Introduction to Software Defined Networks

Dino López Pacheco – dino.lopez@univ-cotedazur.fr

The main objective of this Lab is to introduce you to the Software Defined Network (SDN) technologies. In order to learn the basics of SDN, this time we will manipulate the Flow Table of SDN software devices through the ovs-ofctl command. We will be using Open vSwitch (ovs) switches, frequently employed in production Data Centers.

For the following exercises, you can work in pairs (== 2 people teams).

As to the teacher is this lab is graded. If so, your mark will depend on the quality of the explanations and/or results you are expected to provide for every exercise. Write all your answers down in a Microsoft Word or LibreOffice Writer file. Then, you submit your report in PDF format through the moodle website.

1. Introduction to the SDN administration commands

First of all, we are going to start familiarizing a bit with the ovs-ofctl command, which allows a manual administration of a SDN equipment.

1. Go to the cours Moodle website and complete the "ovs-ofctl" activity.
    a. You have all the session to complete this activity
    b. be sure to correctly answer the questions by visiting the man pages of the ovs-ofctl command: https://www.openvswitch.org/support/dist-docs-2.5/ovs-ofctl.8.txt and http://www.openvswitch.org/support/dist-docs/ovs-ofctl.8.txt

2. First contact with a SDN switch

Let's try to use the ovs-ofctl command. For the next exercises, please take into account that all SDN switches have been configured to listen for OpenFlow connections at the 6653 TCP port.

2. Verify if you have a "~/pox/" directory. If not, install the POX controller as follows
    a. In your home directory, clone the POX repository with the command "~$ git clone https://github.com/noxrepo/pox.git"
    b. If you successfully cloned the POX repository, you must have now a "pox" directory in your home directory.
3. Download and deploy the network from the "2sdnswitches.imn" file.
4. In this topology, the SDN switches are expected to work in in-band or out-of-band mode? Note that the hosts "hX" are the users' computers and of course, they are not aware about the SDN network/controller. Provide a brief argument.

As the user packets from the hosts will travel on the red links only and the OpenFlow traffic will be carried out by the blue links, clearly users' packets and OpenFlow packets will not share the same links. We have hence an out-of-band configuration.

5. Double-click on the controller to open a terminal. Get the characteristics of "sdn1" using the appropriate ovs-ofctl command.

   a. Briefly say what is the number of Flow Tables supported by the SDN switch, the supported actions and the characteristics of every vNIC of the switch.

   b. The switch identifier is known as the datapath id (dpid) in OpenFlow. Provide the dpid you have found.

   c. Also, provide the command you have used.

   "ovs-ofctl show tcp:192.168.0.100:6633". dpid = 0000000000aa0000. The current version of Open vSwitch supports several actions (e.g. set_vlan_vid, set_vlan_pcp, strip_vlan, etc). We can see also the 3 ports available at our switch. Note that each port (whose label appears between parenthesis) there is an associated port ID. In this specific case, port 1 == eth0, however, 1 could be any of the other ports. There are 254 available tables.

6. At this moment, there is not a controller application running on the controller server. Double-click at the controller server to obtain a terminal. Launch the controller with the command "/home/user/pox/pox.py --verbose forwarding.l2_pairs" and verify that both SDN switches connect to the controller. A few seconds might be needed before the SDN switches connect to the controller.

   a. Here, POX executes the l2_pairs network application. In this Net.App., each PacketIn is used to learn the exact location of the sender at the switch (that is, the @MAC - port association). Hence, once a pair is known (e.g. if the controller knows that node A is reachable through port 1, and node B through port 2), the controller install the proper Flow Entries at the Flow Table of the switch (e.g. Rule 1: forward every packet coming from A and going to B by port 2. Rule 2: forward every packet coming from B going to A by port 1).

   b. If you see that the SDN switches connect to the controller, continue with the following exercises. Otherwise, redo carefully the previous steps to solve the problem.

7. Show the Flow Table at switches sdn1 and sdn2. Verify that the Flow Tables are empty. Which command must you execute at the controller server for this aim?

   "ovs-ofctl dump-flows tcp:192.168.0.100:6633" and "ovs-ofctl dump-flows tcp:192.168.0.101:6633"

8. Open one wireshark instance per virtual host and capture the traffic. Then, with the arping command (**to install at the VM** if needed "sudo apt install iputils-arping"), execute a request from h1 to h2 (e.g. arping -c1 10.0.0.11) and show the entries at the Flow Table of sdn1 and sdn2.

a. The arping command above will generate an ARP request (in broadcast mode). Which hosts got the ARP request?

Every host: h1, h2 and h3

b. Explain with your own words every entry at sdn1 and sdn2 (ignore the cookie parameter however). Feel free to read the ovs-ofctl manpage as needed.

0 entries at sdn2. 2 entries at sdn1. Flow entry 1 says that every packet matching a source MAC address equal to 00:00:00:00:00:01 and going to MAC address 00:00:00:00:00:02 must be forwarded trough port "eth1" . Flow entry 2 says a similar thing, inversing however the source and destination MAC addresses. This time, the output port will be "eth0".

c. Which host got the ARP reply (sent in unicast mode)?

h1 only

d. Based on you understanding of SDN, explain

i. if the ARP request triggered (and when) a Packet-In, Packet-Out and Flow Mod at sdn1

the Flow Table should be empty at sdn1 when the ARP request arrived. Therefore, the ARP request was sent to the controller (PacketIn), who decided to broadcast (send the packet through all other ports, except the one that received the ARP request) with a PacketOut. No FlowMod as there is not enough information to add a L2 pair rule.
Note that the controller learned the position of h1 at sdn1

ii. if the ARP request triggered (and when) a Packet-In, Packet-Out and Flow Mod at sdn2

same as point i). The controller learned the position of h1 at sdn2.

iii. if the ARP reply triggered (and when) a Packet-In, Packet-Out and Flow Mod at sdn1

the Flow Table should still be empty at sdn1 when the ARP reply arrives. Hence, the ARP reply is sent to the controller (PacketIn). This time however, the controller knows the exact location of h1 and learned the location of h2 with the new PacketIn. The controller inserts two flow entries at sdn1 (FlowMod) and forward the ARP reply by the right port with a PacketOut.

iv. if the ARP reply triggered (and when) a Packet-In, Packet-Out and Flow Mod at sdn2

sdn2 never saw an ARP reply.

9. Remove the flow entries from sdn1. Which command should you execute?

One possibility: “ovs-ofctl del-flows tcp:192.168.0.100:6633 dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:02” and “ovs-ofctl del-flows tcp:192.168.0.100:6633 dl_src=00:00:00:00:00:02,dl_dst=00:00:00:00:00:01”

10. Verify from the controller that all flow entries have been removed. Then, execute 5 ARP requests with the command “arping -c5 10.0.0.11”. The output of the command will show the delay between the transmission of the request and the reception of the reply. Explain why the first request gets a much higher delay than the subsequent requests/replies.

    Because of the SDN mechanisms. In absence of rules, the execution of PacketIn, PacketOut and FlowMod will considerably delay the retransmission of packets. Once the FlowEntries are installed, the delay hence decreases.

If you have finished with this part, execute Ctrl+C to end the execution of POX and close all the terminals from the virtual network, leaving moreover the virtual network running.

3. Manual administration of a SDN switch with an external controller

1.1. Basics

Now you will play the Network Application role.

11. Download the dummy.py file and save it in /home/user/pox/ext/. Then, execute the POX controller at the controller server: “/home/user/pox/pox.py --verbose dummy”.

    a. Open a terminal for h1, h2 and h3, and as for the previous section, run wireshark.

12. Execute an arping from h1 to h2 with option “-c1”. How many packets were captured by h1, h2 and h3? Explain your observations. According to the log messages at the controller, does the arping command generates some PacketIn at the controller?

    Since there are no rules, except the one sending packets to the controller, only the ARP request at h1 must be captured by tcpdump at h1. Only the ARP request arriving at sdn1 generates a PacketIn.

13. From the controller server, with the ovs-ofctl command, add the following Flow Entry to sdn1 “in_port=1,idle_timeout=30,actions=output:2”. Execute again an arping from h1 to h2 with the “-c1” option. How many packets were captured by h1 and h2 and why?

    The ARP request must be captured at h1 and h2 (meaning that the packet is forwarded). However, the ARP reply is not captured at h1

14. How many seconds should you wait for the previous Flow Entry to be removed at sdn1? Wait until the FlowTable is flushed. Then, continue with the next exercise.

    30 seconds, starting from the last time the rule has seen a hit.

15. From the controller server, with the ovs-ofctl command, add the following Flow Entries to sdn1 “in_port=3,actions=output:2” and “in_port=2,actions=output:3”. Then, stop the controller with Ctrl+C. Wait a few dozen of seconds, then execute from h3 an arping to h2 (“arping -c1 10.0.0.11”).

    a. Explain your observations.

    The arping should succeed. Note that we have installed some flow entries at sdn1 and not at sdn2. Hence, while the controller runs, the arping will not succeed as our controller never installs flow entries at sdn2

    b. Look at the Flow Table of sdn1 and sdn2. Explain if the switches are configured in secure or standalone mode (Open vSwitch only supports the secure and standalone failures mode), and how you did deduce that.

    Looking at the flow table of sdn1, we can see that the previous installed flow entries are still there. Since the controller is gone and the flow entries remain there, the sdn switch is for sure in secure mode (the smart mode isn’t currently supported by Open vSwitch). When the controller is gone, sdn2 starts forwarding packets in a normal mode (without the help of a controller). The normal mode in Open vSwitch makes a switch to behave as a legacy switch.

16. Close all wireshark instances. Start again the controller with the dummy network application and remove all entries from sdn1.

17. Manually (i.e. using the ovs-ofctl command), emulate a legacy switch at sdn1. A legacy switch builds a Forwarding Data Base (FDB) that maps a port number to a reachable MAC address. This way, when the switch receives a packet with an already learned MAC Address, the frame is immediately forwarded through the appropriate port.

    a. As an example, you must create a rule to forward through port 1 any packet with the 00:00:00:00:00:01 destination MAC address.

    b. Broadcasted packets (destination MAC address FF:FF:FF:FF:FF:FF) must be flooded by any port, except the one where the packet is received. Look at the ovs-ofctl manpage how to write a “flood” action and at the website address http://www.openvswitch.org/support/dist-docs-2.5/ovs-ofctl.8.txt , “Flow Syntax” section, how to match a MAC address (destination or source)

    Flow Table at sdn1:

```
 cookie=0x0, duration=92.248s, table=0, n_packets=0, n_bytes=0, dl_dst=00:00:00:00:00:01 actions=output:eth0
 cookie=0x0, duration=70.062s, table=0, n_packets=0, n_bytes=0, dl_dst=00:00:00:00:00:02 actions=output:eth1
 cookie=0x0, duration=59.202s, table=0, n_packets=0, n_bytes=0, dl_dst=00:00:00:00:00:03 actions=output:eth2
 cookie=0x0, duration=12.813s, table=0, n_packets=0, n_bytes=0, dl_dst=ff:ff:ff:ff:ff:ff actions=FLOOD
```

Flow Table at sdn2:

```
 cookie=0x0, duration=44.805s, table=0, n_packets=0, n_bytes=0, dl_dst=00:00:00:00:00:03 actions=output:eth0
 cookie=0x0, duration=30.998s, table=0, n_packets=0, n_bytes=0, dl_dst=00:00:00:00:00:01 actions=output:eth1
 cookie=0x0, duration=22.778s, table=0, n_packets=0, n_bytes=0, dl_dst=00:00:00:00:00:02 actions=output:eth1
 cookie=0x0, duration=7.925s, table=0, n_packets=0, n_bytes=0, dl_dst=ff:ff:ff:ff:ff:ff actions=FLOOD
```

18. Probe with a ping that h1, h2 and h3 are all reachable by each other. Add some screenshots with the flow table rules and the ping output

    It should work. You're expected to provide screenshots or copy/paste your terminal lines here.

19. (Optional – to get ready for the exam – no solution provided here). Block all the UDP traffic only at sdn2.