

Lab 2: Software-Defined Networking (Ryu Controller)

Contents

Experiment Task Design	3
Submission	3
Profile Setup	3
Create a profile for an SDN controller	3
Installing Ryu Controller	4
Setup a topology profile for the experiment	4
Conducting the Lab	6
Install Open vSwitch and setup bridges on all nodes.	6
Ping and dump flows	7
Appendix	7
OVS commands:	7

Experiment Task Design

This lab aims to let students understand the basic concept of SDN in CloudLab. This lab uses Ryu Controller, one of the open-source SDN controllers. Based on the topology that the student creates in the first lab (Lab 1), she or he can push flow rules for routing through Ryu.

Submission

Take screenshots of all the steps involved and explain in one or two paragraphs. Study the flows rules below and explain the meaning of the printed flow rules on the terminal.

Students can refer the link (<http://docs.cloudlab.us/cloudlab-tutorial.html>) for more details about creating profiles on CloudLab. Students should have an account of either CloudLab or GENI or any other federated services like EmuLab to access CloudLab. CloudLab login page: <https://www.cloudlab.us/login.php>

Profile Setup

Create a profile for an SDN controller

Students create a profile with a single node as shown in Figure 1. Use ‘**Ubuntu 18**’ as the OS and select “any” for the hardware type. Accept the topology and then click on the **create** button. Please give appropriate descriptions and then instantiate the experiment by clicking on the **instantiate** button. You will be asked to select a cluster. Select the available cluster and then click the **“Finish”** button. Once the experiment is online, students need to install Ryu controller on the node.

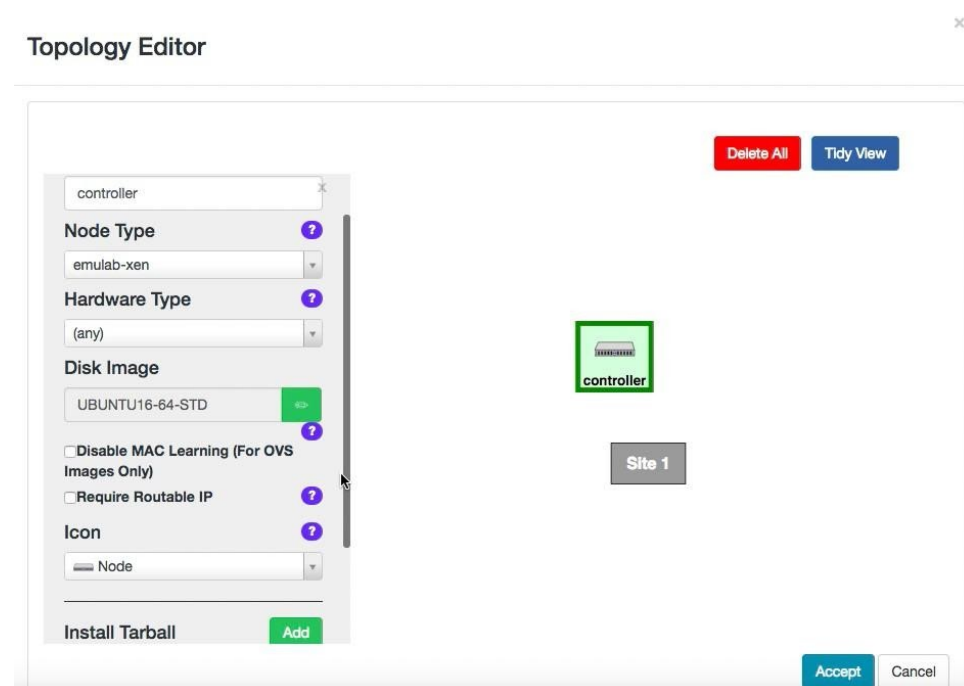


Figure 1: Setup a single node for an SDN controller. (Note that if you check “Require Routable IP,” the controller can be assigned with a public IP address that other nodes can access. If not, the controller will have a private IP address.)

Installing Ryu Controller

- 1) Open a new terminal.
- 2) Run **"ifconfig"** and note down the IP address. This IP address will be used whenever the topology is needed to be setup for an experiment.
- 3) To install Ryu controller, perform the following steps:

Get sudo user privileges

```
sudo su
```

Update APT repo

```
apt-get update
```

Download Ryu repo from Github

```
git clone https://github.com/osrg/ryu.git
```

Install dependencies

```
apt install gcc python-dev libffi-dev libssl-dev libxml2-dev libxslt1-dev zlib1g-dev
```

```
apt install python3-pip
```

Install Ryu Controller:

```
cd ryu
```

```
pip3 install -r tools/pip-requirements
```

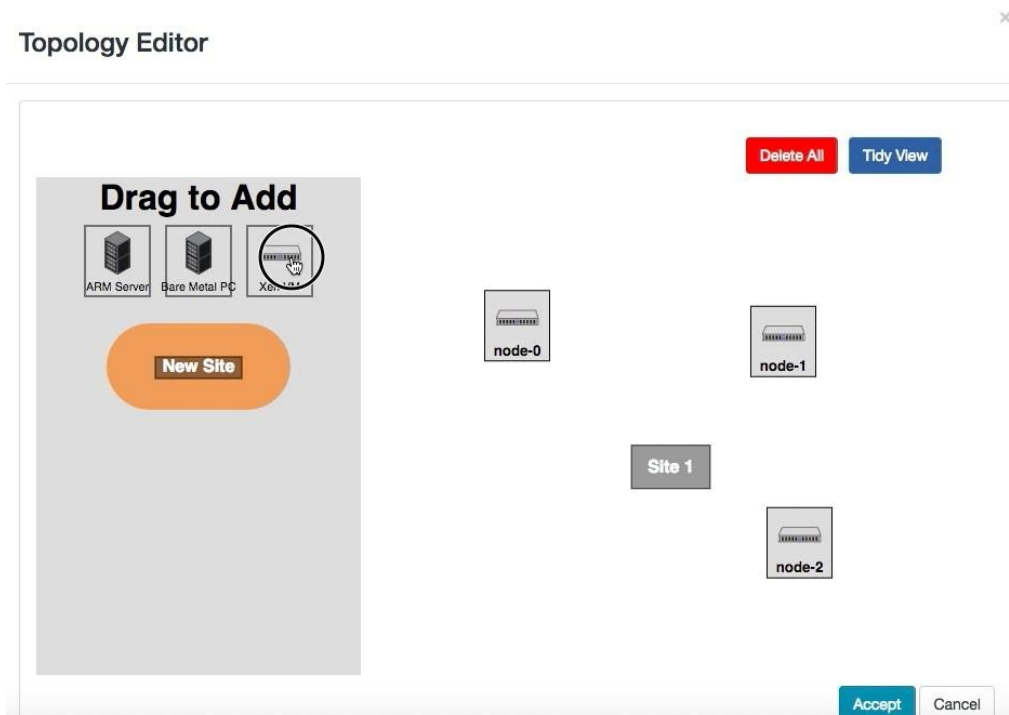
```
python3 setup.py install
```

(Notes) You can use `"sudo /bin/sh set_ryu.sh"` run all those commands to save your installation time. After installation, run `"cd "` to switch to the destination directory.

- 4) Start the controller: **"ryu-manager --verbose yourapp.py"**.
The `"verbose"` option prints the debug output. All the default applications are in `"ryu/app/"` directory. For example, **"ryu-manager --verbose ryu/app/simple_switch_13.py"**

Setup a topology profile for the experiment

- 1) Create a profile with 4 Xen VMs.



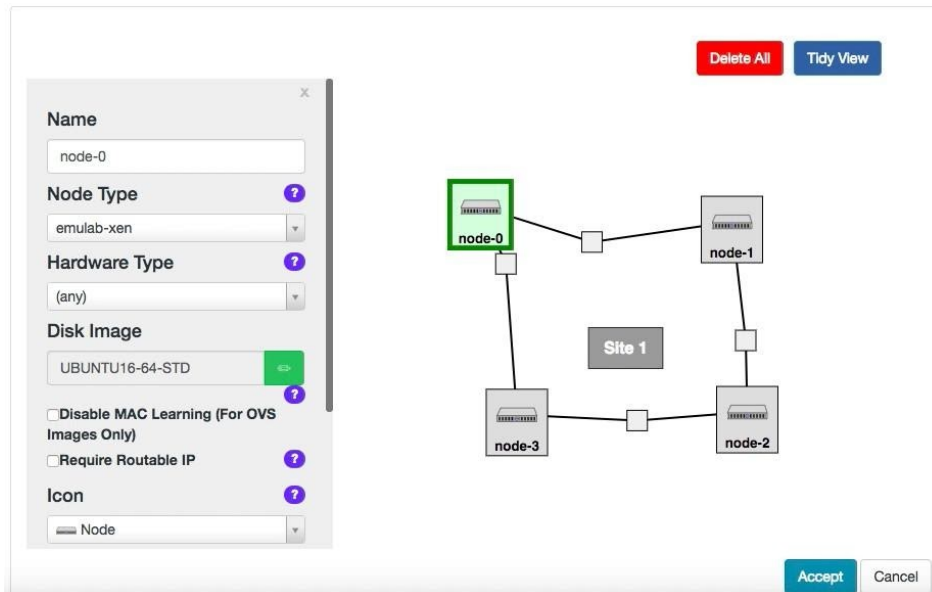
2) The following details should be given for each node:

OS: Ubuntu 16

Hardware Type: Any

Node Type: emulab-xen

Check "Require Routable IP"



3) Provide the details of each link. Select 'Ethernet' as the 'Link type' for this lab.

Note: Enable OpenFlow and insert the IP address that we noted from the SDN Controller into the empty textbox below the 'Enable OpenFlow' checkbox. An example of the command would be: "tcp:128.110.99.138:6653". The IP address here is obtained from the SDN Controller (Ryu controller node).

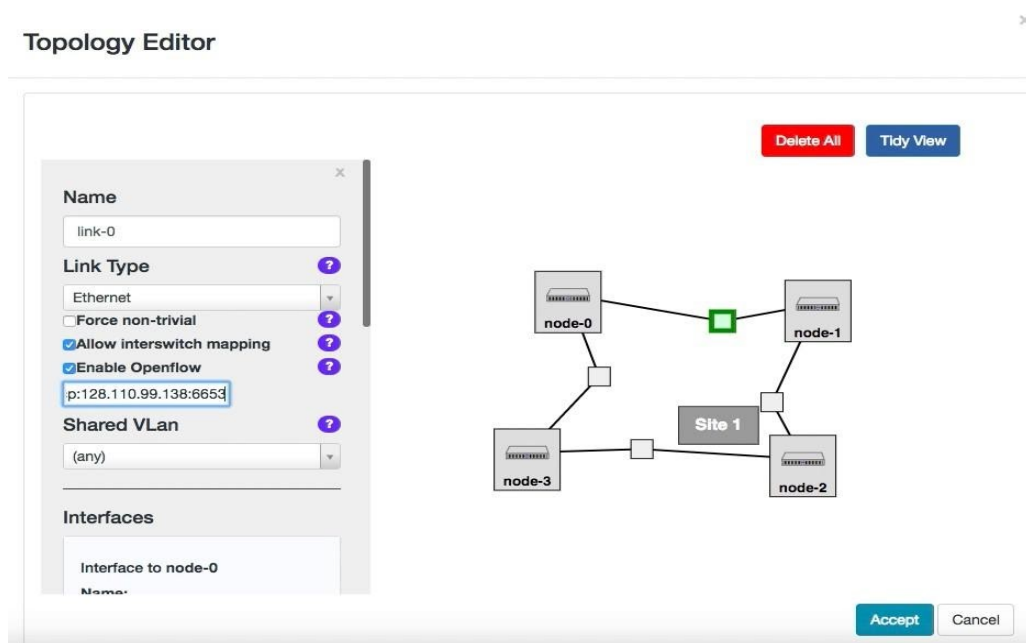


Figure 4: Provide the details of each link. Select 'Ethernet' for 'Link type'.

Also select "Enable OpenFlow" and provide an appropriate IP address and the port number.

4) After putting in the required details of the topology, students can then proceed to starting their experiment.

Conducting the Lab

Install Open vSwitch and setup bridges on all nodes.

All the links in our topology are connected to the SDN controller (i.e. Ryu controller). To check flow rules installed by the controller for routing, we will setup a bridge on all nodes and connect them to the Ryu controller. The controller will then learn the new topology and send appropriate flow rules to the switches.

- 1) Run “**sudo apt-get install openvswitch-switch**” to install OpenVSwitch
- 2) Use the following commands to setup a bridge on each node and connect it to SDN controller:

```
sudo su
ovs-vsctl add-br <bridge_name>
ovs-vsctl add-port <bridge_name> eth1
ovs-vsctl add-port <bridge_name> eth2
ifconfig eth1 0
ifconfig eth2 0
ovs-vsctl set bridge <bridge_name> stp_enable=true
ovs-vsctl set-controller <bridge_name> tcp:<controller_IP_Address>:6653
ifconfig <bridge_name> 10.10.10.1 netmask 255.255.255.0 up
```

These steps must be run on **all nodes** of the topology. Each bridge should have its own bridge name along with a different IP address. For this experiment, we will use the following bridge names for all 4 nodes: “**ovs-lan1**”, “**ovs-lan2**”, “**ovs-lan3**”, “**ovs-lan4**”. Give an IP to each of them as “**10.10.10.1**”, “**10.10.10.2**”, “**10.10.10.3**”, “**10.10.10.4**” from *node 1* to *node 4* respectively. (Note that users need to disable for “stp_enabl” with RYU controller because)

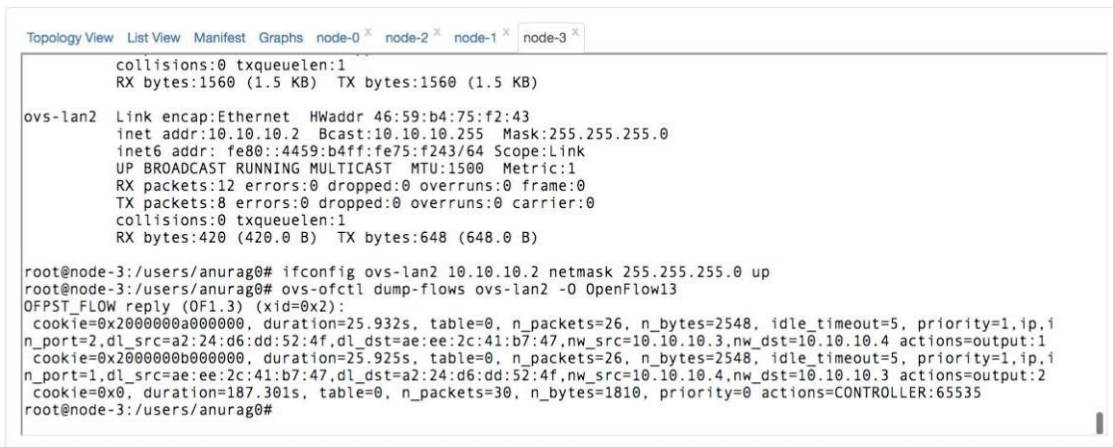
The following example below shows a sample of the command for **ovs-lan1**:

```
sudo su
ovs-vsctl add-br ovs-lan1
ovs-vsctl add-port ovs-lan1 eth1
ovs-vsctl add-port ovs-lan1 eth2
ifconfig eth1 0
ifconfig eth2 0
ovs-vsctl set bridge ovs-lan1 stp_enable=true
ovs-vsctl set-controller ovs-lan1 tcp:155.98.37.38:6653
ifconfig ovs-lan1 10.10.10.1 netmask 255.255.255.0 up
```

(Notes) You can use “**sudo /bin/sh set_ovs.sh eth1 eth2 155.98.37.38 10.10.10.1**” to run the above commands to save your time. The last IP is for **ovs-lan1**, the last second IP is for the **controller**.

Ping and dump flows

Once all the configuration settings are finished, the **Ping** command from **node-0** to **node-2** will start working. Students can use **"tcpdump -i eth1"** on **node-1** and **node-2** to check the routing path for the **Ping** command. In addition, student can check the flow rules on all the 4 nodes using the following command, **"ovs-ofctl dump-flows <bridge_name> -O OpenFlow13"** by replacing <bridge_name> with the configured bridge names for each node.



```
Topology View List View Manifest Graphs node-0 x node-2 x node-1 x node-3 x
collisions:0 txqueuelen:1
RX bytes:1560 (1.5 KB) TX bytes:1560 (1.5 KB)

ovs-lan2 Link encap:Ethernet HWaddr 46:59:b4:75:f2:43
inet addr:10.10.10.2 Bcast:10.10.10.255 Mask:255.255.255.0
inet6 addr: fe80::4459:b4ff:fe75:f243/64 Scope:Link
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
RX packets:12 errors:0 dropped:0 overruns:0 frame:0
TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1
RX bytes:420 (420.0 B) TX bytes:648 (648.0 B)

root@node-3:/users/anurag0# ifconfig ovs-lan2 10.10.10.2 netmask 255.255.255.0 up
root@node-3:/users/anurag0# ovs-ofctl dump-flows ovs-lan2 -O OpenFlow13
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x2000000a000000, duration=25.932s, table=0, n_packets=26, n_bytes=2548, idle_timeout=5, priority=1,ip,1
 n_port=2,d_l_src=a2:24:d6:dd:52:4f,d_l_dst=ae:ee:2c:41:b7:47,nw_src=10.10.10.3,nw_dst=10.10.10.4 actions=output:1
 cookie=0x2000000b000000, duration=25.925s, table=0, n_packets=26, n_bytes=2548, idle_timeout=5, priority=1,ip,1
 n_port=1,d_l_src=ae:ee:2c:41:b7:47,d_l_dst=a2:24:d6:dd:52:4f,nw_src=10.10.10.4,nw_dst=10.10.10.3 actions=output:2
 cookie=0x0, duration=187.301s, table=0, n_packets=30, n_bytes=1810, priority=0 actions=CONTROLLER:65535
root@node-3:/users/anurag0#
```

Figure 5: Flow rules inserted by the SDN controller

Appendix

OVS commands:

ovs-vsctl: Used to configure the ovs-vswitchd configuration database (known as ovs-db)

Example: To delete a bridge: **"ovs-vsctl del-br ovs-lan1"**

ovs-ofctl: A command line tool to monitor and control OpenFlow switches

Example: To print the OVS flow rules **"ovs-ofctl dump-flows ovs-lan2 -O OpenFlow13"**