|  |
| --- |
| School of electrical engineering and computer sciences |
| Distance Vector Routing |
| Semester Project Report |
|  |
| **EE353: Computer Networks** |
| **4 June 2017** |

|  |
| --- |
|  |

## Submitted By:

Sohaib Zahid (126491)

Shahzad Nasiruddin Virani (139414)

# Introduction:

Our project is basically an implementation of the Distance-Vector Routing protocol in Python. For this implementation, we have used various concepts including the Bellman-Ford algorithm, socket programming and multi-threading.

# Techniques and Methodology used for implementation:

We have implemented our program in Python. We have defined 2 classes in our program that will be used throughout (‘Neighbor’ and ‘Path’).

The definitions of these 2 terms are as follows:

1. Neighbor: The link between any two nodes that are directly connected to each other.
2. Path: The link between any two nodes that may or may not be directly connected to each other.

\*To further clarify: Every neighbor is a path but every path is not necessarily a neighbor as well.

Our program contains several threads, some of these are permanent (are active throughout the program) and some of these are temporary (are only active for some time).

The permanents threads are as follows:

* Listening: This thread listens for “Distance Vectors” sent by the neighbors, and creates the temporary Bellman-Ford thread when an update is detected.
* Menu: This thread acts as the main interface for the program. The interface (menu) presents the user with 4 options:
* To view the shortest paths to the other routers in the networks
* To view the whole distance vector table
* To edit the link cost(s)
* To quit the program
* Timeout Checking: This thread determines whether a router is dead or not. After every second it checks if all the neighbors are listening at their designated ports. If an exception is found, the router is considered to be dead.

The temporary threads are as follows:

* Bellman-Ford: This thread executes the Bellman-Ford algorithm on the distance vectors it has received from the neighbors. It is started whenever an updated Distance Vector is received from a neighbor.
* SendDV: This thread sends the router’s Distance Vector to all the neighbors. It is created every time an update in the Distance Vector is detected.

To cater for link cost changes and re-instantiation of routers, the routers send their link cost in addition to the Distance Vector to the neighbors, so that the neighbors can appropriately update the link cost.

# How relevant problems in development combatted:

Count-to-Infinity: When the link costs were increased or a router failed, the network was unable to converge due to count-to-infinity problem. To counter this problem, we used the technique of **‘Split Horizon with Reverse Poisoning’.**

Count-to-Infinity Extended: In some rare situations we saw that this problem occurred in larger loops instead of 2 routers. In order to resolve this problem, we did not execute Bellman-Ford algorithm immediately after a router failure. Instead, the distance to the dead router was advertised as infinity first. Through this, the neighbors of the router got to know that the path is outdated. Hence, the path to the dead router got expired (set as infinity) and the router looping problem was resolved.

Change in Link Costs: When link costs were changed through a given interface, there was no way for the other neighbor to find out about this change. To make the cost of link consistent in both directions, as soon as a change is made, the router sends the updated link cost in addition to the Distance Vector to the concerned neighbor. The neighbor can then update the cost to the new value.

Router Re-instantiation: Whenever a router is created, it sends its link cost in addition to the Distance Vector to all the neighbors. When the neighbors receive this packet, they change the link cost to this router from ‘infinity’ to the specified value.

# Conclusion:

The project was an ideal way to practically apply the theoretical concepts we have learned. Through the project, we learned how to solve a real-life networking issue. Moreover, we also learned the intricacies of distance-vector routing protocols and their significance in modern networks.