**Part 3 (Layer-3 Shortest-Path Switching Routing) Testing and Debugging**

We tested out implementation primarily using the provided Mininet topologies and the ‘pingall’ and ‘iperf’ commands. We verified the following:

* Correct computation and installation of forwarding rules based on Dijkstra’s algorithm.
* Bidirectional connectivity for all hosts after topology updates.
* Handling of topology changes by inspecting the re-installation of flow rules in response to link events.
* Dumped flow tables to ensure rules were being correctly installed.
* Verified logs for topology update messages and checked that rules matched expected source-destination IP pairs.

For example, we ran the generated jar file using “java -jar FloodlightWithApps.jar -cf shortestPathSwitching.prop” and tested it against several of the provided topologies using the provided ‘run\_minitest’ Python script:

**Single host topology:**

A computer screen shot of a program

AI-generated content may be incorrect.

**Linear topology**

A computer screen shot of a program

AI-generated content may be incorrect.

**Tree topology**

A screenshot of a computer program

AI-generated content may be incorrect.

**Sample floodlight logs:**

A screenshot of a computer screen

AI-generated content may be incorrect.

For debugging:

* We used extensive logging to track path computations and rule installations.
* Added temporary print statements to the receive() method and topology update handlers so I could track how Dijkstra’s algorithm was computing paths and which ports were being selected. This helped confirm the logic was working as expected.

**Part 4 (Load Balancer) - Testing and Debugging**

* Testing focused on ARP handling, TCP SYN interception, and connection redirection to backend servers:
* Verified correct ARP replies by inspecting ‘arp -n’ and ensuring the VIP was associated with the expected virtual MAC.
* Used ‘curl http://<insert-VIP>’ from a host to generate TCP SYN packets and checked Floodlight logs to ensure ‘receive()’ was invoked.
* Checked that backend server received the SYN with the VIP as the original destination and the packet was rewritten correctly.
* Verified connection-specific flow rules were installed with the correct timeouts.
* Validated that non-SYN packets to the VIP triggered TCP RST responses.
* Manually flushed and reset IP/MAC settings to rule out stale ARP or routing entries when debugging.

For debugging:

* Inserted ‘System.out.println’ and ‘log.info()’ statements into the ‘receive()’ method and ARP handling block.
* Used ‘tcpdump’ within Mininet hosts (‘mnexec’) to inspect whether ARP requests were being sent and replies received.
* Confirmed backend MAC addresses using ‘getHostMACAddress()’ and verified with device manager outputs.