

Extra Credit Project 2: RIP (Routing Information Protocol)

Document Version 2.0, 4/22/04

UPDATE REGARDING TESTING:

As I said in class, there will be no test box to test your RIP implementation, and you do not have to interoperate with other RIPs (though I'd give you some extra credit if you do). You have to develop your own test scenarios, and then take me through a demo that convinces me that your RIP works properly. You should run multiple of your IP boxes (each as a separate process listening on separate UDP ports), configure them into some kind of topology, and be able to show me the routing messages and resulting RIBs and FIBs when you modify the network (add and delete nodes). You do not actually have to send packets through the network, though a little extra credit if you do. Basically if I am convinced that the RIBs and FIBs are correct, then that is sufficient.

NOTE THE TEXT BELOW IS NOT CHANGED FROM THE ORIGINAL POSTING. SO THE ABOVE PARAGRAPHS SUPERCEDE ANYTHING BELOW.

Worth 3 MEng project credits. Must be a single-person project.

Goal: To implement a working RIP.

Task: You must build a near-fully functional RIP routing protocol, as described in RFC 2453. You do not need to implement authentication (section 4.1), route tags (section 4.2), next hop (section 4.4), and the compatibility switch (section 5.1). Everything else, however, should be implemented.

Your implementation should define infinity as 16 hops, and should use the timer values suggested by RFC 2453.

Your implementation will run over the UDP module you created in project 2 (real RIP runs directly over UDP), and interface with the forwarding table you created in project 3.

Test: We will test your RIP by actually routing packets over a network composed of your IP boxes. Specifically, you will start some number of IP boxes (probably around 30 of them), each of which will contact the test box. The test box will then tell each IP box which interfaces to create, and whether each interface is a subnet with hosts, or another router. In the latter case, the interfaces will be with other instances of your IP box.

Specifically the test box will send the following types of commands to the IP boxes (via the send_config sockets):

Create router link	This command will cause a link to be established between two IP boxes (i.e. the IP box will create a UDP iface socket to emulate the link interface). The IP box will establish this link and start running RIP over it.
Delete router link	
Add subnet	This command will cause the IP box to establish an iface

	socket with the test box to emulate a subnet. There will be a subnet address associated with this subnet, which of course will have to be advertised to other IP boxes using RIP.
Delete subnet	

The test box will create and delete links and subnets at will. The test box will also insert packets into the subnet interfaces, and expects to receive these packets via the appropriate subnet interfaces.

This is illustrated below. Here we see that the test box has asked each IP box to establish two links, as shown. The IP boxes on the right and left have subnets with subnet numbers 10.1.2/24 and 10.1.1/24 respectively. After creating these links, the IP Boxes would run RIP and establish routing tables. Later, if the test box sends an IP packet to the IP box on the right destined to 10.1.1.1, we would expect the IP boxes to forward the packet through the network of IP boxes and deliver it to subnet 10.1.1/24 (i.e. transmit the packet to the test box via that subnet interface).

Note: later on we will give you more details as to the exact test process. For now this should be enough to get you started.

