Workshop in Communication Networks

Exercise 0 – Get to Know The Tools

General Guidelines:

Submission deadline is **Sunday, March 9, 23:55 (moodle server time).**
Submit your answers as PDF and Python files, packed into a single ZIP file.
Pack your files in a ZIP file named ex0-YourName1-YourName2.zip,
for example: ex0-LoisLane-ClarkKent.zip
Post the ZIP file in the submission page in course website

- No late submission will be accepted!
- Document your code. Place your names at the top of every source file, as well as in the PDF with the answers.

There is a question at the end of each part in this exercise (except for part 1). They are framed in a black line. You should answer these questions and submit them in the PDF file, along with the final code you write in this exercise. This is a trivial exercise that. You must submit it with thoughtful answers in order to pass the course.

Part 1: Install Mininet

Download the latest mininet VM from here:
https://bitbucket.org/mininet/mininet-vm-images/downloads
(version 2.1.0 from September 20, 2013 – either i386 or amd64 according to the computer you use)

Follow the instructions from the lecture to install a virtual machine hypervisor and setup the Mininet VM in it. (The default username and password of the Mininet VM is: mininet)

If you do this on a Windows host machine, install another VM with a fresh installation of Linux Ubuntu, to easily control the Mininet VM, and make sure to configure NAT networking between the two VMs.

Find the IP address of each VM using **ifconfig**.

Part 2: Play With Mininet (take 1)

A.
In the management machine (either the Ubuntu VM or your native linux machine), open two terminal windows (In mac, open a terminal window and an xQuartz window).

In one of the windows, copy the **of_lecture.py** file from course website to the VM:
(from the directory where this file is stored)
scp of_lecture.py mininet@<Mininet VM IP address>:~/pox/pox/samples

In each one of the terminal windows, connect to the Mininet VM as follows:
ssh -XY mininet@<Mininet VM IP address>
One window will be used for running the controller (POX), and the other for running Mininet.
(Mac users – use the xQuartz window for running Mininet)

Run POX on the sample class:
(In the POX terminal window)
./pox.py log.level --DEBUG samples.of_lecture

Run Mininet:
(In the Mininet terminal window)
sudo mn --topo single,3 --mac --switch ovsk --controller remote

(This will run Mininet on a topology of a single switch (named s1) and three hosts (named h1, h2, h3) connected to it.

B.
In the Mininet terminal window, type:
xterm h1 h2 h3
This will open three xterm windows, each one is a terminal for each host in our network.

In the xterms of h2 and h3, type:
tcpdump -XX -i h?-eth0
(Replace ? with 2 or 3 according to the host number)

In the xterm of h1, type:
ping –c 1 h2
Wait for it to complete.

Look at the xterm windows of h2 and h3, do you see traffic there? Do you get a response for the ping in h1? ***Explain.***

Part 3 – Using Wireshark

Wireshark is installed on the Mininet VM by default. To run it, open another terminal window (in Mac: xQuartz window) and **ssh** to it with X forwarding as explained before.

Type:
sudo wireshark &

Wireshark will run. Ignore the warnings and go to Capture -> Interfaces
Select the **any** interface and click **Start**.

In the xterm window of h1, type:
ping -c 1 h2
Wait for it to complete.

Stop the wireshark capture (Capture -> Stop).

In the Filter field in the toolbar, write: **of.pktin** and click **Apply**. This filters the captured packets only to OpenFlow PacketIn packets.

How many PacketIn packets do you see? Look at the “Frame Data” of the PacketIn packets (select a packet, expand “OpenFlow Protocol”, expand “Packet In”, then expand “Frame Data”). What type of packets was sent to controller? ***Explain.***

Part 4 – Make the Controller Control (take 1 – controller see all packets)

A. Look at **of_lecture.py**. Edit the code in method **act_like_hub(…)** so that when a packet arrives, it tells the switch to act like hub and flood it. That is, as directed in the comments, uncomment the line that starts with: *self.send_packet(…)*
You may take a look at the code of **send_packet** method. It will be helpful later.

B. Restart Mininet and POX. Reopen xterm windows for h1, h2, h3 and rerun tcpdump for h2, h3.

C. Start a new capture in Wireshark just as you did before.

D. In the xterm window of h1, type:
ping -c 4 h2
Wait for it to complete.

E. Stop the wireshark capture.

F. If the filter is not already set, set the display filter field to **of.pktin** and click **Apply**, just as we did in the previous part.

How many PacketIn packets do you see now? What are the types of packets that were sent to controller? Why these packets were sent to controller? What do you see in the xterm windows of h2, h3? Is there a difference? Why? ***Explain.***

Part 5 – Make the Controller Control (take 2 – switches act like hubs)

The line you uncommented in **of_lecture.py** caused the controller to order switch(es) to flood every packet they send to it. Since they send all packets (as they have no rules installed), they flood all packets.

We will now make the controller install a flood rule for each input port, such that only the first packet on this port will get to controller, and then it will install a rule on the switch to flood such packets in future.

Installing the rule in OpenFlow is done by sending an **ofp_flow_mod** message to the switch.
The example method **send_flow_mod_by_in_port(…)** creates such a message and sends it to the controller.

A. Implement **add_flood_rule_to_flowtable(…)** so that it will send a flood rule to the switch using the given sample method **send_flow_mod_by_in_port**. Follow instructions in the comments.

B. Comment out the line you uncommented in part 4 and uncomment the last line in **act_like_hub**. It will now use your implementation of **add_flood_rule_to_flowtable**.

C. Restart Mininet and POX. Reopen xterm windows for h1, h2, h3 and rerun tcpdump for h2, h3.

D. Start a new capture in Wireshark just as you did before.

E. In the xterm window of h1, type:
ping -c 4 h2
Wait for it to complete.

F. Stop the wireshark capture.

G. If the filter is not already set, set the display filter field to **of.pktin** and click **Apply**.

How many PacketIn packets do you see now? What are the types of packets that were sent to controller? Is there a difference from what you saw in part 4? Why? What do you see in the xterm windows of h2, h3? Is there a difference? Why? ***Explain.***

Submit your updated of_lecture.py along with the PDF file with your answers.