HS2020: 11072 Advanced Networking and Future Internet
Practical Exercise 2

Lucas Pacheco
lucas.pacheco@inf.unibe.ch

October 19, 2020

0 Documentation

Don't hesitate to contact if you have questions about the assignments. However, please refer to the documentations first to see if your question can be found in the there:

- Mininet Docs (http://mininet.org/walkthrough/)
- Ryu Docs (https://book.ryu-sdn.org/en/Ryubook.pdf)
- Openflow Tutorial (https://github.com/mininet/openflow-tutorial/wiki)

1 Instructions

This exercise has the goal of introducing you to SDN concepts and programmable networks in real life. During this, you are going to deploy different network topologies and integrate your network with SDN controllers.

In the controllers, you will run and modify the flow rules, analyze the flow tables for the switches in the network, and write your own rules.

1.1 VM Configuration

The experiment will require the following tools:

1. Virtualbox (https://www.virtualbox.org/wiki/Downloads)
2. Mininet (Included in the VM)
3. Ryu SDN controller (Included in the VM)

For the exercise, you will be provided an Ubuntu VM in which to run the experiments. If you have an ubuntu installation, you may install Mininet and Ryu directly. However, that is under your own responsibility.

You can download the VM in the link https://bit.ly/37fPIGa. After downloading, extract the VM from the zip file and import it into VirtualBox.

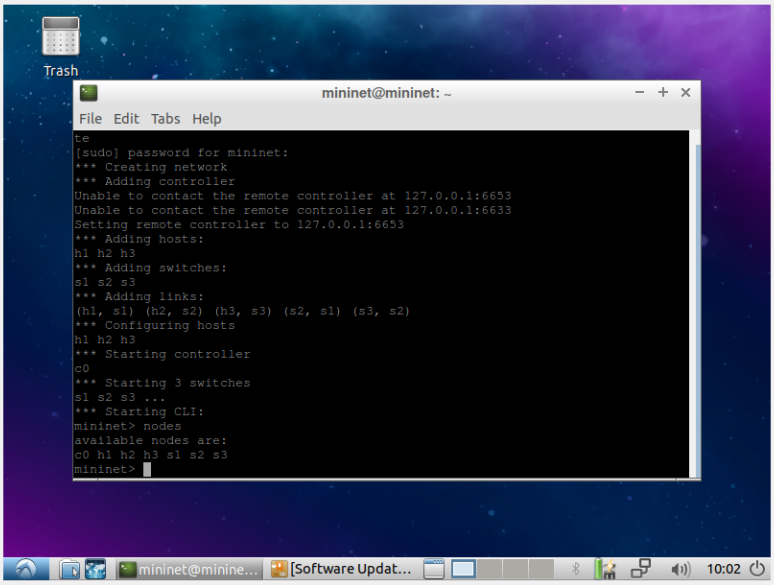
1.2 Directories Listing

1. start up the VM and log in with the credentials user:mininet password:mininet
2. open a terminal window and execute the command (if asked for a passwork, use "mininet"):

```
sudo mn --topo linear,3 --controller remote --switch ovsk --mac
```

3. In the home directory execute "ls", notice that a directory named "ryu" is present, that is the SDN controller. When required to run it, navigate into this directory with "cd ryu/"

The output for the command should look like the following image:



From this, you can see all the nodes created in the topology (hX are the hosts, cX are the SDN controllers, sX are the SDN-enabled switches). Notice

that the program listens for the controller in port 6653. This is because we specified a remote controller in the command. Let's run the topology once more with a controller enabled.

Kill the process with Ctrl+D, run the command "sudo mn -c" for cleanup, and open a new terminal window (you should have two windows open).

On the second window, go into the ryu directory and execute the example program for this class with the command:

```
python3 bin/ryu-manager ryu/app/anfi.py
```

Now run again the pingall test, were the nodes reachables for each other?

1.3 Program Explanation

Recall from your class. The SDN architecture often assumes the presence of a Northbound API and a Southbound API. The controller assumed the role of interconnecting both to enable the programmable networks. At the northbound API, we have applications written in human-readable programming languages. The controller interprets such programs (such as the ryu-manager) and transmits OpenFlow commands at the Southbound API to OpenFlow-enabled switches.

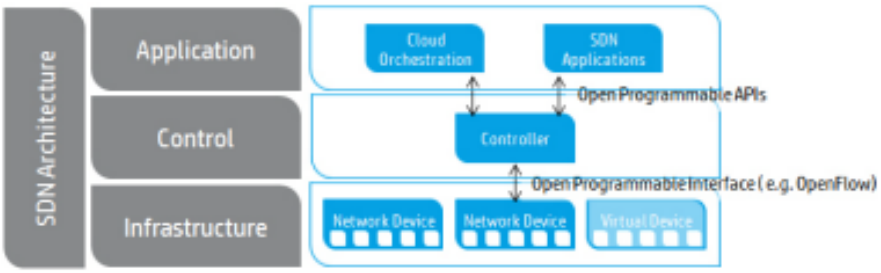


Figure 1: Source: https://sreeninet.wordpress.com/2014/08/03/sdn-commercial-applications/

The program provided consists of a simple switching application. Its functioning is as follows:

1. Packet is received by the switch (event)
2. If the flow is not present in the table for the switch, add flow.
3. If the flow is present, a control message does not need to be added. It is handled in the control plane.

1.4 Your Tasks :)

1. Start the following mininet topologies, and run the "pingall" command on the Mininet CLI in each case.

   ```
   sudo mn --topo linear,3 --mac --controller remote --switch ovsk
   sudo mn --topo single,3 --mac --controller remote --switch ovsk
   sudo mn --topo tree,3 --mac --controller remote --switch ovsk
   ```

   How many of the nodes were reachable by each other?

   Now, before running each topology, navigate to the ryu directory in another terminal ("cd ryu/") and start up the controller with the command:

   ```
   python3 bin/ryu-manager ryu/app/anfi.py
   ```

   How many of the nodes were now reachable?

2. Print the flow table for a switch in the network before and after the pingall command is run (you need the controller running as well). Can you explain the new flows added in terms of the packets sent in the network?

   Tip: print the table with the command, change s3 for the swicth you're analyzing.

   ```
   sudo ovs-ofclt -O openflow13 dump-flows s3
   ```

3. In the anfi.py program, which tells the controller how to deal with the incoming flows, add a rule to drop the packets incoming from host h1 (in any of the topologies), include the program in your submission. **Notice that dropping a packet is the default action for an openflow switch, and only if a flow is explicitly added the data plane forwards a packet. Check the Ryu docs chapter seven for more information on flow matching**.

2 Submission

Submit the assignment on ILIAS via a ZIP file through an upload task on the practical exercises directory. Deadline is until 31/10/2020 at 13:55. The ZIP should include:

- Modified anfi.py file.
- PDf with the textual answers to the tasks and descriptions (if you have screenshots and images include in the PDF for organization).

3 Appendix: Topologies

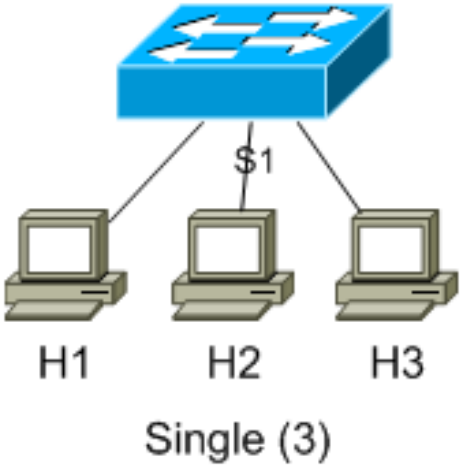


Figure 2: single,3 Topology

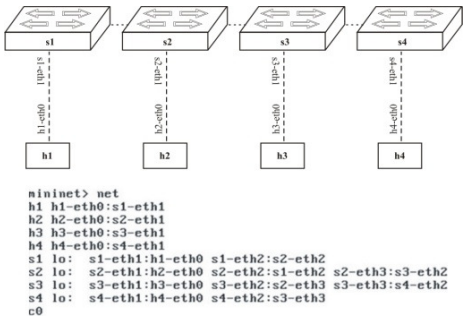


Figure 3: linear,4 Topology

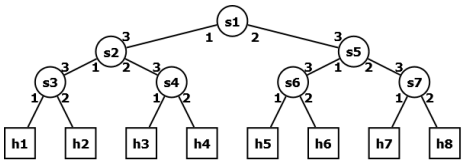


Figure 4: tree,3 Topology