OpenFlow-based Adaptive Routing for Wireless Networks

Alok Kulkarni
akulkar4@ncsu.edu

Angelyn Arputha Babu John
ababujo@ncsu.edu

Jignesh Darji
jndarji@ncsu.edu

Nishad Sabnis
nsabnis@ncsu.edu

`https://sites.google.com/a/ncsu.edu/openflow-based-adaptive-routing-for-wireless-networks`
November 5th, 2015

1 Introduction

1.1 Problem Statement

Designing a system to enable adaptive routing in a wireless network in order to make a comparative analysis of the throughput efficiency of ground nodes versus aerial nodes.

1.2 Problem Description

We aim to implement OpenFlow based adaptive routing in an ad-hoc network by monitoring the link quality between wireless nodes. We anticipate that in such a network which offers multiple wireless routes between two end-points, the fluctuations in the RF link qualities between the endpoints will play an important role in determining the best end to end path. Determining the wireless link quality between each and every inter-connected node and making routing decisions based on this information constitute the two major parts of the problem.

We plan to make aerial nodes a part of the network which will be used for testing. Aerial nodes have their own set of advantages and disadvantages. They are less susceptible to electromagnetic interferences and can beam wifi over a large area if the antennae are powerful enough. However, the number of aerial nodes and naturally the number of available links through such nodes is likely to be lesser due to the low prevalence of such nodes. These trade offs need to be accounted for while making the routing decisions as well. The final aim is to ensure that the flow tables are dynamically modified to ensure effective end to end packet transmission.

2 Components

2.1 Platforms for the project

- OpenFlow v1.3
- Ubuntu
- SDN Controller

2.2 Areas for the project

- Link Quality Monitoring
- Adaptive Routing

2.3 Major Components

2.3.1 Wireless Ground Nodes

The ground nodes will be laptops. They will have the following features:

- At least one wireless interface
- At least one interface to connect to the host
- Open vSwitch module installed
- One of the ground nodes will be the controller

2.3.2 Aerial Nodes

The aerial nodes will have the capacity to go up till 30 feet and beem signals from above. The key features of the aerial nodes are:

- At least one wireless interface
- BeagleBone black Linux boards
- Open vSwitch kernel module installed

2.3.3 Software Components

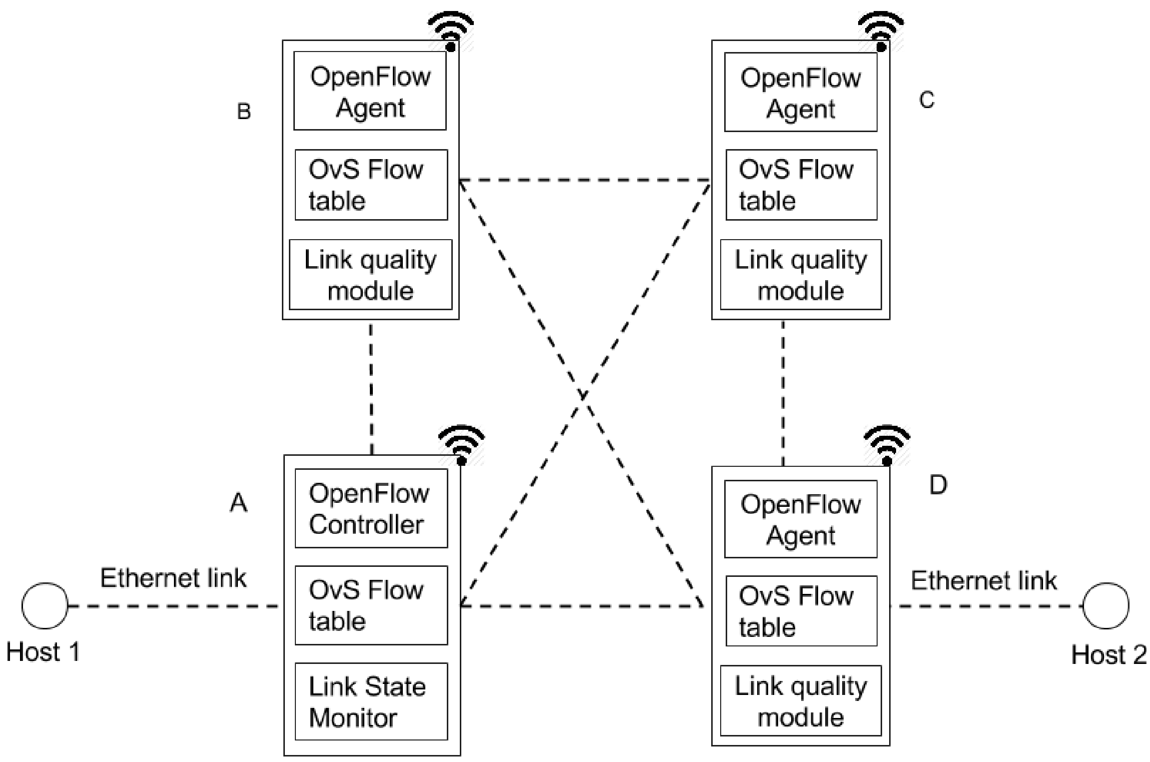
The software components will help determine the link quality and the optimum path, and they will configure the network with the optimal path.

- Link Quality Information module
- OpenFlow Control module

3 Design

3.1 Overview

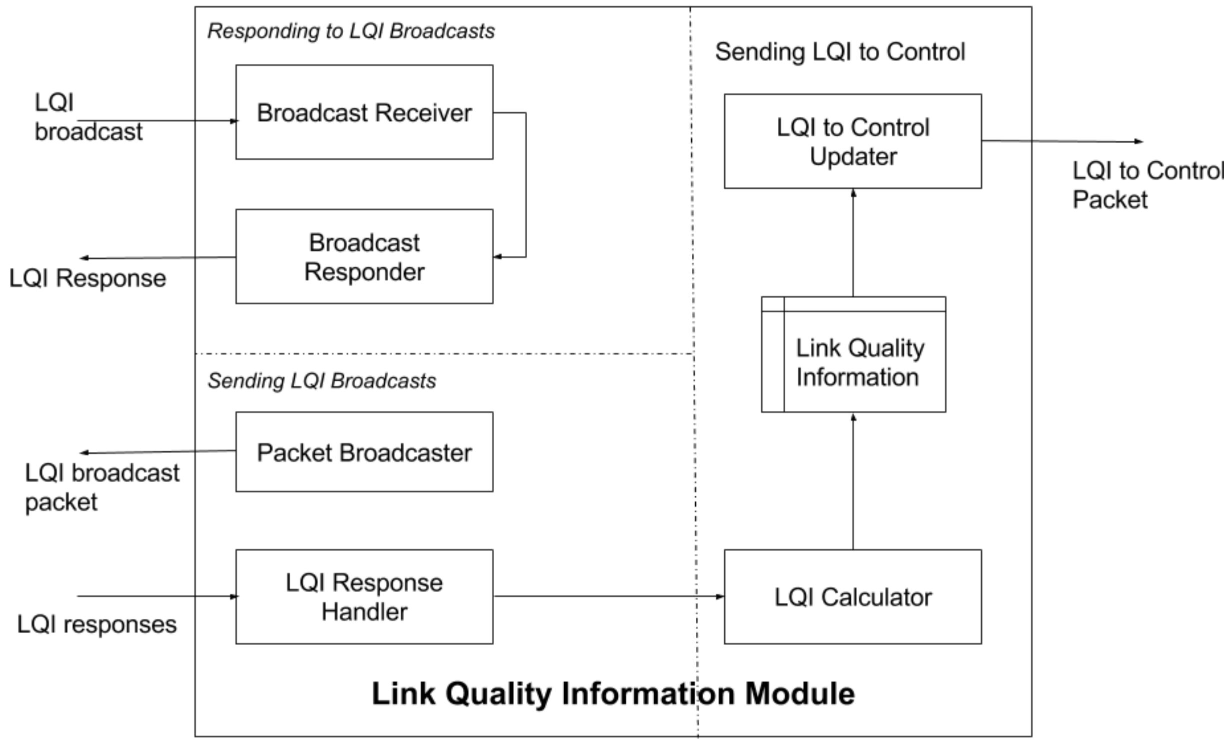
Figure 1: Design Overview



The above figure describes the overview of the components constituting this system. There exists a wireless ad-hoc network of aerial and ground nodes. One of the nodes acts as an SDN controller and the others act as agents. A link quality information (LQI) module is running on all the nodes in the network and this information is forwarded to the OpenFlow controller which makes adaptive routing decisions. The controller will use this information to compute the optimum end-to-end route between the endpoints. These routes will then be configured into the nodes using OpenFlow. The nodes will have OVS running on them which will configure the routes sent by the Controller.

3.2 Link Quality Information Module

Figure 2: Components in the Link Quality Information Module



This module is responsible for estimating the point-to-point link quality between nodes. It will run on all the nodes in the topology and give a periodically updated link quality estimate to the SDN controller. This information can then be used by the SDN controller to update the topology and find the optimal path. The module is made up of two units:

- *Request unit*: It sends out the UDP broadcast and receives the responses for these broadcasts (Packet Broadcaster, Response Handler).
- *Response unit*: It receives the UDP broadcast and responds to it (Broadcast Receiver, Broadcast Responder).

3.2.1 Packet Broadcaster

The LQI packet broadcaster will send out a UDP broadcast on the ad-hoc network. The broadcast is directed at a specific UDP port to differentiate it from normal UDP traffic. The packet broadcaster will send out a broadcast packet after a 10 second interval. The UDP frame will look as follows:

Source Port	<specific destination port>
Length	Checksum
"LQI Hello Packet" <payload>	

3.2.2 Broadcast Receiver

Will listen on a specific port for LQI Hello packets and trigger the Broadcast Responder once it has verified the frame format for each.

3.2.3 Broadcast Responder

Will prepare a unicast response for the broadcast source. The response will be sent out to a specific UDP port and will have the following format.

Source Port	<specific destination port>
Length	Checksum
"LQI Response Packet" <payload>	

3.2.4 Response Handler

The LQI Response Handler will listen at a specific port for any LQI responses, parse the LQI packet for obtaining the SSI, source and destination IP and MAC, and forward the packet to the LQI calculator. The module will have a timeout within which it expects to receive all the responses and will update the LQI Calculator after the timeout. The radiotap header (LINKTYPE_IEEE802_11_RADIOTAP) is an alternative to the normal 802.11 header (LINKTYPE_IEEE802_11) and and contains some extra information about the wireless network along with the normal 802.11 header. The radiotap header itself consists of a set of defined fields from which antenna signal (SSI) is to be extracted. The position of the Radiotap header in the frame is as shown below.

Frame(Physical Layer)	**Radiotap Header**	IP	UDP	UDP Payload

3.2.5 LQI Calculator

This module will get the SSI, source and destination IP and MAC address values from the Response Handler for all the responses it receives. These value will be converted and stored in an LQI table in a format which can be further sent to the controller.

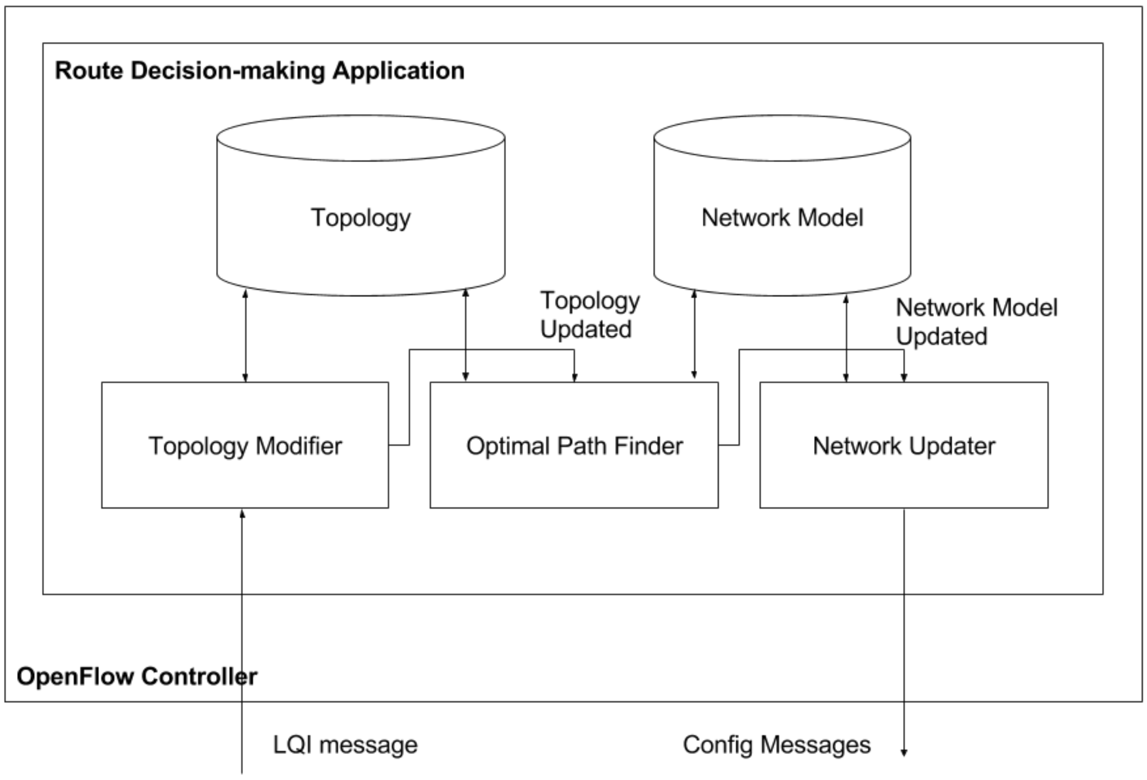
3.2.6 LQI to Control Updater

This module will refer to the values stored by the Calculator in LQI information table and prepare a payload that will be sent to the controller. This packet will be used by the controller to create a topology and find the optimal path. The format of the payload is as follows:

Source Port	Destination Port	Length	Checksum
SRC_IP			
SRC_MAC			Padding
Neighbor 1 IP			
Neighbor 1 MAC			SSI(1 byte)
Neighbor 2 IP			
Neighbor 2 MAC			SSI(1 byte)

3.3 OpenFlow Control Module

Figure 3: Components in the OpenFlow Control application



3.3.1 Topology Modifier(TM)

Function: Generate and maintain the topology of the network and intimate Optimal Path Finder about topology change.

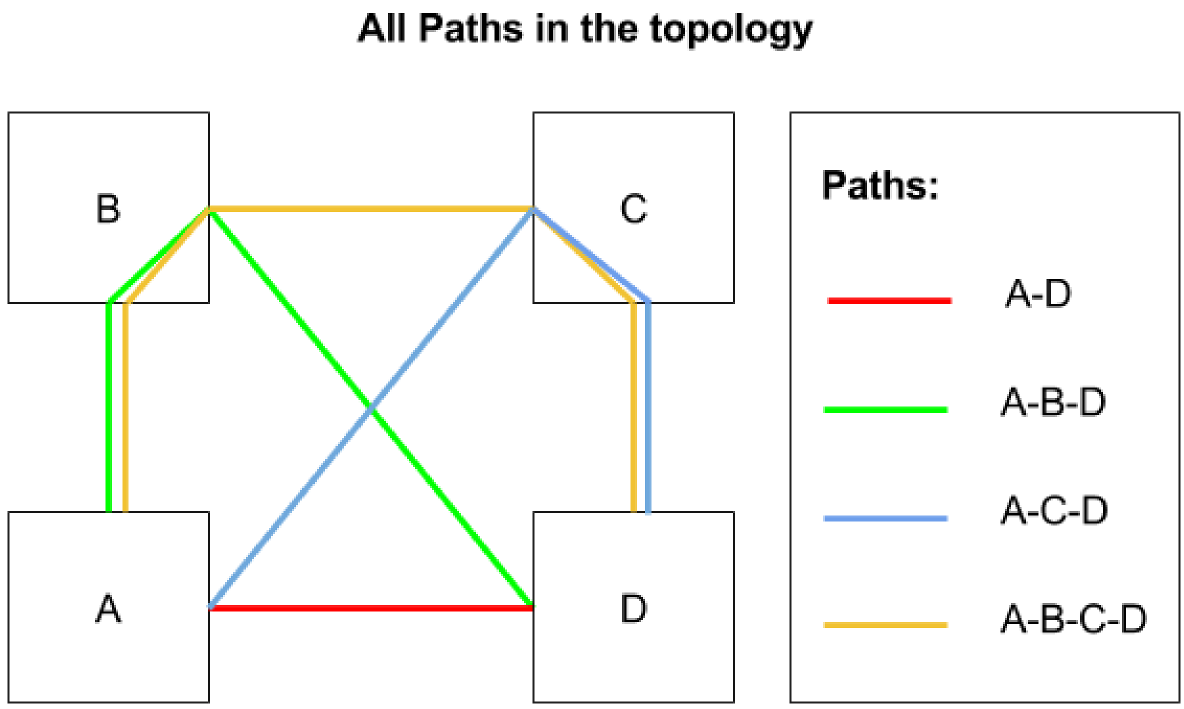
- Topology Construction: TM module accept the LQI messages and constructs the adjacency lists for all the nodes in the network. The node data structure will contain following information be as follows:
  - Node IP
  - Node MAC
  - Adjacent Nodes(IP and MAC) along with cost to reach them
- *Node Removal*: For each node, we shall have a running timer. If we do not get N consecutive LQI messages from a node, we shall remove the node from the topology. Calculation of N will be done later. We cannot have a high N because the network convergence time would be affected. We cannot keep it low because that will mean a lot of reconfigurations by the Network Updater if the node was not out of network or down but its LQI messages were dropped for a transient duration due to other factors.
- *Topology Updates*: The TM module will be invoked to update the topology when one of the following event occurs in the network:
  - *Node Addition*: When a node is added, it will be added to the adjacency list and cost to reach it from other nodes, and costs from that new node to other nodes need to be updated.
  - *Node Removal*: When a node is removed from the network(See Node Removal above), we need to remove its entry.
  - *Cost Updates*: When the LQI message from a node is received with different costs, the adjacency list of that node shall be update.
- *Callback to Optimal Path Finder*: When the topology is updated, it will send a callback to the OPF module.

3.3.2 Optimal Path Finder(OPF)

Function: Find the lowest cost path between the end hosts.

- *Limited Paths*: Since the topology is limited - 4 nodes and 2 of them connect to the end hosts - we have limited end-to-end paths between the 2 edge nodes.

Figure 4: Paths between edge nodes

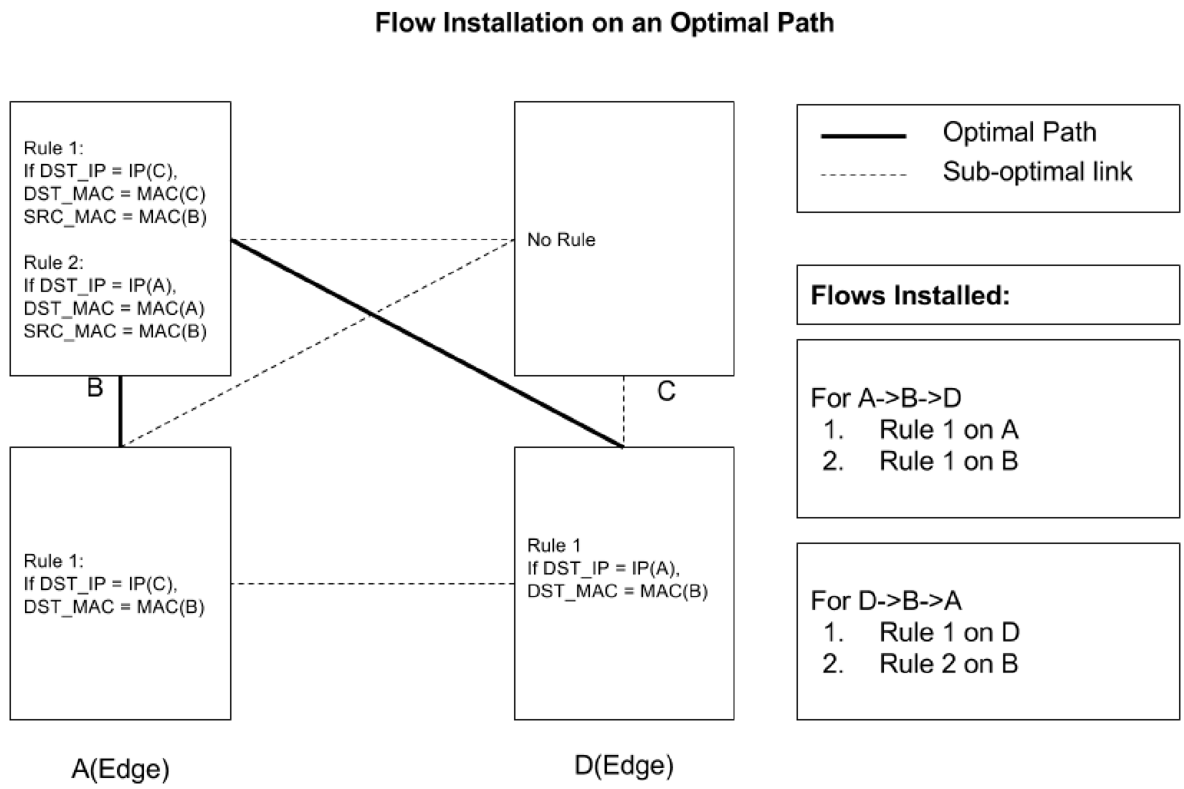


- *Path Selection*: OPF will calculate path costs for all the 4 paths and select the least cost path.
- *Callback to Network Updater*: If the new path is different from the previous path, we send a callback to the Network Updater.

3.3.3 Network Updater(NU)

- *Problem*: Since the network is wireless ad-hoc and all the nodes can generally connect to each other, the destination node will accept the packet directly. We need to enforce a method so that the packet follows the optimal path calculated by the OPF and not the direct path between edge nodes.
- *Algorithm*:
  - *Edge Nodes*: Install single rule on edge nodes to modify the DST_MAC to its next nodes MAC. Here, next node is the adjacent node in the optimal path.
  - *Internal Nodes*: Install two rules on internal nodes to modify the DST_MAC as well as SRC_MAC for both the directions. A simple illustration will clarify this step.
- *Example*: In the below illustration, the edge nodes are A and D and the optimal path is A-B-D. To enforce forwarding along A-¿B-¿D through the optimal path, A sends to B by changing the DST_MAC to Bs MAC and then B sends to D by changing the DST_MAC to Ds MAC and SRC_MAC to Bs MAC. Similar rules on D and B are required to enable for D-¿B-¿A.

Figure 5: Flow Rule Illustration



4 Per-member Responsibilities

Tasks	**Angelyn Arputha Babu John**	**Jignesh Darji**	**Nishad Sabnis**	**Alok Kulkarni**
Node Setup	Implement	Implement	Implement	Implement
Creation of ad-hoc network	Review	Review	Implement	Implement
LQI Message Handling	Review	Review	Implement	Implement
LQI Calculator	Review	Review	Review	Implement
LQI to Control Updater	Review	Review	Implement	Review
Topology Modifier	Implement	Implement	Review	Review
Optimal Path Finder	Implement	Implement	Review	Review
Network Updater	Implement	Implement	Review	Review

5 Test Plan

Test	**LQI Module**	**Topology Modifier(TM)**	**Optimal Path Finder (OPF)**	**Network Updater(NU)**
Add a node with low path cost	Connected nodes should transmit cost with the new node to the LQI messages	Add node to Topology; intimate OPF	Add to Network Model; intimate NU	Update affected nodes
Add a node with high path cost	Connected nodes should transmit cost with the new node to the LQI messages	Add node to Topology; intimate OPF	Network Model remains same	NA
Remove node from Network Model	LQI messages will not contain costs to this node	Remove node from Topology; intimate OPF	Modifies Network Model; intimate NU	Update affected nodes
Remove node not part of Network Model	LQI messages will not contain costs to this node	Remove node from Topology; intimate OPF	Network Model remains same	NA
Increase associated path cost of a node in Network Model	Cost to this node should be increased in the LQI messages	Update path costs in Topology; intimate OPF	Modifies Network Model; intimate NU	Update affected nodes
Increase associated path cost of a node NOT in Network Model	Cost to this node should be increased in the LQI messages	Update path costs in Topology; intimate OPF	Network Model remains same	NA
Decrease associated path cost of a node in Network Model	Cost to this node should be decreased in the LQI messages	Update path costs in Topology; intimate OPF	Network Model remains same	NA
Decrease associated path cost of a node NOT in Network Model	Cost to this node should be decreased in the LQI messages	Update path costs in Topology; intimate OPF	Modifies Network Model; intimate NU	Update affected nodes

6 Demo Plan

1. Once the ad-hoc network is setup, the controller will generate the Network Model and configure the rules in all the nodes for the optimal path.
2. A display utility for the Network Model will print the current Network Model.
3. We'll display that these rules have been correctly configured in the nodes by connecting to each node and displaying their rule tables.
4. Thus, on changing the distance between the nodes, we plan to show that the routes between the two endpoints change dynamically based on the link strength.