**SCHOOL OF DIGITAL MEDIA AND INFOCOMM TECHNOLOGY**

**DIPLOMA IN INFOCOMM SECURITY MANAGEMENT**

**Year 2**

**ST2614 PROGRAMMING USING PERL AND C**

**ASSIGNMENT 2 REPORT**

**By: Cheong Ren Hann, 1230068, DISM/FT/2A/02**

The basic layout of this project is focused on the implementation of the extraction and processing of traceroute / tracecert information.

The program starts by opening a file containing raw trace data and extracts the ipaddresses from that file and assigns it to a node that is linked with others depending on the topology of the nodes. The nodes are also linked to other via a linked list and then further sorted in order of ascending order of ipaddresses and then displayed on the screen.

The next part is then the processing of user commands and displaying data according to input provided. Data provided will be based on what the user queries which is then checked by the program and the nodes are then searched for the appropriate response.

The architecture of how the nodes are stored in the program is using linked lists. In this case, there are 3 separate links in each node, thereby making it a triply linked list; however it functions as a usual doubly linked list with an extra link for sorting and ordering purposes. The reason for the extra link is because, the doubly links are not the usual 1 – 1 links, instead a node can have multiple previous and next nodes; this results in a potentially complicated web in a large scale. As such it is necessary to have a single link that can pass through every node only once for display or other calculations.

The whole program has therefore 3 main parts: extraction of information, sorting/processing of information and handling user queries.

Firstly, the file is opened and read line by line. The program looks for certain patterns that signify an ipaddress within the line, namely numbers and “.”. In the case of hidden addresses “\*\*\*”, it is kept as such.

Secondly, after successfully picking out the addresses, the program then checks whether this address has been assigned a node previously, if not then it gets assigned a new node or else the existing node gets modified. During this process, the program also takes note of the previous nodes that were assigned / modified and updates the previous node link in the node. Also, the previous node’s next link gets updated accordingly. In addition, the nodes also are connected by id order based on when they were created by the 3rd link.

After processing the nodes, the program proceeds to sort the nodes by ascending order of addresses. How it does this is by finding the numerical value of each address and sorting based on that. The order link of the nodes then gets reassigned to the new order after the sort.

The nodes are then formatted and printed on the screen and into the output file. The program then prints the menu and waits for the user to query. All user queries are sanitised and checked, failing that the program informs the user of an invalid query.

Thirdly, there are 3 options for the user to query. The first two accepts both ipaddress and node id as input. The first two locates and prints the previous and next hop(s) of a node. These first attempt to locate the node indicted by the user, if found the program proceeds to print out the required data. The program differentiates an ipadress query from an id query by looking for “.” in the query.

As for the last query, it accepts an integer, n, and attempts to find the nodes n hops away. It starts by locating all possible first hops and works from there by recursion until n number of hops away and prints out the respective nodes.