COMS E6998-8 **Homework 2** Prof. Li Erran Li
Software Defined Networking September 24, 2013

Frenetic Controller Implementation Using Automatic Sequential Composition.
Due Date: 10/12/2013 18:00:00 EST

1 Overview

In this homework, you are asked to write a set of Frenetic modules and compose them to implement a Frenetic controller. [1].

2 Background Concepts

Frenetic [1] is a functional reactive programming language for SDN operating at a packet level of abstraction. Frenetic is designed to solve a major OpenFlow/NOX programming issue, a difficulty on compositing controller module features. This is achieved by abstracting the details of Openflow protocols, providing a simplified syntax, and enabling modular program development. Its syntax incorporates high-level, programmer-centric packet-processing operators, which makes composition easy.
A *monitor* is a feature that periodically watches how many packets or how long data have been transfered to a specified target (e.g. all the packets from a host 1). In frenetic, the monitor function, as most other primitive features, can be called easily. For example, *monitorLoad(10, "Packets From H1")* will print the monitor result every 10 seconds along with the given message. For more details, please use this as reference (https://github.com/frenetic-lang/frenetic/wiki/A-NCManual).

3 Question Specification

We have 5 questions. Each solution should be located in Qx where x is the question number. Use a MAC address of 00:00:00:00:00:0x to recognize host x.

Q1 Write a simple hub that floods all packets.
$sudo\ mn --topo\ single, 4 --controller\ remote --mac$

Q2 Copy Q1 and expand it to work as a hub and a simple monitor that counts all packets going to or leaving hosts with a odd number for the last digit of their MAC address.
$sudo\ mn --topo\ linear, 6 --controller\ remote --mac$

Q3 Copy Q1 file and modify to block HTTP (port 80) traffic between hosts 2 and 4.
$sudo\ mn --topo\ single, 4 --controller\ remote --mac$

Q4 Make Q3's firewall feature an independent module (file). Write a main module that "include"s the module and works as a firewalling repeater by composition.
$sudo\ mn --topo\ linear, 3 --controller\ remote --mac$

Q5 Implement a firewalling and monitoring learning switch. For the firewall feature : 1) block all arp packets, 2) block any HTTP packets among host 3,4,5, and 3) allow all the rest packets. For monitor, show the overall status (i.e. all the packets under the network). Use 'learn' function to call the learning switch.
$sudo\ mn --topo\ tree, depth = 2, fanout = 3 --controller\ remote --mac$

4 Preparation Steps

1. Download a VM image from "http://www.cs.umass.edu/ arjun/download/frenetic.ova".
2. Open the image from your VM environment (VMWare, VirtualBox, or etc.) and power on.
3. After the image finishes booting, login with frenetic / frenetic.
4. Change password ASAP.
5. Follow Frenetic Tutorial [2] Section 6 to 9.
6. Clone hw2 git repo by typing 'git clone https://github.com/jcybha/SDN_hw2.git'.

5 Work Steps

1. Since Frenetic follows OCaml syntax, if you have not programmed in OCaml before please read this "http://ocaml.org/tutorials/basics.html"
   and this "http://ocaml.org/tutorials/structure_of_ocaml_programs.html".
2. It is also recommended to go over this interactive tutorial on OCaml "http://try.ocamlpro.com"
3. Change working directory by 'cd SDN_hw2'.
4. Modify the files in 'Q1' according to the specification above and launch Frenetic 'frenetic Main.nc' in the directory Q1.
5. Test your code by launching mininet. Please use the give mininet command for each question.
6. Finish all the questions on Qx directories.
7. Create an archive by typing './make_archive.sh'. You should be able to see an archive file, named 'sdn-hw2-UNI.bz2' (Note that it will include Qx directory only.)
8. Upload the archive file to the courseworks' hw2 page.

6 Grading Rules

Maximum Grade: 100
Minimum Grade: 0

Implementation for each question +20

Coding style -1 per case
Use case failure -5 per case
Late submission -$\sum_{i=1}^{d} 50/i$, where d is the round-up number of days beyond the deadline.

7 Late Policy

Follows the late submission guide in Grading Rules Section.

References

[1] Nate Foster et al. Frenetic: a network programming language. *SIGPLAN Not.*, 46(9):279–291, September 2011.

[2] https://github.com/frenetic-lang/frenetic/wiki/Frenetic Tutorial. Frenetic tutorial. 2013.