**Johnny Sandiford 300365133**

**Lab 3 Report**

Before starting with task 1 the SimpleSwitch class needed to know the IP addresses and mac addresses of each host. As we are not implementing discover this was simple hardcoded in has value pairs, the host number paired with its associated information [ipv4\_addr, ipv4\_addr\_asInt, mac\_addr].

Please not the commented areas for each task as the code implemented is different in each task.

*Key Task 1*

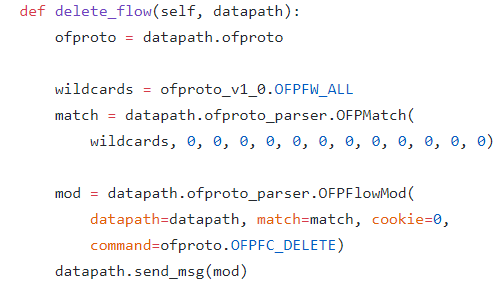
*Modify simple\_switch\_13.py to include logic to block traffic between host 2 and host 3.*

The first task requires the blocking of traffic between host 2 and 3. Before approaching this task the packet\_in\_handler and flows must be understood. The handling of a packet is expressed through the event handler method packet\_in\_handler, it has event ev as a parameter which contains the msg data.

The skeletal code contains code for the flooding and forwarding of packets based on mac\_to\_port entries, however it currently allows the installation of flows between hosts 2 and 3.

Flows are a way to rout packets without the added overhead of doing lookups and handling packets individually for forwarding by the switch. It is commonly used when handling streaming data to ensure smoothness and prevent bugs and jitteriness. RYU allows the installation of flows between end points to by pass the packet\_in\_handler method. Therefore, the flow must be deleted.

The way the flow is deleted follows the convention outlines in <https://github.com/osrg/ryu/blob/master/ryu/app/simple_switch_stp.py> under the delete\_flow method. Not the following code from this documentation:

**** The match structure basically represents a wrapped package specifying the input port the packet came in on. This combined with the MAC of the destination expresses a basic flow of data through the switch. A wildcard here is used to specify the removal of such flow. The mod uses OSFPlowMod to modify the datapath.

It is important to not here the data path is class object describing the openflowswitch.

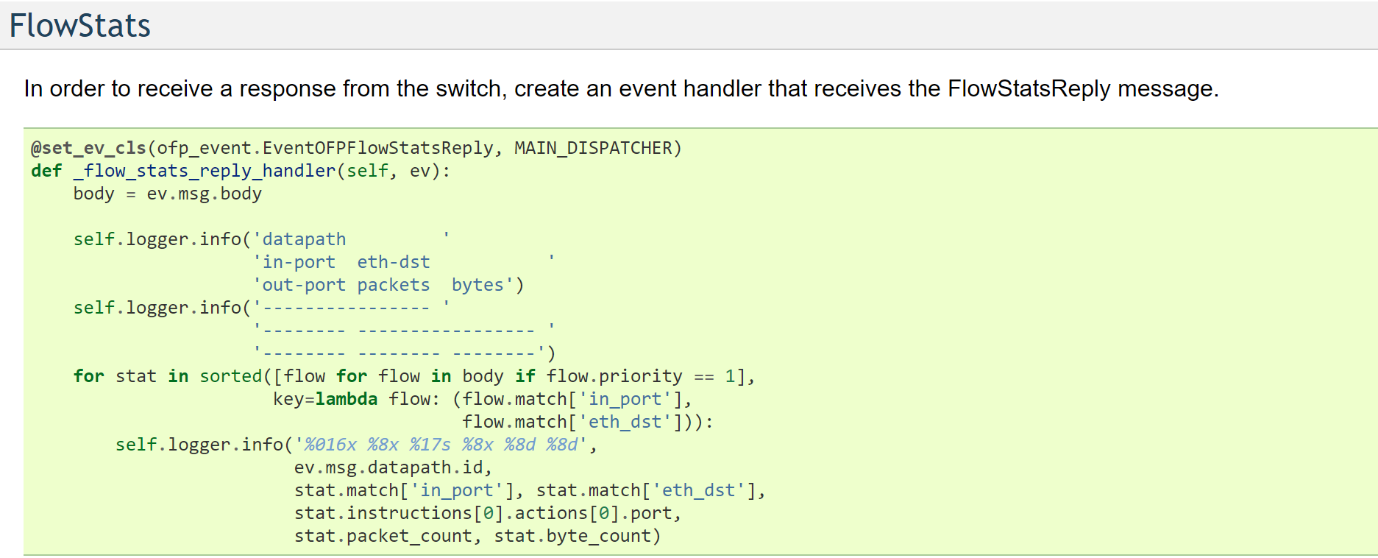
*Key Task 1: Testing*

|  |
| --- |
|  |

*Key Task 2*

*Extend simple\_switch\_13.py to count all traffic going to and originating from host1*

We could implement a simple counter that counted the number of packets coming from host 1. This would involve looking into the packet data and doing a match for host1 eth. Instead we can use EventOFPFlowStatsReply in the SimpleMonitor13 class.



This will involve defining a method to make a OFFlowStatsequest to receive the data for the flow originating at host1 and the flow where host1 is the destination. The reason for doing it this way is than rather than just get the count, we can get the number of bytes passing between hosts in the flow.

It is important here to discuss the @set\_ev\_cls class decorator, this decorator is used to get stats information from the Datapath. This method must also be modified to ensure we are only getting the stats where host1 is either the source or the destination.

*Key Task 2: Testing*