

**School of Engineering & School of Computer Science and Statistics**

**3D3– Computer Networks**

|  |  |
| --- | --- |
| **Group Number** | 16 |
| **Title** | Assignment 2 Report |
| **Date of Submission** | 12/04/19 |

|  |  |
| --- | --- |
| **Member Name** | **Member Student Number** |
| Adam McQuade | 16326767 |
| Ailish Conneely | 16320336 |
| Cian Levy | XXXXXXXX |
| Edmond Cheng | XXXXXXXX |

By submitting this Laboratory Report we understand that we are agreeing with the following:

* We agree that this is all our own work and has not been taken from the work of others save where appropriately referenced in the body of the assignment.
* We have read and we understand the plagiarism provisions in the General Regulations of the University Calendar for the current year, found at http://www.tcd.ie/calendar.
* We have also completed the Online Tutorial on avoiding plagiarism ‘Ready Steady Write’, located at http://tcd-ie.libguides.com/plagiarism/ready-steady-write."

– Implementation description (structures, messages,routing/forwarding tables, etc).

- Difficulties that you faced and how you resolved them.

**Implementation**

The following classes were implemented in our solution to the task

* **Datagram**
* Encapsulates the datagrams transmitted between routers.
* Used by the router class to both construct and deconstruct datagrams.
* From the sender point of view the class will be used to multiplex the payload data and datagram header.
* From the receiver point of view the class will be used to demultiplex the payload data and header fields.
* **Distance-vector**
* This class encapsulates the distance vectors and their data.
* Used by the router to add all the data required to advertise a new link to a vector (buildDV) which can be sent to the other routers.
* When a router receives a new link advertisement the data will be extracted and parsed by this class.
* **Log**
* Responsible for recording: changes to the routing table, datagrams received and routed.
* The log will write to a file called "routing-outputX.txt" (where X is the router ID) which will be stored in the "resources" folder.
* Storing the state of the table prior to an update requires that the previous state is stored somewhere.
* When recordTableUpdate is called, "recordTableUpdate" will be one update behind. After the required records are written to the file previousTable can be overwritten with the current table state.
* **Route-table**
* Implements the Bellman-Ford algorithm which finds the least-cost path between routers.
* When router a router advertises a new link, checks the existing link cost.
* Table is updated if the new cost is less than the current.
* Returns a flag (e.g. 0x7FFFFFFF) if there are multiple updates, causing all links to be advertised again
* Returns -1 if there are no updates. If only one link was updated return the location of the link in the vector.
* **Router**
* Effectively implements a dynamically updated lookup table.
* When the router receives a datagram whose destination is X, the link in position X in the "table" vector specifies which router the datagram should be forwarded to based on the current least cost path.
* When a new link is advertised the path cost is calculated based on the values in the vector. If the new costis less than the current cost the link will be updated.