EE 555 Major Project, Fall 2019

Deadline – Dec 3rd, 2019 (midnight)

Overview

The objective of this project would be to build an SDN controller. You will be having a dumb/simple packet forwarding switch and an intelligent controller. The dumb packet forwarding switch will be talking to the controller to make forwarding decisions. Based on how you program the controller, the packet forwarding switch will either act as a router or a switch. The controller and the packet forwarding switch will interact with each other using the OpenFlow protocol. We will be using Mininet network emulation software to emulate the testbeds and networks to test and validate the controller and the switch interaction.

In case of SDN, where the above described combination of controller and packet forwarding switches are used, the interaction of switch with controller for every packet would be expensive in terms of processing delay. If the packet forwarding switch will talk to controller for every packet, then the latency and delay in the network will increase. So, OpenFlow gives us an ability to push rules from controller to packet forwarding switches. In this way, the packet forwarding switch need not talk to controller upon every packet arrival. These rules are pushed from the controller to the packet forwarding switch using the OpenFlow protocol. So, it is essential for you to understand the semantics and usage of the OpenFlow protocol in order to successfully complete this project.

For executing Sceanrio1, Scenario 2 and Scenario 3 of the project you will be following the instructions and scenarios described at the following github link:

https://github.com/mininet/openflow-tutorial/wiki

First steps

As described above, we will be using the Mininet emulation software for this project. A very customized and trimmed version of Linux is available with Mininet installed in it at the github link above. You will be needing Oracle VM Virtual Box, the VM image that has Mininet software, some SSH applications like putty or Mobaxterm, X11 display connectors like Xming etc., to execute the lab. Please go through installing required software section of the tutorial at the above link.

Please go through all the instructions of set up virtual machine carefully.

Note: Please ensure that you have the host-only network configured properly in your Oracle VM Virtual Box, as the host only network will be used to SSH into your VM from your Host Operating System (Windows / MAC). As mentioned above, the Mininet VM is a highly trimmed version of Linux and you will not have GUI and it will be cumbersome and difficult to access the VM from the virtual machine console. So, please ensure that your host-only network is setup properly and you can SSH into your VM from your guest OS (Windows / MAC).

Once all the required software are installed and the VM is setup, you can start with the project. We would also recommend everyone to go through these links.

Virtual Box Specific Instructions

Learn Development Tools

Once you are done with all the above steps, you can start working on the project.

The next thing to do is to understand two sets of python files.

1. The files that contain the description of the topology.
2. The file that contains the code for controller.

To do various scenarios, you need to emulate different type of topologies. The topology files you will write need to be stored in the location "~/mininet/custom/".

The controller python files should be stored in the location "~/pox/pox/misc/".

You will be performing a total of 4 scenarios in this project.

Scenario 1

In this scenario you will be designing the controller in such a manner that the packet forwarding switch will be acting as a Layer 2 switch. In this scenario you will have 3 hosts connected to the switch and the three hosts will be in the same LAN (Local Area Network). Your controller must first learn the switch port to MAC mapping of the hosts. This mapping helps the controller to determine which host is attached to a given port of the switch. Only when controller determines this information, it can push the rules to packet forwarding switch in such a way that the packet forwarding switch will act as a Layer 2 switch.

You need to complete the section of Create a Learning Switch in order to complete this scenario of the project.

Some Key Points to think of:

- What is the first message that a host sends out into network when it wants to talk to another host in the same LAN?
- Can you learn the MAC address from this first message?
- Can you learn MAC address of both the hosts through these first messages?
- If you can learn the MAC address then, what fields among the 10 available fields in the OpenFlow protocol will you use to make a rule that can achieve the Layer 2 functionality?
- Try to answer these questions and you should be able to implement the functionality of a Layer 2 switch.

Scenario 2 and Scenario 3

Now, having implemented the learning switch, you need to turn the controller and the packet forwarding switch combo into a Layer 3 router now. The design of the controller logic decides the behaviour of the packet forwarding switch. In this case, you need to make the packet forwarding switch to act as a router and this will be the most important aspect of all the three scenarios viz., scenario 2, scenario 3, scenario 4.

The scenarios are as follows:

Scenario 2:

In order to complete this scenario, you need to complete the Router Exercise section in the github tutorial.

In scenario 2, you need to make the packet forwarding switch act as a router with three interfaces attached to it. All the three interfaces of the packet forwarding switch will be in different subnets. For each of the host, configure the router interface IP address as the default gateway (Think why? And think what a default gateway for a host is?). The router should have a static routing table with the network IDs and the corresponding interface to use for the corresponding network IDs.

Note: In case any host tries to ping an unknown destination that is not part of the topology, in that case, the router should send an ICMP destination host unreachable message back to the host that initiated the ping.

Scenario 3:

In order to complete this scenario, you need to complete the Advanced Topology section in the github tutorial. This is essentially an extension to the Router Exercise. In this section you will have two packet forwarding switches connected by a single point to point link. One packet forwarding switch has two hosts attached to it with both the hosts in the same network. The second packet forwarding switch has one host attached to it.

The following are some good questions to ask yourselves to be able to complete scenarios 2 and 3:

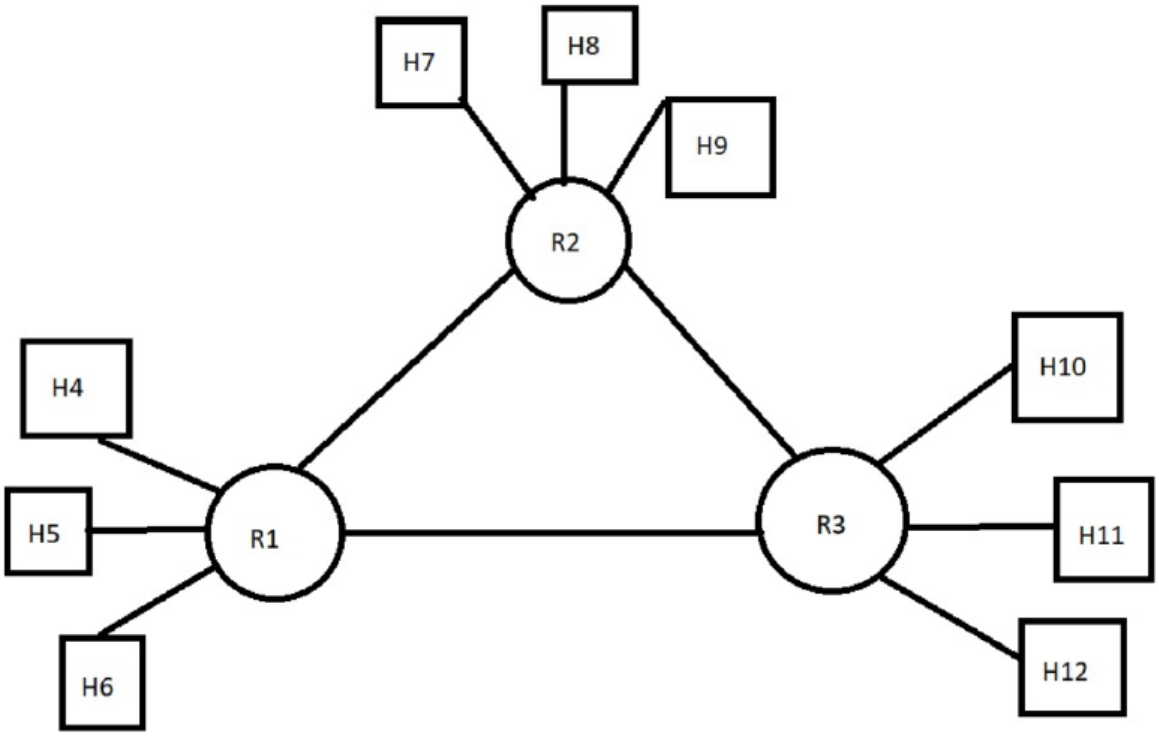
- What does a host do when it wants to send a packet to another host that is not part of the same LAN?
- If a packet comes to a router and the router does not have the MAC address of the destination host, then what does it do? (Think about queuing the packet, while router tries to learn the MAC address of the destination IP address in the packet)
- What rules should the controller push to the packet forwarding switch so that it looks like the packet forwarding switch is acting like a router? (Again, think of which fields in the available 10 fields will you use while forming rules)

Note: In case any host tries to ping an unknown destination that is not part of the topology, in that case, the router should send an ICMP destination host unreachable message back to the host that initiated the ping.

Scenario 4

The topology for this scenario is as shown below:

This is an extension to the scenario 3 and this has a loop in the topology. In this scenario you will have 3 routers connected to each other in a full mesh topology. Each router will have 3 hosts attached to it. You need to forward the packet internally if the destination host address belongs to the same LAN. If the destination is of not the same LAN, then you need to forward the packet to the nearest router. The topology for the scenario is as shown below.



Note: Be careful to ensure that your packets do not get lost in a loop. You are not allowed to turn down any link for the purpose of this scenario. All the links should be up and running while we execute the scenario. You are required to push the rules corresponding to various packets in this scenario from the controller to the packet forwarding switch. We will be checking the installed rules in the packet forwarding switches using the commands described in github portal to dump the flows.

You can assign any set of IP addresses for the routers and hosts you wish. But ensure that they are all in private IP address range and the hosts and routers in the same LAN have IP addresses from the same subnet. Do not forget to mention the IP address details in your readme file. You will loose points if you don't clearly mention the IP address details of the hosts and routers clearly in the readme file.

Deliverables for various scenarios:

Scenario 1:

For scenario 1, you don't need to create any topology file. Scenario 1 uses the default scenario with one packet forwarding switch and three hosts connected to it. But you need to write the controller code. Some part of the controller code is provided to you for convenience in the file - "~/pox/pox/misc/of_tutorial.py". Some part of the code is missing in this file and you need to complete the code to make the controller and packet forwarding switch combo to achieve Layer 2 switch functionality.

Scenario 2, 3, 4:

For scenario 2, 3 and 4, you need to create two python files. One file that has the configuration of the topology and the other file that contains the code for the controller.

The names of the various files that needs to be submitted are as below:

Scenario	Topology File	Controller code file
Scenario 1	Not Applicable	of_tutorial.py

Scenario 2	topology2.py	controller2.py
Scenario 3	topology3.py	controller3.py
Scenario 4	topology4.py	controller4.py

You need to have a readme file that contains detailed instructions on how to emulate different scenarios and what commands to run and when to run them. Readme file should contain details for all the scenarios. The readme file should be named "readme.txt" and it should be a common file for all the four scenarios i.e., the details of all the files and instructions corresponding to all the scenarios should be submitted in a single readme file. The readme should be a txt file.

Readme

- List of files that are being submitted.
- Location at which files should be copied to execute the scenarios. Please mention the location for each file individually, so that it will be clear to us while verifying your submission.
- Commands to execute the scenarios.
- Support included for each scenario. By support we mean what functionalities are implemented for completing each scenario.
- Your name and your teammates name.
- Your USC email ID and your teammates USC email ID

Please make sure that you write the readme file properly because we will be executing the instructions mentioned by you in the readme file as it is. So, if you write incorrect instructions or if some instructions will be missing that are necessary for successful execution of scenario, then you will be penalized.

Report

You need to support a report that contains the approach followed by you for various scenarios. Describe the high-level design of your controller for each scenario and list out any problems you faced during design, implementation or testing phases. Keep the report simple and succinct. The report must be in PDF format

Testing and Grading Criteria

The below is the grading criteria and the tests that are used to grade the project.

Scenario 1: (10 points)

Scenario	Points
h2 ping h3	2
pingall	2
iperf TCP between h2 and h3	2
iperf UDP between h2 and h3	2
rules installation in switch	2

Scenario 2: (20 points)

Scenario	Points
h1 pings its default GW	2
h1 pings router IP2	2

h1 pings router IP3	2
h1 pings h2	2
h1 pings h3	2
pingall	2
h1 pings unknown host	2
iperf TCP between h1 and h3	2
iperf UDP between h1 and h3	2
rules installation in switch	2

Scenario 3: (30 points)

Scenario	Points
h3 ping h4	2
h3 ping h5	2
h3 ping s1	2
h3 ping s2	2
h5 ping s1	2
h5 ping s2	2
pingall	2
iperf TCP between h3 and h4	2
iperf UDP between h3 and h4	2
iperf TCP between h3 and h5	2
iperf UDP between h3 and h5	2
h3 ping unknown host	2
h5 ping unknown host	2
rules installation in switch 1	2
rules installation in switch 2	2

Scenario 4: (30 points)

Scenario	Points
h4 ping h5	2
h4 pin h7	2
h4 ping h10	2
h4 ping R1	2
h4 ping R2	2
h4 ping R3	2
Pingall	2
iperf TCP between h4 and h5	2
iperf UDP between h4 and h5	2
iperf TCP between h4 and h7	2
iperf UDP between h4 and h7	2
h4 ping an unknown host	2
rules installation in switch 1	2
rules installation in switch 2	2
rules installation in switch 3	2

Readme – 5 points

Report – 5 points

So, the project will be for a total of 100 points.

Bonus Scenario – Firewall

In this scenario, you can take the basic scenario 1 topology. In addition to having the basic Layer 2 functionality, the packet forwarding switch should also act as a Firewall in this case. It should block TCP iperf packets and allow any other kind of traffic between the hosts.

This scenario will be for 10 points. The testcases that will be used to grade this scenario are as follows:

Scenario	Points
h2 ping h3	2
pingall	2
iperf TCP block	2
iperf UDP pass	2
rules in switch	2

The maximum score of the entire project will be 100 points. This bonus section will help you to compensate failed testcases in the first four scenarios. If you get full score in the first four scenarios and if you get the bonus scenario correct, your total score would still be 100.

Submission guidelines

- The project should be worked in a team of 2 (maximum allowed people in a team is also 2). If you are not able to find a teammate, then you can work alone. But please let us know before hand if you are going to be working alone instead of working in a team.
- The deadline for the project submission is December 3rd, 2019 (midnight) for both on-campus students and DEN students.
- This deadline is a hard deadline and the drop box will close automatically after this deadline and we will not any submissions post the deadline.
- You need to zip all the files and name the zip as below:
- partner1firstname_partner2firstname_EE555_Fall_2019.zip
- For ex: If me and Muhammed were to team up, then our file name would be ramnath_muhammed_EE555_Fall_2019.zip
- Please make the zip file with above name and submit all your files to the drop box.

Some useful information

- The below are the links for some basic python tutorials.
- http://docs.python.org/tutorial/introduction.html
- https://www.learnpython.org/
- We recommend you start early on the project, because you need to look at considerable amount existing code before you start writing your own code.
- Do not assume anything. If you are not sure of anything, whatever it be, just drop an e-mail or make a piazza post or come visit us in office hours. Do not implement something by

assuming this/that might be the case and please don’t make things complicated for both us and you. We will be happy to help you always and all you need to do is reach out to us.

- Ensure that all the files are present before submission. If you miss files for one scenario, then it can cost you a lot of points.
- Also, once submission is done, try verifying your submission in your teammates virtual machine or by creating a new virtual machine. That will ensure that the project is implemented successfully, and the submission is also right.