**Software-Defined Networking Assignment 2 2020-21:**

**“Design, create and debug an SDN network consisting of multiple network elements (SDN switches and controller).”**

## Description

Write a Mininet script in Python to create a small data centre network using a spine-leaf topology. The script should take a parameter ‘n’, which corresponds to the number of switches at each level of the topology. The topology should include 2 hosts connected to each leaf switch, and a remote controller. Name the switches ‘S1’, ‘S2’,… , and the hosts ‘H1’, ‘H2’, … The switches should be Open vSwitch instances, and the remote controller must be Pox.

The example topology below would be created by running the script with parameter ‘n’ equal to 4. The controller is not shown.

**S1**

**S2**

**S3**

**S4**

**S5**

**S6**

**S7**

**S8**

**H1**

**H2**

**H3**

**H4**

**H5**

**H6**

**H7**

**H8**

Write an application for the controller to proactively add rules that don’t time out for the following functions:

* H1 and H2 should be able to exchange any kind of traffic with each other (reachability functionality).
* H3 and H4 should never be able to communicate with each other (traffic isolation, similar to VLAN functionality).
* H1 should be able to telnet and SSH to H5, but no other traffic should be allowed through (stateless firewall functionality).

Rules should be reactively configured to achieve the following functions, and the rules should have an idle timeout of 30 seconds:

* H1 should be able to telnet and SSH to H6, and H6 should be able to send any kind of traffic to H1 (stateful firewall functionality).
* For HTTP traffic going from H1 to H7/H8, every second flow should go to H7, and every other flow to H8 (load balancer functionality).

You will need to think about ARP. You can set static ARP entries on all hosts, or you can handle ARP requests in your own code, or use an existing controller application to deal with ARP.

You must submit a .zip file (named <student-number>-<firstname.surname>-assign2.zip) including only the following files:

* sdntopo2.py – a script to create the Mininet topology.
* policy.py – a Pox script to implement the policies listed above. Your code must be meaningfully commented.
* output.txt or output.docx – something that shows that the functions are achieved by your code (e.g. the commands to run your code, and screenshots of the output of testing with hping3 and tcpdump).

## Other Information

* Submission is through Canvas (<https://cit.instructure.com>) only.
* Assignment value: 40%
* Submission date: 20 Dec 2020 (end week 12)

## Marking

* Correctly following submission instructions –> 2 \* 3 = 6%
  + Correct naming of files,
  + Correct file formats – py, txt (or doc/docx) & zip
* Mininet script (sdntopo2.py) – (20%)
  + Scripting style (indentation, loops, parameters, clean code) –> 4 \* 2 = 8%
  + Correctly build topology (switches/hosts/links/controller present, named correctly –> 4 \* 3 = 12%
* Pox script (policy.py) – (61%)
  + Scripting style (indentation, clean code, meaningful comments) – 3 \* 3 = 9%
  + Proactive policy (in ConnectionUp: new rule, match criteria, action + timeout, sent to switches) –> 4 \* 3 \* 3 = 36%
    - Reachability rule
    - Traffic isolation rule
    - Stateless firewall rule
  + Reactive policy (in PacketIn: new rule, match criteria, action + timeout, sent to switch) -> 4 \* 2 \* 2 = 16%
    - Stateful firewall rule
    - Load balancer rule
* ARP consideration (static in Mininet script, dynamic in Pox script, existing Pox application) – 4%
* Evidence of successful operation of scripts (output.txt – topology created; proactive / reactive behaviour, Pox output) –> 3 \* 3 = 9 %

[Changes for next time: