Assignment 2: SDN

Overview

In this assignment, you will be required to simulate a simple cloud networking platform. The Mininet network emulator will be used to build a virtual network as required. And you need to program the OpenFlow controller POX (of course, you can choose other preferred controllers) to implement two common applications in multi-tenants cloud environment: Firewall and Virtual Network Slicing.

The Mininet network emulation environment and POX controller should run on different machines (we recommend you to use two virtual machines) and they communicate via LAN (or a virtual LAN if you use two virtual machines in a single physical machine).

For both tasks, you may need to use iperf to generate some traffic to test your code.

For details of Mininet, POX and iperf, please refer to the document: “A short walk-through of Mininet and POX.pdf”

For any technique problems with both Mininet and POX, please contact the tutor of this assignment: **Mr. Yuxiang ZHANG** (Email: 894240400@qq.com)

Task 1: Network Building

In this part, your task is to build a simple virtual network in the Mininet network emulation environment. The network consists of 4 servers, 4 switches and 8 links that connect them together. The topology of the network is shown in Figure 1.

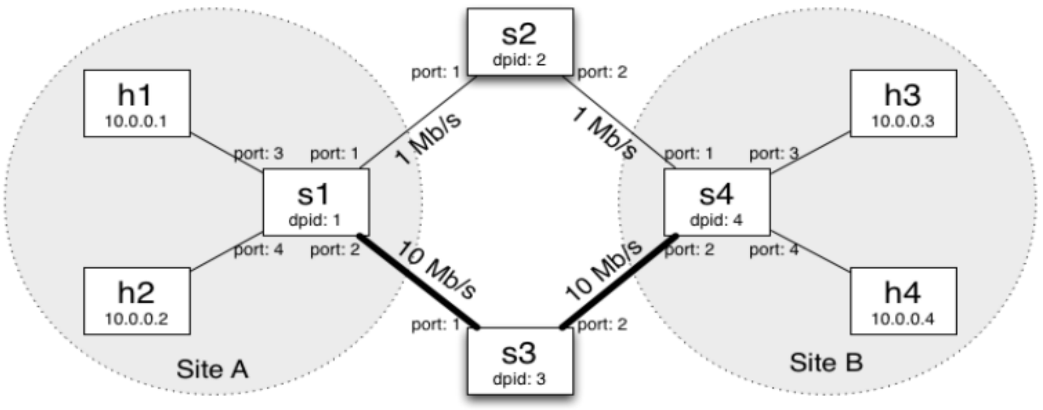

Figure 1: Network with 4 servers, 4 switches and 8 links.

You need to build a network whose topology is described in Figure 1, where *h* represents servers and *s* represents switches. Notice that the configurations of the links that connect the 4 servers to their nearest switches are not specified. The only requirement is they should not be the bottleneck. It is recommended you write a python script to do the job.

You will be provided with a template of script, based on which you can add your own code to set up the network.

Task 2: Firewall

In this task, you need to implement a layer-2 firewall application using the POX controller. You are required to block the communication between h1 and h4.

The basic idea is when a connection between a switch and a controller is up, the application installs flow entries to disable the traffic between each pair of the MAC addresses in the list. To make things simple, you can install the firewall rules on all switches in the network.

Task 3: Virtual Network Slicing

In this part, your task is to slice the network based on the application that is sending the traffic. In principle, SDN networks can be sliced based on any attributes.

The topology of the network is the same as in Task 1. However, you need to treat the video and non-video traffic differently. Between site A and site B, all the video traffic will go through the high bandwidth path (video slice), and all other traffic will go through the low bandwidth path (non-video slice). To simplify this task, we assume that all the video traffic use TCP port 80.

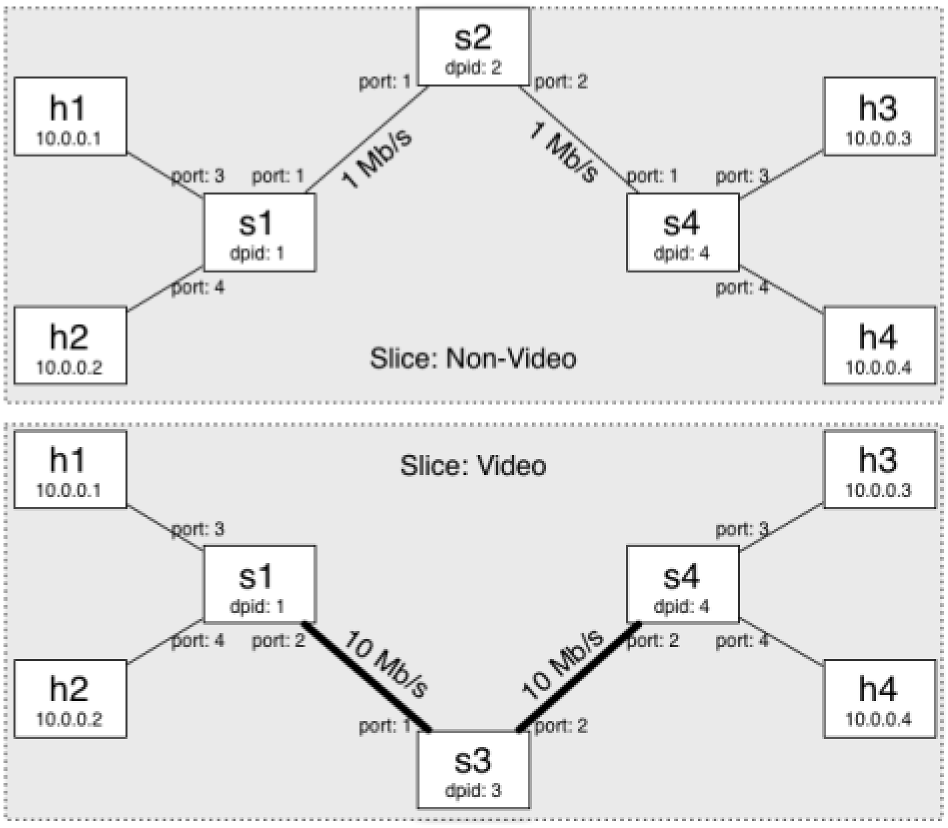

Figure 2: Sliced network traffic.

For illustration, Figure 2 depicts the sliced network, in which the video traffic flows through the path S1-S3-S4, and the rest traffic is forwarded to the path S1-S2-S4.

Handling Conflicts

POX supports multiple concurrently running applications. However, it is possible that one application will be in conflict with another application. In this assignment, the network slicing application should not allow communication which is blocked by the firewall application. You are to make sure that no conflicting rules are installed by two different applications. More specifically, the blocking pairs and the communicating pairs can be chosen among all servers without conflicts between them.

Hint: Such conflict can be handled by setting different priority value to different rules. OpenFlow message structure ofp_flow_mod has an attribute called priority.

Understanding the provided files

mininetSlice.py: The template of scripts which creates the network topology for this assignment.
controller.py: A skeleton for the two applications. launch() function has been implemented to register a new component.

Submissions:

Please submit the following documents to the tutor (894240400@qq.com) before the deadline:

1. **All related source code**;
2. **Report**: describe steps and results of each tasks, as well as your comments if any. For task results, you can use figures or tables. Remember to provide some necessary descriptions if there are any figures or tables in your reports.