

The assignment is similar to assignment 1, but now we have 2 switches instead of 1.

You need to build an OpenFlow controller that implements two types of applications: firewall and implementation of quality of service.

**You need to complete the two files controller assignment2.py and topo\_Assignment2.py, as explained at the end of this text.**

All rules need to be installed dynamically, i.e., when packet arrive (**no rule hardcoding and at the node startup is allowed**), i.e. the controller operates in reactive mode.

**If a packet should be blocked, then a drop rule should also be installed** so that additional similar packets are dropped without going to the controller.

All installed rules should have a **soft timeout** of 40 seconds

**The rules are the following**, which are the same given for the first assignment:

* Data from H1 to H3 on TCP port 40 should be capped at 30 Mb/s
* Data from H1 to H2 on TCP port 60 should be capped at 150 Mb/s
* Data from H1 to H4 and H4 to H1 should be uncapped
* Data from H2 to H4 should be capped at 200 Mb/s
* Data from H3 to H4 and from H4 to H3 should be blocked
* Notice that H3 to H1, H2 to H1, H4 to H2 should be allowed in order for the transfer to work and uncapped.

To help you coding you will receive two template files, with some of the code already implemented. You only need to add your code where stated in these files (**you will see a => sign**):

* topo\_assignment2.py: this is the file where you can add code to define the topology above, add the ovs-vsctl QoS rules. This file also runs the iperf tests, but these have already been coded for you.

To run this code you will type: *sudo python3 topo\_assignment2.py*

* controller\_assignment2.py: this is the file where you add your code to define the controller behaviour.

To run this colde you will type: *./pox.py log.level --DEBUG misc.controller\_assignment2*

Additional notes:

* POX does not support meters, this is why we will use ovs-vsctl to install QoS rules (do this in the topo\_assignment2.py file).

When you run the files, you now that your work is correct if the results you get match the capacity you get from iperf, as defined by the rules above. Notice that you will not get exact match, as it is an emulation.

The POX library is available at: <https://github.com/att/pox>

These classes will be particularly useful:

<https://github.com/att/pox/blob/master/pox/openflow/libopenflow_01.py>

<https://github.com/att/pox/tree/master/pox/lib/packet>