Routing Information Protocol

Mattias Jansson, KTHNOC

fimblo@sunet.se

February 4, 2004

Contents

1 General

- Routing Information Protocol
- RIP is an implementation of the Distance-Vector protocol.
- RIP uses the Bellman-Ford Algorithm to calculate its routes.
- Bellman 1957, Ford and Fulkerson 1962
- RIPv1 is specified in RFC 1058
- RIPv2 is specified in RFC 2453

2 The Distance-Vector Protocol

- RIP uses the Bellman Ford Algorithm to calculate the network topology.
- Each router sends a list of distance-vectors each of its neighbours periodically.
- The metric must be a positive integer. This metric measures the cost to get to the destination. In RIP, this cost describes number of hops.

2.1 The Distance-Vector

A distance-vector is a vector containing the following elements (*destination, cost, source*) where:

- `destination` is the destination network
- `cost` is the sending router's metric to the destination + 1
- `source` is the sending router's id.

One distance-vector is created for each entry in the routing table, and the entire set of distance-vectors are sent periodically.

2.2 Bellman-Ford's algorithm

```
BELLMAN-FORD(G, w, s)                    ; Given a graph G, a weight function w,
                                           ; and a source node s
    for each vertex v in V[G]            ; For every node v in G
       d[v] := inf                       ; Set distance to v to a large number
       p[v] := nil                       ; Set parent of v to nil
    d[s] := 0                            ; Set the source's distance to itself to 0
    for each vertex in V[G]              ; For every node v in G
      for each edge (u,v) in the set E[G] ; for all edges
        if d[v] > d[u] + w(u, v)           ; if the distance to a node can be updated
           d[v] := d[u] + w(u, v)          ; update the v's distance
           p[v] := u                       ; update the v's parent
```

$p[v]$ is used to build the tree, $d[v]$ contains the cost to get to v. This version of Bellman-Ford is adapted for a weighted, directed, *non-negative* graph. This algorithm runs at $\mathrm{O}(|V||E|)$.

2.3 Route update example

- At cold start each router knows only its own directly connected nets
- The cost to directly connected nets are 0

RTA

Dest	Cost	Via
a	0	-

RTB

Dest	Cost	Via
b	0	-

RTC

Dest	Cost	Via

Route update example (con't)

- After each router has made one update

RTA			RTB			RTC		
Dest	Cost	Via	Dest	Cost	Via	Dest	Cost	Via
a	0	-	b	0	-	b	1	rtb
b	1	rtb	a	1	rta	a	1	rta

Route update example (con't)

- After each router has done two updates

RTA			RTB			RTC		
Dest	Cost	Via	Dest	Cost	Via	Dest	Cost	Via
a	0	-	b	0	-	b	1	rtb
b	1	rtb	a	1	rta	a	1	rta
b	1	rtb	a	1	rta	b	1	rtb
a	2	rtb	b	2	rta	a	2	rtb
b	2	rtc	b	2	rtc	a	1	rta
a	2	rtc	a	2	rtc	b	2	rta

Route update example (con't)

- The newly arrived distance-vectors *(destination/cost/source)* are checked against the current routing table:
  - A *dest* that has no match in the table is unique and is added.
  - If the *dest* matches an entry it must replace it iff:
    * . . . the source is the same.

      or
    * . . . the source is different and the cost is better.

Route update example (con't)

- The final table is achieved. Yay!

RTA			RTB			RTC		
Dest	Cost	Via	Dest	Cost	Via	Dest	Cost	Via
a	0	-	b	0	-	b	1	rtb
b	1	rtb	a	1	rta	a	1	rta

- But is it really this simple?

2.4 Count-to-Infinity problem

- Network *a* becomes unreachable, and RTA sends out *(a inf rta)* to its neighbours.
- Since there is a timing issue here, the following can occur:
  - RTA sends *(a inf rta)* to RTB and RTC
  - RTC updates its table to *(a inf rta)*
  - Before RTB has received *(a inf rta)*, RTC sends *(a 2 rtc)* to RTB and RTA causing them both to update their tables
  - RTC receives the original *(a inf rta)* from RTA
  - This can cause all routers to spread the poisonous entry until it reaches *infinity* in all routers.

3 RIP's implementation of DV

- RIP needs to deal with some of the shortcomings of Distance-Vector protocols.
- These are:
  - Infinity
  - Split Horizon and Poison Reverse
  - Triggered Updates
  - Timers

3.1 The Concept of Infinity

Counting to infinity can take a long time.

Thus infinity is set to something not quite as eternal as infinity, namely 16.

This limits the diameter of the routing domain to 15, and also makes counting to infinity a little faster.

3.2 Split Horizon

With Split Horizon activated, a router omits sending routes back to the router it learned them from.

This helps in avoiding a process of mutual deception, where two routers tell each other that they can reach destination X via each other.

Split Horizon MUST be supported by all running versions of RIP.

3.3 Split Horizon with Poisoned Reverse

With this scheme activated, a router behaves in the same way as in plain Split Horizon, but instead of not sending information back, it sends a route update with a metric of 16 (unreachable) to the router it got the route from.

Split Horizon with Poisoned Reverse SHOULD be supported in all implementations of RIP, though it is legal to have a knob to turn it off.

Split Horizon with Poisoned Reverse (con't)

In a network with two routers, we have now eliminated routing loops.

However, in a situation with three, fully meshed routers, we still can get loops.

Can we avoid this?

3.4 Triggered Updates

No. But we can minimize the probability of a Count-to-Infinity using triggered updates.

- Triggered Update means that an incoming update message triggers the router to make its own update. RIP does its updates by sending out its distance-vectors to all its neighbours.

3.5 Flash Updates

- On a Cisco box, a router that cold-starts broadcasts a request packet to all its neighbours. Every neighbour responds by immediately unicasting a reply containing its distance-vector.
- In Cisco-ese, this functionality together with the normal triggered update functionality is called a *Flash Update*.

3.6 RIP's timers

There are two sets of timers in RIP:

The RFC way

The Cisco way

3.6.1 RIP's counters- according to the RFC

Update The time between each update. Default is 30 seconds modified by a small random time to avoid synchonization problems.

Time-out If no updates are received, wait this long after the last successful update before marking the route for deletion. Default: 180s

Garbage-collection When a route has been marked for deletion, its metric is set to 16, and after this amount of time it is removed from memory. Default: 120s.

3.6.2 RIP's counters- according to Cisco

Update The time between each update. Default is 30s modified by a small random time to avoid synchonization problems.

Invalid If no updates are received, wait this long after the last successful update before marking the route invalid. Default: 180s

Holddown When a route has become invalid, don't accept updates about this route for this period of time, Default: 180s

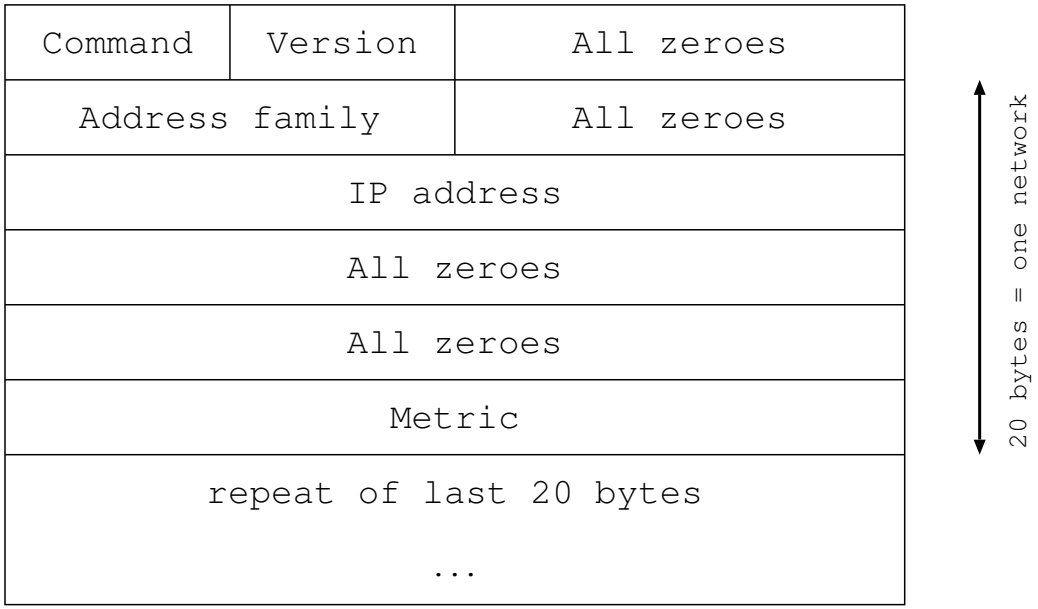
Flush If a route has been invalid for this time, flush (remove) it from memory. Default: 240s.

4 RIP protocol details

- RIP sends its updates over UDP
- RIP listens to updates on port 520.
- Currently there are two versions of RIP, RIPv1 and RIPv2
- RIPv1 uses broadcast, RIPv2 uses multicast
- In contrast to RIPv1, RIPv2 supports subnet masks and a simple authentication mechanism.

4.1 RIPv1

RIP version 1



4.1.1 RIPv1 header explanation

Command Type of command. Only 1 (Request) and 2 (Response) are used

Version Version of RIP. In this case RIPv1.

Below is the 20 bytes that describes what is advertised. These 20 bytes are repeated for each network.

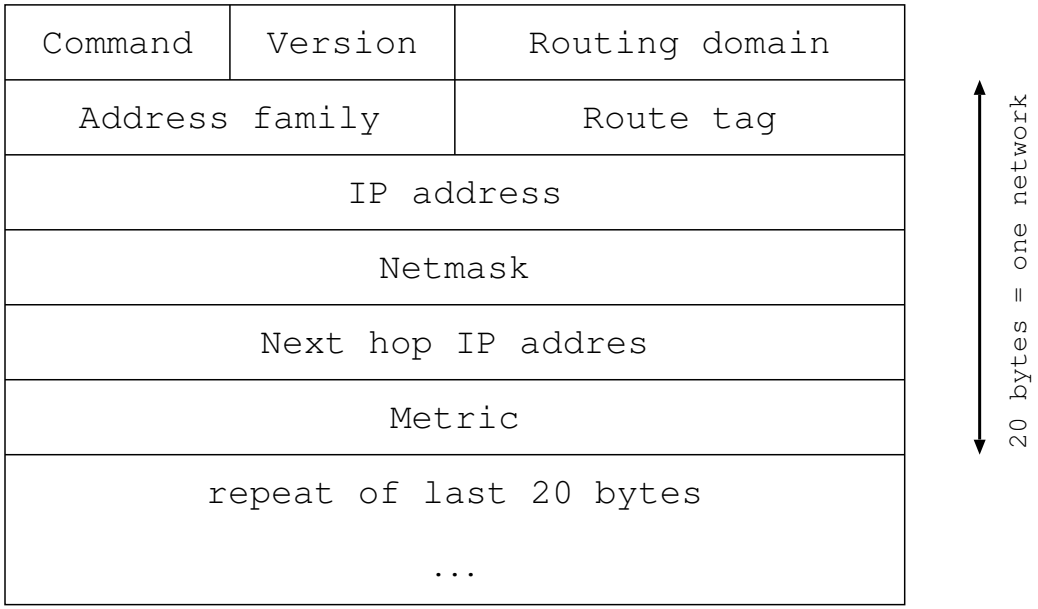
Address family What kind of address is advertised. 2 = IP

IP address The prefix that is advertised.

Metric The metric cost to the prefix.

4.2 RIPv2

RIP version 2



4.2.1 RIPv2 header explanation

Command Type of command. Only 1 (Request) and 2 (Response) are used

Version Version of RIP. In this case RIPv2.

Routing domain Points out which process of RIP this message belongs to if there are more then one instance running on a machine.

Next slide explains the 20 bytes that describes what is advertised. These 20 bytes are repeated for each network.

RIPv2 header explanation (con't)

Address family What kind of address is advertised. 2 = IP

Route tag If RIP is used to support an EGP this field contains the AS number

IP address The prefix that is advertised.

Netmask The netmask of the prefix in the previous field.

Next hop IP address The IP address of the next hop of the advertised prefix

Metric The metric cost to the prefix.

5 Some limitations in RIP

- The UDP packets has a maximum sixe of 512 bytes. This leaves room for 25 prefixes. If there are more prefixes to advertise we must use multiple UDP datagrams
- The diameter of a network that runs RIP cannot be larger than 15.
- In large networks, RIP traffic can consume a substantial amount of the bandwith. The more changes in the topology, the more flash updates, the more bandwidth is consumed.

6 So why use RIP?

- RIP is easy to implement →
  - Many different implementations.
  - The protocol has been rigorously tested by many teams of developers.
- In a small network, RIP has very little overhead in terms of bandwidth, memory consumption, processor load, etc.