Project 1

Submitted by : Nandini Krupa Krishnamurthy

Part 1: CloudLab

Task:

To create a simple network topology in CloudLab. This simple topology includes four instances of XEN VM linked together. Next a topology should be instantiated. After this, ping test is carried out to see if topology is created successfully. This project is divided into 3 steps:

I. Creating a profile
II. Starting the experiment
III. Testing the connectivity by pinging all nodes.

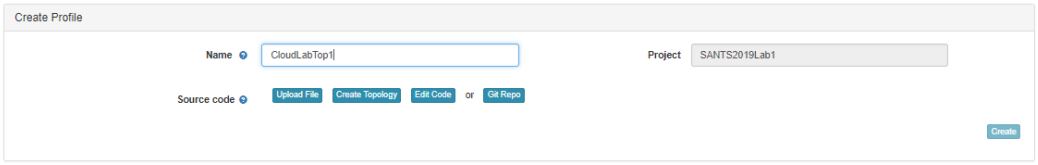
The steps along with the screenshots are as attached.

I. Creating a profile

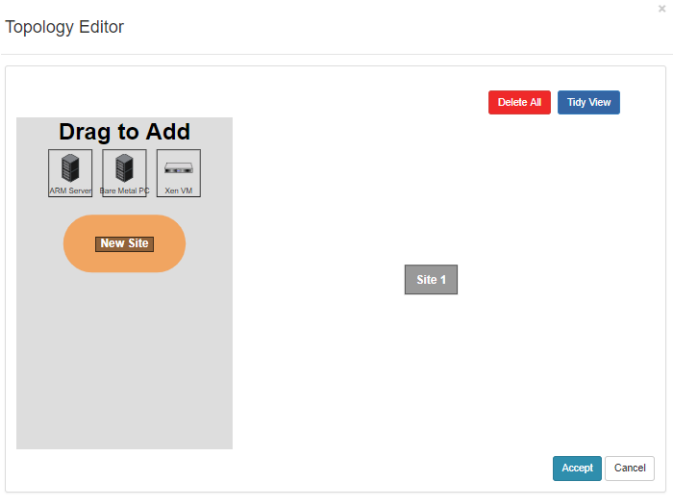
The profile is created as explained and the screenshot is as attached.

To create a profile the following steps are followed.

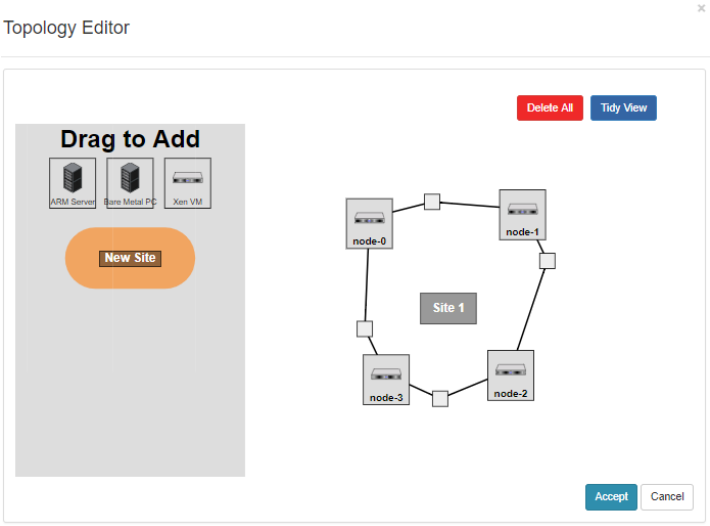
1. In the experiments tab, “Create Experiment Profile” is chosen.
   When the dialogue box opens, the details as shown below are input:



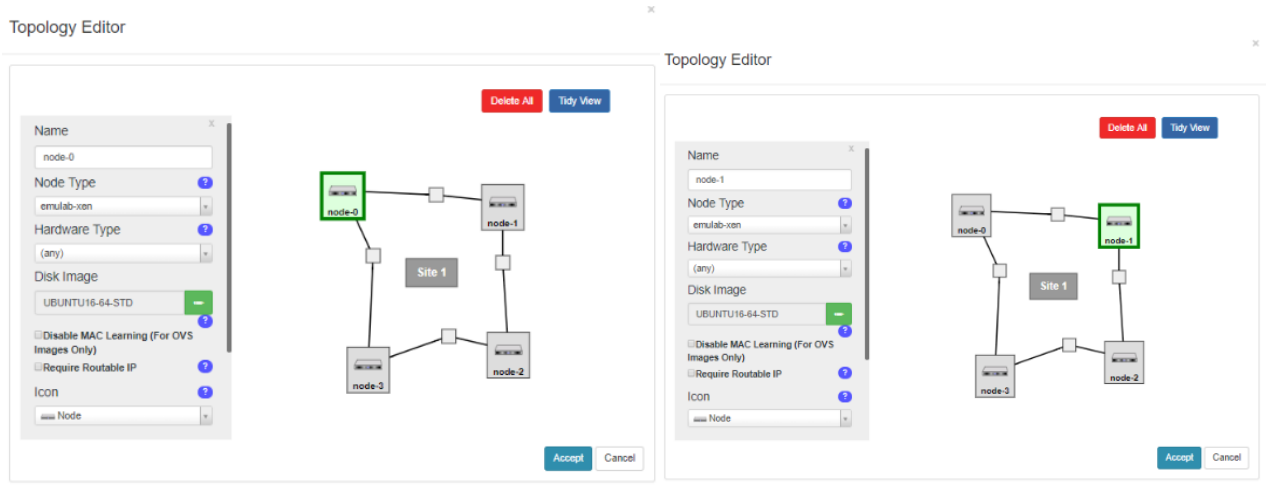
2. “Create topology” is selected as we are not importing or editing from any other source. Topology editor opens as shown below:

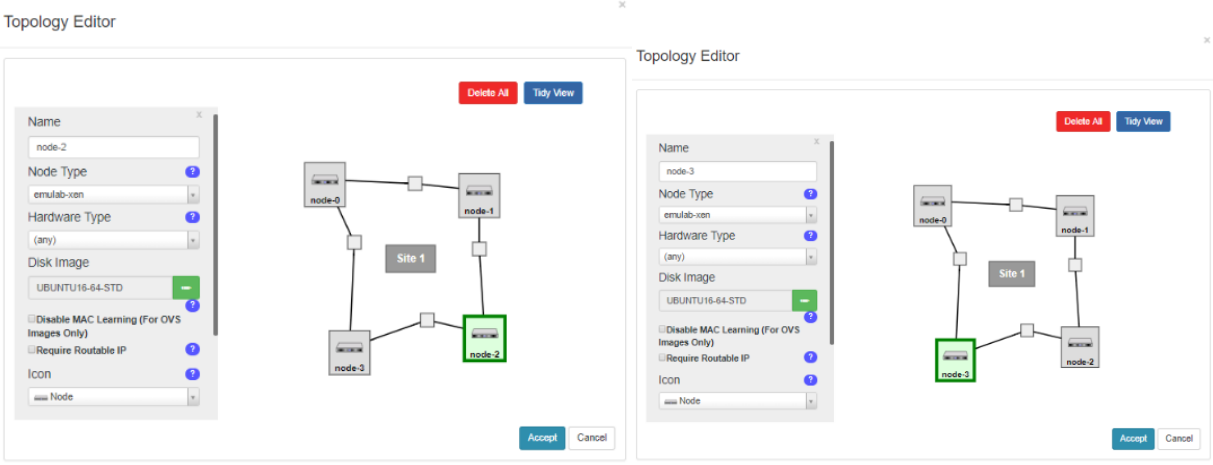


3. Four Xen VMs are dragged and dropped on the editor and connected as shown below. We see there will be 4 links connecting all the four nodes.

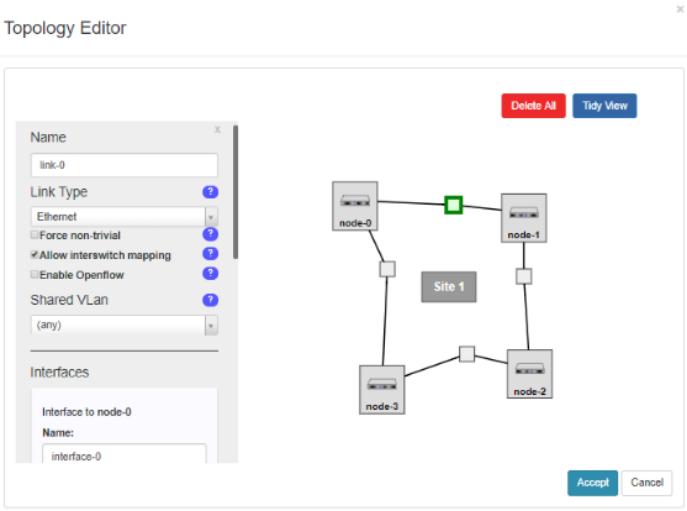


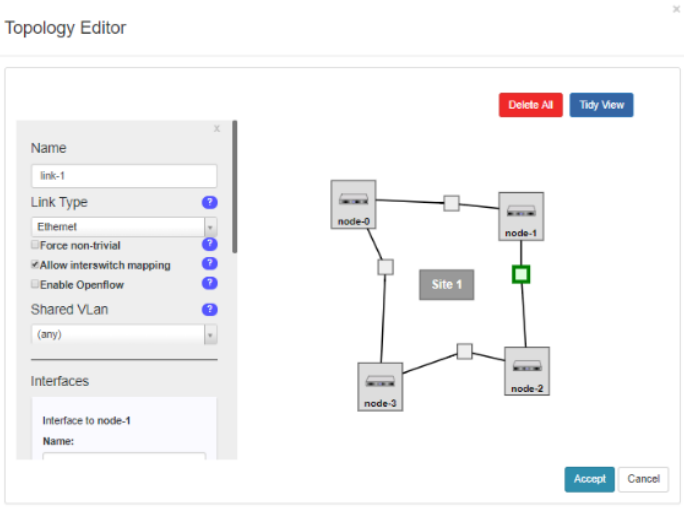
4. For each of the node, the following details are given:
   a. Node Type: emulab-xen
   b. Hardware Type: (any)
   c. Disk Image: UBUNTU16-64-STD

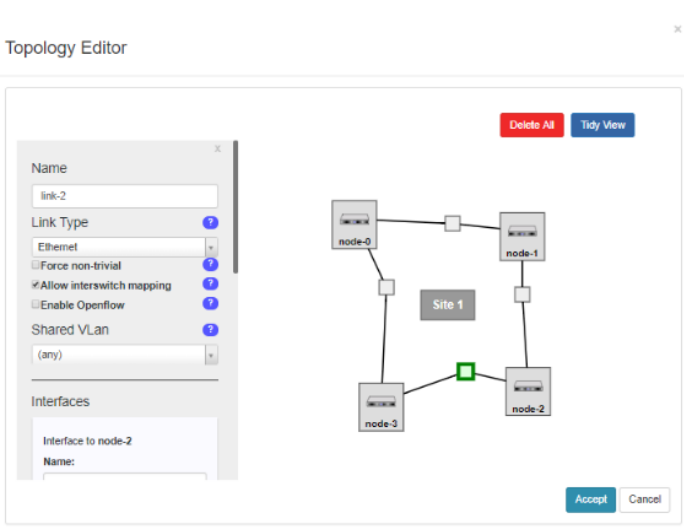


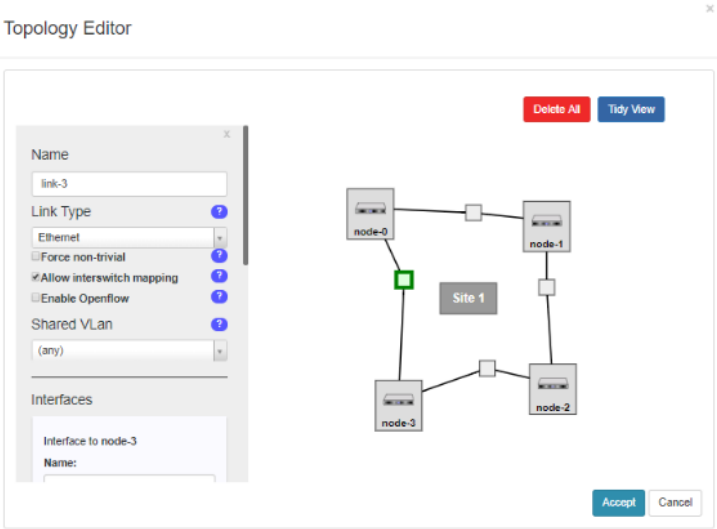


5. For each of the links, the following details are selected:
   a. Link Type: Ethernet
   b. Select Allow interswitch mapping

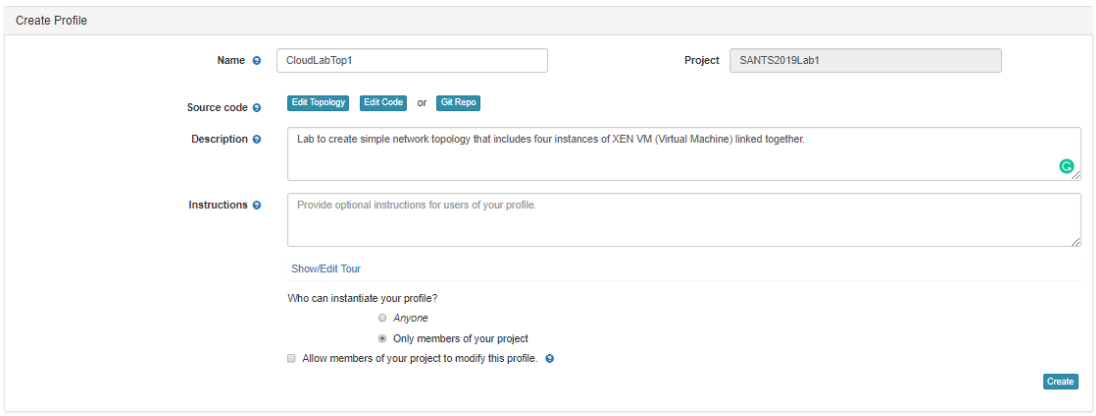






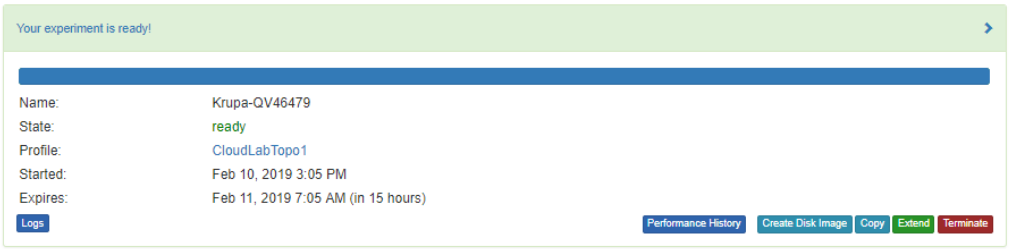


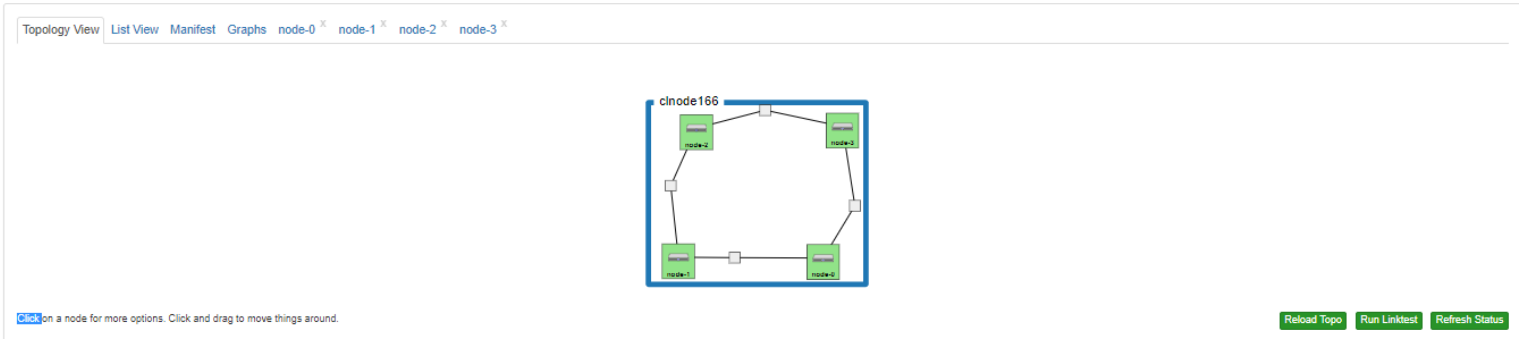
6. The description and optional instructions are provided. After everything is done we procced to create .

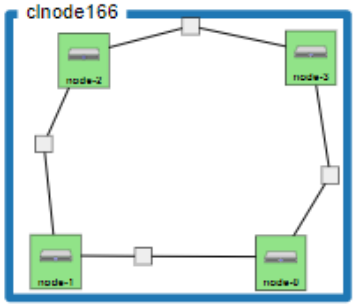


II. Starting the project

Topology View Screenshot:

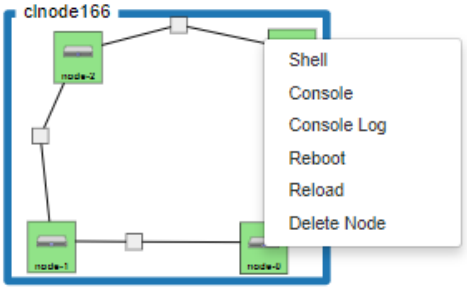




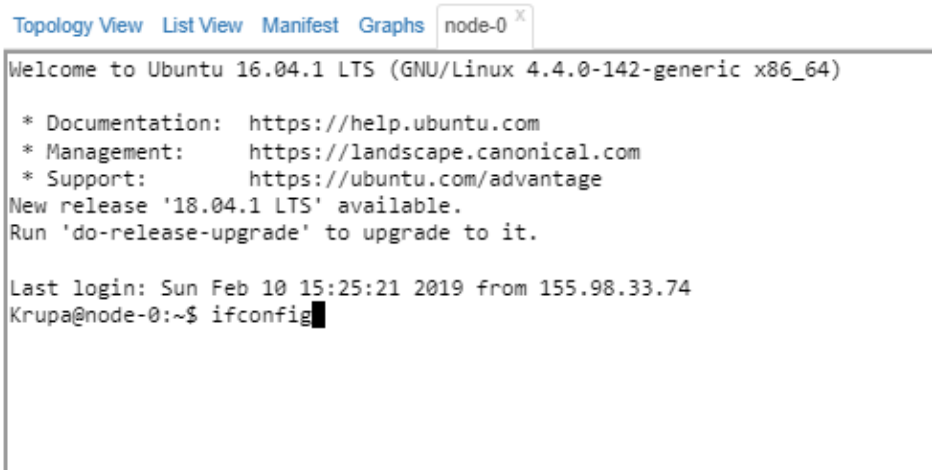


III. Testing Connectivity

1. To test connectiving we need the IP address of each node. Therefore first each node is selected to access “Shell”.

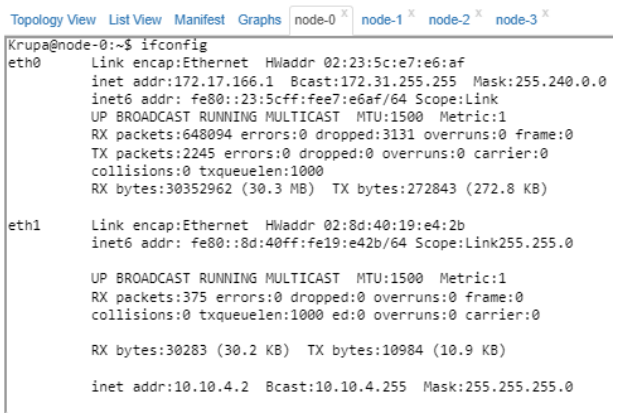


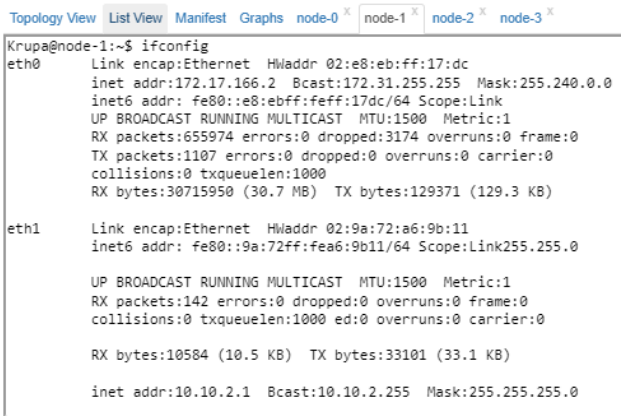
2. In the shell "ifconfig" is typed to find the IP address of the node.

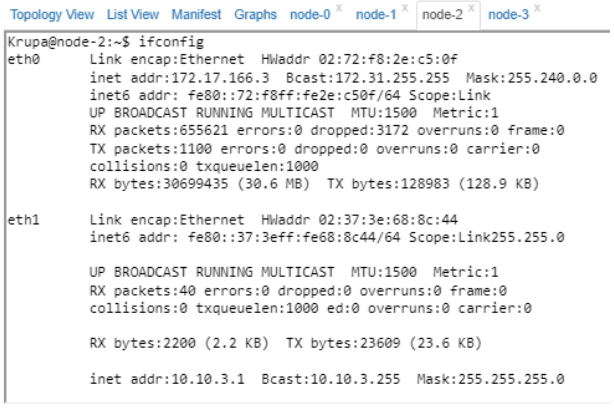


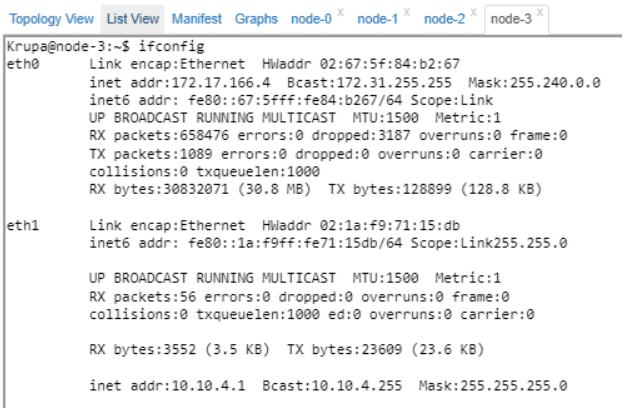
By performing this step, the IP addresses of all the nodes are found out which are as below:

a. Node0: 10.10.4.2
b. Node1: 10.10.2.1
c. Node2: 10.10.3.1
d. Node3: 10.10.4.1



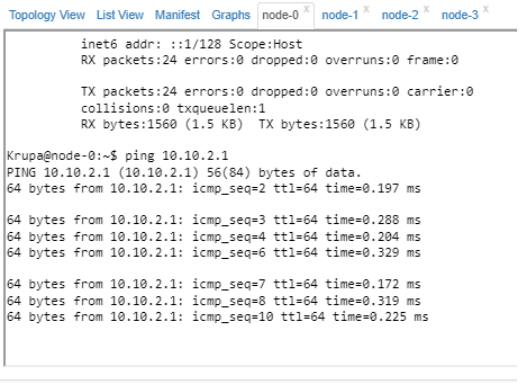


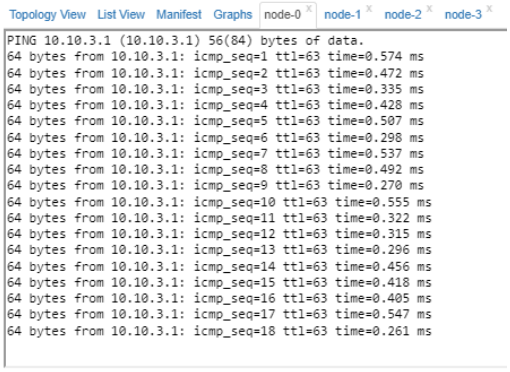




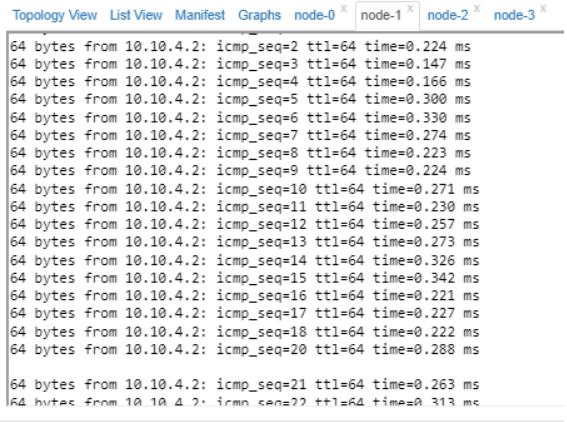
3. Now to check connectivity, ping test is carried out. In this, from a shell of node the command – "ping (ip address)" is executed. For example, to test connectivity from node0 to node1, "ping 10.10.2.1" is executed .

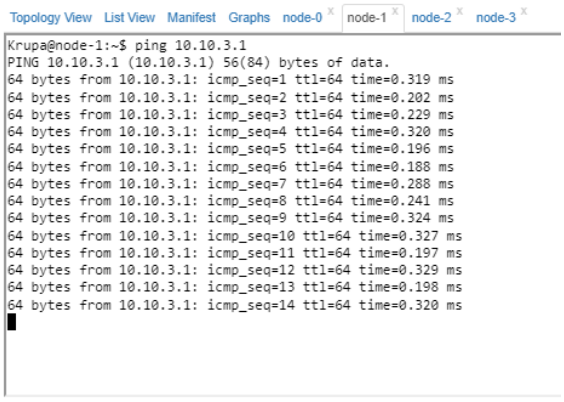
The following screenshots show the connectivity tests performed on node0 to all other nodes.

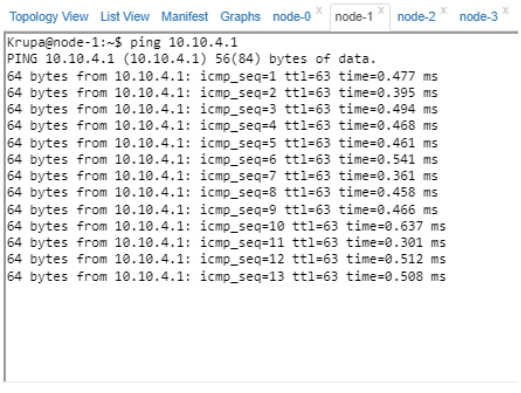




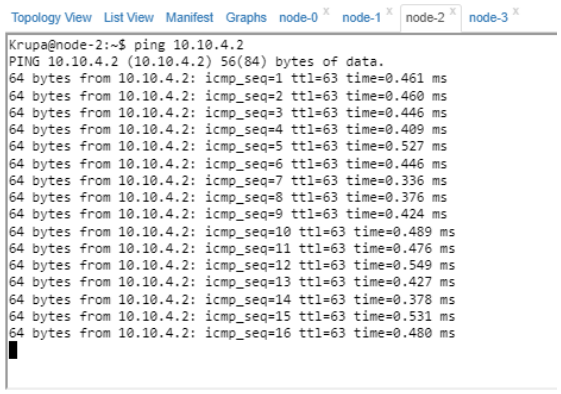
The following screenshots show the connectivity tests performed on node1 to all other nodes.

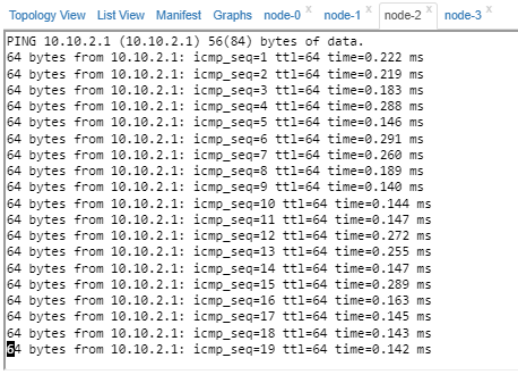


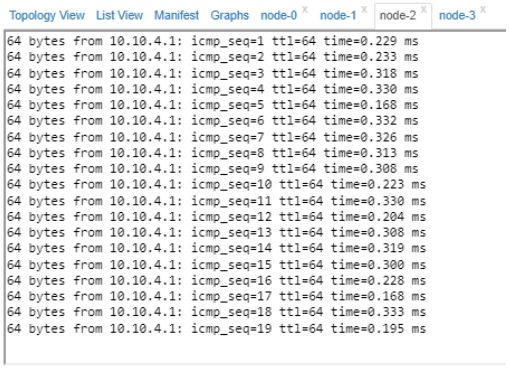




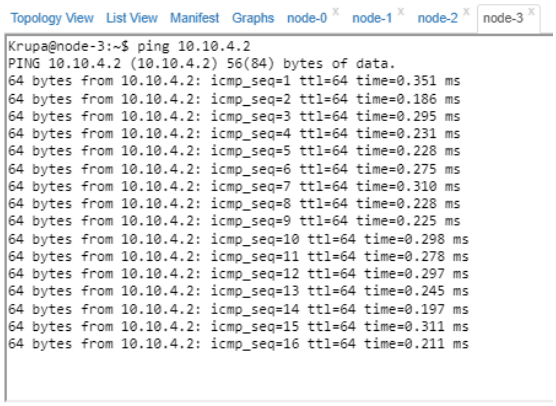
The following screenshots show the connectivity tests performed on node2 to all other nodes.

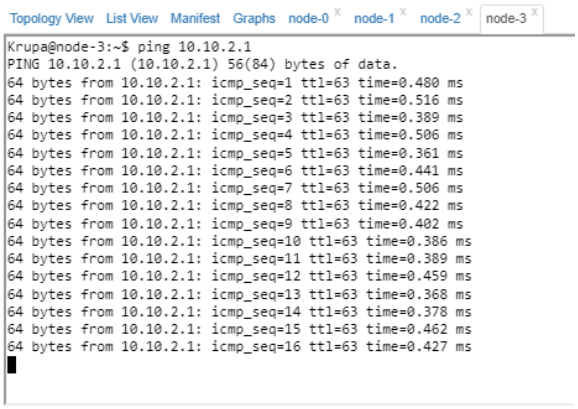


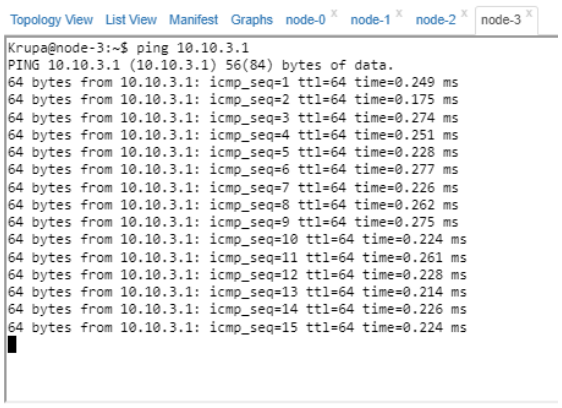




The following screenshots show the connectivity tests performed on node2 to all other nodes.







Project 2: SDN Basic

Task:

To create a basic SDN by using Floodlight in CloudLab

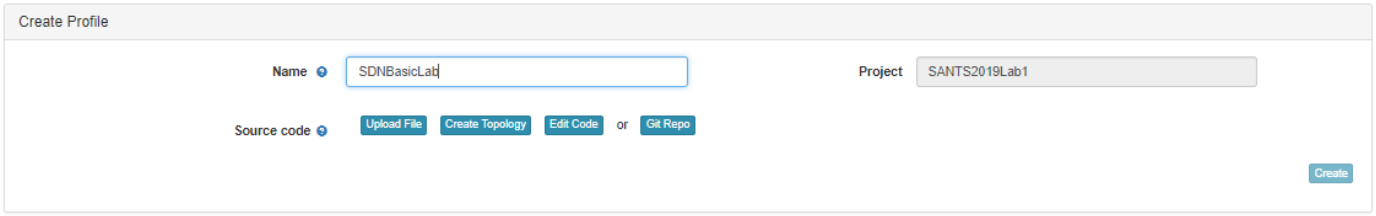
Steps:

Step1: Create Profile for SDN controller

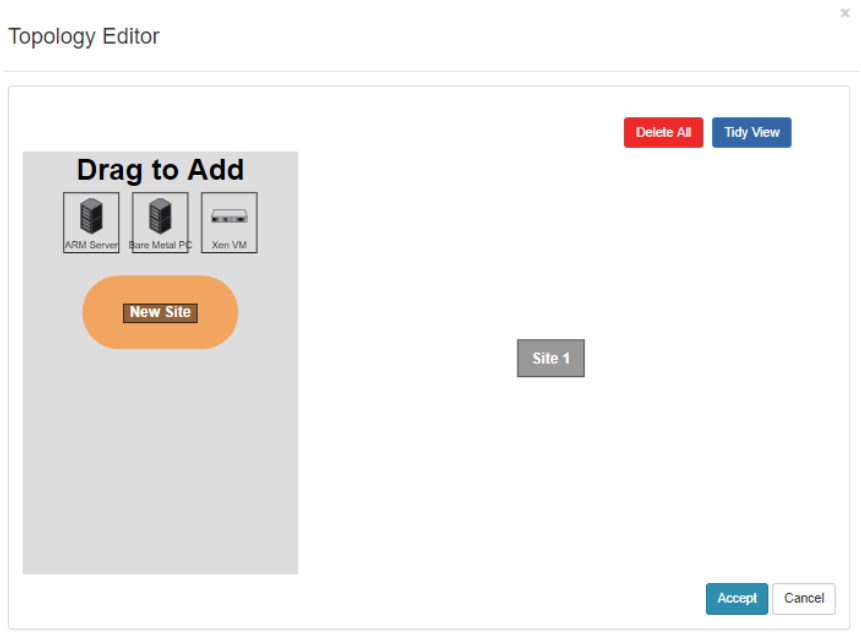
The profile is created as explained and the screenshot is as attached.
To create a profile the following steps are followed.

1. In the experiments tab, “Create Experiment Profile” is chosen.

When the dialogue box opens, the details are provided as shown below:

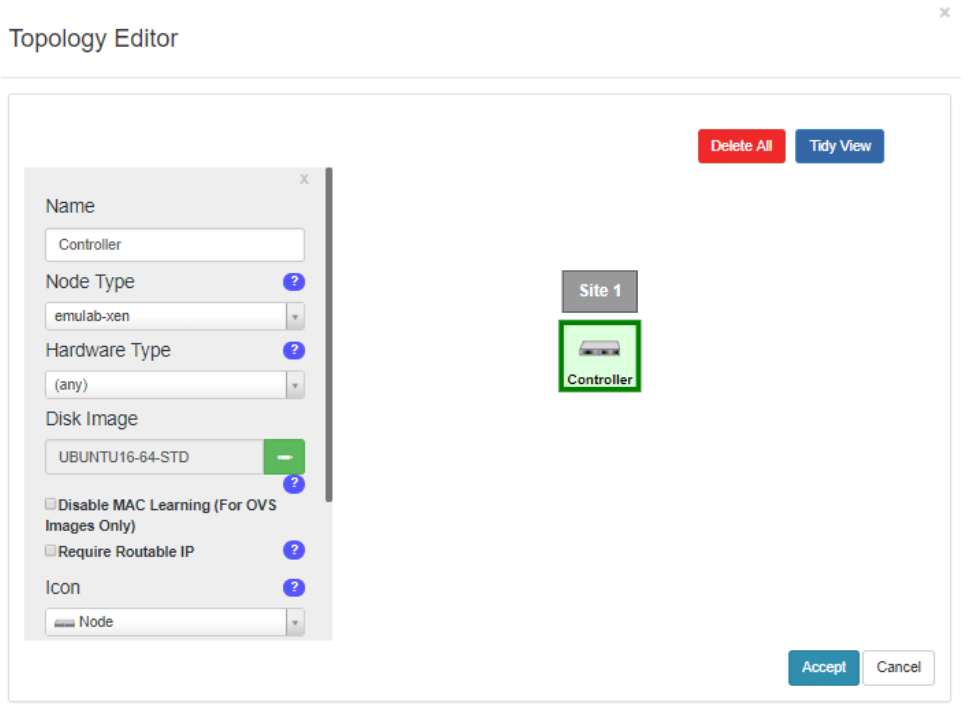


2. “create topology” is selected as we are not importing or editing from any other source. Topology editor opens as shown below:

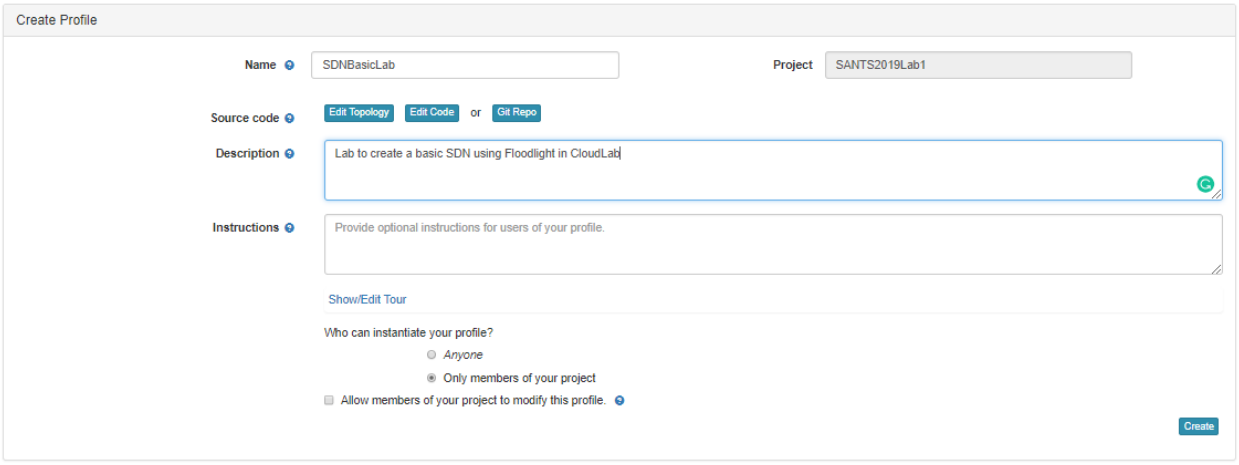


3. One Xen VMs is dropped on the editor and as shown below:

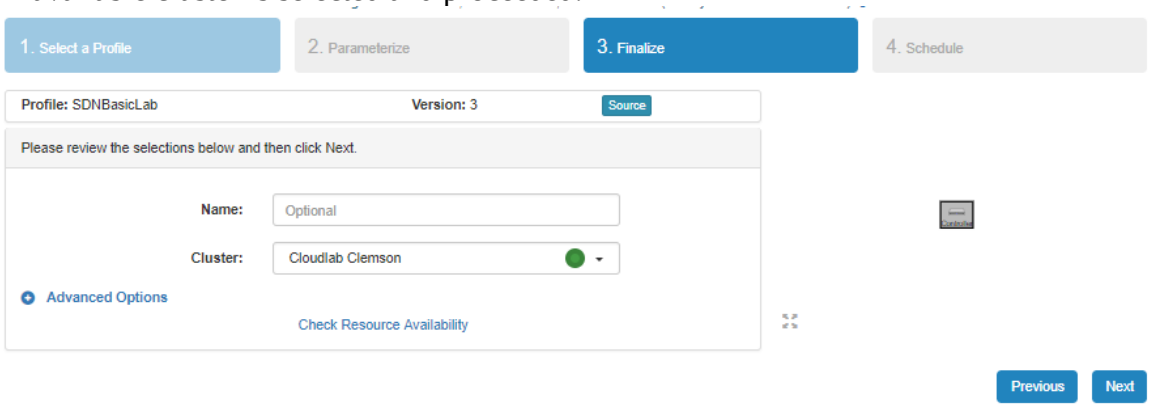
4. For the node, the details are given as below:
   d. Name: Controller
   e. Node Type: emulab-xen
   f. Hardware Type: (any)
   g. Disk Image: UBUNTU16-64-STD



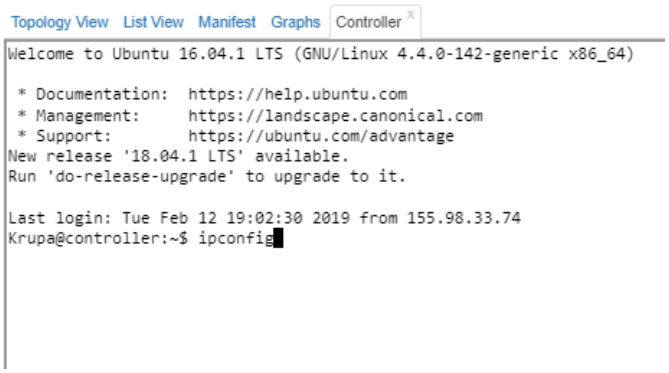
5. An appropriate description is provided. After everything is done the profile is created and then instantiated.

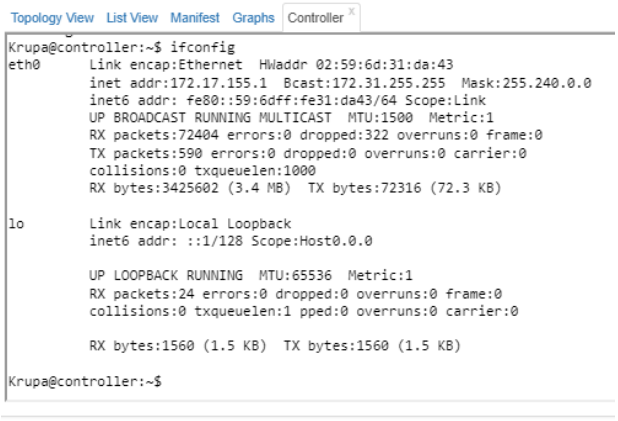


6. An available cluster is selected and proceeded.



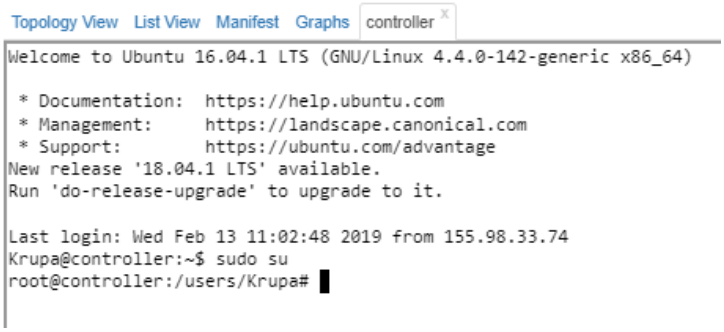
7. In the shell of the controller and command “ifconfig” is executed to find the ipaddress of the controller.



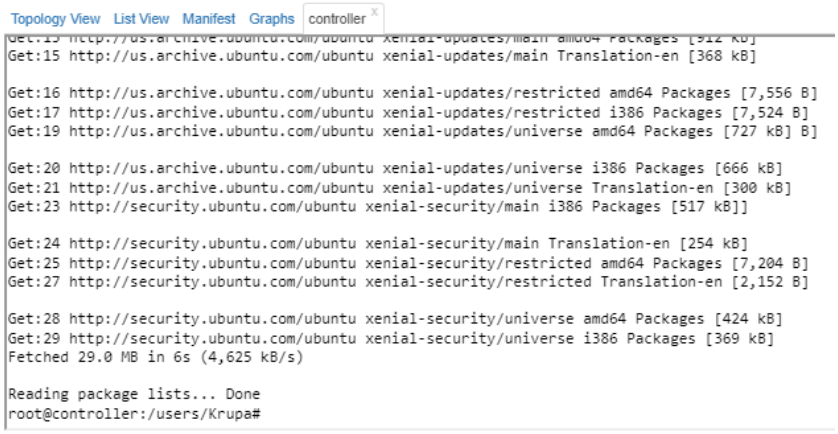


8. The following commands, as given in the manual are carried out. These install floodlight on the controller.

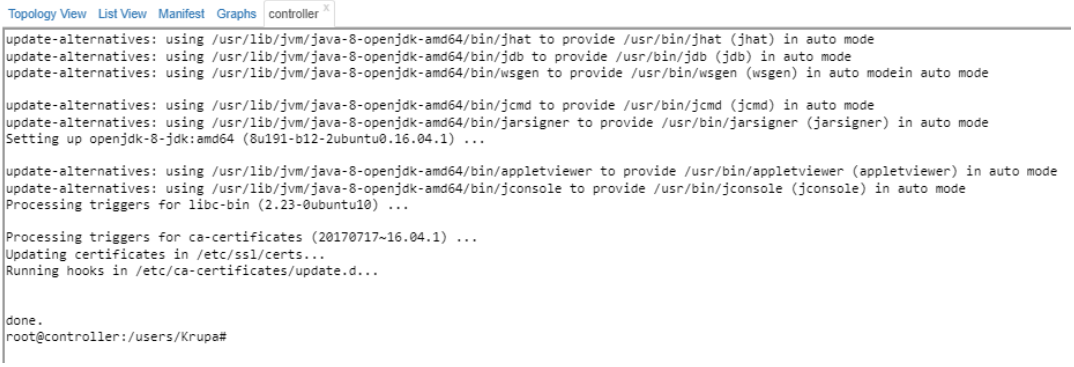
a. sudo su (gives the superuser privileges)



b. apt-get update (to download packages from their repositories and update them to the latest versions)



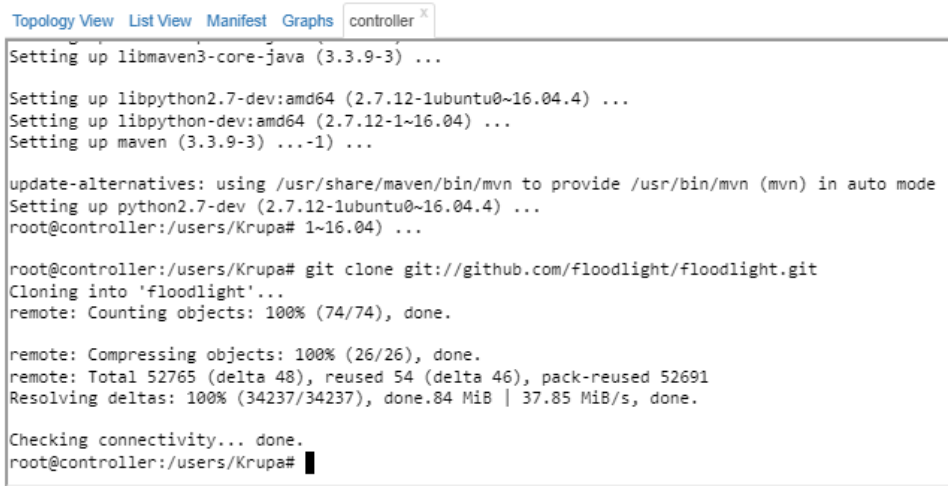
c. apt-get install default-jdk (installs default JDK)



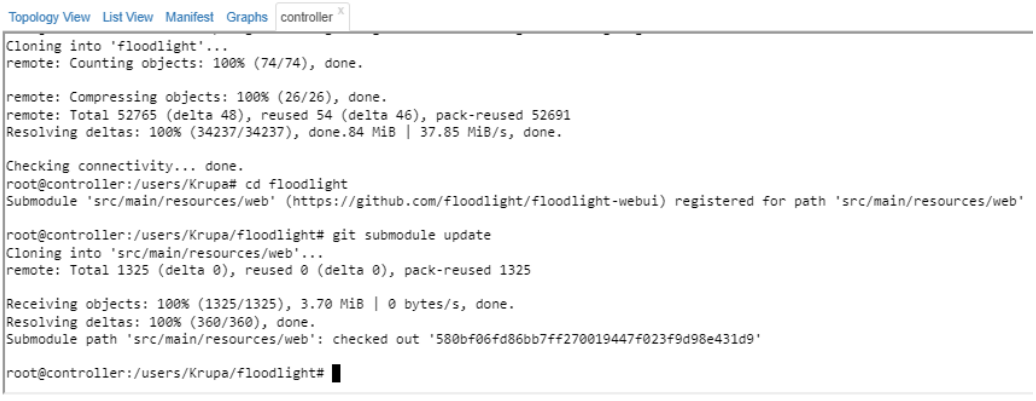
d. apt-get install default-jre (install default Java Runtime Environment)

e. apt-get install build-essential ant maven python-dev (to build essential)

f. git clone git://github.com/floodlight/floodlight.git (cloning repository)



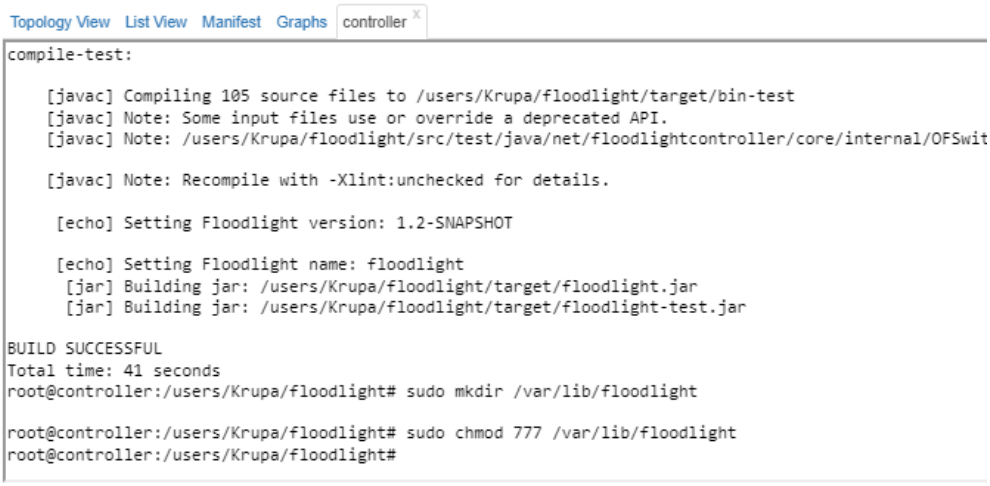
- cd floodlight
- git submodule init
- git submodule update



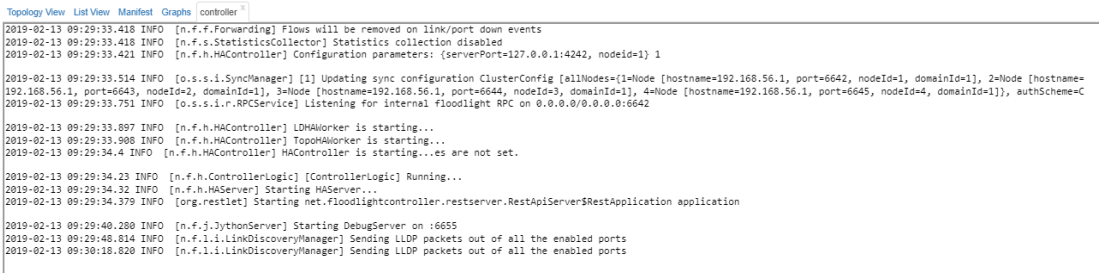
- ant

Note: Build may fail if package javafx.util does not exist, as was the case here. so package javafx.util does not exist is installed using “apt install openjfx”.

- sudo mkdir /var/lib/floodlight
- sudo chmod 777 /var/lib/floodlight



9. java -jar target/floodlight.jar (to start the controller to send packets)

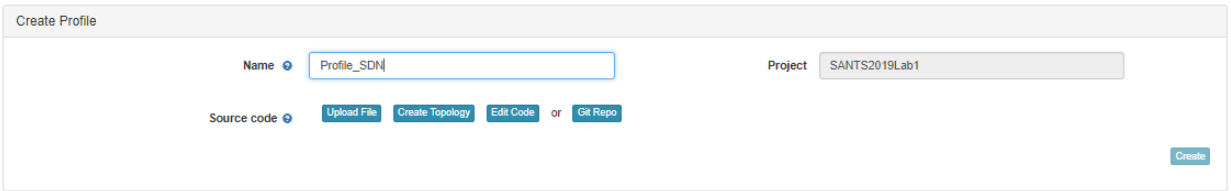


Now that the controller is ready, we will move ahead to creating a profile for the nodes.

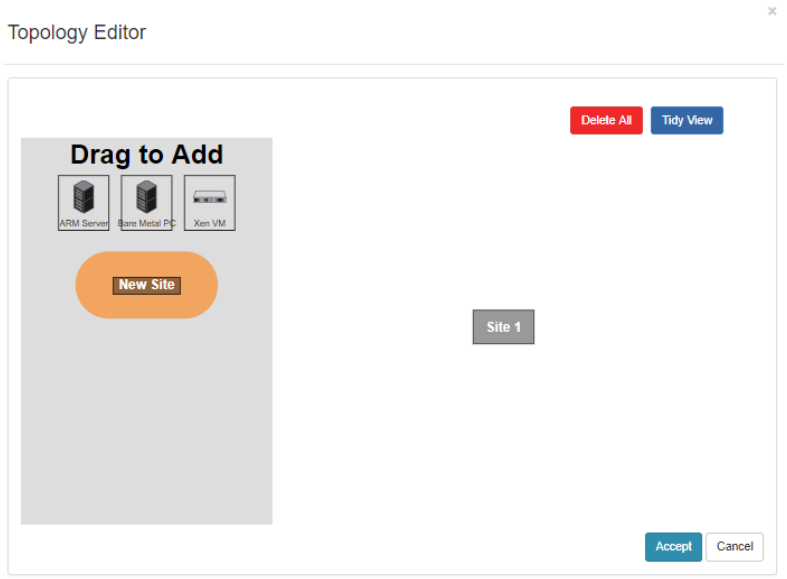
Step2: Create Profile for SDN controller

To create a profile the following steps are followed.

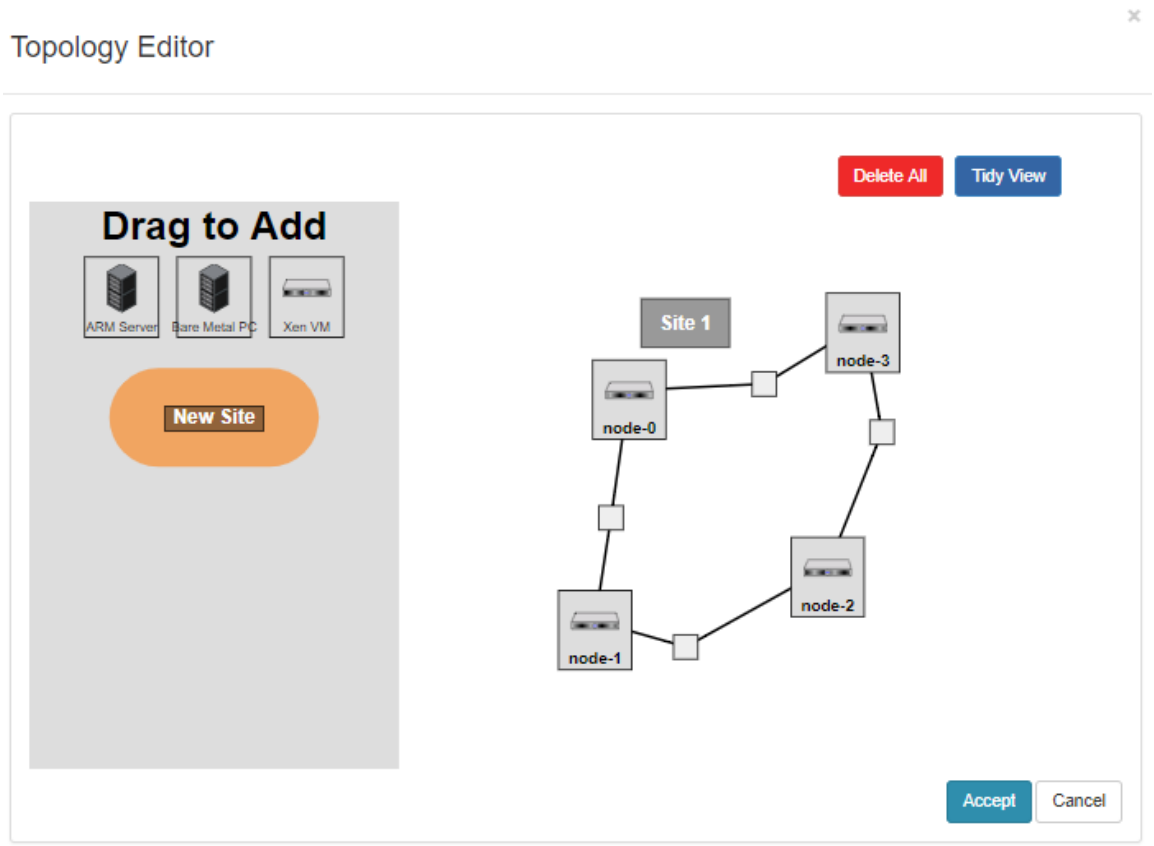
1. In the experiments tab,"Create Experiment Profile" is chosen.



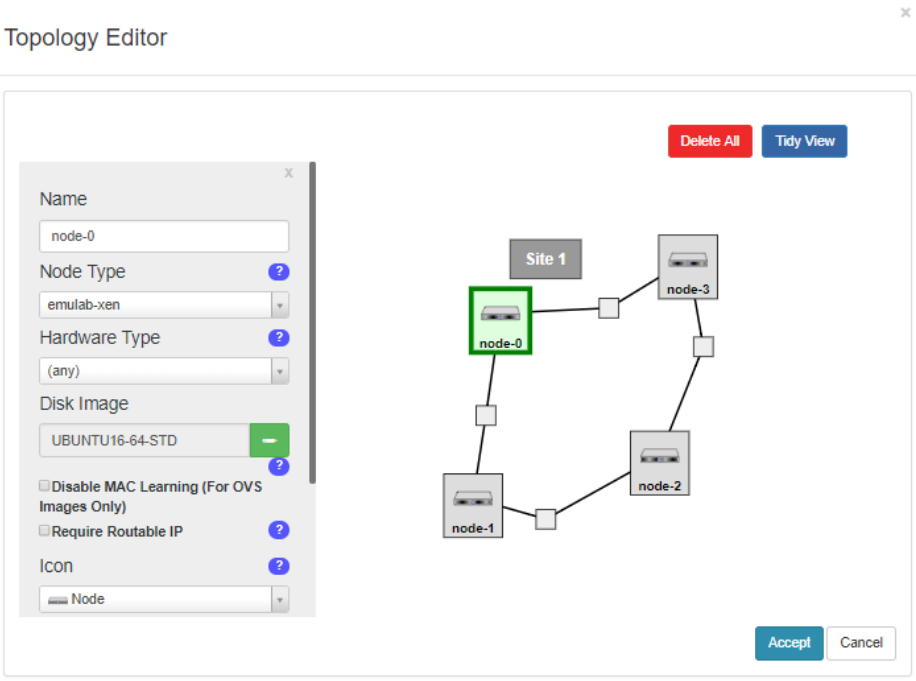
2. “create topology” is selected as we are not importing or editing from any other source. Topology editor opens as shown below:



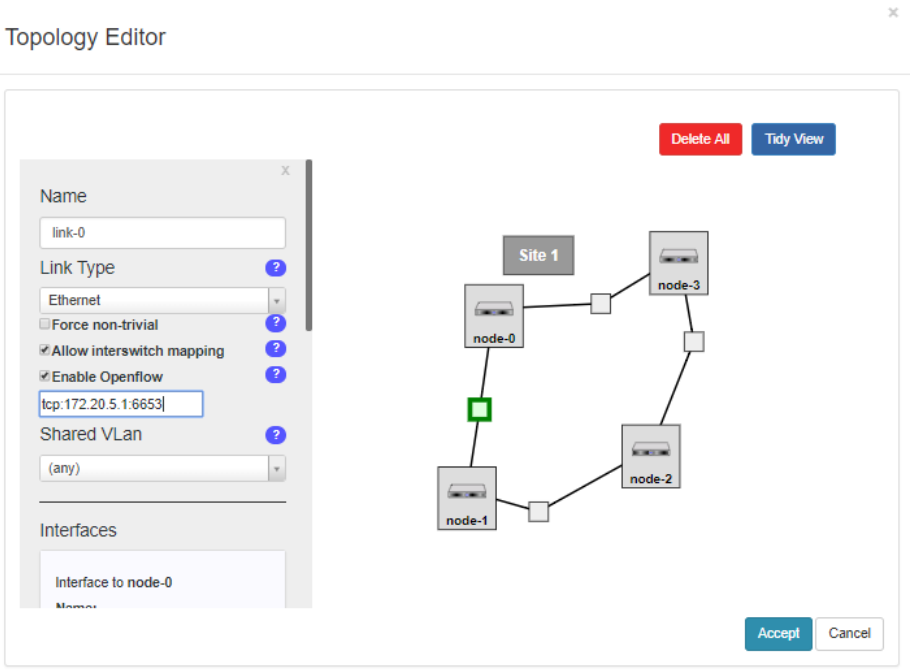
3. Four Xen VMs on imported on to the editor and connected as shown below. We see there will be 4 links connecting all the four nodes.



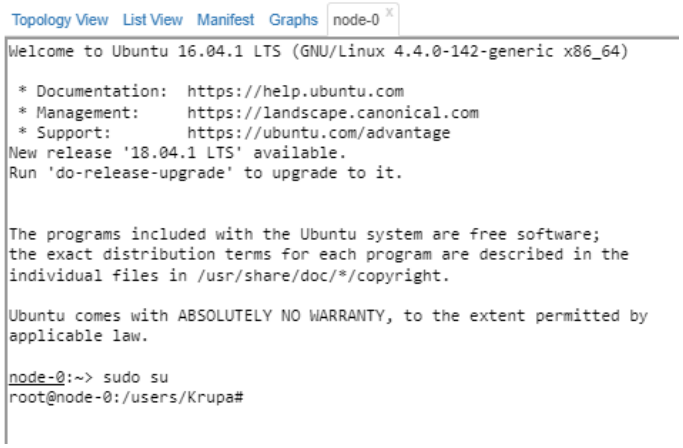
4. For each of the node, the following are selected:
   a. Node Type: emulab-xen
   b. Hardware Type: (any)
   c. Disk Image: UBUNTU16-64-STD



5. For each of the links, the following are selected and accepted:
   c. Link Type: Ethernet
   d. Select Allow interswitch mapping
   e. Select Enable OpenFlow and give IP address of the controller as shown. tcp:172.17.55.1:6653



6. An appropriate description is provided. After everything is done, the profile is created and then instantiated.
7. The process as before is followed to find a proper available cluster.
8. OpenVSwitch is installed and bridges on all nodes are setup using the following commands. These are used to connect to the SDN Controller. It is taken care to perform all these steps on all nodes with a unique bridge name and a unique IP address. The following commands are shown by taking node0 as example from the experiment.
   a. sudo su (for availing superuser access)

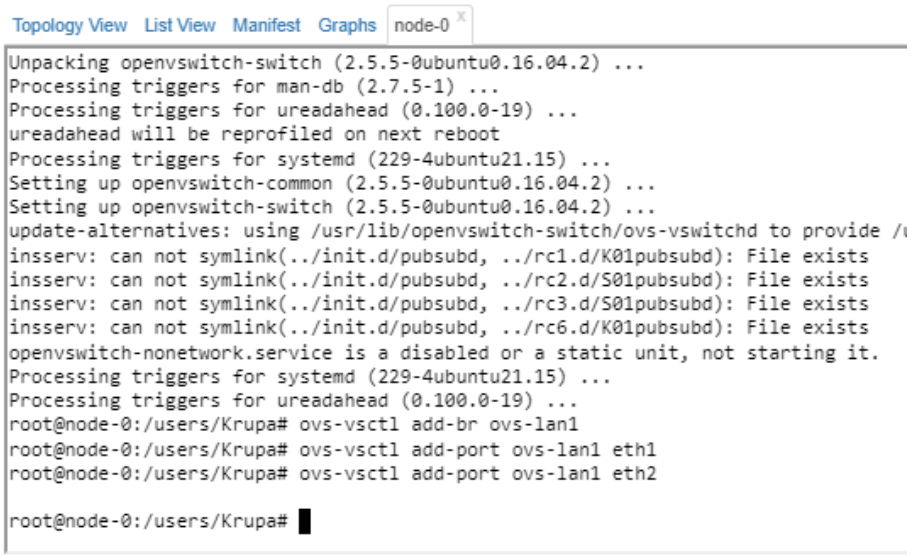


   b. apt-get update

   c. apt install openvswitch-switch

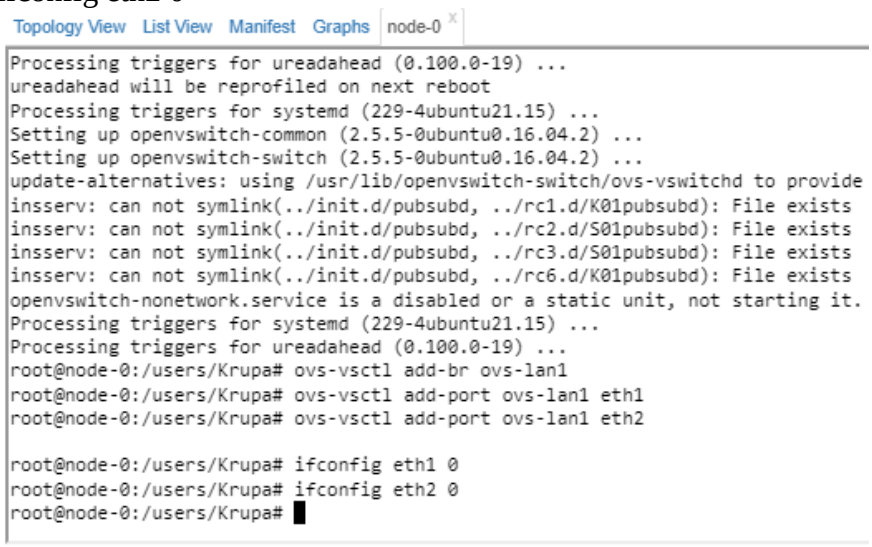
   d. ovs-vsctl add-br ovs-lan1 (to add a bridge)
   e. ovs-vsctl add-port ovs-lan1 eth1

f. ovs-vsctl add-port ovs-lan1 eth2

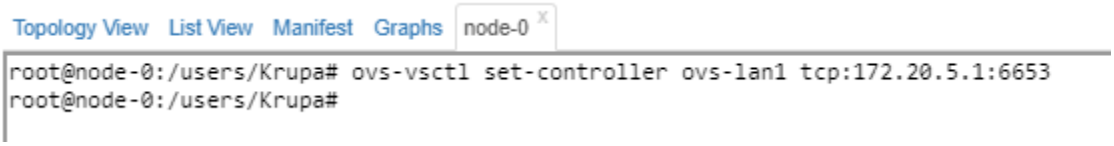


g. ifconfig eth1 0

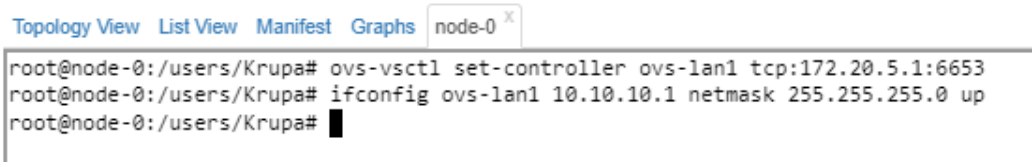
h. ifconfig eth2 0



i. ovs-vsctl set-controller ovs-lan4 tcp: tcp:172.20.5.1:6653



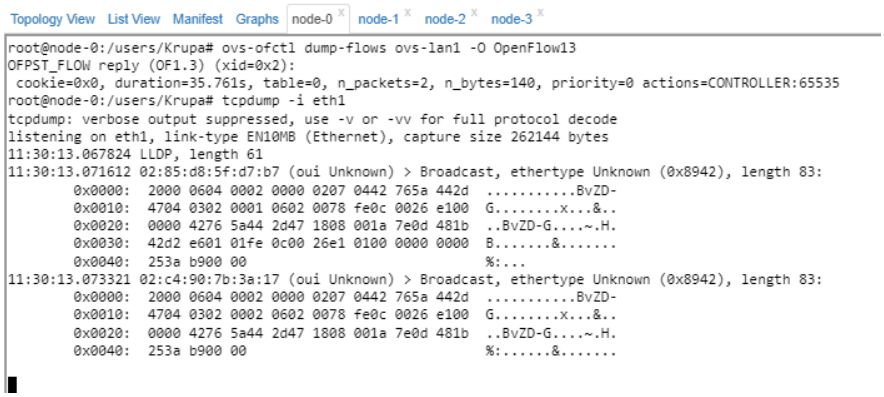
j. ifconfig ovs-lan1 10.10.10.1 netmask 255.255.255.0 up



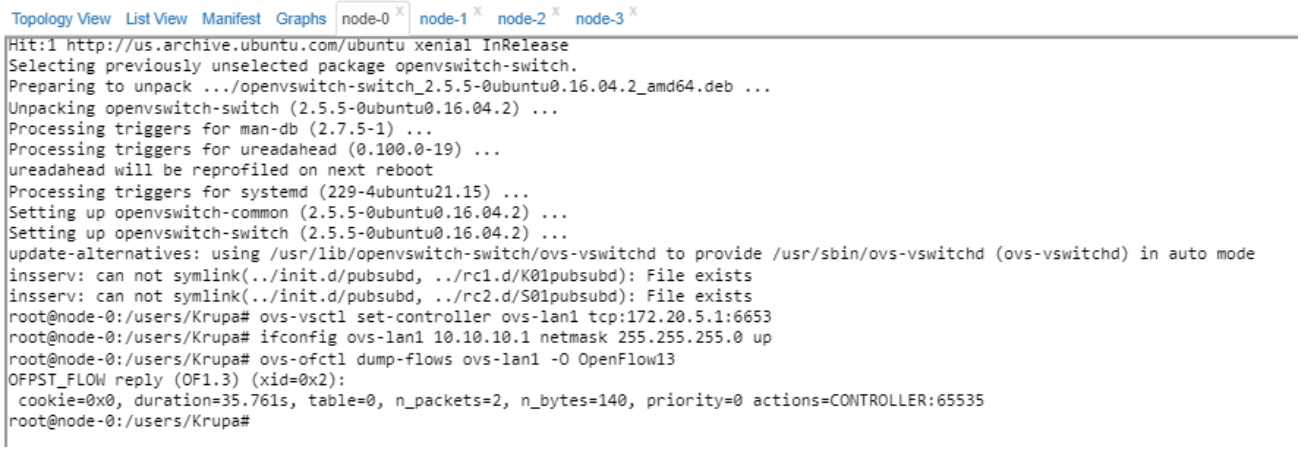
9. The steps from a-j are carried out on all nodes.

10. Ping test can be done using the command " tcpdump -I eth1". We may get an error saying –" tcpdump: eth1: That device doesn't support monitor mode"

This error can be solved by using “tcpdump -i eth1”



11. Finally, the flow rules are checked using the command - ovs-ofctl dump-flows <bridge_name> -O OpenFlow13.



Explanation of flow rules and tcpdump:

1. tcpdump -I eth1

   tcpdump is a packet analyser. When used, it shows the packed being sent or received in a network. It is also used to display TCP/IP address. -I denoted that’s eth1 is an interface, So in this command, basically tcpdump is used to monitor the TCP/IP address of the packets and also to analyse/monitor all the packets being transmitted over the network. It outputs all the packets to the interface eth1. Hence this shall help in problems related to packet flow.

2. Flow rules in step 6
   The flow rules given in step6 is completely dependent on “ovs-ofctl”. this command is used to monitor and created Openflow switches. This command can also be used to understand the current state of OVS. and also, this works for any available OpenFlow switch. So when“ ovs-ofctl dump-flows <bridge_name> -O OpenFlow13” is used, it showsthe status of the bridge (in our experiment bridge names are provided as lan1, lan2, lan3 and lan4 for node0, node1, node2 and node3 respectively) and helps controller keep a track of the packet flow in the form of flow rules or flow tables..