Lab 11. SDN Application Development Lab (IP Blacklist Blocking App)

Contents

Experiment Task Design

This lab aims to let students understand the working of IP blacklisting in Floodlight controller. This lab uses Floodlight, one of the open-source SDN controllers. The Floodlight controller used for this lab has inbuilt blacklist IP module and the working of the module is explained.

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 to 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 16**' 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 Floodlight on the node.

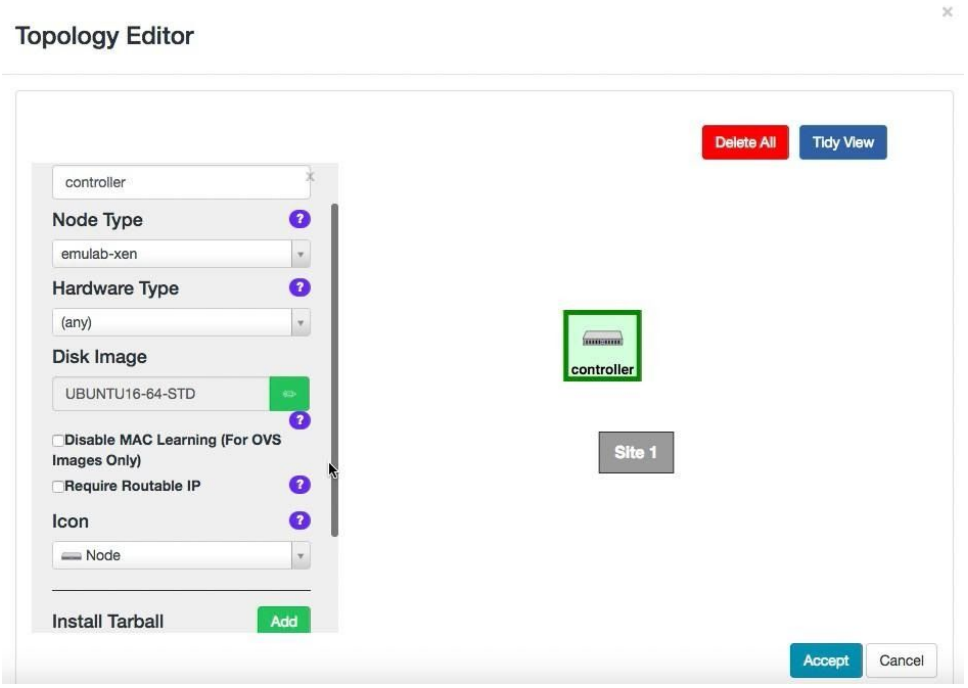

Figure 1: Setup a single node for an SDN controller. (Note that if you check "Require Routable IP," the controller can be assigned with a public IP address that other nodes can access. If not, the controller will have a private IP address.)

Installing Floodlight

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 being setup for an experiment.

3) To install Floodlight, perform the following steps:

Get sudo user privileges
sudo su
Update APT repo
apt-get update
Install Java
apt-get install default-jdk -y
apt-get install default-jre -y
apt install openjfx
Install dependencies
apt-get install -y build-essential ant maven python-dev
Install Floodlight:
git clone --recursive https://github.com/anc15791/floodlight
cd floodlight
ant
sudo mkdir /var/lib/floodlight
sudo chmod 777 /var/lib/floodlight

4) Start the controller: "**java -jar target/floodlight.jar**".

Setup a topology profile for the experiment

1) Drag and drop 3 nodes and create a profile with 3 Xen VMs as shown in the figure.

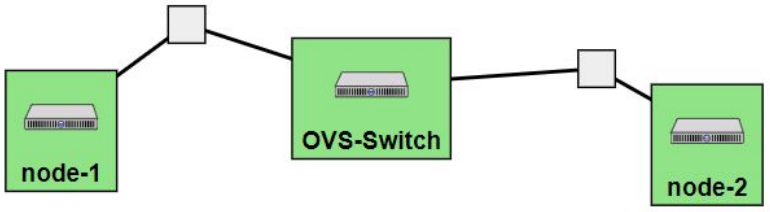

Fig 2: Topology

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

OS: Ubuntu 18-64-STD
Hardware Type: Any
Node Type: emulab-xen
Check "Require Routable IP" only for OVS-switch

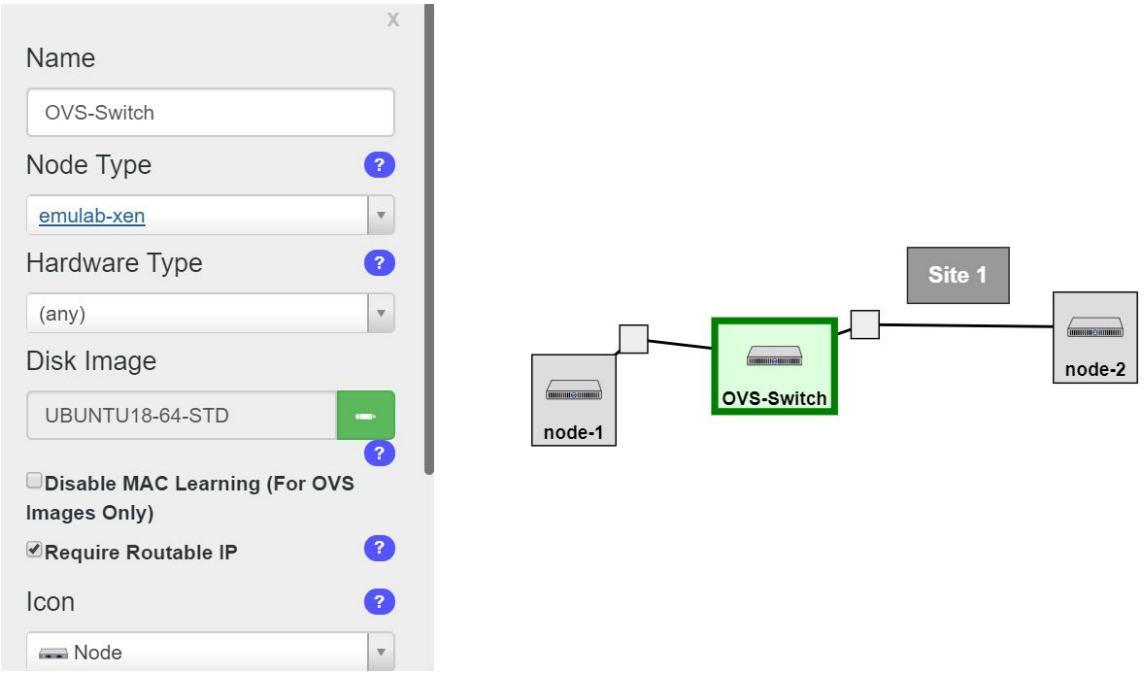

Fig 3: Node configuration

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:155.98.37.91:6653**". The IP address here is obtained from the SDN Controller (floodlight node).

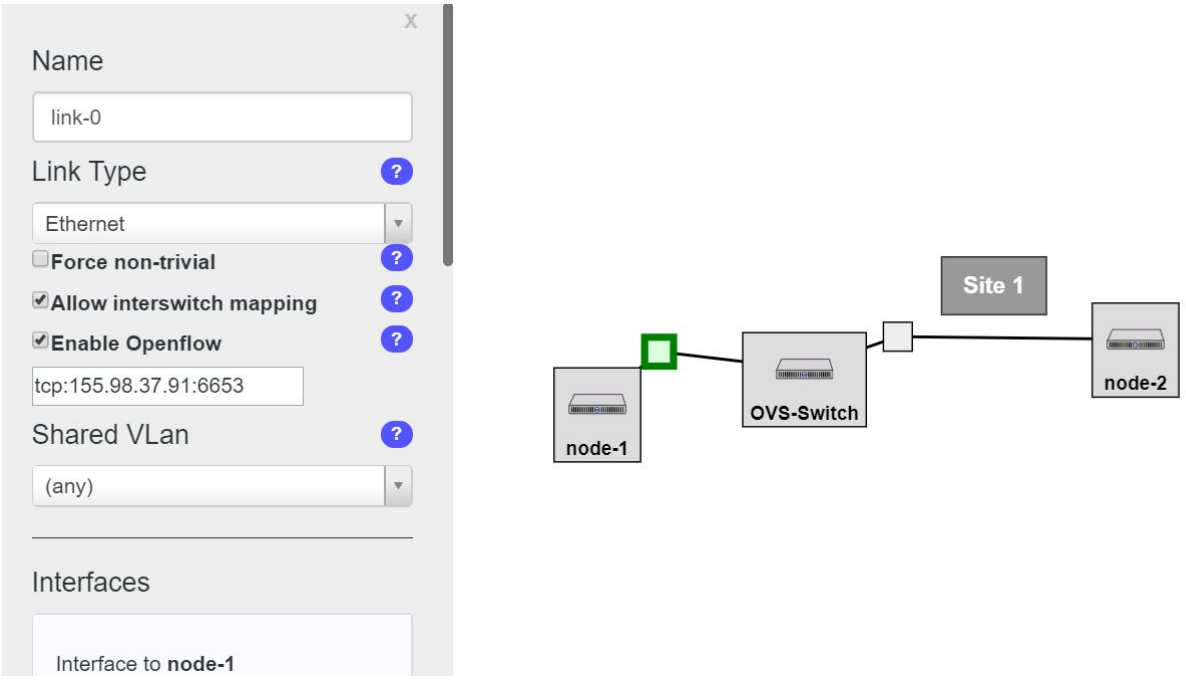

Figure 4: Provide the details of each link. Select 'Ethernet' for 'Link type'. Also select "Enable OpenFlow" and provide an appropriate IP address and the port number.

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

NOTE: The topology can be created by uploading the provided xml file (profile.xml) at **Create Experiment Profile** and the topology can be viewed by clicking **Edit Topology**

Conducting the Lab

Install Open vSwitch and setup bridges on OVS-switch nodes.

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

1) Run "**sudo apt-get update"** and **"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-controller <bridge_name> tcp:<controller_IP_Address>:6653
ifconfig <bridge_name> 10.10.10.1 netmask 255.255.255.0 up
```

The name of the bridge can be **ovs-lan**. The commands can be written as follows:

```
sudo su
ovs-vsctl add-br ovs-lan
ovs-vsctl add-port ovs-lan eth1
ovs-vsctl add-port ovs-lan eth2
ifconfig eth1 0
ifconfig eth2 0
ovs-vsctl set-controller ovs-lan tcp:155.98.37.91:6653
ifconfig ovs-lan 10.10.10.1 netmask 255.255.255.0 up
```

Change IP addresses of Node1 and Node2

Type the following commands on corresponding nodes to change their IP addresses.

Node1: **sudo ifconfig eth1 10.10.10.2/24 up**
Node2: **sudo ifconfig eth1 10.10.10.3/24 up**

Test the Connectivity between nodes

Ping Node2 from Node1 by running "**ping 10.10.10.3**"

Test IP Blacklist Functionality

Get command

This command prints all the blacklisted IP's.
Command: **curl -s http://localhost:8080/wm/ipblacklist/ipblacklist/json | python -mjson.tool**

```
root@node-0:~# curl -s http://localhost:8080/wm/ipblacklist/ipblacklist/json | python -mjson.tool
[
    "10.0.2.16",
    "10.0.2.15"
]
root@node-0:~#
```

Post command

This command is used to append any IP's to the list of blacklisted IP's.
Command: **curl -X POST -d '{"ip":"10.10.10.3"}' http://localhost:8080/wm/ipblacklist/ipblacklist/json**

```
root@node-0:~# curl -X POST -d '{"ip":"10.10.10.3"}' http://localhost:8080/wm/ipblacklist/ipblacklist/json
{"status" : "ip added", "ip" : "10.10.10.3"}root@node-0:~#
```

Test the connectivity between the nodes using the ping program and check the output on the controller console.

Delete command

This command is used to delete any IP address in the blacklisted IP's.
Command: **curl -X DELETE -d '{"ip":"10.10.10.3"}' http://localhost:8080/wm/ipblacklist/ipblacklist/json**

```
root@node-0:~# curl -X DELETE -d '{"ip":"10.10.10.3"}' http://localhost:8080/wm/ipblacklist/ipblacklist/json
{"status" : "ip deleted", "ip" : "10.10.10.3"}root@node-0:~#
```

Test the connectivity again and check the controller console.

Result

Throughout the changes done to the blacklist IPs, the output can be observed on the console when any IPs are added or deleted or viewed the blacklist. The response action can also be observed on the console output.

Below is the screenshot when Node2 IP address is not added to blacklist and packet is forwarded to the destination.

```
2020-05-17 18:34:19.452 INFO  [n.f.i.IPBlacklist] Destination IP: /10.10.10.3
2020-05-17 18:34:19.452 INFO  [n.f.i.IPBlacklist] Destination IP not in blacklist,forwarding packet.
2020-05-17 18:34:19.465 INFO  [n.f.i.IPBlacklist] Destination IP: /10.10.10.2
2020-05-17 18:34:19.466 INFO  [n.f.i.IPBlacklist] Destination IP not in blacklist,forwarding packet.
2020-05-17 18:34:32.17 INFO  [n.f.l.i.LinkDiscoveryManager] Sending LLDP packets out of all the enabled ports
```

Below is the screenshot when Node3 IP address is added as a blacklist IP and the packets are dropped.

```
2020-05-17 18:36:18.798 INFO  [n.f.i.IPBlacklist] reading file: /users/priganta/floodlight/src/main/java/net/floodlightcontroller/ipblacklist/blacklist.
txt
2020-05-17 18:36:18.802 INFO  [n.f.i.IPBlacklist] reading file: /users/priganta/floodlight/src/main/java/net/floodlightcontroller/ipblacklist/blacklist.
txt
2020-05-17 18:36:32.173 INFO  [n.f.l.i.LinkDiscoveryManager] Sending LLDP packets out of all the enabled ports
2020-05-17 18:36:44.400 INFO  [n.f.i.IPBlacklist] Destination IP: /10.10.10.3
2020-05-17 18:36:44.402 INFO  [n.f.i.IPBlacklist] black list ip matched, dropping: /10.10.10.3
2020-05-17 18:36:44.407 INFO  [n.f.i.IPBlacklist] Setting actions
2020-05-17 18:36:44.411 INFO  [n.f.i.IPBlacklist] Dropping
2020-05-17 18:36:45.420 INFO  [n.f.i.IPBlacklist] Destination IP: /10.10.10.3
2020-05-17 18:36:45.421 INFO  [n.f.i.IPBlacklist] black list ip matched, dropping: /10.10.10.3
2020-05-17 18:36:45.422 INFO  [n.f.i.IPBlacklist] Setting actions
2020-05-17 18:36:45.422 INFO  [n.f.i.IPBlacklist] Dropping
2020-05-17 18:36:46.444 INFO  [n.f.i.IPBlacklist] Destination IP: /10.10.10.3
2020-05-17 18:36:46.447 INFO  [n.f.i.IPBlacklist] black list ip matched, dropping: /10.10.10.3
2020-05-17 18:36:46.448 INFO  [n.f.i.IPBlacklist] Setting actions
2020-05-17 18:36:46.448 INFO  [n.f.i.IPBlacklist] Dropping
```

The messages when the list of Blacklist IPs is accessed can be found as "reading file". The action taken on the packet is "dropping".

Working

We have created a simple IP blacklist module in floodlight. It will check if destination IP is one of the blacklist, if true then set a drop flow else forward the packet.

We will have following files:

1. IPBlacklist.java: This class implements the IFloodlightModule and IOFMessageListener which are required to create a module and listen to the OpenFlow messages in floodlight. We will override several methods that are required for writing a module and remaining we will leave empty which are required to create a REST service and API.
2. FlowMgr.java: This class will keep all methods to perform dropping, forwarding, flooding actions. For this module, we will only need to add the dropping method.
3. BlacklistMgr.java: This class will have all methods to read and write into the blacklist file. For this module, we only need to read from a file and send a list back to calling method.
4. Blacklist.txt: This file has the list of IP's we want to blacklist
5. Other classes like "ipblacklistWebRoutable" creates a restlet for our module to expose its REST API.
6. "ipblacklistSerializer" is used for serializing/de-serializing json objects.
7. "ipblacklistResource" creates the API handlers for GET, POST, DELETE methods.

All these files will be inside the ipblacklist package in "src/main/java/net/floodlightcontroller" directory.
To write a module and understand the code follow the floodlight tutorial at
"https://floodlight.atlassian.net/wiki/spaces/floodlightcontroller/pages/1343513/How+to+Write+a+Module "
IP blacklist Code repository is at:
https://github.com/anc15791/floodlight/tree/master/src/main/java/net/floodlightcontroller/ipblacklist

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**"