CNT 4713 – Project 3

Overview:

The goal of this project is to build a simulation environment for experimenting with software-defined networks (SDN). The environment includes (i) a Mininet Virtual Machine, for simulating a network with switches and hosts; and (ii) a FloodLight controller, running on the host machine.

Detailed instructions for setting up the experiment environment:

Step 1: download and install the necessary software

1.1 Download & install Virtualbox

1.2 Download the Mininet VM image from this web-page:
https://github.com/mininet/openflow-tutorial/wiki/Installing-Required-Software
Choose "Virtual Machine Image (OVF format, 64-bit, Mininet 2.2.0)" or "Virtual Machine Image (OVF format, 32-bit, Mininet 2.2.0)" based on your machine and OS.

1.3 Depending on your OS, you may need to install other software listed on the above web-page, i.e., *Xming* and *Putty* for Windows, and *XQuartz* for Mac.
Download the compiled Floodlight controller V 1.0 from Moodle. Alternatively you can download the source code here (not compiled): http://floodlight-download.projectfloodlight.org/files/floodlight-source-1.0.zip

1.4 Download Eclipse from here:
https://eclipse.org
Note: you will need Java JRE in order to run Eclipse. If Java is not currently installed, you will need to download and install it first.

Step 2: Setup and start Mininet VM

2.1 Import the Mininet VM image into Virtualbox. Follow the instructions here:
http://www.brianlinkletter.com/set-up-mininet/
Starting from "Import the virtual machine into VirtualBox" , all the way to the end.
Note: if you are on Windows, you may want to also take a look at the "Windows" section of the following page:
https://github.com/mininet/openflow-tutorial/wiki/Set-up-Virtual-Machine

Step 3: Setup and start Floodlight

3.1 Build floodlight (Note: you will NOT need to do this if you use the pre-compiled Floodlight posted on Moodle. You will need to do this if you use the source code.)

cd floodlight-1.0

(Note: floodlight-1.0 is the directory of Floodlight source code, assuming you have not renamed it.)

ant eclipse

(note: to run the above command, make sure you have *apache ant* on your system. If not, simply download the binary from here: http://ant.apache.org/bindownload.cgi
The binary should work out of box for Linux and MAC systems. No installation is needed. Simply replace the above command with:
<path-prefix>/apache-ant-1.9.4/bin/ant eclipse)

Alternative: a pre-compiled copy of Floodlight is available in Moodle.

3.2 Import the Floodlight project (downloaded from Moodle or built from source code in the previous step) into Eclipse. Follow these instructions:

- ***./eclipse &*** from the command line *or* find the icon to click to launch it Eclipse.
- Go to Preference-> Java -> compiler, and make sure the *Compiler compliance level* option is set at **1.7**. Note that neither 1.8 nor 1.6 (or lower) settings will work for this project.
- File -> Import -> General -> Existing Projects into Workspace. Then click "Next".
- From "Select root directory" click "Browse". Select the floodlight-1.0 directory.
- Check the box for "Floodlight". No other Projects should be present and none should be selected.
- Click Finish.

3.3 Create the FloodlightLaunch target and run (inside Eclipse):

- Note: make sure you have Java JDK 1.7 or higher installed on your system. For MAC, you may need to manually select the right Java JRE by going to Prefernce -> Java -> Installed JREs and making sure you have selected a version of 1.7 or higher. The default MAC Java version is 1.6 which will not work for this project.
- Click Run->Run Configurations
- Right Click Java Application->New
- For Name use FloodlightLaunch
- For Project use Floodlight
- For Main use net.floodlightcontroller.core.Main
- Click Apply
- Run

Step 4: Construct a network in Mininet, and connect the network to Floodlight

4.1 Start Mininet & Floodlight

4.2 SSH into Mininet, and do the following:

sudo mn --topo single,3 --mac --switch ovsk --controller remote,ip=192.168.56.1,port=6653

This will create the following topology in Mininet: a single switch and three hosts all connecting to the switch. This will also connect the switch to the Floodlight controller. Note that the specified IP in the above command corresponds to the IP of the host machine in the "host-only" network; and that the specified port number is the port that the Floodlight is listening on, which is displayed in the Floodlight output. You might need to change them based on your actual setup.

To learn more about the command line arguments, simply do ***sudo mn –h***

4.3 open a browser and type the following URL to see a live view of the network: http://192.168.56.1:8080/ui/index.html

Note that the IP specified is again IP of the host machine in the "host-only" network. The port should always be 8080.

Step 5: Explore Mininet

The following walkthrough is a good guide for exploring Mininet
http://mininet.org/walkthrough/

Step 6: Learn about the REST API

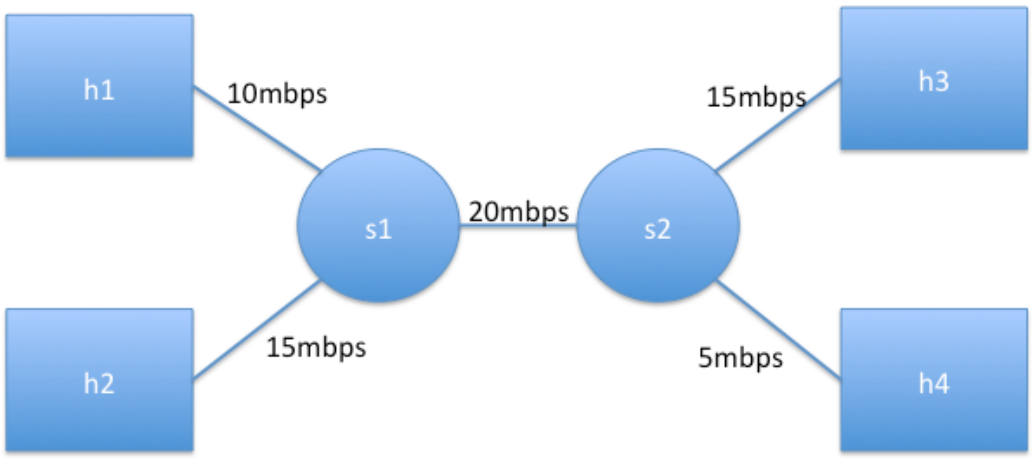
Read the list of REST APIs here:
http://www.openflowhub.org/display/floodlightcontroller/Floodlight+REST+API
The particularly important one for this project is Firewall API:
http://www.openflowhub.org/display/floodlightcontroller/Firewall+REST+API

References:

- Mininet: http://mininet.org
- Floodlight: http://www.projectfloodlight.org/floodlight/
- Iperf: https://iperf.fr/

Exercise

1. Create the following topology in Mininet: (Note: s1 and s2 are switches, h1, h2, h3 and h4 are hosts. **The bandwidth of each link is also given and must be implemented.**) Take a screenshot of the Mininet output that shows the topology is correctly set up.



(Hint: you will need to create a simple python file to do this. Refer to the walkthrough in Step 5 above, look for the "*Custom Topologies*" section to see an example Python script. You can modify the example script to create the topology shown above. You will also need to include the "bw=X" setting when adding links in the script, where X is a number such as 20, 15, 10 or 5. Finally, when starting Mininet using the "sudo mn… " command, you will need to add "--link=tc" to enforce the bandwidth setting in the resulting network. tc stands for "traffic control".)

2. Connecting the above Mininet topology to Floodlight. Test the connectivity among the hosts, using the **pingall** command. Include screenshot of the output.
3. *Immediately After* finishing the pingall test, take a screenshot of the network topology as displayed by Floodlight (refer to Step 4.3).
4. Now use **iperf** to confirm the bandwidth between h1 and h4, between h2 and h3; between h1 and h2, and between h3 and h4. Include the screenshots of all the iperf output.
5. Using the REST Firewall API, implement the following access-control policy: the communication between h1 and h3, and between h2 and h4, should be blocked; communication between any other pairs of hosts are allowed. Use **pingall** again to verify and include the screenshot of the output.

Submission Guide:

Please submit all of the following:

1. a txt, doc or pdf file that describes your experience in following the Instructions in this handout for setting up the experiment environment, particularly any

problem that you have encountered and how you resolved them. Be sure to detail any issue that you could not resolve.

2. The python file that you used to create the topology for the exercise.
3. All the screenshots you have been asked to take in the Exercise section. Make sure the resolution is high enough so that the screenshots can be easily read!
4. The REST Firewall API commands that you used to implement the access-control policy in the exercise.

Grading Criteria:

Item	Percentage
Set up and run Mininet with the specified custom topology	20%
Implement connectivity as demonstrated by pingall output	20%
Floodlight correctly illustrates the topology	20%
Correctly implement the link bandwidth, as demonstrated by iperf output	20%
Correctly implement the firewall policy	20%

Plagiarism will be reported to the university for academic dishonesty.