Experiment Task Design:

This lab aims to let students understand the basic SDN by using Floodlight in CloudLab. Based on the topology that students create in the first lab (Lab 1), students can push flow rules through Floodlight, one of open-source SDN controllers.

Submission:

Take screenshots of all the steps involved and explain in one or two paragraphs. Study the flow rules that are printed in step 6 and explain its meaning.

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
Conducting the labs

Step 1: Create Profile for SDN controller

Create a profile with a single node. Use ‘Ubuntu 16’ as the OS and node type as XEN VM. Accept the topology and then create. Give appropriate description and then instantiate. You will be asked to select a cluster. Select the one that is available and click finish. Once the experiment is online proceed to installation of floodlight.

**Figure 1: Setup a single node for SDN controller. Remember to check “Require Routable IP” so that other nodes can access the controller.**
Step 2: Install Floodlight

Open a shell window of the controller node. Use “ifconfig” and note down the IP address. We will need this IP address while setting up the topology of the experiment.

To install floodlight, run the following commands in sequential order:
- Get sudo user privileges: “sudo su”
- Update APT repo: “apt-get update”
- Install java 8: “apt-get install default-jdk” and “apt-get install default-jre”. This will install java 8 on Ubuntu 16.
- Install dependencies: “apt-get install build-essential ant maven python-dev”
- Install Floodlight:
  - git clone git://github.com/floodlight/floodlight.git -b v1.2
  - cd floodlight
  - git submodule init
  - git submodule update
  - ant
  - sudo mkdir /var/lib/floodlight
  - sudo chmod 777 /var/lib/floodlight
- Start the controller: “java -jar target/floodlight.jar”

Step 3: Setup profile for the experiment.

Create a profile with 4 Xen VMs. You can select UBUNTU16 as your choice of Operating System. Hardware type can be any. Node type will be emulab-xen. For the links select Link type as Ethernet. Give a name to the topology with a description. Once the topology has been created click on Accept and then create. In the next window instantiate the profile.

An important step is to select enable OpenFlow for all links and give appropriate SDN controller IP:Port. E.g. “tcp:128.110.99.138:6653”.

Topography Editor

Figure 2: Add nodes to your topology. In this case 4 XEN VMs.
Figure 3: Give details of each node by selecting Node type, Hardware type, and Disk image. Remember to check “Required Routable IP” so that they can connect to the SDN controller.

Figure 4: Provide the details of each link. Select ‘Ethernet’ as the ‘Link type’ for this lab. Also enable OpenFlow and provide appropriate ip address.

Step 4: Start Experiment:

To start an experiment, select an available cluster. For this lab select the one that is available. You can check the availability by hovering on the Green dot next to each cluster name. Click on finish when done. It will take some time for the profile to boot up. Once the experiment has started you can check connectivity using ping.
Step 5: Install OpenVSwitch and setup bridges on all nodes.

All the links in our topology are connected to the SDN controller (floodlight). To check what flows are pushed by the controller to route the traffic in the network we will setup a bridge on all nodes and connect them to Floodlight controller. Controller will then learn the new topology and send appropriate flow rules.
Install OpenVSwitch:
- “sudo apt-get update”
- “sudo apt-get install openvswitch-switch”

Use the following commands to setup a bridge on each node and connect it to SDN controller. These steps have to be performed on all nodes. Each bridge should get its own bridge name and a different IP address. For the 4 bridges you can use “ovs-lan1”, “ovs-lan2”, “ovs-lan3”, “ovs-lan4”. These are the bridge names. Give IP to each of them as “10.10.10.1”, “10.10.10.2”, “10.10.10.3”, “10.10.10.4”. In example below eth1 and eth2 are part of the LAN network.

```
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-controller <bridge_name> tcp:<controller_IP_Address>:6653
ifconfig <bridge_name> 10.10.10.1 netmask 255.255.255.0 up
```

Example:

```
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-controller ovs-lan1 tcp:128.110.99.138:6653
ifconfig ovs-lan1 10.10.10.1 netmask 255.255.255.0 up
```

Step 6: Ping and dump flows

Once all the configuration is finished, ping from node-0 to node-2 will start working. Students can use “tcpdump -i eth1” on node-1 and node-2 to check which path the ping takes. Also check the flow rules on all the 4 nodes using “ovs-ofctl dump-flows <bridge_name> -O OpenFlow13” by replacing <bridge_name> with appropriate bridge that you created on each node.

![Figure 5: Flow rule inserted by the SDN controller](image-url)
All the fields that are visible in the above flow entry are set by the Floodlight controller. API’s can be used to set these entries as well. The default entry inserted is to send the packet to the controller.

Appendix

OVS commands:

**ovs-vsctl**: Used for configuring 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 for monitoring and administering OpenFlow switches

Example:

To print the OVS flow rules “*ovs-ofctl dump-flows ovs-lan2 -O OpenFlow13*”

FAQ:

Q: Why I see unlimited ARP packets when I use TCPDUMP?
A: This is because all the nodes and switches make a cycle. Floodlight cannot handle the ARP requesting and replying in this situation gracefully. To solve this problem, you need to unplug one interface from the cycle. E.g., on node1, you use “*ovs-vsctl del-port eth1*”. Then wait a few seconds, plug the interface again using “*ovs-vsctl add-port ovs-lan1 eth1*”. 