T. Pusateri
Juniper Networks

INTERNET DRAFT
Obsoletes: RFC 1075
draft-ietf-idmr-dvmrp-v3-03
September 1996
Expires: February 1997

Distance Vector Multicast Routing Protocol

Status of this Memo

This document is an Internet-Draft. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as ``work in progress.''

To learn the current status of any Internet-Draft, please check the ``1id-abstracts.txt'' listing contained in the Internet-Drafts Shadow Directories on ftp.is.co.za (Africa), nic.nordu.net (Europe), munnari.oz.au (Pacific Rim), ds.internic.net (US East Coast), or ftp.isi.edu (US West Coast).

Abstract

DVMRP is an Internet routing protocol that provides an efficient mechanism for connectionless datagram delivery to a group of hosts across an internetwork. It is a distributed protocol that dynamically generates IP multicast delivery trees using a technique called Reverse Path Multicasting (RPM) [Deer90]. This document is an update to Version 1 of the protocol specified in RFC 1075 [Wait88].

1. Introduction

1.1. Reverse Path Multicasting

Datagrams follow multicast delivery trees from a source to all members of a multicast group [Deer89], replicating the packet only at necessary branches in the delivery tree. The trees are calculated and updated dynamically to track the membership of individual groups. When a datagram arrives on an interface, the reverse path to the source of the datagram is determined by examining a unicast routing table of known source networks. If the datagram arrives on an interface that would be used to transmit unicast datagrams back to the source, then it is forwarded to the appropriate list of downstream interfaces. Otherwise, it is not on the optimal delivery tree and should be discarded. In this way duplicate packets can be filtered when loops exist in the network topology. The source specific delivery trees are automatically pruned back as group membership changes or leaf routers determine that no group members are present. This keeps the delivery trees to the minimum branches necessary to reach all of the group members. New sections of the tree can also be added dynamically as new members join the multicast group by grafting the new sections onto the delivery trees.

1.2. IP-IP Tunnels

Because not all IP routers support native multicast routing, DVMRP includes direct support for tunneling IP Multicast datagrams through routers. The IP Multicast datagrams are encapsulated in unicast IP packets and addressed to the routers that do support native multicast routing. DVMRP treats tunnel interfaces in an identical manner to physical network interfaces. More information on encapsulated tunnels can be found in [Perk96].

1.3. Document Overview

Section 2 provides an overview of the protocol and the different message types exchanged by DVMRP routers. Those who wish to gain a general understanding of the protocol but are not interested in the more precise details may wish to only read this section. Section 3 explains the detailed operation of the protocol to accommodate

developers needing to provide interoperable implementations. Included in Appendix A, is a summary of the DVMRP parameters. A section on DVMRP support for tracing and troubleshooting is the topic of Appendix B. Finally, a short DVMRP version compatibility section is provided in Appendix C to assist with backward compatibility issues.

2. Protocol Overview

DVMRP can be summarized as a "broadcast & prune" multicast routing protocol. It performs Reverse Path Forwarding checks to determine when multicast traffic should be forwarded to downstream interfaces. In this way, minimum spanning trees can be formed to reach all group members from each source network of multicast traffic.

2.1. Neighbor Discovery

Neighbor DVMRP routers can be discovered dynamically by sending Neighbor Probe Messages on local multicast capable network interfaces and tunnel pseudo interfaces. These messages are sent periodically to the All-DVMRP-Routers IP Multicast group address. This address falls into the range of IP Multicast addresses that are to remain on the locally attached IP network and therefore are not forwarded by multicast routers. Protocol messages sent across tunnels should be encapsulated in the same way as data packets to allow for navigation through firewalls.

Beginning with Version 3 of DVMRP outlined in this document, each Neighbor Probe message should contain the list of Neighbor DVMRP routers for which Neighbor Probe messages have been received. In this way, Neighbor DVMRP routers can ensure that they are seen by each other. Care must be taken to interoperate with older implementations of DVMRP that do not include this list of neighbors. It can be assumed that older implementations of DVMRP will safely ignore this list of neighbors in the Probe message. Therefore, it is not necessary to send both old and new types of Neighbor Probes.

2.2. Source Location

When an IP Multicast datagram is received by a router running DVMRP, it first looks up the interface of the DVMRP unicast route back to the source of the datagram. This interface is called the upstream

interface. If the datagram arrived on the correct upstream interface, then it is a candidate for forwarding to one or more downstream interfaces. If the datagram did not arrive on the anticipated upstream interface, it is discarded. This check is known as a reverse path forwarding check and must be performed by all DVMRP routers.

In order to ensure that all DVMRP routers have a consistent view of the unicast path back to a source, a unicast routing table is propagated to all DVMRP routers as an integral part of the protocol. Each router advertises the network number and mask of the interfaces it is directly connected to as well as relaying the routes received from neighbor routers. DVMRP allows for an interface metric to be configured on all physical and tunnel interfaces. When a route is received, the metric of the upstream interface over which the datagram was received must be added to the metric of the route being propagated. This adjusted metric should be computed before the route is compared to the metric of the current next hop gateway.

Although there is certainly additional overhead associated with propagating a separate unicast routing table, it does provide two nice features. First, since all DVMRP routers are using the same unicast routing protocol, there are no inconsistencies between routers when determining the upstream interface (aside from normal convergence issues related to distance vector routing protocols). By placing the burden of synchronization on the protocol as opposed to the network manager, DVMRP reduces the risk of creating routing loops or blackholes due to disagreement between neighbor routers on the upstream interface.

Second, by propagating its own unicast routing table, DVMRP makes it convenient to have separate paths for unicast vs. multicast datagrams. Although, ideally, many network managers would prefer to keep their unicast and multicast traffic aligned, tunneled multicast topologies may prevent this causing the unicast and multicast paths to diverge. Additionally, service providers may prefer to keep the unicast and multicast traffic separate for routing policy reasons as they experiment with IP multicast routing and begin to offer it as a service. For these benefits, DVMRP has chosen to accept the additional overhead of propagating unicast routes.

2.3. Dependent Downstream Routers

In addition to providing a consistent view of source networks, the exchange of unicast routes in DVMRP provides one other important feature. DVMRP uses the unicast route exchange as a mechanism for upstream routers to determine if any downstream routers depend on

them for forwarding from particular source networks. DVMRP accomplishes this by using a well known technique called "Poison Reverse". If a downstream router selects an upstream router as the best next hop to a particular source network, this is indicated by echoing back the route to the upstream router with a metric equal to the original metric plus infinity. When the upstream router receives the report and sees a metric that lies between infinity and twice infinity, it can then add the downstream router from which it received the report to a list of dependent routers for this source.

This list of dependent routers per source network built by the "Poison Reverse" technique will provide the foundation necessary to determine when it is appropriate to prune back the IP source specific multicast trees.

2.4. Building Multicast Trees

As previously mentioned, when an IP multicast datagram arrives, the upstream interface is determined by looking up the interface that would be used if a datagram was being sent back to the source of the datagram. If the upstream interface is correct, then a DVMRP router will forward the datagram to a list of downstream interfaces.

2.4.1. Adding Leaf Networks

Initially, the DVMRP router must consider all of the remaining IP multicast capable interfaces (including tunnels) on the router. If the downstream interface under consideration is a leaf network (has no other IP multicast routers on it), then the IGMP local group database must be consulted. DVMRP routers can easily determine if a directly attached network is a leaf network by keeping a list of all routers from which DVMRP Router Probe messages have been received on the interface. Obviously, it is necessary to refresh this list and age out entries received from routers that are no longer being refreshed. The IGMP local group database is maintained by an elected IP multicast router on each physical, multicast capable network. The details of the election procedure are discussed in [Fenn96]. If the destination group address is listed in the local group database, then the interface should be included in the list of downstream interfaces. If there are no group members on the interface, then the interface can be pruned from the candidate list.

2.4.2. Adding Non-Leaf Networks

Initially, all non-leaf networks should be included in the downstream interface list when a forwarding cache entry is first being created. This allows all downstream routers to be aware of traffic destined for a particular (source, group) pair. The downstream routers will then have the option to prune and graft this (source, group) pair to and from the multicast delivery tree as requirements change from their downstream routers and local group members.

2.5. Pruning Multicast Trees

As mentioned above, routers at the edges with leaf networks will prune their leaf interfaces that have no group members associated with an IP multicast datagram. If a router prunes all of its downstream interfaces, it can notify the upstream router that it no longer wants traffic destined for a particular (source, group) pair. This is accomplished by sending a DVMRP Prune message upstream to the router it expects to forward datagrams from a particular source. Recall that a downstream router will inform an upstream router that it depends on the upstream router to receive datagrams from particular source networks by using the "Poison Reverse" technique during the exchange of unicast routes. This method allows the upstream router to build a list of downstream routers on each interface that are dependent upon it for datagrams from a particular source network. If the upstream router receives prune messages from each one of the dependent downstream routers on an interface, then the upstream router can in turn prune this interface from its downstream interface list. If the upstream router is able to prune all of its downstream interfaces in this way, it can then send a DVMRP Prune message to its upstream router. This continues until the minimum tree necessary to reach all of the receivers remains. Since IP multicast routers may be restarted at any time and lose state information about existing prunes, it is necessary to limit the life of a prune and periodically resume the flooding procedure. This will reinitiate the prune mechanism and the cycle will continue. When a router decides to prune one of its downstream interfaces, it will set a timer to indicate the lifetime of the prune. If all of its downstream interfaces become pruned off the multicast delivery tree and a DVMRP Prune message is sent upstream, the lifetime of the prune will be equal to the minimum of the remaining lifetimes of the pruned interfaces.

Pruning downstream interfaces is also necessary to prevent duplicate packets from arriving on a multi-access network when there are

parallel paths back to a source. Since the routers use the "Poison Reverse" technique during unicast route exchange, they will establish which router will forward multicast traffic to the shared network and prune the appropriate downstream interfaces based on the metrics of the unicast routes exchanged.

2.6. Grafting Multicast Trees

Once a tree branch has been pruned from a multicast delivery tree, packets from the pruned (source, group) pair will no longer be forwarded. However, since IP multicast supports dynamic group membership, new hosts may join the multicast group. In this case, DVMRP routers will need a mechanism to undo the prunes that are in place from the host back to the first branch that was pruned from the multicast tree. This is accomplished with a DVMRP Graft message. A router will send a Graft message to its upstream neighbor if a group join occurs for a group that currently has pruned sources. Separate Graft messages must be sent to the appropriate upstream neighbor for each source that has been pruned. Since there would be no way to tell if a Graft message sent upstream was lost or the source simply quit sending traffic, it is necessary to acknowledge each Graft message with a DVMRP Graft Ack message. If an acknowledgment is not received within a Graft Time-out period, the Graft message should be retransmitted. Duplicate Graft Ack messages should simply be ignored.

3. Detailed Protocol Operation

This section contains a detailed description of DVMRP. It covers sending and receiving of DVMRP messages as well as the generation and maintenance of IP Multicast forwarding cache entries.

3.1. Protocol Header

DVMRP packets are encapsulated in IP datagrams, with an IP protocol number of 2 (IGMP) as specified in the Assigned Numbers RFC [Reyn94]. All fields are transmitted in Network Byte Order. DVMRP packets use a common protocol header that specifies the IGMP [Fenn96] Packet Type as hexadecimal 0x13 (DVMRP). A diagram of the common protocol header follows:

```
 0         8          16               23
+---------+---------+--------------------+
| Type    |  Code   |      Checksum      |
|(0x13)   |         |                    |
+---------+---------+----------+---------+
|     Reserved      |  Minor   | Major   |
|                   | Version  |Version  |
+-------------------+----------+---------+
```

Figure 1 - Common Protocol Header

A Major Version of 3 and a Minor Version of 0xFF should be used to indicate compliance with this specification. The value of the Code field determines the DVMRP packet type. Currently, there are codes allocated for DVMRP protocol message types as well as protocol analysis and troubleshooting packets. The protocol message Codes are:

Code	Packet Type	Description
1	DVMRP Probe	for neighbor discovery
2	DVMRP Report	for unicast route exchange
7	DVMRP Prune	for pruning multicast delivery trees
8	DVMRP Graft	for grafting multicast delivery trees
9	DVMRP Graft Ack	for acknowledging graft messages

Table 1 - Standard Protocol Packet Types

There are additional codes used for protocol analysis and troubleshooting. These codes are discussed in Appendix B. The Checksum is the 16-bit one's complement of the one's complement sum of the DVMRP message. The checksum of the DVMRP message should be calculated with the checksum field set to zero.

3.2. Probe Messages

When a DVMRP router is configured to run on an interface (physical or tunnel), it sends local IP Multicast discovery packets to inform other DVMRP routers that it is operational. These discovery packets are called DVMRP Probes and they serve three purposes.

1. Probes provide a mechanism for DVMRP routers to locate each other. DVMRP sends a list of detected neighbors in the Probe message. This list of DVMRP neighbors provides a foundation for neighbor prune list. If no DVMRP neighbors are found, the network is considered to be a leaf network. A DVMRP router should discard all other protocol packets from a neighbor until it seen its own address in the neighbors Probe list.

2. They provide a way for DVMRP routers to determine the capabilities of each other. This may be deduced from the major and minor version numbers in the Probe packet or directly from the capability flags. These flags were first introduced to allow optional protocol features. This specification now mandates the use of Generation IDs and pruning and, therefore, provides no optional capabilities. Other capability flags were used for tracing and troubleshooting and are no longer a part of the actual protocol. These are now defined in an appendix.

3. Probes provide a keep-alive function in order to quickly detect neighbor loss. DVMRP probes sent on each multicast capable interface configured for DVMRP SHOULD have an interval of 10 seconds. The neighbor time-out interval SHOULD be set at 35 seconds. This allows fairly early detection of a lost neighbor yet provides tolerance for busy multicast routers. These values MUST be coordinated between all DVMRP routers on a physical network segment.

3.2.1. Router Capabilities

In the past, there have been many versions of DVMRP in use with a wide range of capabilities. Practical considerations require a current implementation to interoperate with these older implementations that don't formally specify their capabilities and are not compliant with this specification. For instance, for major versions less than 3, it can be assumed that the neighbor does not support pruning. The formal capability flags were first introduced in an well known implementation (Mrouted version 3.5) in an attempt to take the guess work out which features are supported by a neighbor. These flags are no longer necessary since they are now a required part of the protocol, however, special consideration is necessary to not confuse older implementations that expect these flags to be set. Appendix C was written to assist with these and other backward compatibility issues.

Only two of the flags were used for actual protocol operation. The other three assigned flags were used for troubleshooting purposes which are now documented in a separate specification. All of the bits marked "U" in the Figure below are now unused. They may be defined in the future and MUST be set to 0. Bit positions 1, 2, and 3 MUST be set to 1 for backward compatibility. They were used to specify the PRUNE, GENID, and MTRACE bits. The first two, PRUNE and GENID, are now required features. The MTRACE bit must be set so existing implementations will not assume this neighbor does not support multicast traceroute. However, since this bit is now reserved and set to 1, newer implementations should not use this bit in the Probe message to determine if multicast traceroute is supported by a neighbor. The bits marked S and L stand for SNMP and LEAF bits. Since they are only used by troubleshooting applications, they have been removed from the DVMRP protocol messages.

```
     0                           8    9    10   S    M    G    P    L
    +--------------------------+----+----+----+----+----+----+----+----+
    |        Reserved          | U  | U  | U  | U  | 1  | 1  | 1  | U  |
    +--------------------------+----+----+----+----+----+----+----+----+
```

Figure 2 - Probe Capability Flags

3.2.2. Generation ID

If a DVMRP router is restarted, it must immediately exchange unicast routing tables with all of its neighbors. If a neighbor doesn't automatically detect that the neighbor has restarted, then it will not send its entire routing table immediately. Instead, it will spread the updates over an entire routing update interval. In order for the neighbor to detect a router that is restarted, a non-decreasing number is placed in the periodic probe message called the generation ID. If a router detects an increase in the generation ID of a neighbor, it should exchange its entire unicast routing table with the neighbor. A time of day clock provides a good source for a non-decreasing 32 bit integer.

If a router detects that a neighbor has changed its generation ID, it should assume that the neighbor has restarted. This means that any prune information received from that router is no longer valid and should be flushed. If this prune state has caused prune information to be sent upstream, a graft will need to be sent upstream just as though a new member has joined below. Once data begins to be delivered downstream, if the downstream router again decides to be pruned from the delivery tree, a new prune can be sent upstream at this time.

3.2.3. Neighbor Addresses

As a DVMRP router sees Probe messages from its DVMRP neighbors, it records the neighbor addresses on each interface and places them in the Probe message sent on the particular interface. This allows the neighbor router to know that its probes have been received by the sending router.

It has been shown that in buggy or malfunctioning multicast implementations, one-way neighbor relationships can form. A router receives a route report from a neighbor and sends back poison reverse

routes but the neighbor does not know about the router. Since the neighbor does not have the router listed as a dependent downstream router, black holes can form.

In order to minimize this condition, a router can delay sending route reports directly to a neighbor until the neighbor includes the routers address in its probe messages.

Since a router should not accept route reports from a neighbor until it has seen its own address in the neighbors probe address list, it is important that a router send a probe with the neighbors address in it before sending the neighbor any route reports. Implementations written before this specification will not wait before sending route reports nor will they ignore reports sent. Therefore, reports from these implementations SHOULD be accepted whether or not a probe with the routers address has been received.

3.2.4. Probe Packet Format

The Probe packet is variable in length depending on the number of neighbor IP addresses included. The length of the IP packet can be used to determine the number of neighbors in the Probe message. The current Major Version is 3. To maintain compatibility with previous versions, implementations of Version 3 must include pruning and grafting of multicast trees. Non-pruning implementations SHOULD NOT be implemented at this time.

```
                        7              15         23          31
              +---------+--------------+----------+---------+
              |  Type   |     Code     |       Checksum     |
              | (0x13)  |    (0x1)     |                    |
              +---------+--------------+----------+---------+
              |         |              |          |         |
              |Reserved | Capabilities |  Minor   | Major   |
              +---------+--------------+----------+---------+
              |                                             |
              |                Generation ID                |
              +---------------------------------------------+
              |                                             |
              |            Neighbor IP Address 1            |
              +---------------------------------------------+
              |                                             |
              |            Neighbor IP Address 2            |
              +---------------------------------------------+
              |                                             |
              |                     ...                     |
              +---------------------------------------------+
              |                                             |
              |            Neighbor IP Address N            |
              +---------------------------------------------+
```

Figure 3 - DVMRP Probe Packet Format

3.2.5. Designated Router Election

Since it is wasteful to have more than a single router sending IGMP Host Membership Queries on a given physical network, a single router on each physical network is elected as the Designated Querier. This election used to be a part of DVMRP. However, this is now handled as a part of the IGMP protocol in version 2 and later. Therefore, DVMRP Version 3 requires the use of IGMP Version 2 or later specifying that the Designated Querier election is performed as a part of IGMP.

3.3. Building Forwarding Cache Entries

In order to create optimal multicast delivery trees, IP Multicast was designed to keep separate forwarding cache entries for each (source network, destination group) pair. Because the possible combinations of these is quite large, forwarding cache entries are generated on demand as data arrives at a multicast router. Since the IP forwarding decision is made on a hop by hop basis (as with the unicast case), it is imperative that each multicast router has a consistent view of the reverse path back to the source network. For this reason, DVMRP includes its own unicast routing protocol.

3.3.1. Determining the upstream interface

When a multicast packet arrives, a DVMRP router will use the internal DVMRP unicast routing table to determine which interface leads back to the source. If the packet did not arrive on that interface, it should be discarded without further processing. Each multicast forwarding entry should cache the upstream interface for a particular source host or source network after looking this up in the DVMRP unicast routing table.

3.3.2. Determining the downstream interface list

The downstream interface list is built from the remaining list of multicast capable interfaces. Any interfaces designated as leaf networks and do not have members of the particular multicast group can be automatically pruned from list of downstream interfaces. The remaining interfaces will either have downstream DVMRP routers or directly attached group members. These interfaces may be pruned in the future if it is determined that there are no group members anywhere along the entire tree branch.

3.4. Unicast Route Exchange

It was mentioned earlier that since not all IP routers support IP multicast forwarding, it is necessary to tunnel IP multicast datagrams through these routers. One effect of using these encapsulated tunnels is that IP multicast traffic may not be aligned with IP unicast traffic. This means that a multicast datagram from a particular source can arrive on a different (logical) interface than

the expected upstream interface based on traditional unicast routing.

The unicast routing information propagated by DVMRP is used exclusively for determining the reverse path back to the source of multicast traffic. Tunnels pseudo-interfaces are considered to be distinct for the purpose of determining upstream and downstream interfaces. The routing information that is propagated by DVMRP contains a list of unicast source networks and an appropriate metric. The metric used is a hop count which is incremented by the cost of the incoming interface metric. Traditionally, physical interfaces use a metric of 1 while the metric of a tunnel interface varies with the distance and bandwidth in the path between the two tunnel endpoints. Users are encouraged to configure tunnels with the same metric in each direction to create symmetric routing and provide for easier problem determination although the protocol does not strictly enforce this.

3.4.1. Route Packing and Ordering

Since DVMRP Route Reports may need to refresh several thousand routes each Report interval, routers MUST attempt to spread the routes reported across the whole route update interval. This reduces the chance of synchronized route reports causing routers to become overwhelmed for a few seconds each report interval. Since the route report interval is 60 seconds, it is suggested that the total number routes being updated be split across multiple Route Reports sent at regular intervals. One implementation splits all unicast routes across 6 Report periods sent at 10 second intervals. Route Reports MUST contain source network/mask pairs sorted first by increasing network mask and then by increasing source network within each possible mask value.

In order to pack more source networks into a route report, source networks are often represented by less than 4 octets. The number of non-zero bytes in the mask value is used to determine the number of octets used to represent each source network within that particular mask value. For instance if the mask value of 255.255.0.0 is being reported, the source networks would only contain 2 octets each. DVMRP assumes that source networks will never be aggregated into networks whose prefix length is less than 8. Therefore, it does not carry the first octet of the mask in the Route Report since, given this assumption, the first octet will always be 0xFF. This means that the netmask value will always be represented in 3 octets. This method of specifying source network masks is compatible with techniques described in [Rekh93] and [Full93] to group traditional Class C networks into super-nets and to allow different subnets of the same

Class A network to be discontinuous. In this notation, the default route is represented as the least three significant octets of the netmask [00 00 00], followed by one octet for the network number [00].

3.4.2. Unicast Route Metrics

For each source network reported, a route metric is associated with the unicast route being reported. The metric is the sum of the outgoing interface metrics between the router originating the report and the source network. For the purposes of DVMRP, Infinity is defined to be 32. This limits the breadth across the whole DVMRP network and is necessary to place an upper bound on the convergence time of the protocol.

As seen in the packet format below, Route Reports do not contain a count of the number of routes reported for each netmask. Instead, the high order bit of the metric is used to signify the last route being reported for a particular mask value. If a metric is read with the high order bit of the 8-bit value set and if the end of the message has not been reached, the next value will be a new netmask to be applied to the subsequent list of routes. This technique uses less octets in the Route Report message.

3.4.3. Unicast Route Dependencies

In order for pruning to work correctly, each DVMRP router needs to know which downstream routers depend on it for receiving datagrams from particular source networks. Initially, when a new datagram arrives from a particular source/group pair, it is flooded to all downstream interfaces that have DVMRP neighbors who have indicated a dependency on the receiving DVMRP router for that particular source. A downstream interface can only be pruned when it has received Prune messages from each of the dependent routers on that interface. Each downstream router uses a method called Poison Reverse to indicate to the upstream router which source networks it expects to receive from the upstream router. The downstream router indicates this by echoing back the source networks it expects to receive from the upstream router with infinity added to the advertised metric. This means that the legal values for the metric now become between 1 and (2 x Infinity - 1) or 1 and 63. Values between 1 and 31 indicate reachable unicast source networks. The value Infinity (32)indicates the source network is not reachable. Values between 33 and 63 indicate that the downstream router originating the Report is depending upon the

upstream router to provide multicast datagrams from the corresponding source network.

3.4.4. Sending Route Reports

Full Route Reports MUST be sent out every Route Report Interval. In addition, flash updates MAY be sent between full route reports. Flash updates can reduce the chances of routing loops and black holes occurring when source networks become unreachable through a particular path. Flash updates need only contain the source networks that have changed. It is not necessary to report all of the source networks from a particular mask value when sending an update.

A DVMRP router should not send a Route Report to a neighbor until it has seen its own address in the neighbors Probe neighbor list. See Appendix C for exceptions.

3.4.5. Receiving Route Reports

After receiving a route report, a check should be made to verify it is from a known neighbor. Neighbors are learned via received Probe messages which also indicate the capabilities of the neighbor. Therefore, route reports from unknown neighbors are discarded.

Some older implementations did not sort the routes contained in the update. Therefore, Version 3 implementations MUST be able to handle these reports.

If a route is not refreshed within 140 seconds (2 x Route Report Interval + 20), then it can be replaced with the next best route to the same source. If, after 200 seconds (3 x Route Report Interval + 20), the route has not been refreshed, then it should be expired.

Each route in the report is then parsed and processed according to the following rules:

A. If the route is new and the metric is less than infinity, the route should be added.

B. If the route already exists, several checks must be performed.

1. New metric < infinity

   a. New metric > existing metric

      If the new metric is greater than the existing metric then check to see if the same neighbor is reporting the route. If so, update the metric. Otherwise, discard the route.

   b. New metric < existing metric

      Update the metric for the route and if the neighbor reporting the route is different, update the neighbor. A flash update should be sent to the new neighbor indicating downstream dependence and to the existing neighbor withdrawing downstream dependence of the route.

   c. New metric = existing metric

      If the neighbor reporting the route is the same as the existing route, then simply refresh the route. If the new neighbor has a lower IP address, then update the route. New and existing neighbors should be notified of any changes in downstream dependencies.

2. New metric = infinity

   a. New gateway = existing gateway

      If the existing metric was less than infinity, the route is now unreachable. Update the route and notify any dependent neighbors of the change.

   b. New gateway != existing gateway

      The route can be ignored since the existing gateway has a metric better than or equal to this gateway.

3. infinity < New metric < 2 x infinity

In this case, the neighbors considers the receiving router to be upstream for the route and is indicating it is dependent on the receiving router.

a. Neighbor on down stream interface

If the neighbor is considered to be on a downstream interface for that route, then the neighbor should be registered as a downstream dependent router for that route.

b. Neighbor not on down stream interface

If the receiving router thinks the neighbor is on the upstream interface, then a routing loop has occured and the indication of downstream dependency should be ignored and the detected loop should be logged.

4. 2 x infinity <= New metric

If the metric is greater than or equal to 2 x infinity, the metric is illegal and the route should be ignored.

3.4.6. Route Report Packet Format

The format of a sample Route Report Packet is shown in Figure 4 below. The packet shown is an example of how the source networks are packed into a Report. The number of octets in each Source Network will vary depending on the mask value. The values below are only an example for clarity and are not intended to represent the format of every Route Report.

```
                7            15            23            31
          +-----------+------------+------------------------+
          |   Type    |    Code    |         Checksum       |
          |  (0x13)   |   (0x2)    |                        |
          +-----------+------------+------------+-----------+
          |       Reserved         |   Minor    |   Major   |
          |                        |  Version   |  Version  |
          +-----------+------------+------------+-----------+
          |  Mask1    |   Mask1    |   Mask1    |    Src    |
          |  Octet2   |   Octet3   |   Octet4   |   Net11   |
          +-----------+------------+------------+-----------+
          |  SrcNet11(cont.)...    |  Metric11  |    Src    |
          |                        |            |   Net12   |
          +------------------------+------------+-----------+
          |  SrcNet12(cont.)...    |  Metric12  |   Mask2   |
          |                        |            |   Octet2  |
          +-----------+------------+------------+-----------+
          |  Mask2    |   Mask2    |        SrcNet21        |
          |  Octet3   |   Octet4   |                        |
          +-----------+------------+------------+-----------+
          |  SrcNet21(cont.)...    |  Metric21  |   Mask3   |
          |                        |            |   Octet2  |
          +-----------+------------+------------+-----------+
          |  Mask3    |   Mask3    |           ...          |
          |  Octet3   |   Octet4   |                        |
          +-----------+------------+------------------------+
```

Figure 4 - Example Route Report Packet Format

3.5. Pruning

DVMRP is described as a flood and prune multicast routing protocol since datagrams are initially sent out all dependent downstream interfaces and then pruned back to only the downstream interfaces that are on a reverse shortest path to a receiver. Prunes are data driven and are sent in response to receiving unwanted multicast traffic at the leafs of the multicast tree rooted at a particular source network.

3.5.1. Leaf Networks

Detection of leaf networks is very important to the pruning process. Routers at the end of a source specific multicast delivery tree must detect that there are no further downstream routers. This detection mechanism is covered above in section 3.2 titled DVMRP Probe Messages. If there are no group members present for a particular multicast datagram received, the leaf routers will start the pruning process by pruning their downstream interfaces and sending a prune to the upstream router for that source.

3.5.2. Source Networks

It is important to note that prunes are specific to a group and source network. A prune sent upstream triggered by traffic received from a particular source applies to all sources on that network. It is not currently possible to prune only one or a subset of hosts on a source network for a particular group. All or none of the sources must be pruned.

3.5.3. Receiving a Prune

When a prune is received, the following steps should be taken:

1. Determine if a Probe has been received from this router recently.

2. If not, discard prune since there is no prior state about this neighbor.

3. If so, make sure the neighbor is capable of pruning (based on received Probe message).

4. Since Prune messages are fixed length, ensure the prune message contains the correct amount of data.

5. Extract the source address, group address, and prune time-out values

6. If there is no current state information for the (source, group) pair, then ignore the prune.

7. Verify that the prune was received from a dependent neighbor for the source network. If not, discard the prune.

8. Determine if a prune is currently active from the same dependent neighbor for this (source, group) pair.

9. If so, reset the timer to the new time-out value. Otherwise, create state for the new prune and set a timer for the prune lifetime.

10. Determine if all dependent downstream routers on the interface from which the prune was received have now sent prunes.

11. If so, then determine if there are group members active on the interface.

12. If no group members are found, then prune the interface.

13. If all downstream interfaces have now been pruned, send a prune to the RPF neighbor on the upstream interface.

3.5.4. Sending a Prune

When sending a prune upstream, the following steps should be taken:

1. Decide if upstream neighbor is capable of receiving prunes.

2. If not, then proceed no further.

3. Stop any pending Grafts awaiting acknowledgments.

4. Determine the prune lifetime. This value should be the minimum of the prune lifetimes remaining from the downstream neighbors and the default prune lifetime.

5. Form and transmit the packet to the upstream neighbor for the source.

3.5.5. Retransmitting a Prune

By increasing the prune lifetime to ~2 hours, the effect of a lost prune message becomes more apparent. Therefore, an implementation MAY want to retransmit prunes messages using exponential backoff for the lifetime of the prune if traffic is still arriving on the upstream interface.

One way to implement this would be to send a prune, install a negative cache entry for 3 seconds while waiting for the prune to take effect. Then remove the forward cache entry. If traffic continues to arrive, a new forwarding cache request will be generated. The prune can be resent with the remaining prune lifetime and a negative cache entry can be installed for 6 seconds. After this, the negative cache entry is removed. This procedure is repeated while each time doubling the length of time the negative cache entry is installed.

3.5.6. Prune Packet Format

In addition to the standard IGMP and DVMRP headers, a Prune Packet contains three additional fields: the source host IP address, the destination group IP address, and the Prune Lifetime in seconds.

The Prune Lifetime is a derived value calculated as the minimum of the default prune lifetime (2 hours) and the remaining lifetimes of of any downstream prunes received for the same cache entry. A router with no downstream dependent neighbors would use the the default prune lifetime.

```
                7            15            23            31
      +-----------+------------+------------------------+
      |   Type    |    Code    |        Checksum        |
      |  (0x13)   |   (0x7)    |                        |
      +-----------+------------+------------+-----------+
      |       Reserved         |   Minor    |   Major   |
      +------------------------+------------+-----------+
      |                 Source Address                  |
      +-------------------------------------------------+
      |                  Group Address                  |
      +-------------------------------------------------+
      |                 Prune Lifetime                  |
      +-------------------------------------------------+
```

Figure 5 - Prune Packet Format

3.6. Grafting

Once a multicast delivery tree has been pruned back, DVMRP Graft messages are necessary to join new receivers onto the multicast tree. Graft messages are sent upstream from the new receiver's first-hop router until a point on the multicast tree is reached. Graft messages are re-originated between adjacent DVMRP routers and are not forwarded by DVMRP routers. Therefore, the first-hop router does not know if the Graft message ever reaches the multicast tree. To remedy this, each Graft message is acknowledged hop by hop. This ensures that the Graft message is not lost somewhere along the path between the receiver's first-hop router and the closest point on the multicast delivery tree.

3.6.1. Grafting Each Source Network

It is important to realize that prunes are source specific and are sent up different trees for each source. Grafts are sent in response to a new Group Member which is not source specific. Therefore, separate Graft messages must be sent to the appropriate upstream routers to counteract each previous source specific prune that was sent.

3.6.2. Sending a Graft

As mentioned above, a Graft message sent to the upstream DVMRP router should be acknowledged hop by hop guaranteeing end-to-end delivery. If a Graft Acknowledgment is not received within the Graft Retransmission Time-out period, the Graft should be resent to the upstream router. The initial retransmission period is 5 seconds. A binary exponential backoff policy is used on subsequent retransmissions. In order to send a Graft message, the following steps should be taken:

1. Verify a forwarding cache entry exists for the (source, group) pair and that a prune exists for the cache entry.

2. Verify that the upstream router is capable of receiving prunes (and therefore grafts).

3. Add the graft to the retransmission timer list awaiting an acknowledgment.

4. Formulate and transmit the Graft packet.

3.6.3. Receiving a Graft

The actions taken when a Graft is received depends on the state in the receiving router for the (source, group) pair in the received Graft message. If the receiving router has prune state for the (source, group) pair, then it must acknowledge the received graft and send a subsequent graft to its upstream router. If the receiving router has some pruned downstream interfaces but has not sent a prune upstream, then the receiving interface can simply be added to the list of downstream interfaces in the forwarding cache. A Graft Acknowledgment must also be sent back to the source of the Graft message. If the receiving router has no state at all for the (source, group) pair, then datagrams arriving for the (source, group) pair should automatically be flooded when they arrive. A Graft Acknowledgment must be sent to the source of the Graft message. If a Graft message is received from an unknown neighbor, it should be discarded.

3.6.4. Graft Packet Format

The format of a Graft packet is show below:

```
            7            15            23            31
+-----------+------------+-------------------------+
|   Type    |    Code    |        Checksum         |
|  (0x13)   |   (0x8)    |                         |
+-----------+------------+------------+------------+
|       Reserved         |   Minor    |   Major    |
+------------------------+------------+------------+
|                 Source Address                   |
+--------------------------------------------------+
|                  Group Address                   |
+--------------------------------------------------+
```

Figure 6 - Graft Packet Format

3.6.5. Sending a Graft Acknowledgment

A Graft Acknowledgment packet is sent to a downstream neighbor in response to receiving a Graft message. Grafts received from unknown neighbors should be discarded but all other correctly formatted Graft messages should be acknowledged. This is true even if no other action is taken in response to receiving the Graft to prevent the source from continually re-transmitting the Graft message. The Graft Acknowledgment packet is identical to the Graft packet except that the DVMRP code in the common header is set to Graft Ack. This allows the receiver of the Graft Ack message to correctly identify which Graft was acknowledged and stop the appropriate retransmission timer.

3.6.6. Receiving a Graft Acknowledgment

When a Graft Acknowledgment is received, the (source, group) pair in the packet can be used to determine if a Graft was sent to this particular upstream router. If no Graft was sent, the Graft Ack can simply be ignored. If a Graft was sent, and the acknowledgment has come from the correct upstream router, then it has been successfully received and the retransmission timer for the Graft can be stopped.

3.6.7. Graft Acknowledgment Packet Format

The format of a Graft Ack packet (which is identical to that of a Graft packet is show below:

```
                 7            15            23            31
         +-----------+------------+------------------------+
         |   Type    |    Code    |        Checksum        |
         |  (0x13)   |   (0x9)    |                        |
         +-----------+------------+------------+-----------+
         |       Reserved         |   Minor    |   Major   |
         +------------------------+------------+-----------+
         |                 Source Address                  |
         +-------------------------------------------------+
         |                  Group Address                  |
         +-------------------------------------------------+
```

Figure 7 - Graft Ack Packet Format

3.7. Interfaces

Interfaces running DVMRP will either be multicast capable physical interfaces or encapsulated tunnel pseudo-interfaces. Physical interfaces may either be multi-access networks or point-to-point networks. Tunnel interfaces are used when there are non-multicast capable routers between DVMRP neighbors. Multicast data traffic is sent between tunnel endpoints using IP-IP encapsulation. The unicast IP addresses of the tunnel endpoints are used as the source and destination IP addresses in the outer IP header. The inner IP header remains unchanged from the original data packet.

Since DVMRP Protocol messages are not encapsulated when sent between tunnel endpoints, they must always be sent directly to the unicast address of the tunneled neighbor.

4. Security Considerations

Security for DVMRP follows the general security architecture provided for the Internet Protocol [Atk95a]. This framework provides for both privacy and authentication. It recommends the use of the IP Authentication Header [Atk95b] to provide trusted neighbor

relationships. Confidentiality is provided by the addition of the IP Encapsulating Security Payload [Atk95c]. Please refer to these documents for the general architecture design as well as the specific implementation details.

5. References

[Atk95a] Atkinson, R., "Security Architecture for the Internet Protocol", RFC 1825, August 1995.

[Atk95b] Atkinson, R., "IP Authentication Header", RFC 1826, August 1995.

[Atk95c] Atkinson, R., "IP Encapsulating Security Payload (ESP)", RFC 1827, August 1995.

[Deer89] Deering, S., "Host Extensions for IP Multicasting", RFC 1112, August 1989.

[Deer90] Deering, S., Cheriton, D., "Multicast Routing in Datagram Internetworks and Extended LANs", ACM Transactions on Computer Systems, Vol. 8, No. 2, May 1990, Pages 85-110.

[Fenn96] Fenner, W., "Internet Group Management Protocol, Version 2", Work In Progress, February 1996.

[Full93] Fuller, V., T. Li, J. Yu, and K. Varadhan, "Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy", RFC 1519, September 1993.

[Perk96] Perkins, C., IP Encapsulation within IP, Work in Progress, May 1996.

[Rekh93] Rekhter, Y., and T. Li, "An Architecture for IP Address Allocation with CIDR", RFC 1518, September 1993.

[Reyn94] Reynolds, J., Postel, J., "Assigned Numbers", STD 0002, October, 1994.

[Wait88] Waitzman, D., Partridge, C., Deering, S., "Distance Vector Multicast Routing Protocol", RFC 1075, November 1988.

6. Author's Address

Thomas Pusateri
Juniper Networks, Inc.
3260 Jay St.
Santa Clara, CA 95051
Phone: (919) 558-0700
EMail: pusateri@jnx.com

7. Acknowledgments

The author would like to acknowledge the original designers of the protocol, Steve Deering, Craig Partridge, and David Waitzman. Version 3 of the protocol would not have been possible without the work of Ajit Thyagarajan and Bill Fenner. Credit also goes to Dave LeRoy and Danny Mitzel for the careful review of this document.

8. Appendix A - Constants & Configurable Parameters

The following table provides a summary of the DVMRP timing parameters:

Parameter	Value (seconds)
Probe Interval	10
Neighbor Time-out Interval	35
Route Report Interval	60
Route Replacement Time	140
Route Expiration Time	200
Prune Lifetime	variable (< 2 hours)
Graft Retransmission Time	5 with exp. backoff

Table 2 - Parameter Summary

9. Appendix B - Tracing and Troubleshooting support

There are several packet types used to gather DVMRP specific information. They are generally used for diagnosing problems or gathering topology information. The first two messages are now obsoleted and should not be used. The remaining two messages provide a request/response mechanism to determine the versions and capabilities of a particular DVMRP router.

Code	Packet Type	Description
3	DVMRP Ask Neighbors	Obsolete
4	DVMRP Neighbors	Obsolete
5	DVMRP Ask Neighbors 2	Request Neighbor List
6	DVMRP Neighbors 2	Respond with Neighbor List

Table 3 - Debugging Packet Types

9.1. DVMRP Ask Neighbors2

The Ask Neighbors2 packet is a unicast request packet directed at a DVMRP router. The destination should respond with a unicast Neighbors2 message back to the sender of the Ask Neighbors2 message.

```
 0         8         16               31
+---------+---------+-------------------+
| Type    |  Code   |      Checksum     |
|(0x13)   | (0x5)   |                   |
+---------+---------+----------+---------+
|     Reserved      |  Minor   | Major   |
|                   | Version  |Version  |
+-------------------+----------+---------+
```

Figure 8 - Ask Neighbors 2 Packet Format

9.2. DVMRP Neighbors2

The format of a Neighbors2 response packet is shown below. This is sent as a unicast message back to the sender of an Ask Neighbors2 message. There is a common header at the top followed by the routers capabilities. One or more sections follow that contain an entry for each logical interface. The interface parameters are listed along with a variable list of neighbors learned on each interface.

```
 0            8              16                      31
+-----------+--------------+-------------------------+
|   Type    |     Code     |         Checksum        |
|  (0x13)   |    (0x6)     |                         |
+-----------+--------------+------------+------------+
| Reserved  | Capabilities |   Minor    |   Major    |
|           |              |  Version   |  Version   |
+-----------+--------------+------------+------------+
|                                                    |
|                   Local Addr 1                     |
+-----------+--------------+------------+------------+
|           |              |            |            |
| Metric 1  | Threshold 1  |  Flags 1   | Nbr Count 1|
+-----------+--------------+------------+------------+
|                                                    |
|                      Nbr 1                         |
+----------------------------------------------------+
|                                                    |
|                       ...                          |
+----------------------------------------------------+
|                                                    |
|                      Nbr m                         |
+----------------------------------------------------+
|                                                    |
|                   Local Addr N                     |
+-----------+--------------+------------+------------+
|           |              |            |            |
| Metric N  | Threshold N  |  Flags N   | Nbr Count N|
+-----------+--------------+------------+------------+
|                                                    |
|                      Nbr 1                         |
+----------------------------------------------------+
|                                                    |
|                       ...                          |
+----------------------------------------------------+
|                                                    |
|                      Nbr k                         |
+----------------------------------------------------+
```

Figure 9 - Neighbors 2 Packet Format

The capabilities of the local router are defined as follows:

Bit	Flag	Description
0	Leaf	This is a leaf router
1	Prune	This router understands pruning
2	GenID	This router sends Generation IDs
3	Mtrace	This router handles Mtrace requests

Table 4 - DVMRP Router Capabilities

The flags associated with a particular interface are:

Bit	Flag	Description
0	Tunnel	Neighbor reached via tunnel
1	Source Route	Tunnel uses IP source routing
2	Reserved	No longer used
3	Down	Operational status down
4	Disabled	Administrative status down
5	Reserved	No longer used
6	Leaf	No downstream neighbors on interface

Table 5 - DVMRP Interface flags

10. Appendix C - Version Compatibility

There have been two previous major versions of DVMRP with implementations still in circulation. If the receipt of a Probe message reveals a major version of 1 or 2, then it can be assumed that this neighbor does not support pruning or the use of the Generation ID in the Probe message. However, since these older implementations are known to safely ignore the Generation ID and neighbor information in the Probe packet, it is not necessary to send specially formatted Probe packets to these neighbors.

There were two minor versions (1 and 2) of major version 3 that did support pruning but did not support the Generation ID or capability flags. These special cases will have to be accounted for.

Any other minor versions of major version 3 closely compare to this specification.

In addition, cisco Systems is known to use their software major and minor release number as the DVMRP major and minor version number. These will typically be 10 or 11 for the major version number. Pruning was introduced in Version 11.

Implementations prior to this specification may not wait to send route reports until probe messages have been received with the routers address listed. Reports SHOULD be sent to these neighbors without first requiring a received probe with the routers address in it as well as reports from these neighbors SHOULD be accepted. Although, this allows one-way neighbor relationships to occur, it does maintain backward compatibility.

Table of Contents