CMPUT 379 - Assignment 2

Objectives

The objective of this assignment can provide hands-on experience in using Linux system API, such as, nonblocking I/O with `poll()`, using `FIFO` for process communication and developed peer-to-peer communication protocol.

Design Overview

- Each component (controller and switcher) has its own handler that are invoking from the main function, keep code modular.
- Keep the code as dry as possible by seperating repeated code into small helper function

Project Status

All required features are implemented with error handling for most important functions, however, some edge cases may not be handled properly.

Testing and Results

1. Start controller with `a2sdn cont 2`
2. Start `sw1` with `a2sdn sw1 t2.dat null sw2 100-110`
3. Observe `controller` and `sw1 have properly handshaken, and flow table and switch information has been properly updated
4. Start `sw2` with `a2sdn sw2 t2.dat sw1 null 200-210`
5. Observe `sw2` and `controller` has properly exchanged `OPEN` and `ACK` packets
6. `sw2` encounters a unknown destination IP in traffic file
7. `sw2` sends a `QUERY` packet to controller
8. `sw2` received `ACK` from controller then relay the packet to `sw1`
9. `sw2` relaied another packet to `sw1`

Acknowledgments

- https://stackoverflow.com/questions/2784500/how-to-send-a-simple-string-between-two-programs-using-pipes
- https://stackoverflow.com/a/1488815

Packet payload design

OPEN

```
OPEN <switch_num> <left_switch> <right switch> <range_low> <range_low>
```

QUERY

```
QUERY <switch_num> <address>
```

ADD

`ADD <switch_num> <left_switch> <right switch> <range_low> <range_low>` OR `ADD <switch_num> <ip>` for no result

RELAY

`RELAY <switch_num>`