

LAB 0

2021

Network Emulation Lab

04/07/2021

University of Idaho

Yifan Zhu



CONTENTS

Introduction	1
Software.....	1
Procedure	1
Part 1	1
Part 2	2
Part 3	3
Part 4	5
Part 5 (OPTIONAL)	6
Feedback.....	9

Introduction

Due to the severe pandemic, some students may not have the chance to go to the UI lab for doing the topology lab. So, I designed this lab for the emulation.

Software

- Workstation Pro 16
- Mininet

Procedure

First, you should get familiar with these commands;

```
Sudo -i
```

```
[pass word] : 123
```

```
Cd /home/ubuntu/mininet/
```

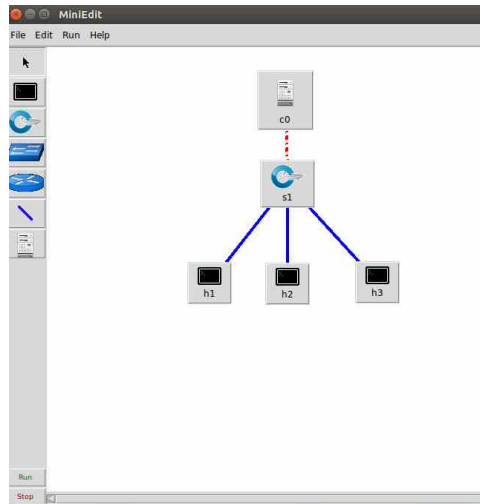
```
Examples/miniedit.py
```

Then you can open the miniedit to create a topology.

Part 1

In this lab, let's make a simple one, the single topology.

See figure 1

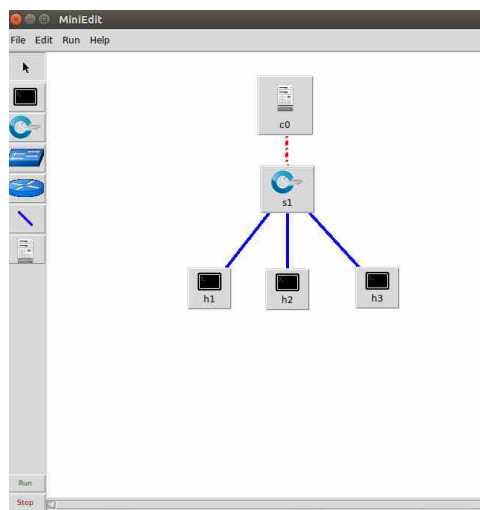


[Figure 1]

Part 2

In this lab, let's make a simple one, the single topology.

See figure 1



[Figure 1]

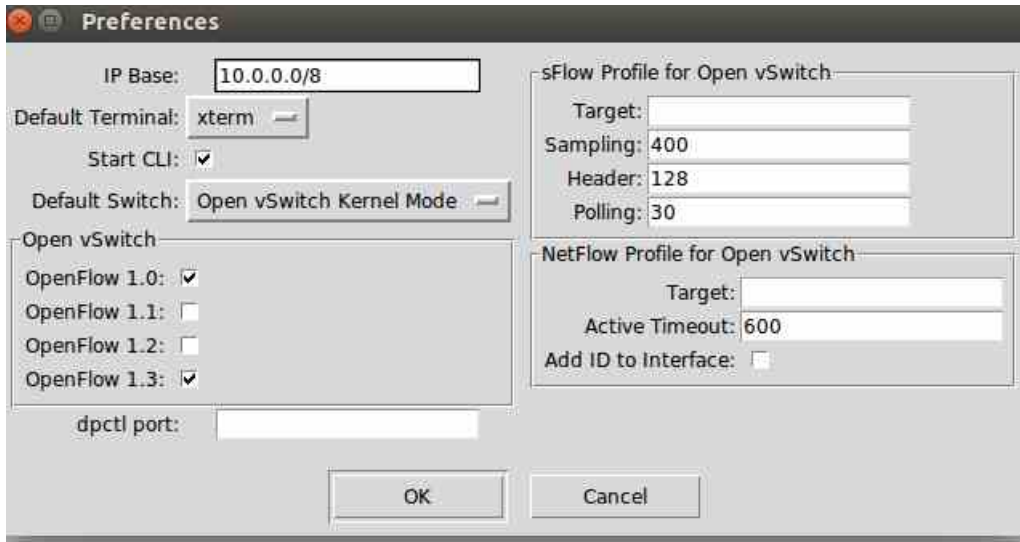
Then click the edit>preference>

As shown in figure 2

In preference session:

1. Check the IP base, that's where you will use in the host, 10.0.0.0/8.

2. Tickle the CLI, so you can make some modifications directly on mininet without opening the miniedit.
3. Choose the default switch as 'Open vSwitch Kernel Mode'.
4. Tickle the openflow 1.0 and 1.3.



[Figure 2]

Part 3

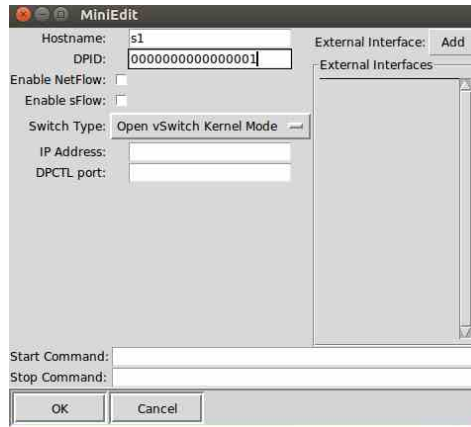
Also, we need to set up the properties in controller, switches and hosts.

- **Controller:** In figure 3, right click on the 'C0', choose 'property'. Change the 'controller type' as Remote Controller.



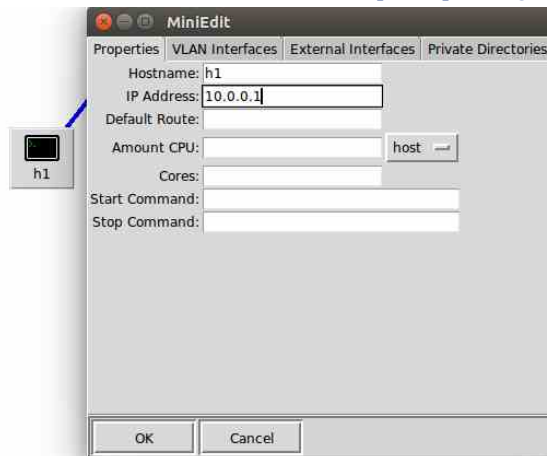
[Figure 3]

- **Switch:** In figure 4, right click on the 'S1', choose 'property'. Choose the default switch as 'Open vSwitch Kernel Mode'. And set the DPIP, it's in 16-digits. So set it as 0000 0000 0000 0001



[Figure 4]

- **Host:** In figure 5, right click on the 'h0', choose 'property'. Set its IP address. Based on the IP address in the preference. It's 10.0.0.1. By the same method, set 'h1' as 10.0.0.2. And so on, I believe you know what to do with the property of 'h3'.



[Figure 5]

Then click the 'save' icon to save your file. And export it as level 2, so that you can start the emulation without launching the miniedit.

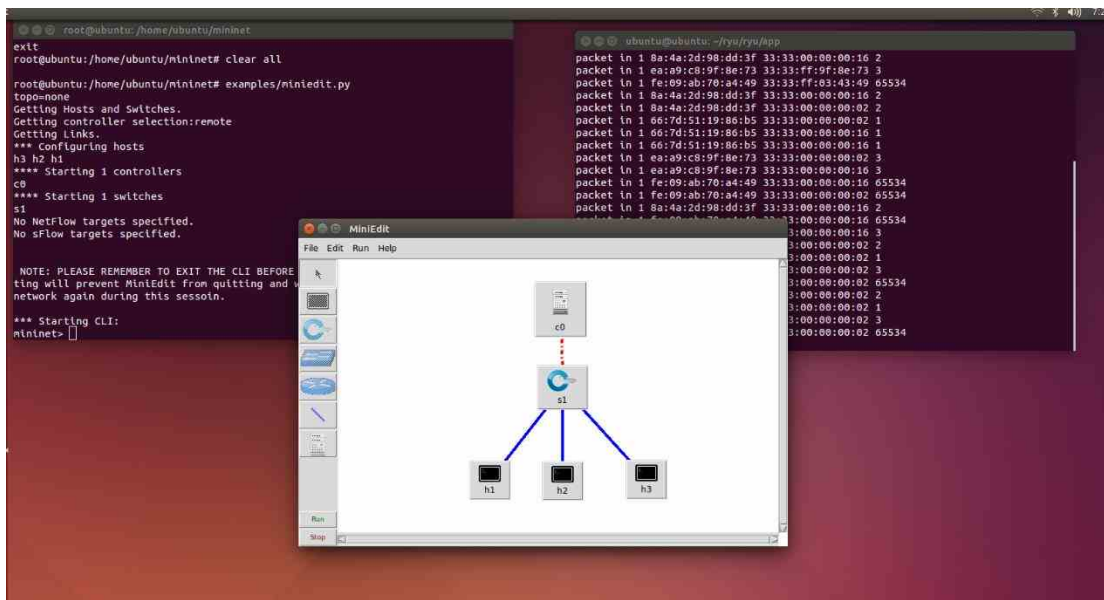
Part 4

Right click the CMD interface, choose the 'Open New Terminal'.

- Cd ryu
- Cd ryu
- Cd app
- Ryu-manager simple_switch.py

Then we run the topology.

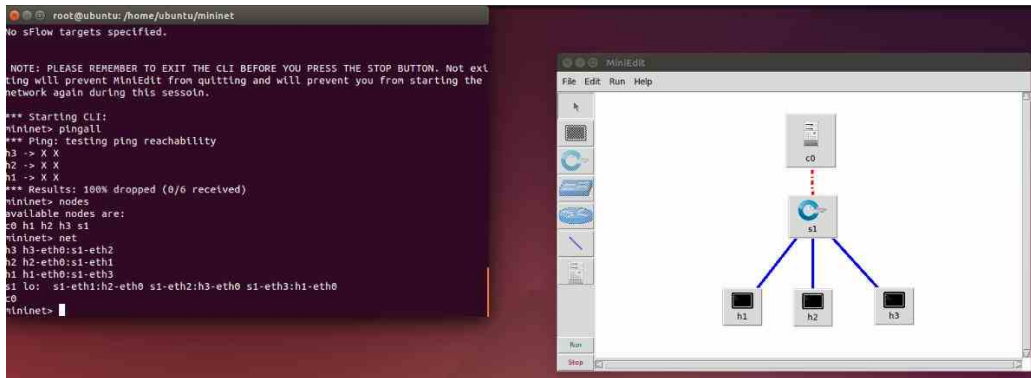
You can see they are recording the data as shown in figure 6.



[Figure 6]

Then type these commands in mininet. Check the connection, the available nodes and connect conditions. See figure 7.

- Pingall
- Nodes
- Net



[Figure 7]

Part 5 (OPTIONAL)

In the previous procedure, we've made the emulations in mininet. As we saved our file. Here I save them as 'lab 0.mn' and 'lab0.py'. We can check the python file in mininet interface.

You may need to check the path, command 'ls' can help you have a quick review of all the files.

Type in vi lab0.py in the file folder where you can find your saved file. Then you can check it. See figure 8.


```
ubuntu@ubuntu: ~/mininet
! /usr/bin/python
from mininet.net import Mininet
from mininet.node import Controller, RemoteController, OVSController
from mininet.node import CPULimitedHost, Host, Node
from mininet.node import OVSKernelSwitch, UserSwitch
from mininet.node import IVSSwitch
from mininet.cli import CLI
from mininet.log import setLogLevel, info
from mininet.link import TCLink, Intf
from subprocess import call

def myNetwork():

    net = Mininet( topo=None,
                  build=False,
                  ipBase='10.0.0.0/8')

    info( '*** Adding controller\n' )
    c0=net.addController(name='c0',
                        controller=RemoteController,
                        ip='127.0.0.1',
                        protocol='tcp',
                        port=6633)

    info( '*** Add switches\n' )
    s1 = net.addSwitch('s1', cls=OVSKernelSwitch, dpid='0000000000000001')

    info( '*** Add hosts\n' )
    h1 = net.addHost('h1', cls=Host, ip='10.0.0.1', defaultRoute=None)
    h3 = net.addHost('h3', cls=Host, ip='10.0.0.3', defaultRoute=None)
    h2 = net.addHost('h2', cls=Host, ip='10.0.0.2', defaultRoute=None)

    info( '*** Add links\n' )
    net.addLink(s1, h1)
    net.addLink(s1, h2)
    net.addLink(s1, h3)

    info( '*** Starting network\n' )
    net.build()
    info( '*** Starting controllers\n' )
    for controller in net.controllers:
        controller.start()

    info( '*** Starting switches\n' )
    net.get('s1').start([c0])
```

[Figure 8]

You can type ':' , then type 'q!' to exit.

So, if you want to directly check it without launching miniedit. Open two terminals.

Launch the ryu manager as I did before.

Type in 'python lab0.py'. (Don't forget to check the root administration)

See figure 9.

```

root@ubuntu: /home/ubuntu/mininet
*** Add links
*** Starting network
*** Configuring hosts
h1 h3 h2
*** Starting controllers
*** Starting switches
*** Post configure switches and hosts
*** Starting CLI:
mininet> nodes
available nodes are:
h0 h1 h2 h3 s1
mininet> net
mininet> s1-eth0
h1-eth0:s1-eth1
h3-eth0:s1-eth3
h2-eth0:s1-eth2
s1 lo: s1-eth1:h1-eth0 s1-eth2:h2-eth0 s1-eth3:h3-eth0
mininet> pingall
*** Ping: testing ping reachability
h1 -> h3 h2
h3 -> h1 h2
h2 -> h1 h3
*** Results: 0% dropped (6/6 received)
mininet>

```

```

ubuntu@ubuntu: ~/ryu/ryu/app
packet in 1 26:75:92:da:e3:6c 33:33:00:00:00:02 3
packet in 1 26:75:92:da:e3:6c 33:33:00:00:00:16 3
packet in 1 5e:d4:22:bb:71:4d 33:33:00:00:00:16 1
packet in 1 26:75:92:da:e3:6c 33:33:00:00:00:16 3
packet in 1 ae:ea:f0:25:bc:8c 33:33:00:00:00:16 2
packet in 1 b2:23:2a:6e:a6:4d 33:33:00:00:00:16 65534
packet in 1 5e:d4:22:bb:71:4d 33:33:00:00:00:02 1
packet in 1 b2:23:2a:6e:a6:4d 33:33:00:00:00:02 65534
packet in 1 ae:ea:f0:25:bc:8c 33:33:00:00:00:02 2
packet in 1 26:75:92:da:e3:6c 33:33:00:00:00:02 3
packet in 1 5e:d4:22:bb:71:4d 33:33:00:00:00:02 1
packet in 1 b2:23:2a:6e:a6:4d 33:33:00:00:00:02 65534
packet in 1 ae:ea:f0:25:bc:8c 33:33:00:00:00:02 2
packet in 1 26:75:92:da:e3:6c 33:33:00:00:00:02 3
packet in 1 5e:d4:22:bb:71:4d ff:ff:ff:ff:ff:ff 1
packet in 1 26:75:92:da:e3:6c 5e:d4:22:bb:71:4d 3
packet in 1 5e:d4:22:bb:71:4d 26:75:92:da:e3:6c 1
packet in 1 5e:d4:22:bb:71:4d ff:ff:ff:ff:ff:ff 1
packet in 1 ae:ea:f0:25:bc:8c 5e:d4:22:bb:71:4d 2
packet in 1 5e:d4:22:bb:71:4d ae:ea:f0:25:bc:8c 1
packet in 1 26:75:92:da:e3:6c ff:ff:ff:ff:ff:ff 3
packet in 1 ae:ea:f0:25:bc:8c 26:75:92:da:e3:6c 2
packet in 1 26:75:92:da:e3:6c ae:ea:f0:25:bc:8c 3

```

[Figure 9]

Let me show you how to directly edit the python file.

For example, I want to add a new host under the s1. We open the 'lab0.py'

Add them under the next the line of h0, also add a new switch position.

Press 'i' or 'a' to start the medications. (Never forget to have the root administration, otherwise you can't edit it!)

It's okay if you are familiar with python or not, just follow the line as already have beore.

Then press ':wq!' to save it.

See figure 10.

```

info( '*** Add switches\n')
s1 = net.addSwitch('s1', cls=OVSKernelSwitch, dpid='0000000000000001')

info( '*** Add hosts\n')
h3 = net.addHost('h3', cls=Host, ip='10.0.0.3', defaultRoute=None)
h1 = net.addHost('h1', cls=Host, ip='10.0.0.1', defaultRoute=None)
h2 = net.addHost('h2', cls=Host, ip='10.0.0.2', defaultRoute=None)
h4 = net.addHost('h4', cls=Host, ip='10.0.0.4', defaultRoute=None)

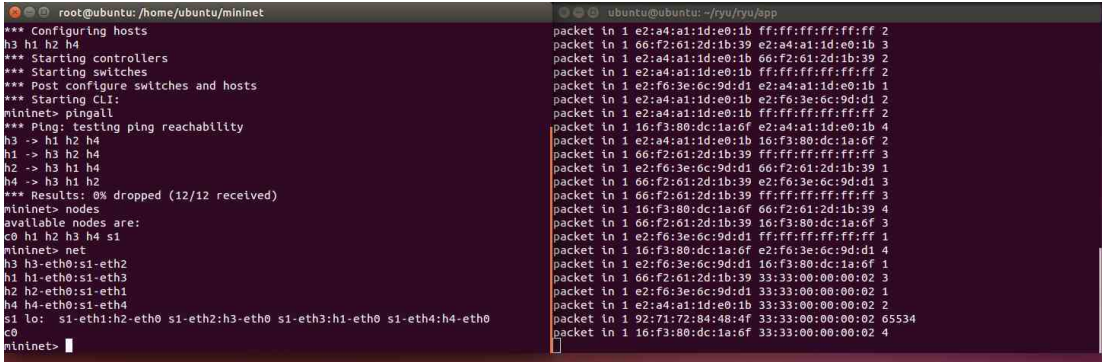
info( '*** Add links\n')
net.addLink(s1, h2)
net.addLink(s1, h3)
net.addLink(s1, h1)
net.addLink(s1,h4)

```

[Figure 10]

Then you can check it by using the procedure before.

See figure 11.



```
root@ubuntu: /home/ubuntu/mininet
*** Configuring hosts
h3 h1 h2 h4
*** Starting controllers
*** Starting switches
*** Post configure switches and hosts
*** Starting CLI:
mininet> pingall
*** Ping: testing ping reachability
h3 -> h1 h2 h4
h1 -> h3 h2 h4
h2 -> h3 h1 h4
h4 -> h3 h1 h2
*** Results: 0% dropped (12/12 received)
mininet> nodes
available nodes are:
c0 h1 h2 h3 h4 s1
mininet> net
h3 h3-eth0:s1-eth2
h1 h1-eth0:s1-eth3
h2 h2-eth0:s1-eth1
h4 h4-eth0:s1-eth4
s1 lo: s1-eth1:h2-eth0 s1-eth2:h3-eth0 s1-eth3:h1-eth0 s1-eth4:h4-eth0
c0
mininet>

ubuntu@ubuntu: ~/ryu/ryu/app
packet in 1 e2:a4:a1:1d:e0:1b ff:ff:ff:ff:ff:ff 2
packet in 1 66:f2:61:2d:1b:39 e2:a4:a1:1d:e0:1b 3
packet in 1 e2:a4:a1:1d:e0:1b 66:f2:61:2d:1b:39 2
packet in 1 e2:a4:a1:1d:e0:1b ff:ff:ff:ff:ff:ff 2
packet in 1 e2:f6:3e:6c:9d:d1 e2:a4:a1:1d:e0:1b 1
packet in 1 e2:a4:a1:1d:e0:1b e2:f6:3e:6c:9d:d1 2
packet in 1 e2:a4:a1:1d:e0:1b ff:ff:ff:ff:ff:ff 2
packet in 1 16:f3:80:dc:1a:6f e2:a4:a1:1d:e0:1b 4
packet in 1 e2:a4:a1:1d:e0:1b 16:f3:80:dc:1a:6f 2
packet in 1 66:f2:61:2d:1b:39 ff:ff:ff:ff:ff:ff 3
packet in 1 e2:f6:3e:6c:9d:d1 66:f2:61:2d:1b:39 1
packet in 1 66:f2:61:2d:1b:39 e2:f6:3e:6c:9d:d1 3
packet in 1 66:f2:61:2d:1b:39 ff:ff:ff:ff:ff:ff 3
packet in 1 16:f3:80:dc:1a:6f 66:f2:61:2d:1b:39 4
packet in 1 66:f2:61:2d:1b:39 16:f3:80:dc:1a:6f 3
packet in 1 e2:f6:3e:6c:9d:d1 ff:ff:ff:ff:ff:ff 1
packet in 1 16:f3:80:dc:1a:6f e2:f6:3e:6c:9d:d1 4
packet in 1 e2:f6:3e:6c:9d:d1 16:f3:80:dc:1a:6f 1
packet in 1 66:f2:61:2d:1b:39 33:33:00:00:00:02 3
packet in 1 e2:f6:3e:6c:9d:d1 33:33:00:00:00:02 1
packet in 1 e2:a4:a1:1d:e0:1b 33:33:00:00:00:02 2
packet in 1 92:71:72:84:48:4f 33:33:00:00:00:02 65534
packet in 1 16:f3:80:dc:1a:6f 33:33:00:00:00:02 4
```

[Figure 11]

It works!

Feedback

Please let us know your feedback, we are not from ECE444/544 class, so we don't know the meaning of these data. We can only provide you the way to do the emulation, but the basic principles are left to you.