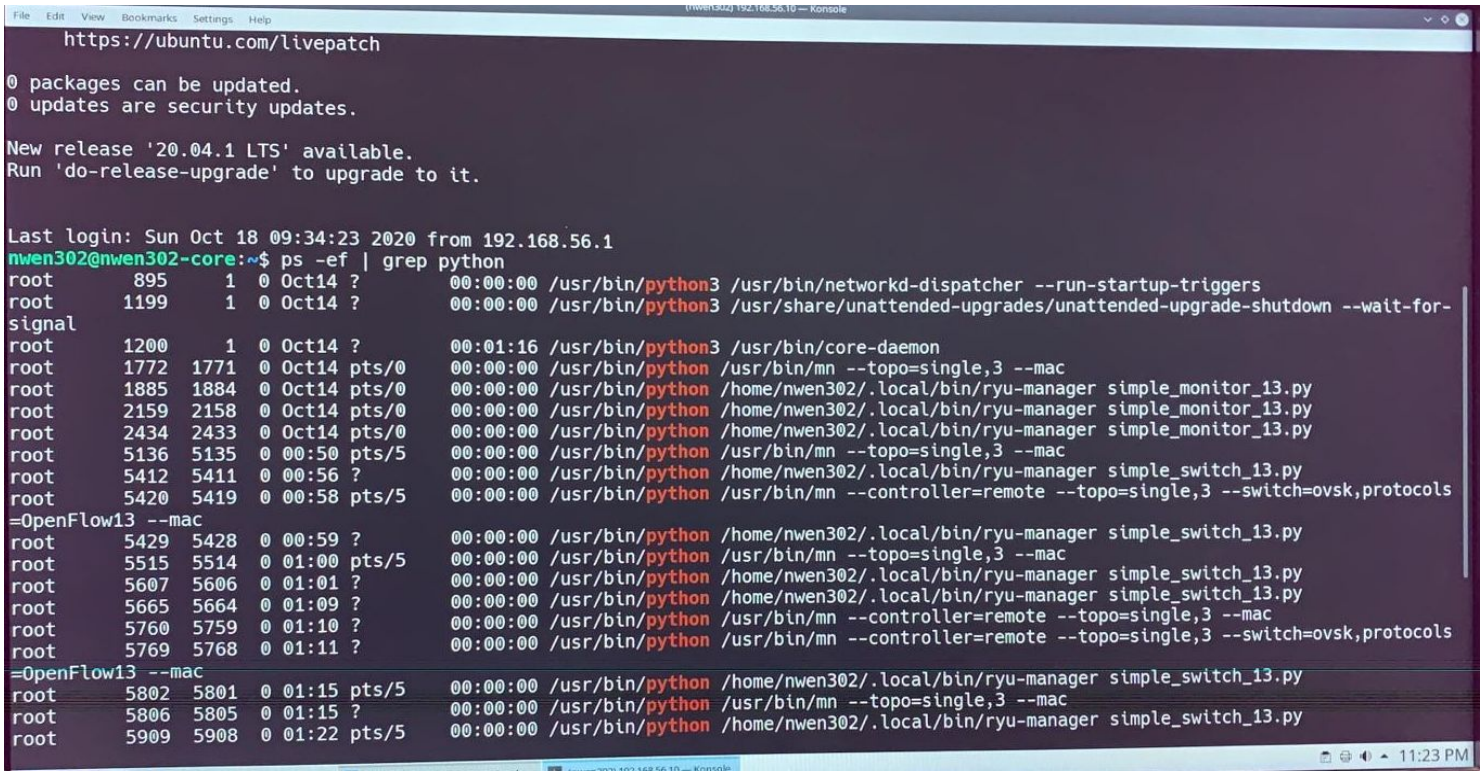


I tried many different ways to fix it, but each method I used did not work. It actually makes it harder or it will cause more error. After searching and looking for a solution to this error, I found this website. This website explains this command below in detail.

```
-------------------------
$ ps -ef |  grep python
-------------------------
```

https://explainshell.com/explain?cmd=ps+-ef+%7C+grep+python

Basically, I had many processes running at the same time.



Therefore, I had to kill each process individually using this command below.

```
----------------------------
% sudo kill -9 <pid-number>
----------------------------
```