A Policy-Based Routing (PBR) Router based on Distance-Vector Algorithm

1. Objectives

Design a DV-based router which simulates a RIP router on the Internet. The router can determine the shortest route based on the policy.

2. Requirements

- Design a program called "router".
- The "router" can be launched many times on one machine. Each launch will start up a new "router" process or thread, meaning that you deploy a new router. Killing a "router" process or thread means you shut down or remove a router.
- Fill and update the routing table of the "router" process using Distance Vector algorithm and specified policy.
- Exchange routing tables among "router" processes by UDP socket, just like RIP do.

3. Routing Table

Suppose the distance between nodes and their adjacent router is 1. The Routing Information Base (Routing table) uses the following structure:

Destination	Route
5	2,3,4

Where:

Destination: 5 is the destination number.

Route: 2,3,4 is the route to the destination 5. It means that the packet will go through 2, 3 and 4 to the destination 5 and 2 is the next node. To support the routing policy, the complete route is recorded in routing table, not the next hop. When transmitting packet, just think the first next node.

4. Lunch-up

To identify routers and their neighbors, when you run the program, it must receive the following parameters:

router ID, myport , port1, port2, port3…

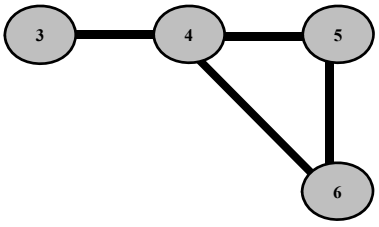
Among them ：

ID：the number of routers, digital 0～9

myport: The UDP port the router use to send and receive packet

port1, port2, port 3, …: The UDP ports that routers' neighbors use

For example, for a given topology like below:



You can run the program like this:

router 3, 3003, 3004
router 4, 3004, 3003, 3006, 3005
router 5, 3005, 3004, 3006
router 6, 3006, 3004, 3005

5. Supported commands

The program should support the following commands:

Command	Description
N	Print activity's adjacent list. Format: 4 5 (in one line, space separated) If having no neighbors，then print "Empty"
RT	Export the routes that reach to every destination node. Each route occupies one line. Format： Destination route 1 4, 3 4 5, 4
D n	Send a data packet with a specified or default TTL value to the destination that the number *n* represents. Each node need to print out the action taken on the data packet. The Packet should reach the destination within the specified time or steps in TTL, otherwise it will be discarded. Assume the destination is 5. **For the source sender,** ● if the destination is the adjacent node, send directly and print out "Direct to 5". ● if there are no any adjacent nodes, print out "No route to 5". **For the node that received the packet,** ● Make TTL minus 1. ● If TTL=0, discard it, and print out "time exceeded 5", otherwise continue to transmit it. ● If the node is the destination, print out "Destination 5". ● If the node is not the destination and the destination is in its base, print out "Forward to 5". ● If the node can't forward the packet (no route exists in its base) , discarded the packet and print out "Dropped 5".
P K n_1 n_2 … n_k	Specified priority route. The route to the destination n_k should include the specified intermediate nodes n_1 n_2 … . n_1 n_2 … n_k **:** IDs of all K nodes and n_k is the ID of destination

	node. Replace possible shortest route with possible priority route after the node receives this command.
R *n*	Refused to pass the node *n*. After the node receives this command, the node ignores all of the updates that contains node *n* in routing update.
S	Display the statistics of route updates.

6. About the time control

- Each router sends out their routing table every 30 seconds (the time value can be adjusted).
- Each router updates its own routing table according to the received routing table.
- Routers must have the ability to detect whether a neighbor is active. If the router does not receive the update from the neighbor in 30*6 seconds (the time value can be adjusted), it is considered that the neighbor is not reachable.

7. About topological structure

- Routers must have the ability to cope with failure and recovery. We assume that the link does not appear to be faulty, and the packet is not lost and don't occur errors.
- If the router is not running at the given time, there is a routing fault. If it restarts the operation, it is considered that the router fault recovery.

Of course, we assume that communication is bidirectional.

8. About routing loop

- Reasonable maximum value should be set in order to avoid infinite route loop.
- Measures should be taken to reduce the survival time of the loop. Measures include：
  - Split Horizon
  - Poison reversion

9. Test

Normal case: in steady state.

Bad news: kill or shutdown one or more router processes or threads. To examine and verify how split horizon and poison-reverse solve the problems.

Good news: restart the killed or launch new router processes or threads.

10. Programming

- Language: any (C, C++, Java, platform independent)
- User Interface：Design by yourself
  - Windows and Graphics (Better)
  - Console by input characters
- It is best to real-timely display the result，such as the process of routing update, the information of routing packet forwarding, etc.

11. Submission

(1) Project Report

Cover Page

- Project name
- Student Number
- Name
- Date
- School and University

Contents

- Objectives
- Description of Distance Vector Algorithm
- Requirement of experiment
- Programing language/Developing platform and tools
- Design ideas
- Data structures
- Implementation (development tools, model, definitions of objects/methods and processes and threads, processing flows, etc.)
- Test, Verification, Result Analysis, Performance Analysis, Screenshot with explanation

(2) Readme.txt

- Plain text format.
- Contain any bugs or issues that you know of in your code.
- It should also indicate how you run your code using command line.

(3) Source Codes

- The source of the program (Must have a program comments).

(4) Running Codes

- The compiled program that can run on Windows or Linux system.