COSC364

Assignment 1
RIP Routing

Tristan Christie
Student ID: 16801546

Clinton Walker
Student ID: 55356003

Contribution per person:

Tristan Christie-50%
Tristan contributed to the project by creating the config file reader script, functions that create update packets, some of the main function, triggered updates, the report, and tests

Clinton Walker-50%
Clinton contributed to the project by creating functions that bind given port numbers to sockets, most of the main function, the routing table and routing table updates, and functions to process incoming packets

Some of the aspects of our overall program that were particularly well done include the config file reader which allows for a wide range of comments in the config file, the Packet creating functions which is in big endian and follows the format given in the RIP specification document.

Some of the aspects of our overall program that we feel could have been improved were the planning of the code, and the code could have had better encapsulation so that the functions would be split into separate files based on the actions they perform.

We have ensured atomicity of event processing by using the select function call to wait for packets, instead of entering a loop until data arrives. There is a similar situation for everything else in the main function, where no waiting, e.g. for timers, is done inside the main loop so when a condition is met, the associated code is completed instantly, which allows for all operations to be isolated from each other.

Some of the weaknesses of the RIP routing protocol we have identified are that the network cannot support more than 15 hops between routers which means this implementation would not be able to support large networks, when a route is removed many triggered updates are sent quickly to ensure consistency between routers which causes a lot of traffic, and it takes a while for the timeout on a router to trigger so traffic could still be sent through a dead router for a while. Count to infinity is still a problem but it is not as drastic anymore as the maximum metric is 15, but this still takes a while because the current implementation would rely on periodic updates for this as triggered updates are reserved for when a route is invalid.

We performed tests on our code to ensure it worked. One of the tests was to make sure the configuration files would check for errors and raise an exception when there was an error. We tested this by adding a few comment lines, blank lines, and incorrectly formatted lines to a config file while keeping all the correct files needed to complete the configuration. The expected outcome was that the files would still load correctly and all the lines that did not match the correct formatting would be treated as comments. After printing the variables that were stored after reading the config files, we concluded that the actual outcome was the same as the expected outcome. We also tested the configuration files by removing lines necessary to set up a router. The expected outcome was that whenever a line needed for setting up a router was missing, the program would raise an error specifying what was missing. This was the actual outcome of the tests with the exception being timer values which default to 30-180-120 when not specified or incorrectly specified, a line is also printed stating the timer values were set to default. Through these tests, an acceptable format was made to allow for comments in configuration files.

To test the routing table format, we set a routers routing table to have route entries of it's neighbours designated in the config file, as we had not implemented the distance vector algorithm yet. The expected result of this was for the routing table to have entries for each neighbour that were consistent with the format of the printed routing table. The actual outcome was consistent with the expected outcome

To test the packet format and the sending of packets, we set up two config files to have each other as neighbours as well as a third router, ran them with the RIP script, and printed the messages they received. The expected outcome was for the received packets to be sent in the correct format, which is specified in the RIP document, and for the routing entry of the third router to have the metric specified in the config file while the other routing entry has a metric of 16 as the receiving router is the next hop for this entry and 16 is treated as infinite in this implementation of the RIP protocol. The actual outcome was consistent with the expected outcome.

We also performed tests on the implementation of the routing protocol and the routing table display. We did this by creating a group of config files that would create a network. We then ran the RIP script with these files and compared the output of the table with the expected output. After confirming the outputs were the same, we then shut down one of the routers to see if the router tables would converge to the expected routing tables. Once the tables had converged to the correct values, we then turned back on the router that was shut down, and checked the tables once again converged to the original values, which they did.

Example configuration file for the example network of Figure 1:

```
router-id 4

input-ports 1028 1031 1038

outputs 1029-4-3 1030-2-5 1039-6-7

timer-values 30-180-120
```