Network Working Group
Internet Draft
Expires 2 Aug 1996

G.M. Meyer
Shiva
S. Sherry
Xyplex

Jan 1996

Triggered Extensions to RIP to Support Demand Circuits
draft-ietf-rip-trigger-rip-01.txt

Status of this Memo

This document is a submission to the RIP Working Group of the Internet Engineering Task Force (IETF). Comments should be submitted to the ietf-rip@xylogics.com mailing list.

Distribution of this memo is unlimited.

Abstract

This document defines a modification which can be applied to Bellman-Ford (distance vector) algorithm information broadcasting protocols - for example IP RIP, Netware RIP or Netware SAP - which makes it feasible to run them on connection oriented Public Data Networks.

This proposal has a number of efficiency advantages over the Demand RIP proposal (RFC 1582).

Acknowledgements

The authors wish to thank Richard Edmonstone of Shiva, Joahanna Kruger of Xyplex, Steve Waters of DEC and Guenter Roeck of Conware for many comments and suggestions which improved this effort.

Conventions

The following language conventions are used in the items of specification in this document:

- MUST -- the item is an absolute requirement of the specification. MUST is only used where it is actually required for interoperation, not to try to impose a particular method on implementors where not required for interoperability.

- SHOULD -- the item should be followed for all but exceptional circumstances.

- MAY or optional -- the item is truly optional and may be followed or ignored according to the needs of the implementor.

  The words "should" and "may" are also used, in lower case, in their more ordinary senses.

Table of Contents

1. Introduction

Routers are used on connection oriented networks, such as X.25 packet switched networks and ISDN networks, to allow potential connectivity to a large number of remote destinations. Circuits on the Wide Area Network (WAN) are established on demand and are relinquished when the traffic subsides. Depending on the application, the connection between any two sites for user data might actually be short and relatively infrequent.

Periodic broadcasting by Bellman-Ford (distance vector) algorithm information broadcasting protocols IP RIP [1], IP RIP V2 [2] or Netware RIP and SAP [3] generally prevents WAN circuits from being closed. Even on fixed point-to-point links the overhead of periodic transmission of RIP - and even more so SAP broadcasts - can seriously interrupt normal data transfer simply through the quantity of information which hits the line every 30 or 60 seconds.

To overcome these limitations, this specification modifies the distance vector protocols so as to send information on the WAN only when there has been an update to the routing database OR a change in the reachability of a next hop router is indicated by the task which manages connections on the WAN.

Because datagrams are not guaranteed to get through on all WAN media, an acknowledgement and retransmission system is required to provide reliability.

The protocols run unmodified on Local Area Networks (LANs) and so interoperate transparently with implementations adhering to the original specifications.

This proposal differs from Demand RIP [4] conceptually as follows:

- If a router has exchanged all routing information with its partner and some routing information subsequently changes only the changed information is sent to the partner.

- The receiver of routes is able to apply all changes immediately upon receiving information from a partner.

These differences lead to further reduced routing traffic and also require less memory than Demand RIP [4]. Demand RIP also has an upper limit of 255 fragments in an update which is lifted in Triggered RIP (which does not use fragmentation).

2. Overview

Multiprotocol routers are used on connection oriented Wide Area Networks (WANs), such as X.25 packet switched networks and ISDN networks, to interconnect LANs. By using the multiplexing properties of the underlying WAN technology, several LANs can be interconnected simultaneously through a single physical interface on the router.

A circuit manager provides an interface between the connectionless network layers, IP and IPX, and the connection oriented WAN, X.25, ISDN etc. Figure 1 shows a schematic representative stack showing the relationship between routing protocols, the network layers, the circuit manager and the connection oriented WAN.

```
          --------------           ---------  ---------
          |    RIP     |           |  RIP  |  |  SAP  |
          --------------           ---------  ---------
                |                      |          |
          --------------               |          |
          |    UDP     |               |          |
          --------------               |          |
                |                      |          |
          --------------             ----------------
          |    IP      |             |     IPX      |
          --------------             ----------------
                |                           |
          -------------------------------------------
          |             Circuit Manager             |
          -------------------------------------------
                           ||||||||||
                           ||||||||||
                   ---------------------------
                   |   Connection Oriented   |
                   |        WAN stack        |
                   ---------------------------
```

A WAN circuit manager will support a variety of network layer protocols, on its upper interface. On its lower interface, it may support one or more subnetworks. A subnetwork may support a number of Virtual Circuits.

Figure 1.   Representative Multiprotocol Router stack

The router has a translation table which relates the network layer address of the next hop router to the physical address used to establish a Virtual Circuit (VC) to it.

The circuit manager takes datagrams from the connectionless network layer protocols and (if one is not currently available) opens a VC to the next hop router. A VC can carry all traffic between two end-point routers for a given network layer protocol (or with appropriate encapsulation all network layer protocols). An idle timer (or some other mechanism) is used to close the VC when the datagrams stop arriving at the circuit manager.

If the circuit manager has data to forward (whether user data OR a routing update) and fails to obtain a VC it informs the routing application that the destination is unreachable (circuit down). The circuit manager is then expected to perform whatever is necessary to recover the link. Once successful, it informs the routing application (circuit up).

In Triggered RIP, routing updates are only transmitted on the WAN when required:

1 When a specific request for a routing update has been received.

2 When the routing database is modified by new information from another interface.

3 When the circuit manager indicates that a destination has changed from an unreachable (circuit down) to a reachable (circuit up) state.

4 And also when a unit is first powered on to ensure that at least one update is sent. This can be thought of as a transition from circuit down to circuit up. It MAY contain no routes or services, and is used to flush routes or services from the peer's database.

In cases 1,3 and 4 the full contents of the database is sent. In case 2 only the latest changes are sent.

Because of the inherent unreliability of a datagram based system, both routing requests and routing responses require acknowledgement, and retransmission in the event of NOT receiving an acknowledgement.

3. The Routing Database

Entries in the routing database can either be permanent or temporary. Entries learned from broadcasts on LANs are temporary. They will expire if not periodically refreshed by further broadcasts.

Entries learned from a triggered response on the WAN are 'permanent'. They MUST not time out in the normal course of events. Certain events can cause these routes to time out.

3.1 Presumption of Reachability

If a routing update is received from a next hop router on the WAN, entries in the update are thereafter always considered to be reachable, unless proven otherwise:

- o If in the normal course of routing datagrams, the circuit manager fails to establish a connection to the next hop router, it notifies the routing application that the next hop router is not reachable through an internal circuit down message.

  The database entries are first marked as temporary and aged normally; Some implementations may choose to omit this initial aging step. The routing application then marks the appropriate database entries as unreachable for a hold down period (the normal 120 second RIP hold down timer).

- o If the circuit manager is subsequently able to establish a connection to the next hop router, it will notify the routing application that the next hop router is reachable through an internal circuit up message.

  The routing application will then exchange messages with the next hop router so as to re-prime their respective routing databases with up-to-date information.

The next hop router may also be marked as unreachable if an excessive number of retransmissions of an update go unacknowledged (see section 6.3).

Handling of circuit up and circuit down messages requires that the circuit manager takes responsibility for establishing (or re-establishing) the connection in the event of a next hop router becoming unreachable. A description of the processes the circuit manager adopts to perform this task is outside the scope of this document.

3.2 Alternative Routes

A requirement of using Triggered RIP for propagating routing information is that NO routing information ever gets LOST or DISCARDED. This means that all alternative routes SHOULD be retained.

It MAY be possible to operate with a sub-set of all alternative routes, but this adds complexity to the protocol - which is NOT covered in this document.

3.3 Split Horizon with Poisoned Reverse

The rules for Split Horizon with Poisoned Reverse MUST be used to determine whether and/or how a route is advertised on an interface running this protocol.

Split Horizon consists of omitting routes learned from a peer when sending updates back to that peer. With Poisoned Reverse instead of omitting those routes, they are advertised as unreachable (setting the metric to infinity).

A route is only poisoned if it is the best route (rather than an inferior alternative route) in the database.

Poison Reverse is necessary because a router may be advertising a route to a network to its partner and then later learn a better route for the same network from the partner. Without Poison Reverse the partner will not know to discard the inferior route learned from the first router.

3.4 Managing Routing Updates

The routing database SHOULD be considered to be a sequence of elements ordered by the time it was last updated. If there is a change in the best route (i.e. a new route is added or a route's metric has changed), the route is reordered and given a new highest sequence number.

Sending updates to a peer consists of running through the database from the oldest entry to the newest entry. Once an entry has been sent and acknowledged it is generally never resent. As new routing information arrives, only the new information is sent.

3.5 Retransmissions

Handling retransmission of updates is simplest if updates are restricted to never having more than one un-acknowledged update outstanding - "one packet in flight". A copy of the update packet can be kept and retransmitted until acknowledged - and then subsequent update packets are sent in turn until the full database (to date) has been sent and acknowledged.

Things become more complicated if several packets are sent in quick succession without waiting for an acknowledgements between packets - "several packets in flight":

- o If packets arrive out of order they could corrupt the peer's database. If the underlying datalink layer bundles several VCs, it MUST guarantee to NOT reorder datagrams.

- o If the elements making up a packet requiring retransmission change because of an alteration in the database, stale incorrect information could be sent (again new information could overtake old information).

To guard against this when 'retransmitting' a packet when the database is in flux the packet MUST be re-created from the database to contain only the subset of routes which currently apply. And if none of the routes still apply, nothing will be 'retransmitted'.

For simplicity of implementation we would advise having only one packet in flight. However if the 'round trip' for a response and acknowledgement is quite long this could significantly delay large updates. See Appendix A for an understanding of the additional complexity of managing several packets in flight.

4. New Packet Types

To support triggered updates, three new packet types MUST be supported. For IP RIP Version 1 [1] and IP RIP Version 2 [2] these are identified by the Command Field values shown:

- 9 - Update Request

- 10 - Update Response

- 11 - Update Acknowledge

For Netware RIP and SAP [3] the equivalent Field to distinguish between packet types is called Operation and these take the same values.

These Command and Operation types require the addition of a 4 octet Update header. All three packet types contain a Version, which MUST be 1. Update Response and Update Acknowledge also have a Sequence Number and a Flush Flag.

4.1 Update Request

The Update Request has the Command/Operation value 9.

It is a request to the peer system to send ALL appropriate elements in its routing database. It is retransmitted at periodic intervals (every 5 seconds) until an Update Response message is received with the Flush flag set.

An Update Request is transmitted in the following circumstances:

- Firstly when the router is powered on.

- Secondly when the circuit manager indicates a destination has been in an unreachable (circuit down) state and changes to a reachable (circuit up) state.

An Update Request may also be sent at other times to compensate for discarding non-optimal routing information or if an Update Response continues to be unacknowledged (see section 6.3).

4.2 Update Response

The Update Response has the Command/Operation value 10.

It is a message containing zero or more routes in an update. It is retransmitted at periodic intervals until an Update Acknowledge is received.

An Update Response message MUST be sent:

- o In response to an Update Request. The Update Response MUST have the Flush flag set. Other Update Responses should NOT be sent until an Update Acknowledge has been received acknowledging the Flush flag.

  The remainder of the database MUST then be sent as a series of Update Responses the Flush flag NOT set.

- o An Update Response with the Flush flag set MUST also be sent at power on to flush the peer's routing table learned from a previous incarnation. This Update Response SHOULD NOT contain any routes. This avoids any possibility of an acknowledgement being received to a response sent BEFORE the unit was restarted causing confusion about which routes are being acknowledged.

Update Response messages continue to be sent any time there is fresh routing information to be propagated.

Each new Update Response is given a different Sequence Number. The Sequence Number only has 'meaning' to the sender of the Update Response. The same Update Response sent to different peers MAY have a different Sequence Number.

An Update Response packet with the Flush flag set MUST be sent to a peer:

- o At power on.

- o In response to an Update Request packet.

- o After transitioning from a circuit down to a circuit up state.

After sending an Update Flush, the full database MUST be sent subsequently.

4.3 Update Acknowledge

The Update Acknowledge has the Command/Operation value 11.

It is a message sent in response to every Update Response packet received. If the Update Response packet has the flush flag set then so should the Update Acknowledge packet.

5. Packet Formats

5.1 Update Header

To support the mechanism outlined in this proposal the packet format for RIP Version 1 [1], RIP Version 2 [2] and Netware RIP and SAP [3] are modified to include an additional small header when using Commands Update Request (9), Update Response (10) and Update Acknowledge (11). Commands are called Operations in Netware.

Update Request (9):

```
0                   1                   2                   3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Version (1)  |               must be zero (3)                |
+-------------------------------+-------------------------------+
```

Update Response (10) and Update Acknowledge (11):

```
0                   1                   2                   3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Version (1)  |   Flush (1)   |     Sequence Number (2)       |
+-------------------------------+-------------------------------+
```

Four octet Update headers, with each tick mark representing one bit. All fields are coded in network byte order (big-endian).

Figure 2. Update Headers.

Version MUST be 1 in all headers. Any packets received for a different Version MUST be silently discarded.

The Sequence Number MUST be incremented every time a new Update Response packet is sent on the WAN. The Sequence Number is unchanged for retransmissions. The Sequence Number wraps round at 65535.

Flush is set to 1 in an Update Response if the peer is required to start timing out its entries - otherwise it is set to zero. Any other values MUST be silently discarded.

The peer returns an Update Acknowledge containing the same Sequence Number and Flush.

5.2 IP Routing Information Protocol Version 1

IP RIP [1] is a UDP-based protocol which generally sends and receives datagrams on UDP port number 520.

To support the mechanism outlined in this proposal the packet format for RIP Version 1 [1] is modified when using Commands Update Request (9), Update Response (10) and Update Acknowledge (11). See Figure 3.

5.3 IP Routing Information Protocol Version 2

IP RIP Version 2 [2] is an enhancement to IP RIP Version 1 which allows RIP updates to include subnetting information.

To support the mechanism outlined in this proposal the packet format for RIP Version 2 [2] is modified when using Commands Update Request (9), Update Response (10) and Update Acknowledge (11). See Figure 4.

5.4 Netware Routing Information Protocol

Netware [3] supports a mechanism that allows routers on an internetwork to exchange routing information using the Routing Information Protocol (RIP) which runs over the Internetwork Packet Exchange (IPX) protocol using socket number 453h.

To support the mechanism outlined in this proposal the packet format for Novell RIP [3] is modified when using Operations Update Request (9), Update Response (10) and Update Acknowledge (11). See Figure 5.

5.5 Netware Service Advertising Protocol

Netware [3] also supports a mechanism that allows servers on an internetwork to advertise their services by name and type using the Service Advertising Protocol (SAP) which runs over the Internetwork Packet Exchange (IPX) protocol using socket number 452h. SAP operates on similar principals to running RIP. Routers act as SAP agents, collecting service information from different networks and relay it to interested parties.

To support the mechanism outlined in this proposal the packet format for Novell SAP [3] is modified when using Operations Update Request (9), Update Response (10) and Update Acknowledge (11). See Figure 6.

```
0                   1                   2                   3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Command (1)   | RIP Version (1)|     must be zero (2)         |
+---------------+---------------+-------------------------------+
```

```
0                   1                   2                   3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Update Header (4)                         |
+-------------------------------+-------------------------------+
```

Update Response then has up to 25 routing entries (each 20 octets):

```
0                   1                   2                   3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Address Family Identifier (2) |      must be zero (2)         |
+-------------------------------+-------------------------------+
|                         IP address (4)                        |
+---------------------------------------------------------------+
|                         must be zero (4)                      |
+---------------------------------------------------------------+
|                         must be zero (4)                      |
+---------------------------------------------------------------+
|                         Metric (4)                            |
+---------------------------------------------------------------+
                                .
                                .
```

The format of an IP RIP datagram in octets, with each tick mark representing one bit. All fields are coded in network byte order (big-endian).

The four octets of the Update header are included in Update Request (Command 9), Update Response (10) and Update Acknowledge (11) packets. They are not present in packet types in the original RIP Version 1 specification.

Figure 3.   IP RIP Version 1 packet format

```
0                   1                   2                   3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Command (1)   |RIP Version (1)|      must be zero (2)         |
+---------------+---------------+-------------------------------+
```

```
0                   1                   2                   3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Update Header (4)                         |
+-------------------------------+-------------------------------+
```

Update Response then has up to 25 routing entries (each 20 octets):

```
0                   1                   2                   3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Address Family Identifier (2) |        Route Tag (2)          |
+-------------------------------+-------------------------------+
|                         IP address (4)                        |
+---------------------------------------------------------------+
|                         Subnet Mask (4)                       |
+---------------------------------------------------------------+
|                         Next Hop (4) - must be zero           |
+---------------------------------------------------------------+
|                         Metric (4)                            |
+---------------------------------------------------------------+
                                .
                                .
```

The format of an IP RIP Version 2 datagram in octets, with each tick mark representing one bit. All fields are coded in network byte order (big-endian).

The four octets of the Update header are included in Update Request (Command 9), Update Response (10) and Update Acknowledge (11) Packets. They are not present in packet types in the original RIP Version 2 specification.

Next Hop MUST be zero, since Triggered RIP can NOT advertise routes on behalf of other WAN routers.

If authentication is used it immediately follows the Update header.

Figure 4.   IP RIP Version 2 packet format

```
0                   1         1
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Operation (2)           |
+---------------+---------------+
```

```
0                   1                   2                   3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Update Header (4)                         |
+-------------------------------+-------------------------------+
```

Update Response then has up to 50 routing entries (each 8 octets):

```
0                   1                   2                   3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Network Number (4)                      |
+---------------------------------------------------------------+
|       Number of Hops (2)      |      Number of Ticks (2)      |
+---------------------------------------------------------------+
                                .
                                .
```

The format of a Netware RIP datagram in octets, with each tick mark representing one bit. All fields are coded in network byte order (big-endian).

The four octets of the Update header are included in Update Request (Operation 9), Update Response (10) and Update Acknowledge (11) packets. They are not present in packet types in the original Novell RIP specification.

Figure 5.   Netware RIP packet format

```
0                   1         1
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Operation (2)           |
+---------------+---------------+
```

```
0                   1                   2                   3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Update Header (4)                         |
+-------------------------------+-------------------------------+
```

Update Response then has up to 8 service entries (each 64 octets):

```
0                   1                   2                   3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Service Type (2)       |                               |
+-------------------------------+                               |
|                        Service Name (48)                      |
|                            .                                  |
                             .
                             .  +-------------------------------+
|                            .  | Network Address (4)           |
+-------------------------------+-------------------------------+
|  Network Address (cont)       |                               |
+-------------------------------+                               |
|                         Node Address (6)                      |
+-------------------------------+-------------------------------+
|      Socket Address (2)       |       Hops to Server (2)      |
+-------------------------------+-------------------------------+
                                .
                                .
```

The format of a Netware SAP datagram in octets, with each tick mark representing one bit. All fields are coded in network byte order (big-endian).

The four octets of the Update header are included in Update Request (Operation 9), Update Response (10) and Update Acknowledge (11) packets. They are not present in packet types in the original Novell SAP specification.

Figure 6.   Netware SAP packet format

6. Timers

Three timers are supported to handle the triggered update mechanism:

- Database timer.

- Hold down timer.

- Retransmission timer.

An optional over-subscription timer MAY also be supported.

6.1 Database Timer

Routes learned by an Update Response are normally considered to be permanent.

When an Update Response with the Flush flag set is received, all routes learned from that next hop router should start timing out as if they had (just) been learned from a conventional Response (Command 2).

Namely each route exists while the database entry timer (usually 180 seconds) is running and is advertised on other interfaces as if still present. The route is then advertised as unreachable while a further hold down timer is allowed to expire.

6.2 Hold down Timer

A hold down timer of 120 seconds is started on a route:

- When the database timer for the route expires.

- When a formerly reachable route changes to unreachable in an incoming response.

- When a circuit down is received from the circuit manager.

While the hold down timer is running routes are advertised as unreachable on other interfaces.

When the hold down timer expires the route MAY be deleted from the database PROVIDING its unreachability has been successfully propagated to all WAN destinations, or the remaining WAN destinations are in a circuit down state. If a route can not be deleted when the hold-down timer expires, it MAY subsequently be deleted when each and every peer is either up-to-date or is in a circuit down state.

If the hold down timer is already running it is NOT reset by any events which would start the hold down timer.

6.3 Retransmission Timer

The routing task runs a retransmission timer:

- An Update Request packet is retransmitted periodically until an Update Flush packet is received. An Update Flush packet is an Update Response packet with the Flush field set. It need not contain routes.

- An Update Response packet is retransmitted periodically until an Update Acknowledge packet is received containing the same Sequence Number.

With call set up time on the WAN being of the order of a second, a value of 5 seconds for the retransmission timer is appropriate.

To prevent against failures in the circuit manager a limit SHOULD be placed on the number of retransmissions. If no response has been received after a configurable length of time (say 180 seconds) routes via the next hop router are marked as unreachable, the hold down timer is started and the entry is advertised as unreachable on other interfaces.

The next hop router may then be polled with Update Requests at a reduced frequency. A suitable poll interval would be of the order of minutes rather than seconds. Alternatively an Update Request could be initiated by administrative action. When a response is received the routers should perform a complete exchange of routing information.

6.4 Over-subscription Timer

Over-subscription is where there are more next hop routers to send updates to on the WAN than there are channels. For example 3 next hop routers accessed by an ISDN Basic Rate Interface (BRI) which can only support 2 calls simultaneously.

To avoid route oscillation routes may NOT be marked unreachable immediately on receiving a circuit down message from the circuit manager. A timeout MAY be used to delay marking the routes unreachable for sufficiently long to allow the calls to 'time division multiplex' over the available channels. A timeout as long as the regular 180 second RIP route timeout MAY be suitable. In general the greater the over-subscription, the longer the time out should be.

Implementations wishing to support over-subscription may implement the delay within the circuit manager or within the routing application.

If the delay is implemented within the routing application the routing entries MUST NOT start timing out during the delay. This allows the circuit up message to be ignored if the timeout after receiving the circuit down has still to expire. This avoids any confusion if the peer had previously issued a Route Flush command and was part way through an update.

7. Security Considerations

The circuit manager is required to be provided with a list of physical addresses to enable it to establish a call to the next hop router. The circuit manager SHOULD only allow incoming calls to be accepted from the same well defined list of routers.

Elsewhere in the system there will be a set of logical address and physical address tuples to enable the network protocols to run over the correct circuit. This may be a lookup table, or in some instances there may be an algorithmic conversion between the two addresses.

The routing (or service advertising) task MUST be provided with a list of logical addresses to which triggered updates are to be sent on the WAN. The list MAY be a subset of the list of next hop routers maintained by the circuit manager.

RIP Version 2 also allows further authentication of Triggered RIP packets.

Appendix A - Implementation Suggestion

This section suggests how the database might be structured to handle Triggered RIP.

Each entry in the database is given a unique route number. Every time a best route to a network changes, a global route number is incremented and the changed route is given the new route number. Note that this route number is completely internal to the router and has no bearing on the Sequence Number sent in Update Responses sent to the peer.

The route number size should be large enough so as not to wrap round - or the routes can be renumbered before it becomes a problem. Re-numbering requires that the database environment is stable (No Update Responses are queued awaiting Acknowledgement)

Is is probably easier to manage the routes if they are also chained together using a pointer to a later (and possibly also a pointer to an earlier) entry which reflect the route number/age.

Performing a complete update then consists of running though the routes from the oldest to the latest and sending them out in Update Responses. Subsequent changes to the database are treated as sending out only the changed entries (from the previous latest to the new latest).

When allowing for several packets in flight care must be taken with retransmissions. An Update Response 'retransmission' MAY be different from the original. When transmitting a sequence of Update Responses each Response packet contains a number of routes which is a 'time-slice' of the database. Initially the time-slice might be represented by a series of routes with consecutive route numbers. Consider sending three Update Responses with Sequence numbers 10,11 and 12 each containing 10 routes:

Sequence Number	Routes represented by Route Numbers
10	101, 102, 103, 104, 105, 106, 107, 108, 109, 110
11	111, 112, 113, 114, 115, 116, 117, 118, 119, 120
12	121, 122, 123, 124, 125, 126, 127, 128, 129, 130

If these Update Responses are NOT acknowledged, but in the meantime the routing database has changed and the routes represented by route numbers 104, 112 - 116 and 127 have changed and been assigned new route numbers 131 - 137, the retransmission will look like:

Sequence Number	Routes represented by Route Numbers
10	101, 102, 103, 105, 106, 107, 108, 109, 110
11	111, 117, 118, 119, 120
12	121, 122, 123, 124, 125, 126, 128, 129, 130
13	131, 132, 133, 134, 135, 136, 137

To perform a retransmission it is VERY IMPORTANT that the retransmission contains only the SUB-SET of route numbers which currently apply. If there are NO suitable routes to send, it is not necessary to send an empty retransmission.

An alternative 'retransmission' strategy is to always use different sequence numbers when resending updates. Consider transmitting packets with sequence numbers 10 through 20 - and responses are received from all packets except those with sequence numbers 14 and 17. In this case only the data in packets 10 through 13 can be considered to be acknowledged. The data from packet 14 onwards MUST be re-sent and given new sequence numbers starting at 21.

References

[1] Hedrick. C., "Routing Information Protocol", RFC 1058, Rutgers University, June 1988.

[2] Malkin. G., "RIP Version 2 - Carrying Additional Information", RFC 1723, Xylogics, November 1994.

[3] Novell Incorporated., "IPX Router Specification", Version 1.20, October 1993.

[4] Meyer. G., "Extensions to RIP to Support Demand Circuits", Spider Systems, February 1994.

Authors' Address:

Gerry Meyer
Shiva
Stanwell Street
Edinburgh EH6 5NG
Scotland, UK

Phone: (UK) 131 554 9424
Fax: (UK) 131 467 7749
Email: gerry@europe.shiva.com

Steve Sherry
Xyplex
295 Foster St.
Littleton, MA 01460

Phone: (US) 508 952 4745
Fax: (US) 508 952 4887
Email: shs@xyplex.com