Compare “learningSwitch” and “intentForward”, state in your opinion, how are they different from each other? Do they behavior differently with varied topology?

Learning Switch

The learning switch application is designed to function as a traditional learning switch within an SDN environment. Its key functionality involves learning the source MAC addresses of incoming packets and dynamically updating the forwarding table to facilitate efficient packet forwarding. The learning switch relies on flooding packets when the destination is unknown and subsequently updating the forwarding table as the network topology is discovered.

When I attempted to use learningSwitch with a ring topology, it was observed that a significant number of packets were often dropped. This behaviour is expected due to the nature of a ring topology, where switches are interconnected in a circular fashion. The learning switch encounters difficulties in finding a proper forwarding path within this circular setup, potentially resulting in packet loops among the switches. In contrast, the intent-forwarding approach is designed to handle such scenarios effectively. Intent forwarding possesses an awareness of the entire network topology, allowing it to virtually eliminate circular dependencies and prevent the issues encountered by learning Switch in a ring topology.

Intent Forward

On guarantee provided by ONOS intent service - it can reroute traffic if alternative paths exist between hosts. However, intent service cannot recover connectivity if a link failure leads to complete disconnection between two hosts.

This helped me understand that while intent service provides built-in fault tolerance by leveraging alternative paths, it cannot inherently ensure resilience if the physical topology itself gets partitioned. The application logic needs to be aware of scenarios where no alternate paths exist between hosts and handle such corner cases appropriately.

Difficulties faced during learning/programming on ONOS and how do you solved them?

The first problem was that this assignment could not even run on my laptop. I had to run it on the laptops in SOC labs which were even harder to use. The tutorial would not run on those PCs either. A lot of times when working on the assignment in the lab it would say insufficient memory left so I had to restart the whole assignment on a different laptop.

Another challenge was facing unfamiliar errors while setting up the assignment. Things like using the right address for ONOS GUI, running *mvn clean install* command had issues, learning about how to do a reinstall sometimes, etc. All these problems were encountered which made this assignment a lot harder to do and I had to refer to Piazza for solutions.

There was also unfamiliarity with the Java API for all the ONOS packages that were being imported in the files. To overcome this challenge I had to read the ONOS documentation to understand the usage and functionality of each API. Along with that VS Code helped to give drop-down options to figure out what each API does and takes in as a parameter.