CHAPTER 23

Routing Information Protocol

Background

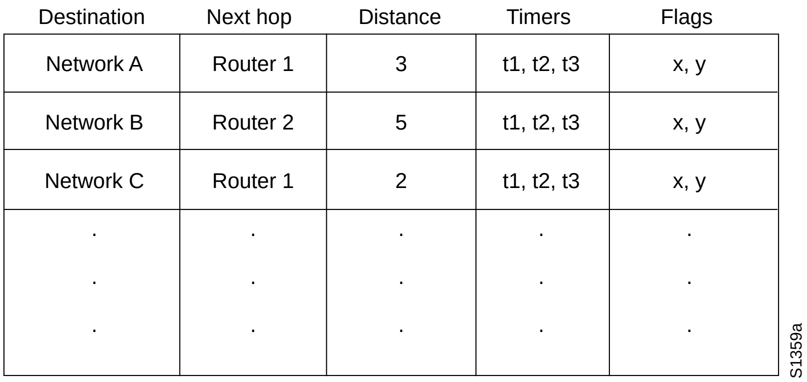
The Routing Information Protocol (RIP) is a routing protocol originally designed for Xerox *PARC Universal Protocol* (where it was called GWINFO) and used in the *Xerox Network Systems* (XNS) protocol suite. RIP became associated with both UNIX and *Transmission Control Protocol/Internet Protocol* (TCP/IP) in 1982 when the Berkeley Software Distribution (BSD) version of UNIX began shipping with a RIP implementation referred to as *routed* (pronounced “route dee”). RIP, which is still a very popular routing protocol in the Internet community, is formally defined in the XNS *Internet Transport Protocols* publication (1981) and in *Request for Comments* (RFC) 1058 (1988).

RIP has been widely adopted by personal computer (PC) manufacturers for use in their networking products. For example, AppleTalk’s routing protocol (*Routing Table Maintenance Protocol*, also known as *RTMP*) is a modified version of RIP. RIP was also the basis for the routing protocols of Novell, 3Com, Ungermann-Bass, and Banyan. The Novell and 3Com RIPs are basically standard Xerox RIP. Ungermann-Bass and Banyan made minor modifications to RIP to serve their own needs.

Routing Table Format

Each entry in a RIP routing table provides a variety of information, including the ultimate destination, the next hop on the way to that destination, and a *metric*. The metric indicates the distance in number of hops to the destination. Other information can also be present in the routing table, including various timers associated with the route. A typical RIP routing table is shown in Figure 23-1.

Figure 23-1 Typical RIP Routing Table



Destination	Next hop	Distance	Timers	Flags
Network A	Router 1	3	t1, t2, t3	x, y
Network B	Router 2	5	t1, t2, t3	x, y
Network C	Router 1	2	t1, t2, t3	x, y
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.

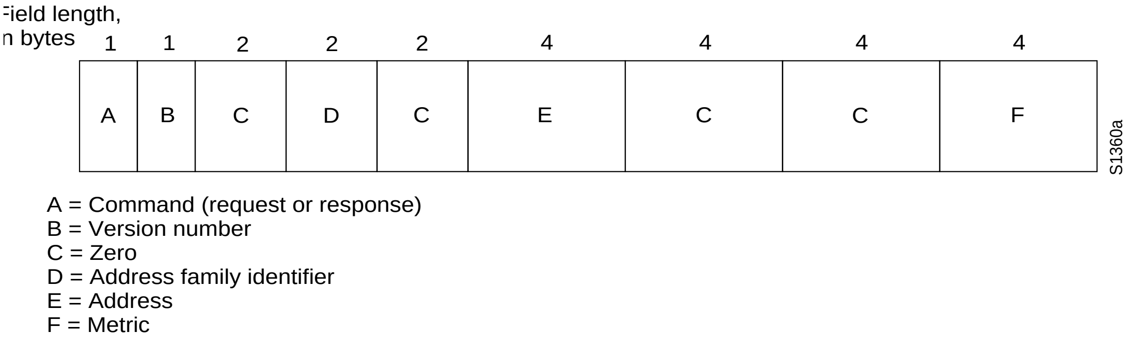
RIP maintains only the best route to a destination. When new information provides a better route, this information replaces old route information. Network topology changes can provoke changes to routes, causing, for example, a new route to become the best route to a particular destination. When network topology changes occur, they are reflected in routing update messages. For example, when a router detects a link failure or a router failure, it recalculates its routes and sends routing update messages. Each router receiving a routing update message that includes a change updates its tables and propagates the change.

Packet Format: IP Implementations

Figure 23-2 shows the RIP packet format for IP implementations, as specified by RFC 1058.

Note Figure 23-2 shows the RIP format used for IP networks in the Internet. Some other RIP variations make slight modifications to the format and/or to the field names listed here, but the basic routing algorithm is functionally the same.

Figure 23-2 RIP Packet Format



The fields of the RIP packet are as follows:

- *Command*—Indicates that the packet is a request or a response. The request command requests the responding system to send all or part of its routing table. Destinations for which a response is requested are listed later in the packet. The response command represents a reply to a request or, more frequently, an unsolicited regular routing update. In the response packet, a responding system includes all or part of its routing table. Regular routing update messages include the entire routing table.
- *Version number*—Specifies the RIP version being implemented. With the potential for many RIP implementations in an internetwork, this field can be used to signal different, potentially incompatible, implementations.
- *Address family identifier*—Follows a 16-bit field of all zeros and specifies the particular address family being used. On the Internet (a large, international network connecting research institutions, government institutions, universities, and private businesses), this address family is typically IP (value = 2), but other network types may also be represented.

- *Address*—Follows another 16-bit field of zeros. In Internet RIP implementations, this field typically contains an IP address.
- *Metric*—Follows two more 32-bit fields of zeros and specifies the *hop count*. The hop count indicates how many internetwork hops (routers) must be traversed before the destination can be reached.

Up to 25 occurrences of the address family identifier field through metric field are permitted in any single IP RIP packet. In other words, up to 25 destinations may be listed in any single RIP packet. Multiple RIP packets are used to convey information from larger routing tables.

Like other routing protocols, RIP uses certain timers to regulate its performance. The RIP *routing update timer* is generally set to 30 seconds, ensuring that each router will send a complete copy of its routing table to all neighbors every 30 seconds. The *route invalid timer* determines how much time must expire without a router having heard about a particular route before that route is considered invalid. When a route is marked invalid, neighbors are notified of this fact. This notification must occur prior to expiration of the *route flush timer.* When the route flush timer expires, the route is removed from the routing table. Typical initial values for these timers are 90 seconds for the route invalid timer and 270 seconds for the route flush timer.

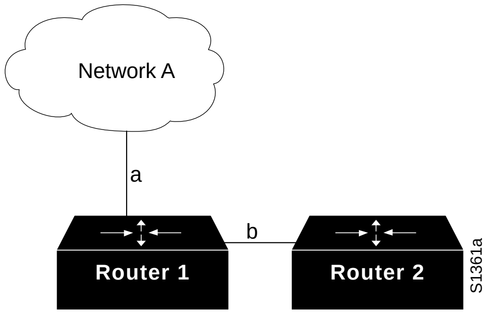
Stability Features

RIP specifies a number of features designed to make its operation more stable in the face of rapid network topology changes. These include a hop-count limit, *hold-downs*, *split horizons*, and *poison reverse updates*.

Hop-Count Limit

RIP permits a maximum hop count of 15. Any destination greater than 15 hops away is tagged as unreachable. RIP's maximum hop count greatly restricts its use in large internetworks, but prevents a problem called *count to infinity* from causing endless network routing loops. The count-to-infinity problem is shown in Figure 23-3.

Figure 23-3 Count-to-Infinity Problem



In Figure 23-3, consider what will happen if Router 1's (R1's) link (link a) to Network A fails. R1 examines its information and sees that Router 2 (R2) has a one-hop link to Network A. Since R1 knows it is directly connected to R2, it advertises a two-hop path to Network A and begins routing all traffic to Network A through R2. This creates a routing loop. When R2 sees that R1 can now get to Network A in two hops, it changes its own routing table entry to show that it has a three-hop path to Network A. This problem, and the routing loop, will continue indefinitely, or until some external boundary condition is imposed. That boundary condition is RIP’s hop-count maximum. When the hop count exceeds 15, the route is marked unreachable. Over time, the route is removed from the table.

Hold-Downs

Hold-downs are used to prevent regular update messages from inappropriately reinstating a route that has gone bad. When a route goes down, neighboring routers will detect this. These routers then calculate new routes and send out routing update messages to inform their neighbors of the route change. This activity begins a wave of routing updates that filter through the network.

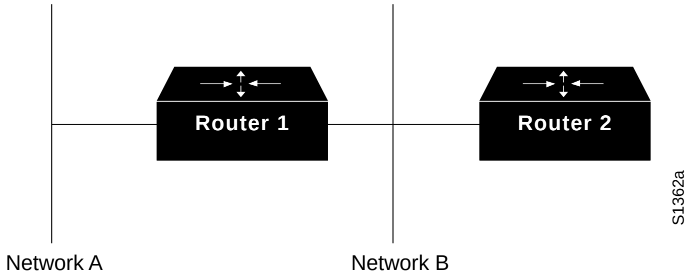
Triggered updates do not instantly arrive at every network device. It is therefore possible that a device that has yet to be informed of a network failure may send a regular update message (indicating that a route that has just gone down is still good) to a device that has just been notified of the network failure. In this case, the latter device now contains (and potentially advertises) incorrect routing information.

Hold-downs tell routers to hold down any changes that might affect recently removed routes for some period of time. The hold-down period is usually calculated to be just greater than the period of time necessary to update the entire network with a routing change. Hold-down prevents the count-to-infinity problem.

Split Horizons

Split horizons derive from the fact that it is never useful to send information about a route back in the direction from which it came. For example, consider Figure 23-4.

Figure 23-4 Split Horizons



Router 1 (R1) initially advertises that it has a route to Network A. There is no reason for Router 2 (R2) to include this route in its update back to R1 because R1 is closer to Network A. The split-horizon rule says that R2 should strike this route from any updates it sends to R1.

The split-horizon rule helps prevent two-node routing loops. For example, consider the case where R1’s interface to Network A goes down. Without split horizons, R2 continues to inform R1 that it can get to Network A through R1. If R1 does not have sufficient intelligence, it might actually pick up R2’s route as an alternative to its failed direct connection, causing a routing loop. Although hold-downs should prevent this, split horizon provides extra algorithm stability.

Poison Reverse Updates

Whereas split horizons should prevent routing loops between adjacent routers, poison reverse updates are intended to defeat larger routing loops. The idea is that increases in routing metrics generally indicate routing loops. Poison reverse updates are then sent to remove the route and place it in hold-down.