Software-Defined Networks Lab

Experiment 2 - Using Open vSwitch as an OpenFlow switch for SDN Research

Release Date: March 27, 2018

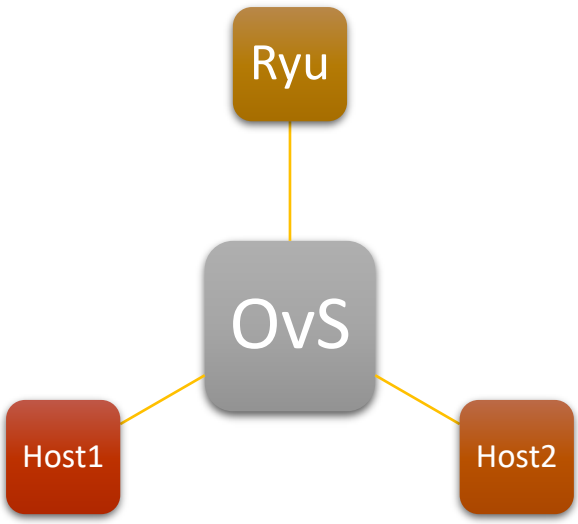
Objective:

In this experiment, you will learn:

- How to use Open vSwitch to emulate an OpenFlow switch by a PC with multiple network interface cards (NICs)

Environment:

1. There will be totally four hosts. One for Open vSwitch, another for controller, and the other two hosts act as traffic sender and receiver.



2. We will use Ryu as our SDN controller in this experiment.
3. Open vSwitch LTS series: openvswitch-2.5.4 is recommended, but you can still use other newer version.
4. We will be using OpenFlow 1.3.

Lab 1 - Build the Environment

Note:

- **It's strongly recommended to run the experiments in virtual machine, e.g., Oracle VM VirtualBox[1], or VMware Workstation[2].**
- **Make sure you have Internet connection.**
- **Make sure you have super user permission.**

In the example, we will use Ubuntu 16.04[3] as our Operating System, but you can also choose other Linux OS.

(A) Setting up Open vSwitch

Open vSwitch is a production quality, multilayer virtual switch, it is designed to enable massive network automation through programmatic extension, while still supporting standard management interfaces and protocols.
Official website: https://www.openvswitch.org/

1. Prepare your building environment.

```
$ sudo apt-get install git autoconf automake libtool
```

2. Get Open vSwitch project using `git` command or using `wget` command. In the example, we will use openvswitch-2.5.4 version.

```
$ git clone -b v2.5.4 https://github.com/openvswitch/ovs.git
```

Or

```
$ wget -qO- http://openvswitch.org/releases/openvswitch-2.5.4.tar.gz | tar xz
```

3. Go to the OvS directory and build Open vSwitch.

```
$ ./boot.sh

$ ./configure

$ make && sudo make install
```

4. Load the kernel module you need.

```
$ sudo /sbin/modprobe openvswitch
```

[1] https://www.virtualbox.org/
[2] https://www.vmware.com/
[3] http://releases.ubuntu.com/16.04/

5. Initialize the configuration database.

```
$ sudo mkdir -p /usr/local/etc/openvswitch

$ sudo ovsdb-tool create /usr/local/etc/openvswitch/conf.db
vswitchd/vswitch.ovsschema
```

6. Run up the daemon (make sure you have loaded the kernel module).

```
$ sudo ovsdb-server \

  --remote=punix:/usr/local/var/run/openvswitch/db.sock \

  --remote=db:Open_vSwitch,Open_vSwitch,manager_options \

  --private-key=db:Open_vSwitch,SSL,private_key \

  --certificate=db:Open_vSwitch,SSL,certificate \

  --bootstrap-ca-cert=db:Open_vSwitch,SSL,ca_cert \

  --pidfile --detach

$ sudo ovs-vsctl --no-wait init

$ sudo ovs-vswitchd --pidfile --detach
```

7. Now your OvS environment should be up and running.

(B) Setting up Ryu Controller

Ryu is another SDN controller written in Python. You can visit their website for more details and tutorial:

- https://osrg.github.io/ryu/
- http://ryu.readthedocs.io/en/latest/getting_started.html

1. Install dependency

```
$ sudo apt-get install python-pip
```

2. Download and install Ryu controller

```
$ git clone git://github.com/osrg/ryu.git

$ cd ryu; pip install .
```

3. Run Ryu web UI controller

```
$ PYTHONPATH=. ./bin/ryu --observe-links \
  ryu/app/gui_topology/gui_topology.py \
  ryu/app/simple_switch_websocket_13.py
```

Or simply

```
$ ryu-manager --observe-links \
  ryu/app/gui_topology/gui_topology.py \
  ryu/app/simple_switch_websocket_13.py
```

4. Now you can see the Ryu web UI as shown below from the web browser:

```
localhost:8080
```

(C) Create a simple topology

1. You need to set up two hosts that connect to OvS.
2. The link setup is shown below:
   a. Create three network adapters on OvS:
      i. One is used to communicate with the controller.
      ii. The others are used to connect to the two hosts.
   b. You should use different LAN segments for different hosts.

For example, the following setup can achieve the requirement:

The Ryu controller uses NAT network to connect to OvS.

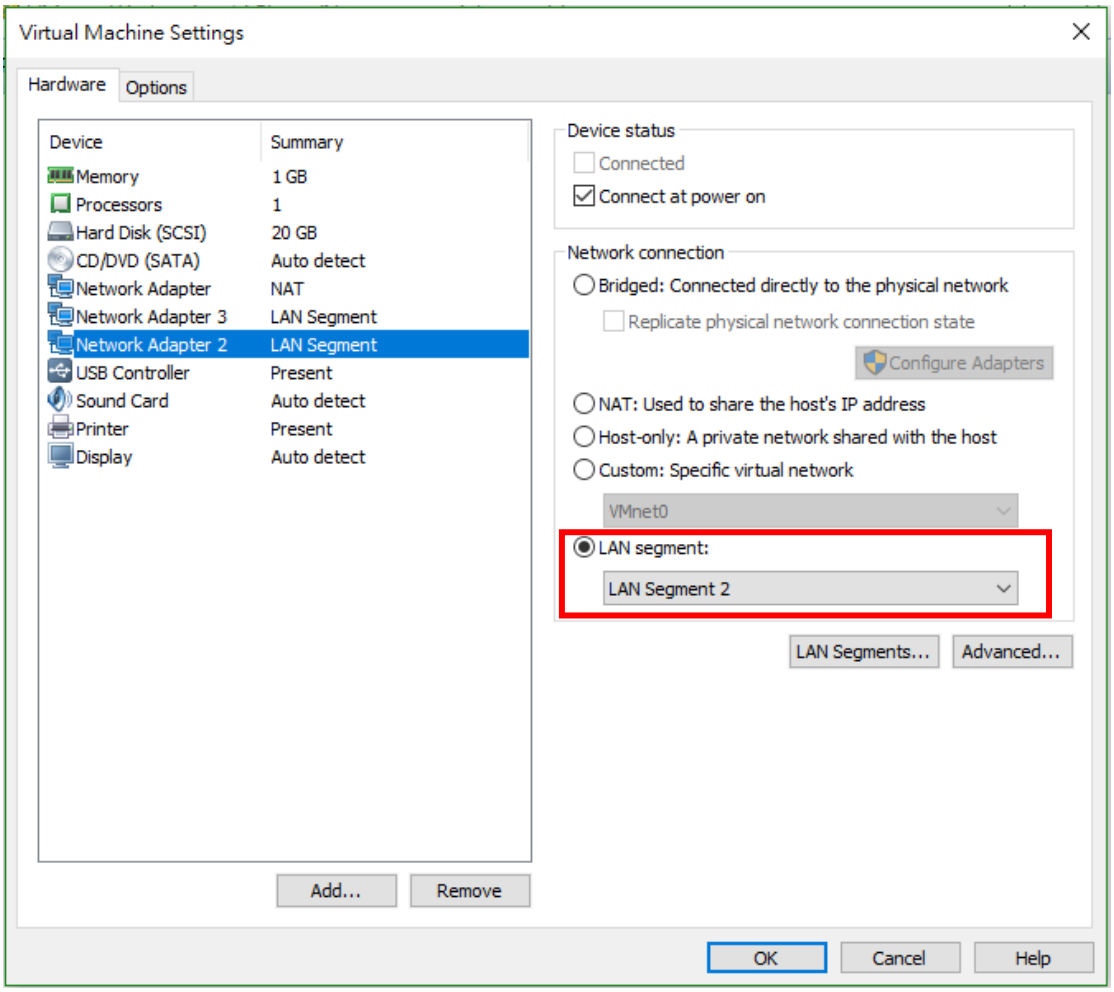
Host1 uses LAN segment 1 to connect to Network Adaptor 2 on OvS.

Host2 uses LAN segment 2 to connect to Network Adaptor 3 on OvS.

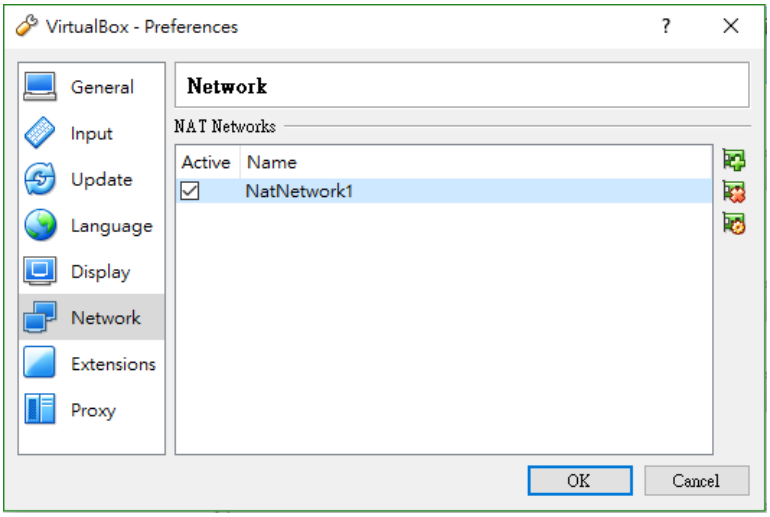
NOTICE that MAC address should be different for each host.

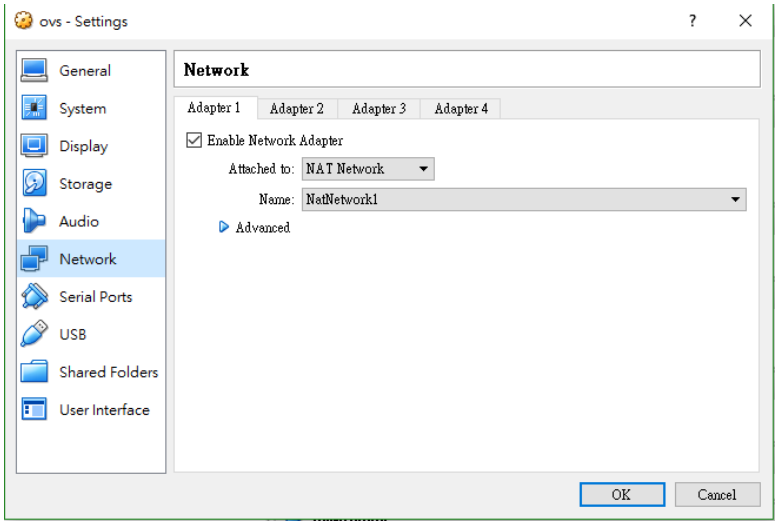
There are some screenshots of VM network GUI setup.

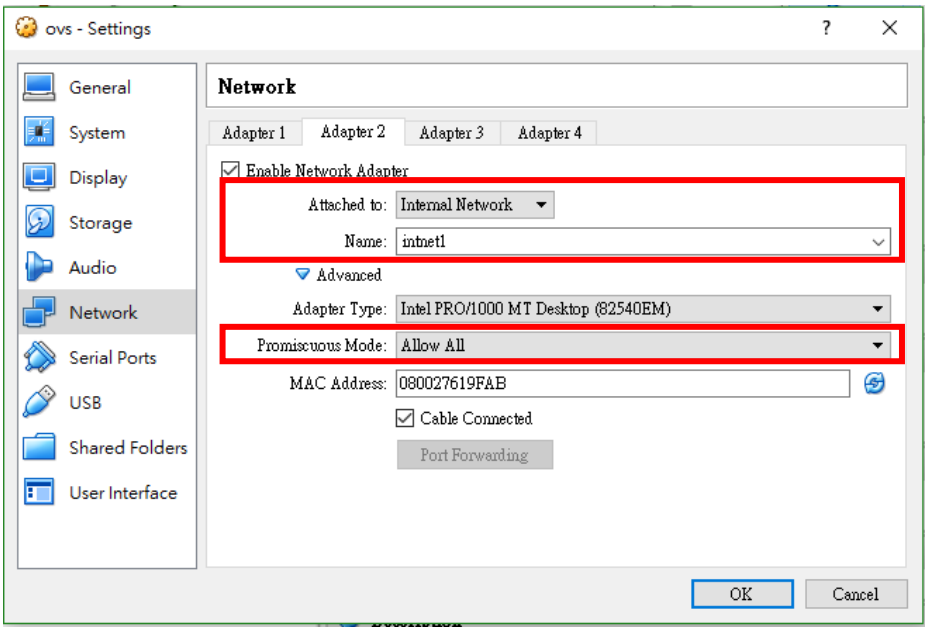
VMWare:



VirtualBox:







NOTICE: You should check the network setting document of the VM provider you used, so that you understand what the differences of all network settings.

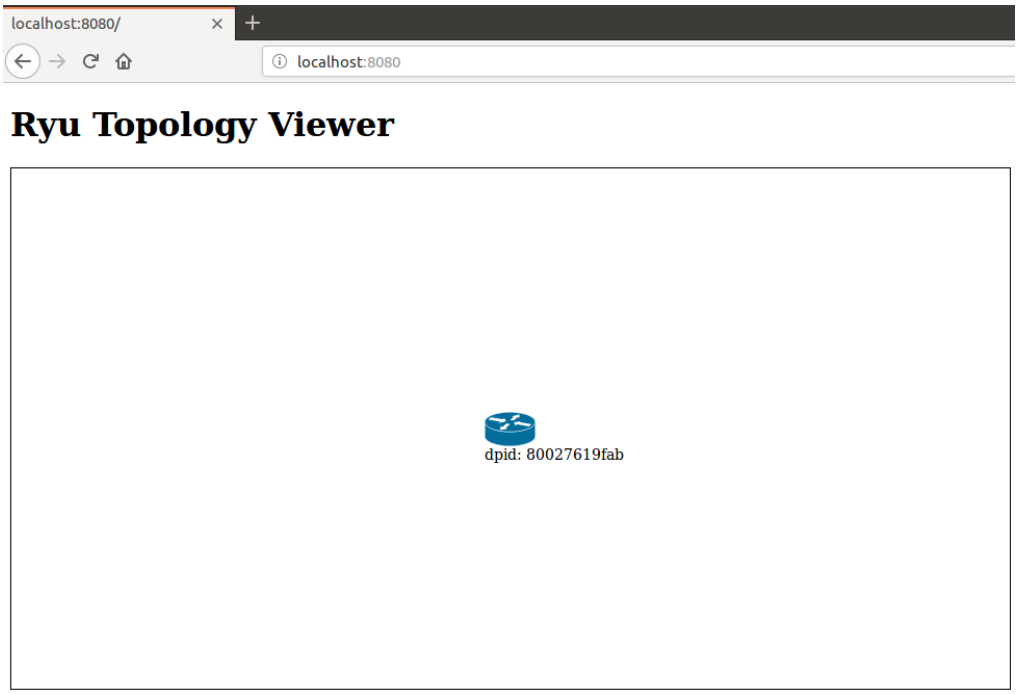
3. The IP address of controller is given by DHCP (or you can manually set the IP). Here we assume that we get 192.168.33.33.
4. The IP address of the two hosts are static. You should set up the IP addresses manually. Assume that we set 192.158.1.11 for host1 and 192.168.2.22 for host2.
5. Set the routing table on the two hosts to enable them to reach the switch. The below example command indicates that 10.0.0.0/8 will go out from "eth0" interface.

```
$ route add -net 10.0.0.0 netmask 255.0.0.0 dev eth0
```

6. Set Open vSwitch to be an OpenFlow switch. You can reference the following commands to set up, you should add `sudo` if you encountered permission problem. (Note: The '#' sign at the front indicates the usage.)

```
$ ovs-vsctl show


# ovs-vsctl add-br <bridge name>
$ ovs-vsctl add-br br0


# ovs-vsctl add-port <bridge name> <interface> (add the interface
connecting to the host to the bridge)
$ ovs-vsctl add-port br0 eth1


# ovs-vsctl set-controller <bridge name> tcp:<ip>:<port>
$ ovs-vsctl set-controller br0 tcp:192.168.33.33:6633


$ ovs-vsctl show
```

7. After you set up the connection to the controller, there should be one switch shown in the Ryu UI.

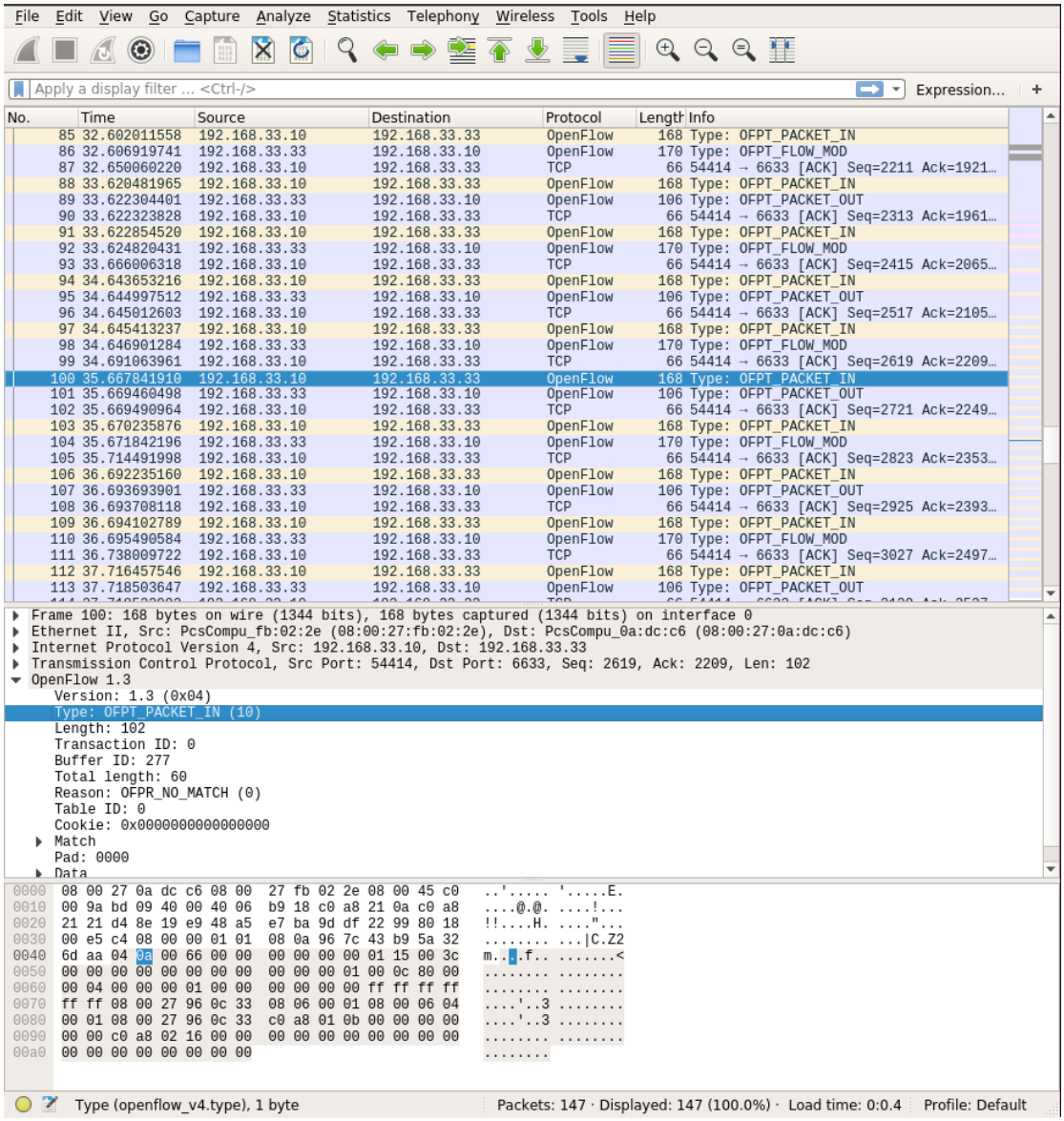


8. On one host, ping the other host to force the controller to learn and insert flow entries into Open vSwitch. (Note: You should make sure you have set the routing rules that goes to the other host.)
9. If you set up and configure OvS correctly, you should receive the ping reply from other host.

Lab 2 - Modify OvS

1. Modify the OpenFlow `flow_mod` function in Open vSwitch (This capability will be very useful for your SDN research in the future.).
2. You should trace the source code of Open vSwitch to identify where the `flow_mod` function is implemented.
3. You need to design a method to purposely let the Open vSwitch not insert the flow entries (which will be received form the OpenFlow controller) into the switch's flow table, except for Table-miss entry.
4. You should use Wireshark[4] to check whether your method can work correctly.
5. We will use iperf to check the correctness of your homework.
6. The expected results should look like the picture below:

[4] Official website: https://www.wireshark.org/



Wireshark Installation

Install from source[5]:

```
$ wget https://1.as.dl.wireshark.org/src/wireshark-2.4.5.tar.xz

$ tar -xJvf wireshark-2.4.5.tar.xz

$ cd wireshark-2.4.5; ./configure

$ make && sudo make install

$ sudo ldconfig
```

Or you can install it from APT:

[5] Related doc: https://www.wireshark.org/docs/wsug_html_chunked/ChBuildInstallUnixBuild.html

```
$ sudo apt-get update

$ sudo apt-get install wireshark
```

Or, install tshark if you prefer using CLI:

```
$ sudo apt-get install tshark
```

Note: Depending on your distribution, the installed version of wireshark/tshark from Ubuntu official repository might not support OpenFlow, see the document and check the version: https://wiki.wireshark.org/OpenFlow

Submission:

1. A pdf file containing:
   a. steps to setup environment (Lab 1),
   b. report that tells how you do it (Lab 2), and
   c. result screenshots,
   d. written in English or Chinese.
2. File(s) you've modified.
3. Compress all the files into a zip file.
4. Name your zip file as "<STUDENT_ID>.zip", e.g., 0556519.zip.
5. Upload the script onto e3.
6. Due date: 2018/04/11.

Hints:

- You can use `grep` command to trace code.

Useful Links:

OvS:
[1] http://roan.logdown.com/posts/191801-set-openvswitch
[2] http://docs.openvswitch.org/en/latest/
Ryu:
[3] http://ryu.readthedocs.io/en/latest/index.html