Programming Assignment 3
Routing Protocols

by Manoj Mariappan

The Experiment involved creating and maintaining routing tables of servers using UDP. The routing table of a server is sent to its neighbors as UDP messages. The messages are constructed using a char buffer i.e. they are just bytes of messages encoded and sent on UDP. Each server listens on a UDP socket waiting for incoming packets of information. The information received is the neighbors own Distance vector. It uses this vector to construct its own vector and then communicate this information further to its neighbors.

I followed the same message format as provided in the Assignment guide. The header of the packets contained number of update fields, Server port, and Server IP. This was followed by 12 Byte sections of node and cost information. I used the header to interpret how much information was being followed. If the value of update count was n then there would be 8 + n * 12 Bytes of information in the UDP buffer.

We load the basic information from the topology file and start a server which uses select statement to switch between user input, incoming UDP messages and timer updates. The user may view the distance vector of the current node which reveals the cost and next hop to the node, may change the topology settings such as alter the cost, disable a link, simulate a crash. This changes the overall topology and affects the individual nodes distance vector.

The different functions have been explained in the code with comments which ultimately serve the above mentioned functions. The important data structures used in the assignment were routing_info at line 20, server_info at line 25 and neighbor_info at 31 in server.h. The routing_info holds cost and next hop information and is used as a vector of vector to store the whole routing table. The routing table is created at line 37 in server.cpp as oRoutingTable. The server_info was used to store the information about the servers, neighbor_info was used to store the information about the neighbors of a node. Another important feature was to maintain a different timer for each neighbor and also maintaining a timer of its own to periodically to send the updates to its neighbors. This is done by using the struct `timeval` of select. We use only one timer to simulate multiple timers by storing time for each node in an array and selecting the next minimum each time a timer interrupt occurs.

The primary functions used in the server.cpp are as follows:

openListenSocket: This opens a socket and listens on it.

GetIpFromBuffer and **addIpToBuffer** : Copies bytes of IP from string to the buffer which is to be transferred over UDP and converts buffer to IP.

UpdateRoutingTable: This does the main job of modifying the routing table based on changes in the topology. This is called after every change in the topology.

SendRoutingUpdate: This send one's own distance vector to all its neighbors. This serves an important function. It starts creating the packet structure at line 196. The buffer starts storing all the information and is finally sent over UDP.

ProcessUserInput: This processes the user input and takes necessary actions. The important function provided to the user are update which helps in modifying costs of existing links, step which triggers a sending of one's own distance vector, packets which lists the count of packets received since this command was last called, display which displays the distance vector of a node, disable which disables a link to specified server and crash which simulates a crash and the application becomes unresponsive.

ProcessIncomingMessage: This processes UDP packets and also modifies the routing table by calling updateRoutingTable in the end.

GetMinimumTime and **decTimerBy**: These are functions used to manage the timer.

RemoveLink: Used to remove a link from neighbor_info vector. This is called when a neighbor has stopped messaging for 3 consecutive timeouts. This doesn't modify the cost of a link to infinity, instead it sets it inactive by modifying the active flag in the neighbor_info struct. In this way when a packet appears later from the link it becomes alive with the previous value.

StartServer: This is the entry point from the main function. It initializes the routing table and enters the loop to switch between various functions using select statement.