Project 2. Distance-Vector Routing

Due: 10/07/2022

1. Introduction

In this project, you will implement a distance-vector routing protocol called RIP. Each router on the network executes RIP to exchange routing information with its neighbors, and based on this information, the router computes the shortest paths from itself to all the other routers.

2. Network Topology Generation

The network will consist of four virtual routers, each of which is a process (or a thread) that runs on a computer connected to a different network. Use the following four computers as the virtual routers:

* queeg: 129.21.30.37
* comet: 129.21.34.80
* rhea: 129.21.37.49
* glados: 129.21.22.196

Their subnet mask is 255.255.255.0 (you can extract the network prefix using this subnet mask). Since a process represents a router, two routers can communicate via sockets. Use the UDP datagram socket. These routers are connected like a ring where each router is connected to two other neighbors, specifically, queeg – comet – rhea – glados – queeg. Ultimately, the routing table computed by each router has entries for these four networks. The cost of each link is given arbitrarily as user input. You create only a router, not a host, where RIP will run.

3. The RIP Routing Protocol

You first read the RIP specification (RFC 2453 for RIP 2) carefully. Be aware that you do not have to implement all the features defined in RFC 2453. Most of what you need can be found in Chapters 3 and 4. Specifically, your program is to support the functionalities described in Section 3.4 (the minimum requirement). You only need 4.3 Subnet mask and 4.4 Next hop from Chapter 4. After reading the RFC, you should be able to answer questions about the details of the protocol. Important questions are:

1) How does RIP broadcast route updates?
2) How does RIP handle incoming messages?
3) How does RIP support CIDR?
4) What does a routing table look like?

RIP has two different types: active RIP and passive RIP. Only a router can run RIP in active mode in advertising its routes to others; a host (passive RIP) listens to RIP messages and use them to update their routing table. Your implementation at least should support 1) active RIP at routers, 2) handling incoming route messages, 3) CIDR, and 4) route message broadcasts. Assume that only RIP 2 is used. Also, set the route update time to 1 second, not 30 seconds as defined in the RFC, to reduce the convergence time.

4. Router Failure

If a router fails, RIP should respond to recover routes if alternate routes are available. To see this effect clearly, implement router failure with some probability (e.g., 10% of the time) or with user command. The neighbors of a failed router should be informed that the failed router is indeed unreachable (distance becomes infinity). This information will be propagated to the entire network through the RIP routing protocol. As you probably know, triggered updates need to be invoked for fast recovery. However, do not implement triggered updaters, but implement split horizon with poisoned reverse to prevent the count-to-infinity problem. In your output, indicate "failure" for a failed router, and mark "split horizon" for route updates invoked from failures.

5. What You Need to Implement

In summary, you will need to implement:

1) Each router sends route updates to its neighbors periodically.
2) Each router should be able to update the routing table based on the route updates that it receives.
3) Your protocol should support CIDR, i.e., classless addressing, with masks.
4) Your protocol should have split horizon with poisoned reverse implemented to prevent the count-to-infinity problem.
5) Each router displays its routing table regularly.

6. Test and Demo

We will use a recorded video that shows a demo of your implementation for testing and grading. For the testing and demo, create the ring network topology of the four computers with neighbor and link cost information. This topology will also be used at the demo for grading. As the routers start exchanging route updates and compute their routing tables, each router prints its routing table periodically (e.g., every 1 second). The output includes the four fields: 1) destination IP address, 2) subnet mask, 3) next hop, and 4) distance. Also, your demo needs to show how router failures are handled. Enable some router failures, and highlight how the routing tables change as they cope with the failures.

7. Submission

Turn in all the files with a readme to the Assignment on MyCourses. Include a link to the demo video (not the video itself) in the readme, so that the TA and I can check.

8. FAQ

1) Is there any format do we have to follow for the output?
   A: No. The output should be the routing table at each router with each entry having those four fields (see Section 5).
2) What failure probability do we need to use?
   A: You are free to use any, e.g., 10%, 1 out of 10 seconds.
3) Do we have to implement any features from RIP 1?
   A: No.
4) Which files do we have to submit?
   A: All the files needed with a readme. No specific format is required.