Project 3: Implementing a Simple Router

Due on 04-15-2024 Midnight (hard deadline)

In this project, you will implement routing between devices on different subnets, and implementing firewalls for certain subnets. The idea is to simulate an actual production network. You will be using ideas from project 1 to help construct the Mininet topology, and then implement the rules allowing for traffic to flow through your network.

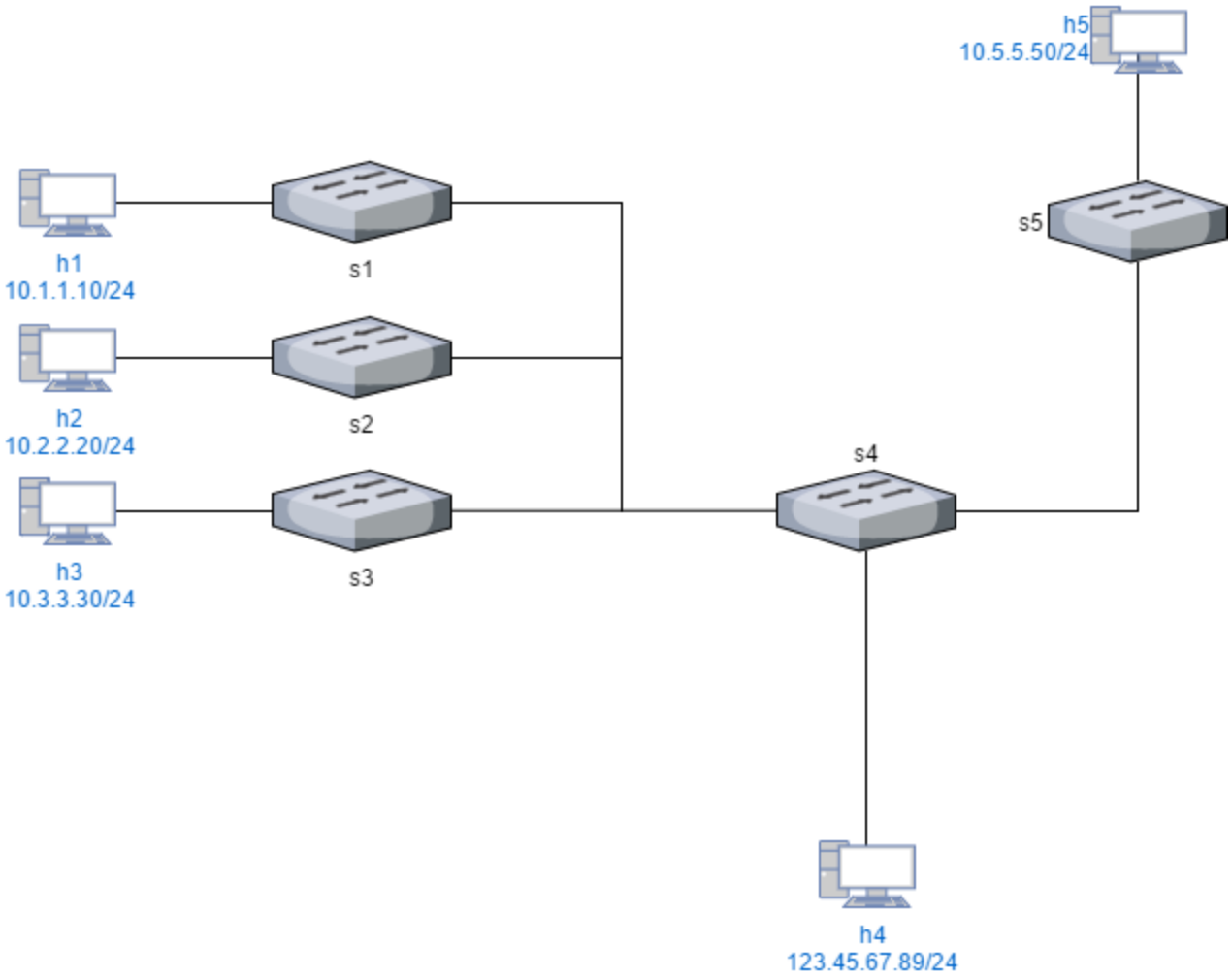
Assignment:

For this project, we will be constructing a 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.

Your device's 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 will look as follows:



Your goal will be 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 of. OFPP_FLOOD). However, you will need to specify specific ports for all IP traffic. You may do this however 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, we will be blocking 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:

Available in a ZIP file here (https://drive.google.com/file/d/1p__xE11ZPjJdoAnYqtIBTM-iJS185wKT/view?usp=drive_link).

We have provided you with starter code (skeleton files) to get you started on this assignment. The controller file (project3controller.py) needs to be placed in ~/pox/pox/misc, and the mininet file (project3.py) should be placed in your home directory (~). This time, you will need to modify both files to meet the project requirements.

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.project3controller**

To run the mininet file, place it in ~ and run the command **sudo python ~/project3.py**

To do project3, you will need to be running both files at the same time (in 2 different terminal windows).

Summary of Goals:

- Create a Mininet Topology (See project 1 for help) to represent the above topology.
- Create a Pox controller with the following features:
    - All hosts 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.

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 prove your implementation)
- 25: All hosts can communicate.
    - 15-point deduction if rules not installed in flow table.
    - 20-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: Demonstration of your results using corresponding command and screenshots.

Please submit a PDF report. You must include screenshots proving 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.