Hilton Truong
1615505
Assignment 3 CMPUT 379 W22

Objectives

The objective of this assignment is to get familiar with developing network switches and communicating peer processes that use signals to get the status of the running process. FIFOs are also used for the communication and the input and output multiplexing for nonblocking input output. The knowledge from class and the labs will be applied throughout the development of the program, allowing for us to develop a program that will act as a controller and switch depending on the command line input given. This assignment also builds on assignment 2 and uses TCP sockets in order to check if the connection has dropped or not.

Design Overview

All implemented functions have basic error checking to see if it's a valid command or not. The program will also run up to a maximum of 7 switches.

The program will cover 2 main commands, one to initialize the controller and one to initialize the switch,

The command to initialize the controller is formatted as such:

./a3w22 master nSwitch

Where nSwitch is the number of switches that you wish to initialize. Once it is started, it will print out an information blurb if anything is received. It will also continually poll the user for two possible inputs. Info and Exit. If the user enters info, it will display the information of the switch like such:

If the user enters exit, the same information as above will display then the program will exit.

Switch information:
[psw1] port1= -1, port2= -1, port3= 100-110
Packet Stats:
Received: HELLO:1, ASK:0
Transmitted: HELLO_ACK:1, ADD:0

The command to initialize the switch is formatted as such:

./a2w22 pswi dataFile (null|pswj) (null|pswk) IPlow-IPhigh

Where: *pswi* is the switch number

dataFile is the desired datafile that needs to be parsed and read

(null|pswj) and *(null|pswk)* are the switches that the switch is connected to

IPlow-IPhigh are the acceptable ipranges that the traffic will be processed.

Similar to the master nSwitch, it will also continually poll the user for two possible inputs. Info and Exit. If the user enters info, it will display the information of the switch like such:

Forwarding table:

[0] (srcIP= 0-1000, destIP= 100-110, action= FORWARD:3, pktCount= 3)

Packet Stats:

Received: ADMIT:3, HELLO_ACK:1, ADD:0, RELAYIN:0

Transmitted: HELLO:1, ASK:0, RELAYOUT:0

A makefile is provided to compile the program from the .cpp file and create a tar file containing the submission.

If the connection to the switch is lost it will display a message:

Lost connection to swi.

Project Status

The project is complete and can handle most of the requirements specified in the assignment pdf. The makefile will compile and provide the executable for the function. All the above commands work, and most commands have some basic error checking, however common sense and basic logic will need to be used as the program does not handle advanced commands such as arrow up and ctrl commands. I could not figure out how to get the delay implemented, however the code is still there however it will only run sometimes.

The hardest part of the implementation was figuring out how to get the parse through the traffic file and get the information sent from switch.cpp to controller.cpp to be recognized.

Sorry if the project doesn't work, I really tried my best. I've outlined the socket stuff that are supposed to be implemented in the file as I understand that part more than assignment 2.

Testing and Results

The testing of this program was done with the guidelines specified on eClass in the example output 1-3 pdfs. Each function was run and tested and made sure that they worked and provided the output that they were supposed to. The only issue that I could not get the delay function to work in ex3.dat. The program will also print what it's received for partial marks. The style of most of the outputs were matched to what was specified in the pdfs. All the functions work as they are intended to if they are not abused.

Acknowledgments

Code provided on the CMPUT-379 eClass – starter.cc was used for a lot of the imports and the general structure of the program

The use of stringstreams was learned from: https://www.cplusplus.com/reference/sstream/stringstream/

The textbook Reference in Advanced Programming in the Unix Environment [APUE 3/E] was used to read up on the functions in the table such as chdir(), fork(), execlp() and etc.

The course notes from Professor Elmallah were also used to implement some of functions.

Other references are made in the code in as comments.