Advanced Computer Networks (2020)

Lab1: Learning Switch

Type: Individual
Maximum points: 5
Submission: Quizzes on Canvas (no code submission needed)
Deadline: Monday November 9, 2020
Contact: vu.acn.ta@gmail.com

The goals of this lab are:

- Get familiar with the Ryu network controller
- Implement a learning switch with Ryu
- Answer questions on Canvas

1. Get familiar with the Ryu network controller

Ryu is a component-based network control framework[1]. Ryu supports various protocols for managing network devices, such as OpenFlow. We can develop network applications with the Ryu controller and emulate inside Mininet. The Ryu controller should be pre-installed in the course VM already. Now, let us test the Ryu controller with a simple switch function.

First, open a terminal and run Mininet with the following command. This starts a network emulation environment with 1 switch and 3 hosts.

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

Second, open a new terminal and run a simple switch controller. This controller will simply flood a packet to all ports except the one that received the packet if the switching table has no record of the destination MAC address and add a record which specifies the source MAC address at the in-port (recall how an L2 learning switch works).

```
$ ryu-manager ryu.app.simple_switch
```

[1] Ryu is one of the network controllers designed for software defined networking (SDN). We will explain how it works in detail when we cover the SDN concept in later lectures.

After running the Ryu controller, you can test the reachability between these three hosts.

```
mininet> pingall
```

You should be able to see packet-in messages at the terminal for the Ryu controller. Note that when you specify a default remote controller for Mininet, the controller is expected to run at `127.0.0.1:6653`. This is handled by Ryu automatically. To verify this, you can first start the Ryu controller and then start Mininet, you can see the following message in the Mininet setup messages:

```
Connecting to remote controller at 127.0.0.1:6653
```

2. Implement a MAC learning switch with Ryu

A learning switch has the following basic functions:

- Learns the MAC address of the host connected to a port and retains it in the forwarding table
- When receiving packets addressed to a host already learned, transfers them to the port connected to the host
- When receiving packets addressed to an unknown host, performs flooding

Now let's use Ryu to implement such a switch and test it in Mininet. You can check the Ryu APIs at: https://ryu.readthedocs.io/en/latest/developing.html. You may use the code template provided by us ("`learning_switch.py`"). The template already includes the basic structure of the controller function. The parts missing is the `packet_in` handler.

You can basically follow the steps below:

- Obtain the datapath ID so you can identify which switch you are dealing with
- Parse the packet into Ethernet frames and obtain the `src` and `dst` MAC addresses
- Record the `src:in_port` so you learn the port for a future destination MAC
- Look for the `dst` MAC in the forwarding table
    - If the `dst` MAC address is found, you obtain the `out_port` from the forwarding table and insert a flow entry `[in_port:dst, out_port]` to the switch
    - Else, you flood the packet to all the ports except the one on which the packet was received

Once you have completed the learning switch, test it with the following commands:

```
$ ryu-manager learning_switch.py
```

In another terminal window, you launch the emulated network with the topology you constructed in Lab0, i.e., `network_bridge.py` here which specifies the bridge topology.

```
$ sh run.sh
```

Test if the learning switch is working as expected:

```
mininet> pingall
```

To check the flow entries that have been installed on the switch, you can use the following command:

```
$ sudo ovs-ofctl dump-flows switch_id
```

where `switch_id` is the name of the switch you want to check, e.g., `s1`.

3. Differentiate ARP from IP

In the above implementation, the switch learns on all packets including the ARP requests and responses. In this exercise, let us differentiate ARP from IP packets, e.g., the switch will not learn any MAC-port mapping from ARP requests/responses. Think about how to handle ARP requests/responses and how to implement it in your learning switch.

Note: The Canvas quizzes for lab1 will be based on this implementation.

4. Answer quizzes on Canvas

After completing the learning switch code and testing it out, please complete the quizzes on Canvas. There are 5 quiz questions each worth **1 point**. You should have your code open and running while filling out the quiz, since there might be questions that require you to modify your code and make observations in the outcome. For this lab, there is no need to submit your code.