**Project 2: Implementing a Simple SDN Controller**

**Due on 11-08-2025 Midnight (hard deadline)**

## Purpose of Assignment

For this project, you will be constructing a network for a small company. You will implement packet forwarding between devices on different subnets in the company network and implement firewalls for certain subnets. You will practice using a Software Defined Networking (SDN) controller to control the behaviors of the switches in the network. In particular, we will use a popular SDN controller, POX. You will program a POX SDN controller (the skeleton code is provided to you) to achieve the requirements specified in this project. You will construct the topology using Mininet (which you have practiced in HW1) and then implement the rules in the controller to allow traffic to flow through your network following the requirements.

Example

Before working on this assignment (see below), please go over “PA2 Practice Questions and Solutions.docx” as well as the accompanying code: (i) practice\_topo.py is the code for constructing a, and (ii) practice\_controller.py is the SDN controller code for achieving the goals specified in the practice questions. Both of them are slightly simpler than what you need to do for this assignment and serve as examples. In addition, our TA, Jie, has kindly recorded a **video** to show you how to run practice\_topo.py and practice\_controller.py, and how to verify that the results are as expected. The video is at

<https://uconn-my.sharepoint.com/:v:/g/personal/minmei_wang_uconn_edu/EYWvqmTBR4NCuCoTppTaL14B2lkiEw-z_mJ-oP-foJIf2A>

Once you have gone over “PA2 Practice Questions and Solutions.docx”, practice\_topo.py, practice\_controller.py, and the video, it will be clear to you what you need to do for this assignment.

## Assignment

Consider the network for a small company. The company has a 3-floor building, with each floor having its own switch and subnet. Additionally, we have a switch and subnet for all the servers in the data center, and a core switch connecting everything together.

The devices’ roles and IP addresses are as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| Device | Mininet Name | IP Address | Description |
| Floor 1 Host | h1 | 10.1.1.10/24 | A computer on floor 1 of the company. |
| Floor 2 Host | h2 | 10.2.2.20/24 | A computer on floor 2 of the company. |
| Floor 3 Host | h3 | 10.3.3.30/24 | A computer on floor 3 of the company. |
| Untrusted Host | h4 | 123.45.67.89/24 | A computer outside our network. We treat this computer as a potential hacker. |
| Server | h5 | 10.5.5.50/24 | A server used by our internal hosts. |

The topology is as follows:

download.png

In the topology, s4 is the core switch; host h1 connects to switch s1, host h2 connects to switch s2, host h3 connects to switch s3, and host h5 connects to switch s5. All the switches, s1, s2, s3 and s5, are connected to the core switch, s4, through different ports of s4. The untrusted host, h4, is connected to switch s4.

Your goal is to allow traffic to be transmitted between all the hosts. In this assignment, you will be allowed (and encouraged) to flood all non-IP traffic (using a destination port of OFPP\_FLOOD). However, you will need to specify specific ports for all IP traffic. You may do this in any way you choose. However, you may find it easiest to determine the correct destination port by using the destination IP address and source IP address, as well as the source port on the switch that the packet originated from. Additional information has been given to you in the do\_final() function to allow you to make these decisions. Please see the comments in the provided code for guidance.

Additionally, to protect our servers from the untrusted Internet, you need to block all IP traffic from the Untrusted Host to the Server. To block the Internet from discovering our internal IP addresses, we will also block all ICMP traffic from the Untrusted Host to anywhere internally.

## Provided Code

We have provided you with starter code (skeleton files) to get you started on this assignment. The controller file (**project2controller.py**) needs to be placed in ~/pox/pox/misc, and the Mininet file for creating the topology (**project2.py**) should be placed in your home directory (~). You will need to modify both files to meet the project requirements. For **project2controller.py, you only need to modify** do\_final() function**. For project2.py, you only need to modify** build() function**.**

You will be using slightly different commands to create the Hosts and Links in the Mininet file to give you more information to make decisions within the Controller file. Additionally, you will notice that you have additional information provided in the do\_final function. This is documented in the comments within the files.

To run the controller, place project3controller.py in the ~/pox/pox/misc directory. You can then launch the controller with the command **sudo ~/pox/pox.py misc.project2controller**

To run the Mininet file, place it in ~ and run the command **sudo python ~/project2.py**

**To do project2, you will need to be running both files at the same time (in 2 different terminal windows).**

## Summary of Goals

* Create a Mininet Topology to represent the specified topology.
* Create a POX controller with the following features:
  + All the hosts are able to communicate, EXCEPT:
    - Untrusted Host cannot send ICMP traffic to Host 1, Host 2, Host 3, or the Server.
    - Untrusted Host cannot send any IP traffic to the Server.

## Testing

You may test with ping commands, xterm windows, and observing packets with Wireshark inside your VM.

## Grading Rubric

Total: 100 points

**Submit both the code and PDF report. Please add comments to your code for better readability.**

30 points: Mininet Topology (use dump command to show your results)

10: Devices successfully created.

10: Links successfully created.

10: IP addresses correct.

50 points: Pox Controller (hint: use pingall, iperf and dpctl dump-flows to show that your implementation works)

25: All hosts can communicate.

15-point deduction if rules not installed in flow table.

10-point deduction if IP traffic is implemented using OFPP\_FLOOD.

15: Untrusted Host cannot send ICMP traffic to Host 1, Host 2, Host 3, Server

10-point deduction if Untrusted Host cannot send ANY traffic to these hosts.

10: Untrusted Host cannot send any IP traffic to Server

20 points: Demonstrate your results using corresponding commands and screenshots.

Please submit a PDF report. You must include screenshots to show that your code works. These screenshots must come from your own code. The code will be tested. Submitting screenshots of someone else’s code can be considered an academic integrity violation.

Partial credit may be awarded for incomplete assignments.