CMPUT379 – Assign 2
Project Report

Brady Pomerleau

Objectives:

To get an introduction to using fifos (named pipes) for interprocess communication, as well as an introduction to Software Defined Networks (SDN).

Design Overview:

- a2sdn
    - program can be executed in 2 modes:
        - controller: the network manager. Invoked from the command line using form:
          a2sdn const [nSwitch]
          where nSwitch is the number of switches to be connected.
        - Switch: router in the network. Invoked from the comman line using form:
          a2sdn sw[id] trafficFile sw[port1]/null sw[port2]/null [range]
          where id is the number of the switch, port 1 is the switch on the left (null if there is none), port2 is the switch on the right(null if there is none), and range is the set of addresses serviced by the switch.
    - Switches are arranged in a line from left to right, starting at 1.
    - A switch starts with one rule (to route its packets to its own serviceable network), and can query additional rules from the controller when it doesn't know what to do with a packet.
    - Switches read from the trafficfile -ignoring comments- and handle the packet either with an existing rule, or with a rule obtained after querying the controller.
    - Packets are passed along the network to the destination if the packet is serviceable by one of the switches in the network.
    - Typing list in either execution mode prints information about the traffic run through the controller/switch.
    - Typing exit prints the same information and then exits the program.
    - Sending SIGUSR1 to process behaves as though list were entered.

Project Status:

All functionality is implemented as described.

Notes:

Controller waits until all switches have connected. This results in no serviceable packets being dropped.

Changes in the network (ie. Switch disconnecting) are not accounted for.

Testing and Results:

The following tests were conducted on university lab computers:

Test	Description	Result
t1.dat	Run t1.dat. Obeserve that handshake occurs and correct stats reported after t1.dat has been parsed.	success
t2.dat	Run t2.dat. Obeserve that queries are responded to with correct rules, packets are not dropped, and correct stats reported after t2.dat has been parsed.	success
t3.dat	Run t3.dat. Obeserve that packets are passed along several routers and correct stats reported after t1.dat has been parsed.	success
List and exit	List and exit perform as intended	success
SIGUSR1 handled	Send SIGUSR1 to various processes mid-execution to verify operating as inteded.	~success didn't have time to implement solution where printing didn't happen during sighandler time.

Acknowledgements:

To complete this assignment I used the following resources:

- Course textbooks:
  - Operating Systems Concepts ver.9
  - Advanced Programming in the Unix Environment ver.3
- various C programming errors troubleshooting accomplished using Google search and forums (primarily stackoverflow.com)