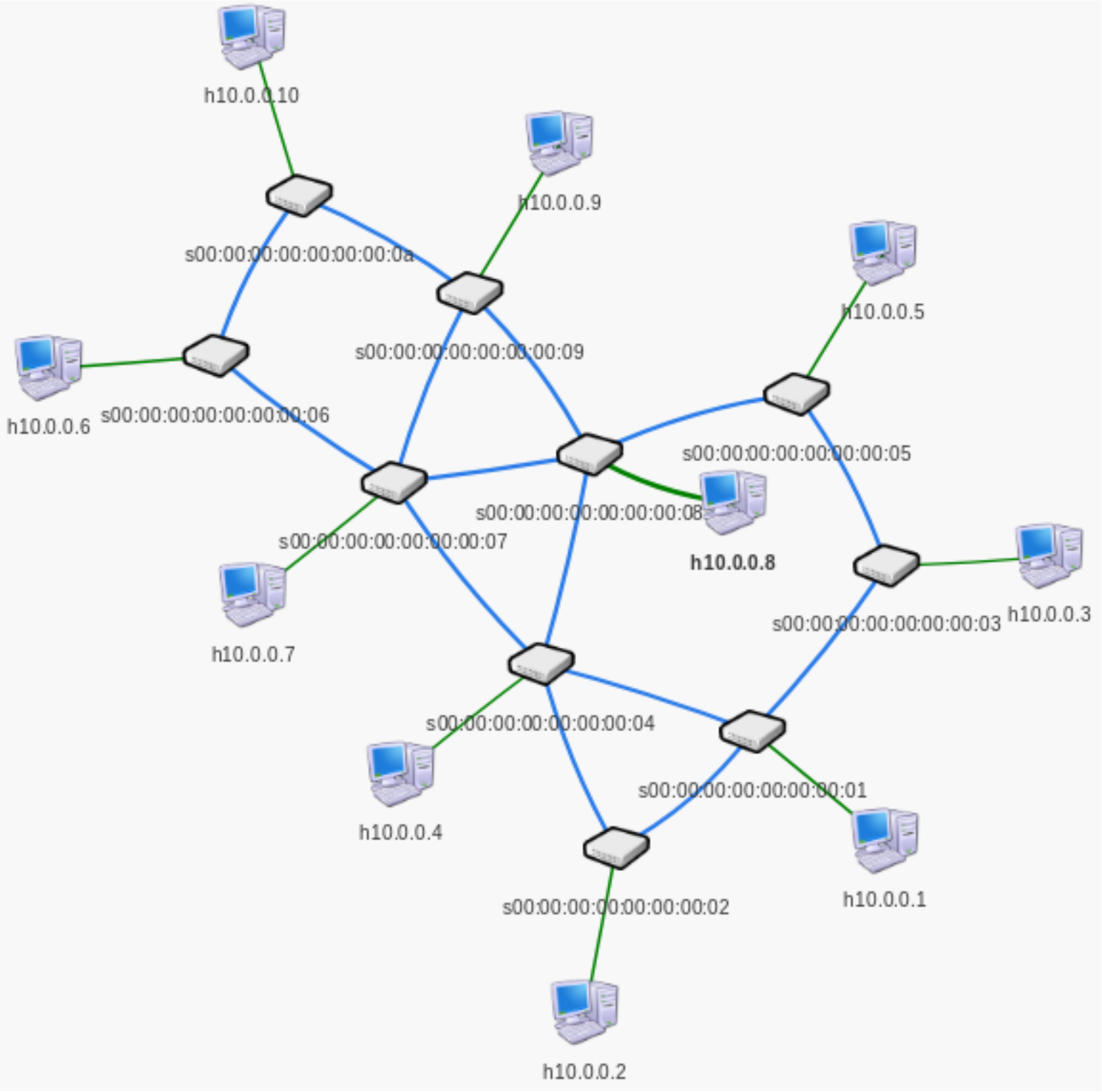
Jaber Daneshamooz, 9325893

Advanced Computer Networks

Software Defined Networking Project 2

Project Report

MPLS with floodlight



The topology of the network

This project contains 2 source codes written in python language. The first one is "project.py" which creates the network topology in Mininet. At the beginning of the code, I created 10 hosts and 10 switches in a for loop naming from h1 to h10 and s1 to s10. Then I connected switches together in a special manner to create the above topology.

In "mpls.py", I created 4 MPLS tunnels and pushed them into floodlight controller.

Flow from h10 to h3: The path contains s10, s9, s8, s5 and s3 switches, both for realtime and non-realtime traffic(same path). Flows with IP-TOS =0x05 are considered realtime and insert in queue number 1 of switches with higher rate and flows with IP-TOS = 0x0a are considered non-realtime(**we can ignore setting IP-TOS for non-realtime and consider it as default, and set higher priority for matching traffic that contain IP-TOS(to prevent a traffic match to 2 different flow entries) flag but I chose to set a value for it**) and insert in queue number 2 of switches with lower rate. The same path is used for both type of traffic.

Flow from h10 to h8: The first path contains s10, s9 and s8 switches. This path is used for realtime traffic with IP-TOS = 0x05 witch will be inserted in queue number 1 with higher data rate. The second path contains s10, s6, s7 and s8 switches. This path is used for non-realtime traffic with IP-TOS = 0x0a witch will be inserted in queue number 2with lower data rate. **Different path for different traffic is used.**

The Openvswitch v1 does not support MPLS but I checked the installed version on my PC by following command and it was version 2.5.5 so I had no problem.

```
$ ovs-vsctl show
```

First of all, you should run the Floodlight controller using:

```
$ java -jar target/floodlight.jar
```

Then, you should run "project.py" to create topology in Mininet using the following command. In this case, you should run switches in userspace mode to have no problem with MPLS because it runs in kernel mode as defualt.

```
$ sudo mn - - custom project.py - - topo mytopo - -
controller=remote,IP=127.0.0.1,port=6653 - - switch= ovsk,datapath=user
```

For putting different types of traffic(realtime and non-realtime) in 2 different queues, we should create 2 different queue with different rate on desired ports. I set 2 queues for nodes that have 2 or more tunnels to the same destination. I used the following command to create 2 queues at output interface of the switches:

```
ovs-vsctl set port s10-eth3 qos=@newqos -- --id=@newqos create qos type=linux-htb other-config:max-rate=200000000 queues:1=@EF queues:2=@AF -- --id=@EF create queue other-config:min-rate=80000000 other-config:max-rate=120000000 -- --id=@AF create queue other-config:min-rate=30000000 other-config:max-rate=80000000
```

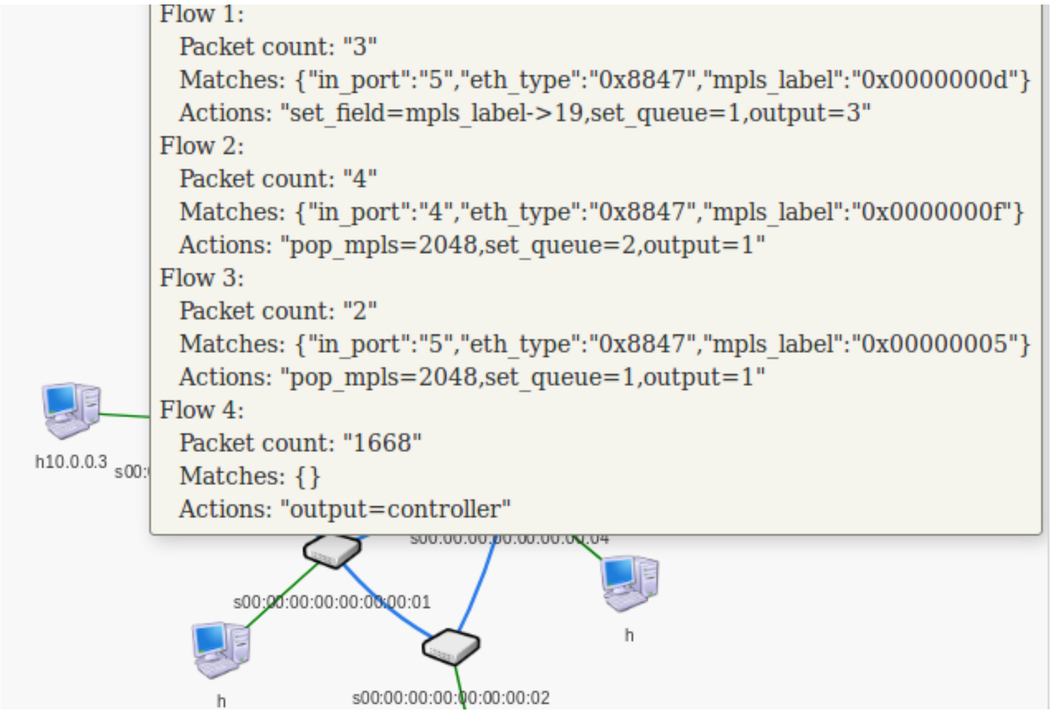
This code creates 2 queue at output interface 3 of switch10, for other switches would be the same and I only change the name of switch and port for creating queue on other switches. This code assumes that max-rate of 200Mbps for that port and sets max-rate of 120Mbps and 80Mbps for EF and AF classes respectively; and min-rate of 80Mbps and 30Mbps for EF and AF class. The EF class is used for realtime traffic(queue number 1) and AF class is used for non-realtime traffic(queue number 2)

After that, you should run the “mpls.py” file to push flows related to creating MPLS tunnels to flow tables of switches, then by using ping with modified header(to set DSCP tag) you can verify that it works correctly.

```
mininet> xterm h10
$ ping 10.0.0.3 -Q <tos>
```

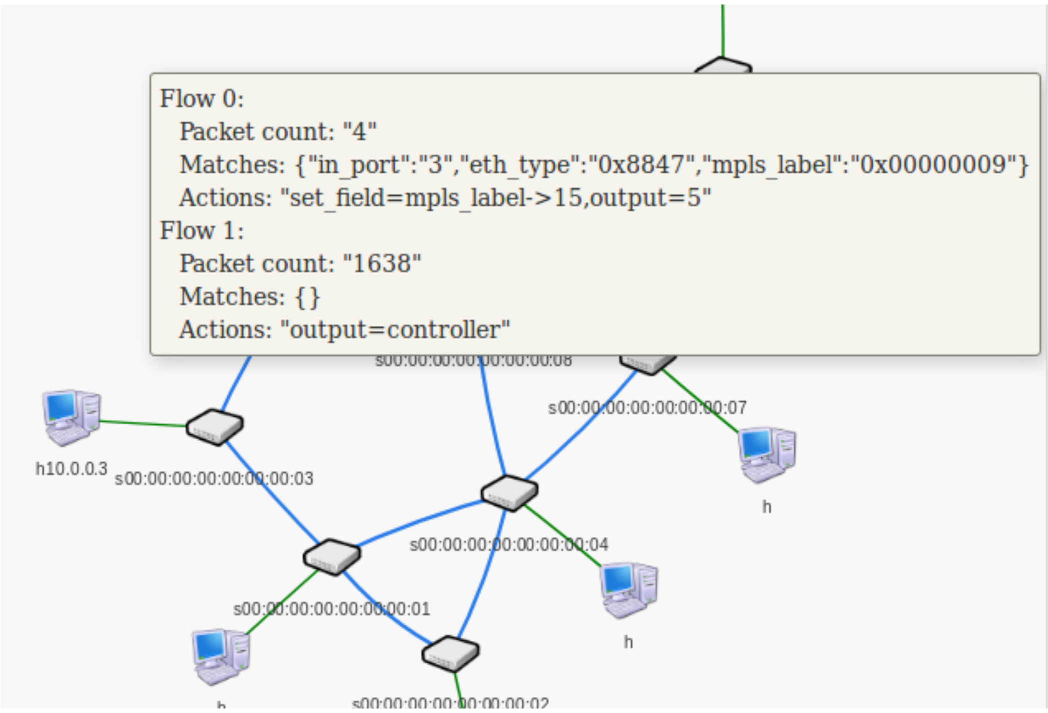
Put value 5 or10 as TOS field to indicate EF and AF traffic. You can also write this value in hexadecimal format: 0x05 or 0x0a.

After this, you are able to ping hosts through specified path and confirm the correctness of the code by looking at floodlight UI and the packet count associated with each flow. You can see that traffic passes through tunnels and different tunnels are used for realtime and non-realtime traffic. Here is a picture of s8 and s7 flow table with count of packets matched this type of flows(tunnel):



```
Flow 1:
  Packet count: "3"
  Matches: {"in_port":"5","eth_type":"0x8847","mpls_label":"0x0000000d"}
  Actions: "set_field=mpls_label->19,set_queue=1,output=3"
Flow 2:
  Packet count: "4"
  Matches: {"in_port":"4","eth_type":"0x8847","mpls_label":"0x0000000f"}
  Actions: "pop_mpls=2048,set_queue=2,output=1"
Flow 3:
  Packet count: "2"
  Matches: {"in_port":"5","eth_type":"0x8847","mpls_label":"0x00000005"}
  Actions: "pop_mpls=2048,set_queue=1,output=1"
Flow 4:
  Packet count: "1668"
  Matches: {}
  Actions: "output=controller"
```

S8



```
Flow 0:
  Packet count: "4"
  Matches: {"in_port":"3","eth_type":"0x8847","mpls_label":"0x00000009"}
  Actions: "set_field=mpls_label->15,output=5"
Flow 1:
  Packet count: "1638"
  Matches: {}
  Actions: "output=controller"
```

s7