Optimizing Network Traffic through Dynamic Load Balancing

Problem Statement:

In networking, the efficient distribution of network traffic is crucial for maintaining optimal performance, minimizing latency, and maximizing resource utilization. Unevenly distributed traffic patterns between a source and destination can lead to suboptimal network conditions, resulting in potential bottlenecks and degraded user experience.

The challenge addressed by this project is to design and implement a dynamic load-balancing solution that intelligently selects the most efficient path between a source and destination based on real-time traffic conditions. The primary issues to be tackled include:

1. **Traffic Imbalance:** Unequal distribution of network traffic across available paths can lead to congestion on certain routes, causing delays and reducing overall network efficiency.
2. **Resource Underutilization:** Inefficient load balancing can result in underutilization of available network resources, limiting the scalability and performance of the overall system.
3. **Dynamic Traffic Variations:** Networks experience dynamic changes in traffic patterns due to varying user demands, application requirements, and network topology alterations. A static load-balancing approach may fail to adapt to these fluctuations.

This project aims to address these challenges by implementing a load-balancing algorithm that dynamically evaluates and selects the shortest path with the minimum traffic between a source and destination. Through this approach, the load balancer seeks to optimize network performance, reduce latency, and enhance overall resource utilization in response to changing network conditions.

Approach:

1. Specify the source node and the destination node for which load balancing is to be performed.

```
Enter Host 1
2

Enter Host 2
3
```

2. Examine all the paths between the source node and the destination node.

3. Utilize Dijkstra's algorithm to determine the shortest paths between the source and destination. This ensures the exclusion of longer paths, limiting the computation to a select few.

```
All Paths

[2, 10, 1]
[2, 21, 1]
```

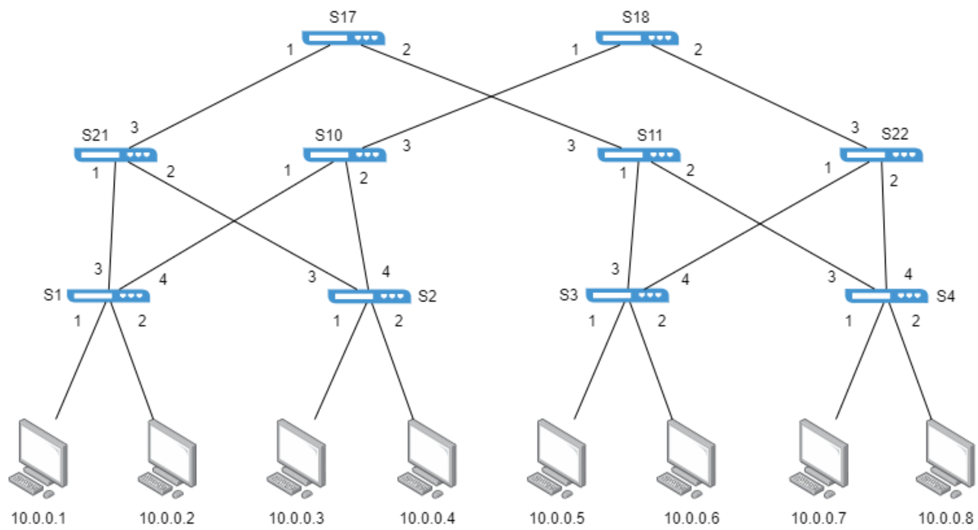
4. Retrieve real-time traffic information for the identified shortest paths.

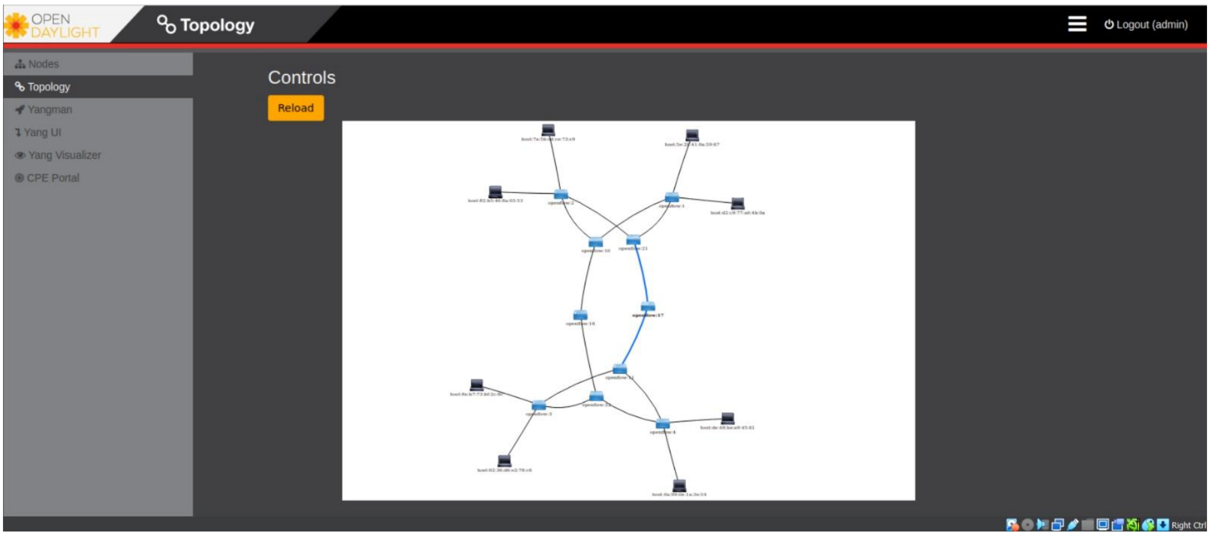
```
Cost Computation....

TxRate...
1172
cost
0
Cost Computation....

TxRate...
1170
cost
2
Final Link Cost
{'2::10::1': 4, '2::21::1': 2}
```

5. Select the path with the least traffic as the best path and insert flow rules into switches along the chosen path.

```
Shortest Path:  2::21::1
```

Testing Topology:





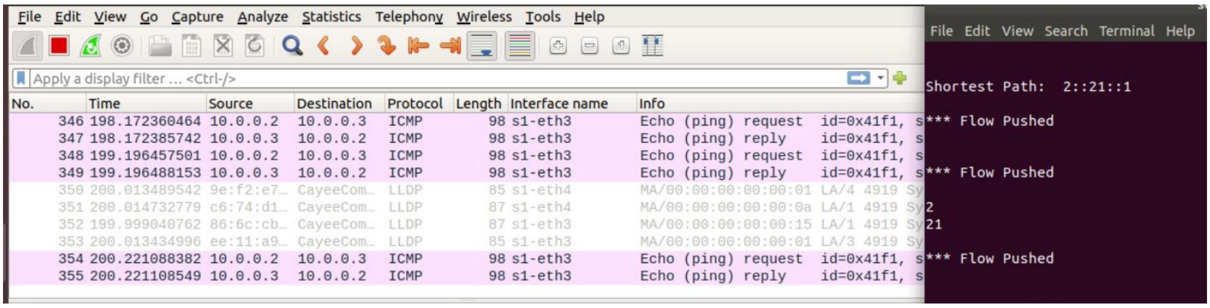
Discussion

When the shortest path(Minimum traffic path) changes, we can see a change of the interface name(port number) of the switch.

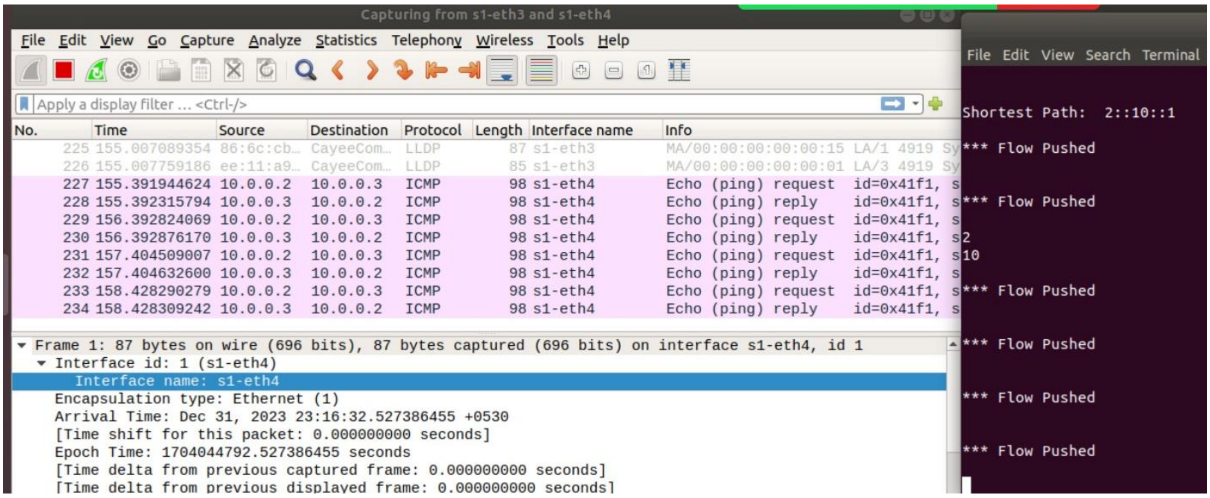
Ex:

Ping 10.0.0.2 to 10.0.0.3

When the shortest path (minimum traffic path) is 2::21::1 (s2-s21-s1) we can see the s1 interface is eth3.



Then the shortest path (minimum traffic path) is changed to 2::10::1 (s2-s10-s1). We can see the s1 interface is eth4.


Capturing from s1-eth3 and s1-eth4
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>
No. Time Source Destination Protocol Length Interface name Info
225 155.007089354 86:6c:cb… CayeeCom… LLDP 87 s1-eth3 MA/00:00:00:00:00:15 LA/1 4919 Sy
226 155.007759186 ee:11:a9… CayeeCom… LLDP 85 s1-eth3 MA/00:00:00:00:00:01 LA/3 4919 Sy
227 155.391944624 10.0.0.2 10.0.0.3 ICMP 98 s1-eth4 Echo (ping) request id=0x41f1, s
228 155.392315794 10.0.0.3 10.0.0.2 ICMP 98 s1-eth4 Echo (ping) reply id=0x41f1, s
229 156.392824069 10.0.0.2 10.0.0.3 ICMP 98 s1-eth4 Echo (ping) request id=0x41f1, s
230 156.392876170 10.0.0.3 10.0.0.2 ICMP 98 s1-eth4 Echo (ping) reply id=0x41f1, s
231 157.404509007 10.0.0.2 10.0.0.3 ICMP 98 s1-eth4 Echo (ping) request id=0x41f1, s
232 157.404632600 10.0.0.3 10.0.0.2 ICMP 98 s1-eth4 Echo (ping) reply id=0x41f1, s
233 158.428290279 10.0.0.2 10.0.0.3 ICMP 98 s1-eth4 Echo (ping) request id=0x41f1, s
234 158.428309242 10.0.0.3 10.0.0.2 ICMP 98 s1-eth4 Echo (ping) reply id=0x41f1, s
Frame 1: 87 bytes on wire (696 bits), 87 bytes captured (696 bits) on interface s1-eth4, id 1
Interface id: 1 (s1-eth4)
Interface name: s1-eth4
Encapsulation type: Ethernet (1)
Arrival Time: Dec 31, 2023 23:16:32.527386455 +0530
[Time shift for this packet: 0.000000000 seconds]
Epoch Time: 1704044792.527386455 seconds
[Time delta from previous captured frame: 0.000000000 seconds]
[Time delta from previous displayed frame: 0.000000000 seconds]
File Edit View Search Terminal
Shortest Path: 2::10::1
*** Flow Pushed
*** Flow Pushed
2
10
*** Flow Pushed
*** Flow Pushed
*** Flow Pushed
*** Flow Pushed