Protocol Independent Multicast–Sparse Mode (PIM-SM): Protocol Specification

Deborah Estrin
Computer Science Dept/ISI
University of Southern Calif.
Los Angeles, CA 90089
estrin@usc.edu

Dino Farinacci
Cisco Systems Inc.
170 West Tasman Drive,
San Jose, CA 95134
dino@cisco.com

Ahmed Helmy
Computer Science Dept.
University of Southern Calif.
Los Angeles, CA 90089
ahelmy@catarina.usc.edu

David Thaler
EECS Department
University of Michigan
Ann Arbor, MI 48109
thalerd@eecs.umich.edu

Stephen Deering
Xerox PARC
3333 Coyote Hill Road
Palo Alto, CA 94304
deering@parc.xerox.com

Mark Handley
Department of Computer Science
University College London
Gower Street
London, WC1E 6BT
UK
m.handley@cs.ucl.ac.uk

Van Jacobson
Lawrence Berkeley Laboratory
1 Cyclotron Road
Berkeley, CA 94720
van@ee.lbl.gov

Ching-gung Liu
Computer Science Dept.
University of Southern Calif.
Los Angeles, CA 90089
charley@catarina.usc.edu

Puneet Sharma
Computer Science Dept.
University of Southern Calif.
Los Angeles, CA 90089
puneet@catarina.usc.edu

Liming Wei [1]
Cisco Systems Inc.
170 West Tasman Drive,
San Jose, CA 95134
lwei@cisco.com

draft-ietf-idmr-PIM-SM-specv2-00.ps

September 9, 1997

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. Internet Drafts may be updated, replaced, or obsoleted by other documents at any time. It is not appropriate to use Internet Drafts as reference material or to cite them other than as a "working draft" or "work in progress."

Please check the I-D abstract listing contained in each Internet Draft directory to learn the current status of this or any other Internet Draft.

[1] The author list has been reordered to reflect the involvement in detailed editorial work on this specification document. The first four authors are the primary editors and are listed alphabetically. The rest of the authors, also listed alphabetically, participated in all aspects of the architectural and detailed design but managed to get away without hacking the latex!

Contents

1 Introduction

This document describes a protocol for efficiently routing to multicast groups that may span wide-area (and inter-domain) internets. We refer to the approach as Protocol Independent Multicast–Sparse Mode (PIM-SM) because it is not dependent on any particular unicast routing protocol, and because it is designed to support sparse groups as defined in [1, 2]. This document describes the protocol details. For the motivation behind the design and a description of the architecture, see [1, 2].

Section 2 summarizes PIM-SM operation. It describes the protocol from a network perspective, in particular, how the participating routers interact to create and maintain the multicast distribution tree. Section 3 describes PIM-SM operations from the perspective of a single router implementing the protocol; this section constitutes the main body of the protocol specification. It is organized according to PIM-SM message type; for each message type we describe its contents, its generation, and its processing.

Sections 3.8 and 3.9 summarize the timers and flags referred to throughout this document. Section 4 provides packet format details.

The most significant functional changes since the January '95 version involve the Rendezvous Point-related mechanisms, several resulting simplifications to the protocol, and removal of the PIM-DM protocol details to a separate document [3] (for clarity).

2 PIM-SM Protocol Overview

In this section we provide an overview of the architectural components of PIM-SM.

A router receives explicit Join/Prune messages from those neighboring routers that have downstream group members. The router then forwards data packets addressed to a multicast group, G, only onto those interfaces on which explicit joins have been received. Note that all routers mentioned in this document are assumed to be PIM-SM capable, unless otherwise specified.

A Designated Router (DR) sends periodic Join/Prune messages toward a group-specific Rendezvous Point (RP) for each group for which it has active members. Each router along the path toward the RP builds a wildcard (any-source) state for the group and sends Join/Prune messages on toward the RP. We use the term *route entry* to refer to the state maintained in a router to represent the distribution tree. A route entry may include such fields as the source address, the group address, the incoming interface from which packets are accepted, the list of outgoing interfaces to which packets are sent, timers, flag bits, etc. The wildcard route entry's incoming interface points toward the RP; the outgoing interfaces point to the neighboring downstream routers that have sent Join/Prune messages toward the RP. This state creates a shared, RP-centered, distribution tree that reaches all group members. When a data source first sends to a group, its DR unicasts Register messages to the RP with the source's data packets encapsulated within. If the data rate is high, the RP can send source-specific Join/Prune messages back towards the source and the source's data packets will follow the resulting forwarding state and travel unencapsulated to the RP. Whether they arrive encapsulated or natively, the RP forwards the source's decapsulated data packets down the RP-centered distribution tree toward group members. If the data rate warrants it, routers with local receivers can join a source-specific, shortest path, distribution tree, and prune this source's packets off of the shared RP-centered tree. For low data rate sources, neither the RP, nor last-hop routers need join a source-specific shortest path tree and data packets can be delivered via the shared, RP-tree.

The following subsections describe SM operation in more detail, in particular, the control messages, and the actions they trigger.

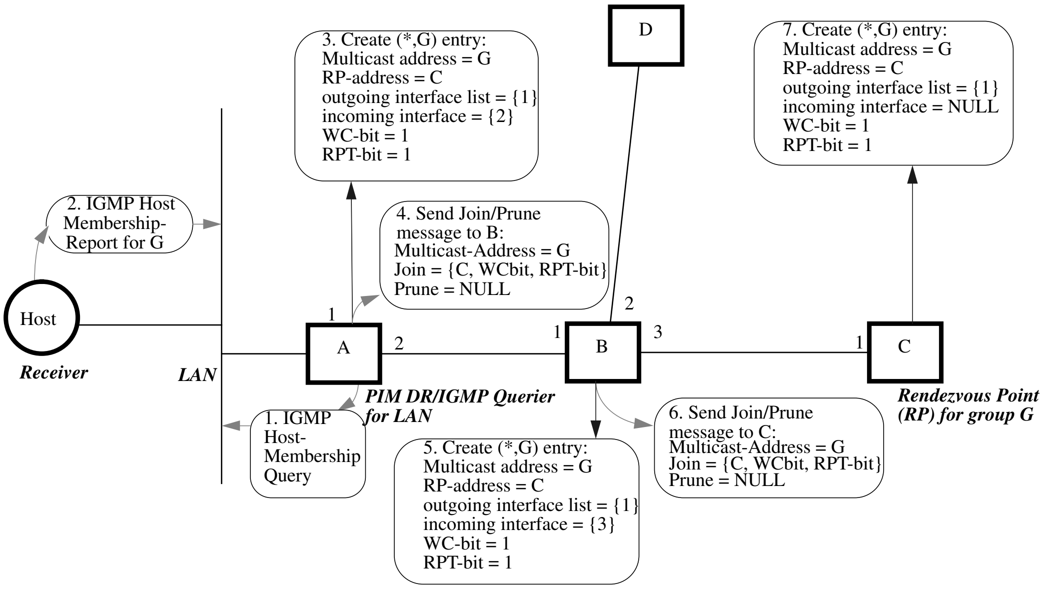


Figure 1: Example: how a receiver joins, and sets up shared tree
Actions are numbered in the order they occur

2.1 Local hosts joining a group

In order to join a multicast group, G, a host conveys its membership information through the Internet Group Management Protocol (IGMP), as specified in [4, 5], (see figure 1). From this point on we refer to such a host as a receiver, R, (or member) of the group G.

Note that all figures used in this section are for illustration and are not intended to be complete. For complete and detailed protocol action see Section 3.

When a DR (e.g., router A in figure 1) gets a membership indication from IGMP for a new group, G, the DR looks up the associated RP. The DR creates a wildcard multicast route entry for the group, referred to here as a (*,G) entry; if there is no more specific match for a particular source, the packet will be forwarded according to this entry.

The RP address is included in a special field in the route entry and is included in periodic upstream Join/Prune messages. The outgoing interface is set to that included in the IGMP membership indication for the new member. The incoming interface is set to the interface used to send unicast packets to the RP.

When there are no longer directly connected members for the group, IGMP notifies the DR. If the DR has neither local members nor downstream receivers, the (*,G) state is deleted.

2.2 Establishing the RP-rooted shared tree

Triggered by the (*,G) state, the DR creates a Join/Prune message with the RP address in its join list and the the wildcard bit (WC-bit) and RP-tree bit (RPT-bit) set to 1. The WC-bit indicates that any source may match and be forwarded according to this entry if there is no longer match; the RPT-bit indicates that this join is being sent up the shared, RP-tree. The prune list is left empty. When the RPT-bit is set to 1 it indicates that the join is associated with the shared RP-tree and therefore the Join/Prune message is propagated along the RP-tree. When the WC-bit is set to 1 it indicates that the address is an RP and the downstream receivers expect to receive packets from all sources via this (shared tree) path. The term RPT-bit is used to refer to both the RPT-bit flags associated with route entries, and the RPT-bit included in each encoded address in a Join/Prune message.

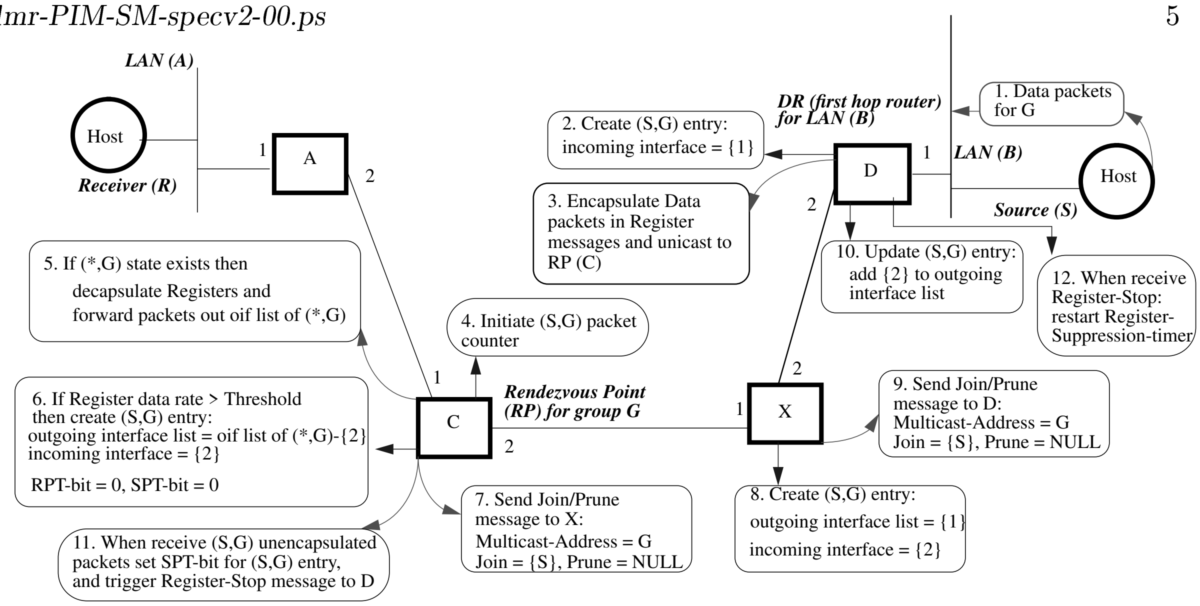


Figure 2: Example: a host sending to a group
Actions are numbered in the order they occur

Each upstream router creates or updates its multicast route entry for (*,G) when it receives a Join/Prune with the RPT-bit and WC-bit set. The interface on which the Join/Prune message arrived is added to the list of outgoing interfaces (oifs) for (*,G). Based on this entry each upstream router between the receiver and the RP sends a Join/Prune message in which the join list includes the RP. The packet payload contains Multicast-Address=G, Join={RP,WC-bit,RPT-bit}, Prune=NULL.

2.3 Hosts sending to a group

When a host starts sending multicast data packets to a group, initially its DR must deliver each packet to the RP for distribution down the RP-tree (see figure 2). The sender's DR initially encapsulates each data packet in a Register message and unicasts it to the RP for that group. The RP decapsulates each Register message and forwards the enclosed data packet natively to downstream members on the shared RP-tree.

If the data rate of the source warrants the use of a source-specific shortest path tree (SPT), the RP may construct a new multicast route entry that is specific to the source, hereafter referred to as (S,G) state, and send periodic Join/Prune messages toward the source. Note that over time, the rules for when to switch can be modified without global coordination. When and if the RP does switch to the SPT, the routers between the source and the RP build and maintain (S,G) state in response to these messages and send (S,G) messages upstream toward the source.

The source's DR must stop encapsulating data packets in Registers when (and so long as) it receives Register-Stop messages from the RP. The RP triggers Register-Stop messages in response to Registers, if the RP has no downstream receivers for the group (or for that particular source), or if the RP has already joined the (S,G) tree and is receiving the data packets natively. Each source's DR maintains, per (S,G), a Register-Suppression-timer. The Register-Suppression-timer is started by the Register-Stop message; upon expiration, the source's DR resumes sending data packets to the RP, encapsulated in Register messages.

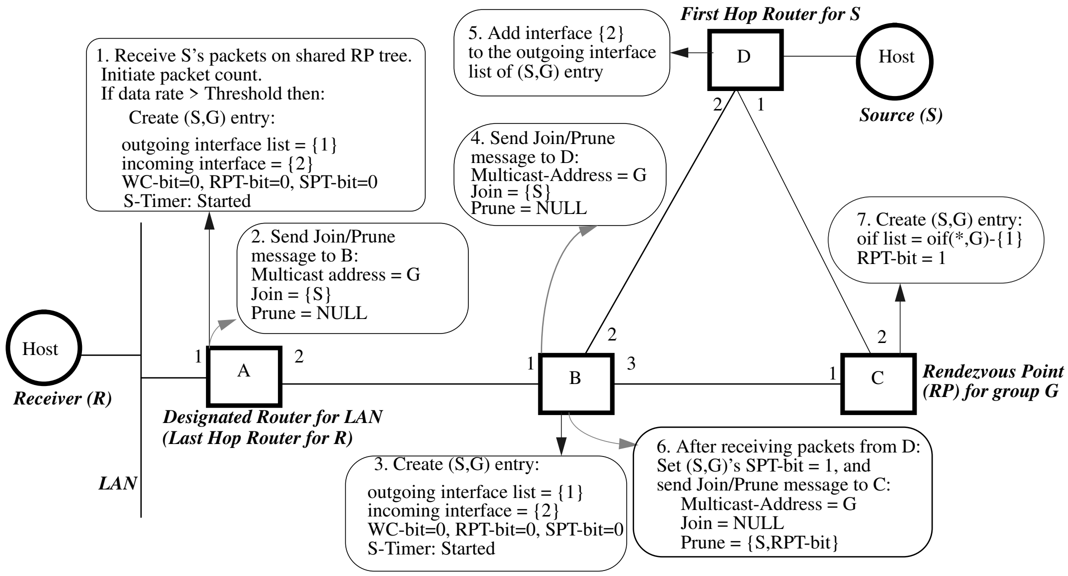


Figure 3: Example: Switching from shared tree to shortest path tree
Actions are numbered in the order they occur

2.4 Switching from shared tree (RP-tree) to shortest path tree (SP-tree)

A router with directly-connected members first joins the shared RP-tree. The router can switch to a source's shortest path tree (SP-tree) after receiving packets from that source over the shared RP-tree. The recommended policy is to initiate the switch to the SP-tree after receiving a significant number of data packets during a specified time interval from a particular source. To realize this policy the router can monitor data packets from sources for which it has *no* source-specific multicast route entry and initiate such an entry when the data rate exceeds the configured threshold. As shown in figure 3, router 'A' initiates a (S,G) state.

When a (S,G) entry is activated (and periodically so long as the state exists), a Join/Prune message is sent upstream towards the source, S, with S in the join list. The payload contains Multicast-Address=G, Join={S}, Prune=NULL. When the (S,G) entry is created, the outgoing interface list is copied from (*,G), i.e., all local shared tree branches are replicated in the new shortest path tree. In this way when a data packet from S arrives and matches on this entry, all receivers will continue to receive the source's packets along this path. (In more complicated scenarios, other entries in the router have to be considered, as described in Section 3).

Note that (S,G) state must be maintained in each last-hop router that is responsible for initiating and maintaining an SP-tree. Even when (*,G) and (S,G) overlap, both states are needed to trigger the source-specific Join/Prune messages. (S,G) state is kept alive by data packets arriving from that source. A timer, Entry-timer, is set for the (S,G) entry and this timer is restarted whenever data packets for (S,G) are forwarded out at least one oif, or Registers are sent. When the Entry-timer expires, the state is deleted. The last-hop router is the router that delivers the packets to their ultimate end-system destination. This is the router that monitors if there is group membership and joins or prunes the appropriate distribution trees in response. In general the last-hop router is the Designated Router (DR) for the LAN. However, under various conditions described later, a parallel router connected to the same LAN may take over as the last-hop router in place of the DR.

Only the RP and routers with local members can initiate switching to the SP-tree; intermediate routers do not. Consequently, last-hop routers create (S,G) state in response to data packets from the source, S; whereas intermediate routers only create (S,G) state in response to Join/Prune messages from downstream that have S in the Join list.

The (S,G) entry is initialized with the SPT-bit cleared, indicating that the shortest path tree branch from S has not yet been setup completely, and the router can still accept packets from S that arrive on the (*,G) entry's indicated incoming interface (iif). Each PIM multicast entry has an associated incoming interface on which packets are expected to arrive.

When a router with a (S,G) entry and a cleared SPT-bit starts to receive packets from the new source S on the iif for the (S,G) entry, and that iif differs from the (*,G) entry's iif, the router sets the SPT-bit, and sends a Join/Prune message towards the RP, indicating that the router no longer wants to receive packets from S via the shared RP-tree. The Join/Prune message sent towards the RP includes S in the prune list, with the RPT-bit set indicating that S's packets must not be forwarded down this branch of the shared tree. If the router receiving the Join/Prune message has (S,G) state (with or without the route entry's RPT-bit flag set), it deletes the arriving interface from the (S,G) oif list. If the router has only (*,G) state, it creates an entry with the RPT-bit flag set to 1. For brevity we refer to an (S,G) entry that has the RPT-bit flag set to 1 as an (S,G)RPT-bit entry. This notational distinction is useful to point out the different actions taken for (S,G) entries depending on the setting of the RPT-bit flag. Note that a router can have no more than one active (S,G) entry for any particular S and G, at any particular time; whether the RPT-bit flag is set or not. In other words, a router never has both an (S,G) and an (S,G)RPT-bit entry for the same S and G at the same time. The Join/Prune message payload contains Multicast-Address=G, Join=NULL, Prune={S,RPT-bit}.

A new receiver may join an existing RP-tree on which source-specific prune state has been established (e.g., because downstream receivers have switched to SP-trees). In this case the prune state must be eradicated upstream of the new receiver to bring all sources' data packets down to the new receiver. Therefore, when a (*,G) Join arrives at a router that has any (Si,G)RPT-bit entries (i.e., entries that cause the router to send source-specific prunes toward the RP), these entries must be updated upstream of the router so as to bring all sources' packets down to the new member. To accomplish this, each router that receives a (*,G) Join/Prune message updates all existing (S,G)RPT-bit entries. The router may also trigger a (*,G) Join/Prune message upstream to cause the same updating of RPT-bit settings upstream and pull down all active sources' packets. If the arriving (*,G) join has some sources included in its prune list, then the corresponding (S,G)RPT-bit entries are left unchanged (i.e., the RPT-bit remains set and no oif is added).

2.5 Steady state maintenance of distribution tree (i.e., router state)

In the steady state each router sends periodic Join/Prune messages for each active PIM route entry; the Join/Prune messages are sent to the neighbor indicated in the corresponding entry. These messages are sent periodically to capture state, topology, and membership changes. A Join/Prune message is also sent on an event-triggered basis each time a new route entry is established for some new source (note that some damping function may be applied, e.g., a short delay to allow for merging of new Join information). Join/Prune messages do not elicit any form of explicit acknowledgment; routers recover from lost packets using the periodic refresh mechanism.

2.6 Obtaining RP information

To obtain the RP information, all routers within a PIM domain collect Bootstrap messages. Bootstrap messages are sent hop-by-hop within the domain; the domain's bootstrap router (BSR) is responsible for originating the Bootstrap messages. Bootstrap messages are used to carry out a dynamic BSR election when needed and to distribute RP information in steady state.

A domain in this context is a contiguous set of routers that all implement PIM and are configured to operate within a common boundary defined by PIM Multicast Border Routers (PMBRs). PMBRs connect each PIM domain to the rest of the internet.

Routers use a set of available RPs (called the *RP-Set*) distributed in Bootstrap messages to get the proper Group to RP mapping. The following paragraphs summarize the mechanism; details of the mechanism may be found in Sections 3.6 and Appendix 6.2.

A (small) set of routers, within a domain, are configured as candidate BSRs and, through a simple election mechanism, a single BSR is selected for that domain. A set of routers within a domain are also configured as candidate RPs (C-RPs); typically these will be the same routers that are configured as C-BSRs. Candidate RPs periodically unicast Candidate-RP-Advertisement messages (C-RP-Advs) to the BSR of that domain. C-RP-Advs include the address of the advertising C-RP, as well as an optional group address and a mask length field, indicating the group prefix(es) for which the candidacy is advertised. The BSR then includes a set of these Candidate-RPs (the RP-Set), along with the corresponding group prefixes, in Bootstrap messages it periodically originates. Bootstrap messages are distributed hop-by-hop throughout the domain.

Routers receive and store Bootstrap messages originated by the BSR. When a DR gets a membership indication from IGMP for (or a data packet from) a directly connected host, for a group for which it has no entry, the DR uses a hash function to map the group address to one of the C-RPs whose Group-prefix includes the group (see Section 3.7). The DR then sends a Join/Prune message towards (or unicasts Registers to) that RP.

The Bootstrap message indicates liveness of the RPs included therein. If an RP is included in the message, then it is tagged as 'up' at the routers; while RPs not included in the message are removed from the list of RPs over which the hash algorithm acts. Each router continues to use the contents of the most recently received Bootstrap message until it receives a new Bootstrap message.

If a PIM domain partitions, each area separated from the old BSR will elect its own BSR, which will distribute an RP-Set containing RPs that are reachable within that partition. When the partition heals, another election will occur automatically and only one of the BSRs will continue to send out Bootstrap messages. As is expected at the time of a partition or healing, some disruption in packet delivery may occur. This time will be on the order of the region's round-trip time and the bootstrap router timeout value.

2.7 Interoperation with dense mode protocols such as DVMRP

In order to interoperate with networks that run dense-mode, *broadcast and prune*, protocols, such as DVMRP, all packets generated within a PIM-SM region must be pulled out to that region's PIM Multicast Border Routers (PMBRs) and injected (i.e., broadcast) into the DVMRP network. A PMBR is a router that sits at the boundary of a PIM-SM domain and interoperates with other types of multicast routers such as those that run DVMRP. Generally a PMBR would speak both protocols and implement interoperability functions not required by regular PIM routers. To support interoperability, a special entry type, referred to as (*,*,RP), must be supported by all PIM routers. For this reason we include details about (*,*,RP) entry handling in this general PIM specification.

A data packet will match on a (*,*,RP) entry if there is no more specific entry (such as (S,G) or (*,G)) and the destination group address in the packet maps to the RP listed in the (*,*,RP) entry. In this sense, a (*,*,RP) entry represents an aggregation of all the groups that hash to that RP. PMBRs initialize (*,*,RP) state for each RP in the domain's RPset. The (*,*,RP) state causes the PMBRs to send (*,*,RP) Join/Prune messages toward each of the active RPs in the domain. As a result distribution trees are built that carry all data packets originated within the PIM domain (and sent to the RPs) down to the PMBRs.

PMBRs are also responsible for delivering externally-generated packets to routers within the PIM domain. To do so, PMBRs initially encapsulate externally-originated packets (i.e., received on DVMRP interfaces) in Register messages and unicast them to the corresponding RP within the PIM domain. The Register message has a bit indicating that it was originated by a border router and the RP caches the originating PMBR's address in the route entry so that duplicate Registers from other PMBRs can be declined with a Register-Stop message.

All PIM routers must be capable of supporting (*,*,RP) state and interpreting associated Join/Prune messages. We describe the handling of (*,*,RP) entries and messages throughout this document; however, detailed PIM Multicast Border Router (PMBR) functions will be specified in a separate interoperability document (see directory, http://catarina.usc.edu/pim/interop/).

2.8 Multicast data packet processing

Data packets are processed in a manner similar to other multicast schemes. A router first performs a longest match on the source and group address in the data packet. A (S,G) entry is matched first if one exists; a (*,G) entry is matched otherwise. If neither state exists, then a (*,*,RP) entry match is attempted as follows: the router hashes on G to identify the RP for group G, and looks for a (*,*,RP) entry that has this RP address associated with it. If none of the above exists, then the packet is dropped. If a state is matched, the router compares the interface on which the packet arrived to the incoming interface field in the matched route entry. If the iif check fails the packet is dropped, otherwise the packet is forwarded to all interfaces listed in the outgoing interface list.

Some special actions are needed to deliver packets continuously while switching from the shared to shortest-path tree. In particular, when a (S,G) entry is matched, incoming packets are forwarded as follows:

1. If the SPT-bit is set, then:
   (a) if the incoming interface is the same as a matching (S,G) iif, the packet is forwarded to the oif-list of (S,G).
   (b) if the incoming interface is different than a matching (S,G) iif , the packet is discarded.
2. If the SPT-bit is cleared, then:
   (a) if the incoming interface is the same as a matching (S,G) iif, the packet is forwarded to the oif-list of (S,G). In addition, the SPT bit is set for that entry if the incoming interface differs from the incoming interface of the (*,G) or (*,*,RP) entry.
   (b) if the incoming interface is different than a matching (S,G) iif, the incoming interface is tested against a matching (*,G) or (*,*,RP) entry. If the iif is the same as one of those, the packet is forwarded to the oif-list of the matching entry.
   (c) Otherwise the iif does not match any entry for G and the packet is discarded.

Data packets never trigger prunes. However, data packets may trigger actions that in turn trigger prunes. For example, when router B in figure 3 decides to switch to SP-tree at step 3, it creates a (S,G) entry with SPT-bit set to 0. When data packets from S arrive at interface 2 of B, B sets the SPT-bit to 1 since the iif for (*,G) is different than that for (S,G). This triggers the sending of prunes towards the RP.

2.9 Operation over Multi-access Networks

This section describes a few additional protocol mechanisms needed to operate PIM over multi-access networks: Designated Router election, Assert messages to resolve parallel paths, and the Join/Prune-Suppression-Timer to suppress redundant Joins on multi-access networks.

Designated router election

When there are multiple routers connected to a multi-access network, one of them must be chosen to operate as the designated router (DR) at any point in time. The DR is responsible for sending triggered Join/Prune and Register messages toward the RP.

A simple designated router (DR) election mechanism is used for both SM and traditional IP multicast routing. Neighboring routers send Hello messages to each other. The sender with the largest network layer address assumes the role of DR. Each router connected to the multi-access LAN sends the Hellos periodically in order to adapt to changes in router status.

Parallel paths to a source or the RP–Assert process

If a router receives a multicast datagram on a multi-access LAN from a source whose corresponding (S,G) outgoing interface list includes the interface to that LAN, the packet must be a duplicate. In this case a single forwarder must be elected. Using Assert messages addressed to '224.0.0.13' (ALL-PIM-ROUTERS group) on the LAN, upstream routers can resolve which one will act as the forwarder. Downstream routers listen to the Asserts so they know which one was elected, and therefore where to send subsequent Joins. Typically this is the same as the downstream router's RPF (Reverse Path Forwarding) neighbor; but there are circumstances where this might not be the case, e.g., when using multiple unicast routing protocols on that LAN. The RPF neighbor for a particular source (or RP) is the next-hop router to which packets are forwarded en route to that source (or RP); and therefore is considered a good path via which to accept packets from that source.

The upstream router elected is the one that has the shortest distance to the source. Therefore, when a packet is received on an outgoing interface a router sends an Assert message on the multi-access LAN indicating what metric it uses to reach the source of the data packet. The router with the smallest numerical metric (with ties broken by highest address) will become the forwarder. All other upstream routers will delete the interface from their outgoing interface list. The downstream routers also do the comparison in case the forwarder is different than the RPF neighbor.

Associated with the metric is a metric preference value. This is provided to deal with the case where the upstream routers may run different unicast routing protocols. The numerically smaller metric preference is always preferred. The metric preference is treated as the high-order part of an assert metric comparison. Therefore, a metric value can be compared with another metric value provided both metric preferences are the same. A metric preference can be assigned per unicast routing protocol and needs to be consistent for all routers on the multi-access network.

Asserts are also needed for (*,G) entries since an RP-Tree and an SP-Tree for the same group may both cross the same multi-access network. When an assert is sent for a (*,G) entry, the first bit in the metric preference (RPT-bit) is always set to 1 to indicate that this path corresponds to the RP tree, and that the match must be done on (*,G) if it exists. Furthermore, the RPT-bit is always cleared for metric preferences that refer to SP-tree entries; this causes an SP-tree path to always look better than an RP-tree path. When the SP-tree and RPtree cross the same LAN, this mechanism eliminates the duplicates that would otherwise be carried over the LAN.

In case the packet, or the Assert message, matches on oif for (*,*,RP) entry, a (*,G) entry is created, and asserts take place as if the matching state were (*,G).

The DR may lose the (*,G) Assert process to another router on the LAN if there are multiple paths to the RP through the LAN. From then on, the DR is no longer the last-hop router for local receivers and removes the LAN from its (*,G) oif list. The winning router becomes the last-hop router and is responsible for sending (*,G) join messages to the RP.

Join/Prune suppression

Join/Prune suppression may be used on multi-access LANs to reduce duplicate control message overhead; it is not required for correct performance of the protocol. If a Join/Prune message arrives and matches on the incoming interface for an existing (S,G), (*,G), or (*,*,RP) route entry, and the Holdtime included in the Join/Prune message is greater than the recipient's own [Join/Prune-Holdtime] (with ties resolved in favor of the higher network layer address), a timer (the Join/Prune-Suppression-timer) in the recipient's route entry may be started to suppress further Join/Prune messages. After this timer expires, the recipient triggers a Join/Prune message, and resumes sending periodic Join/Prunes, for this entry. The Join/Prune-Suppression-timer should be restarted each time a Join/Prune message is received with a higher Holdtime.

2.10 Unicast Routing Changes

When unicast routing changes, an RPF check is done on all active (S,G), (*,G) and (*,*,RP) entries, and all affected expected incoming interfaces are updated. In particular, if the new incoming interface appears in the outgoing interface list, it is deleted from the outgoing interface list. The previous incoming interface may be added to the outgoing interface list by a subsequent Join/Prune from downstream. Join/Prune messages received on the current incoming interface are ignored. Join/Prune messages received on new interfaces or existing outgoing interfaces are not ignored. Other outgoing interfaces are left as is until they are explicitly pruned by downstream routers or are timed out due to lack of appropriate Join/Prune messages. If the router has a (S,G) entry with the SPT-bit set, and the updated iif(S,G) does *not* differ from iif(*,G) or iif(*,*,RP), then the router resets the SPT-bit.

The router must send a Join/Prune message with S in the Join list out any new incoming interfaces to inform upstream routers that it expects multicast datagrams over the interface. It may also send a Join/Prune message with S in the Prune list out the old incoming interface, if the link is operational, to inform upstream routers that this part of the distribution tree is going away.

2.11 PIM-SM for Inter-Domain Multicast

Future documents will address the use of PIM-SM as a backbone inter-domain multicast routing protocol. Design choices center primarily around the distribution and usage of RP information for wide area, inter-domain groups.

2.12 Security

All PIM control messages may use IPsec [6] to address security concerns. Security mechanisms are likely to be enhanced in the near future.

3 Detailed Protocol Description

This section describes the protocol operations from the perspective of an individual router implementation. In particular, for each message type we describe how it is generated and processed.

3.1 Hello

Hello messages are sent so neighboring routers can discover each other.

3.1.1 Sending Hellos

Hello messages are sent periodically between PIM neighbors, every [Hello-Period] seconds. This informs routers what interfaces have PIM neighbors. Hello messages are multicast using address 224.0.0.13 (ALL-PIM-ROUTERS group). The packet includes a Holdtime, set to [Hello-Holdtime], for neighbors to keep the information valid. Hellos are sent on all types of communication links.

3.1.2 Receiving Hellos

When a router receives a Hello message, it stores the network layer address for that neighbor, sets its Neighbor-timer for the Hello sender to the Holdtime included in the Hello, and determines the Designated Router (DR) for that interface. The highest addressed system is elected DR. Each Hello received causes the DR's address to be updated.

When a router that is the active DR receives a Hello from a new neighbor (i.e., from an address that is not yet in the DRs neighbor table), the DR unicasts its most recent RP-set information to the new neighbor.

3.1.3 Timing out neighbor entries

A periodic process is run to time out PIM neighbors that have not sent Hellos. If the DR has gone down, a new DR is chosen by scanning all neighbors on the interface and selecting the new DR to be the one with the highest network layer address. If an interface has gone down, the router may optionally time out all PIM neighbors associated with the interface.

3.2 Join/Prune

Join/Prune messages are sent to join or prune a branch off of the multicast distribution tree. A single message contains both a join and prune list, either one of which may be null. Each list contains a set of source addresses, indicating the source-specific trees or shared tree that the router wants to join or prune.

3.2.1 Sending Join/Prune Messages

Join/Prune messages are merged such that a message sent to a particular upstream neighbor, N, includes all of the current joined and pruned sources that are reached via N; according to unicast routing Join/Prune messages are multicast to all routers on multi-access networks with the target address set to the next hop router towards S or RP. Join/Prune messages are sent every [Join/Prune-Period] seconds. In the future we will introduce mechanisms to rate-limit this control traffic on a hop by hop basis, in order to avoid excessive overhead on small links. In addition, certain events cause triggered Join/Prune messages to be sent.

3.2.1.1 Periodic Join/Prune Messages A router sends a periodic Join/Prune message to each distinct RPF neighbor associated with each (S,G), (*,G) and (*,*,RP) entry. Join/Prune messages are only sent if the RPF neighbor is a PIM neighbor. A periodic Join/Prune message sent to a particular RPF neighbor is constructed as follows:

1. Each router determines the RP for a (*,G) entry by using the hash function described. The RP address (with RPT and WC bits set) is included in the join list of a periodic Join/Prune message under the following conditions:
   (a) The Join/Prune message is being sent to the RPF neighbor toward the RP for an active (*,G) or (*,*,RP) entry, and
   (b) The outgoing interface list in the (*,G) or (*,*,RP) entry is non-NULL, or the router is the DR on the same interface as the RPF neighbor.
2. A particular source address, S, is included in the join list with the RPT and WC bits cleared under the following conditions:
   (a) The Join/Prune message is being sent to the RPF neighbor toward S, and
   (b) There exists an active (S,G) entry with the RPT-bit flag cleared, and
   (c) The *oif* list in the (S,G) entry is not null.
3. A particular source address, S, is included in the prune list with the RPT and WC bits cleared under the following conditions:
   (a) The Join/Prune message is being sent to the RPF neighbor toward S, and
   (b) There exists an active (S,G) entry with the RPT-bit flag cleared, and
   (c) The *oif* list in the (S,G) entry is null.
4. A particular source address, S, is included in the prune list with the RPT-bit *set* and the WC bit cleared under the following conditions:
   (a) The Join/Prune message is being sent to the RPF neighbor toward the RP and there exists a (S,G) entry with the RPT-bit flag set and null *oif* list, or
   (b) The Join/Prune message is being sent to the RPF neighbor toward the RP, there exists a (S,G) entry with the RPT-bit flag cleared and SPT-bit set, and the incoming interface toward S is different than the incoming interface toward the RP, or
   (c) The Join/Prune message is being sent to the RPF neighbor toward the RP, and there exists a (*,G) entry and (S,G) entry for a directly connected source.
5. The RP address (with RPT and WC bits set) is included in the prune list if:
   (a) The Join/Prune message is being sent to the RPF neighbor toward the RP and there exists a (*,G) entry with a null *oif* list (see Section 3.5.2).

3.2.1.2 Triggered Join/Prune Messages In addition to periodic messages, the following events will trigger Join/Prune messages if as a result, a) a new entry is created, or b) the oif list changes from null to non-null or non-null to null. The contents of triggered messages are the same as the periodic, described above.

1. Receipt of an indication from IGMP that the state of directly-connected- membership has changed (i.e., new members have just joined 'membership indication' or all members have left), for a group G, may cause the last-hop router to build or modify corresponding (*,G) state. When IGMP indicates that there are no longer directly connected members, the oif is removed from the oif list if the oif-timer is not running. A Join/Prune message is triggered if and only if a) a new entry is created, or b) the oif list changes from null to non-null or non-null to null, as follows :
   (a) If the receiving router does not have a route entry for G the router creates a (*,G) entry, copies the oif list from the corresponding (*,*,RP) entry (if it exists), and includes the interface included in the IGMP membership indication in the oif list; as always, the router never includes the entry's iif in the oif list. The router sends a Join/Prune message towards the RP with the RP address and RPT-bit and WC-bits set in the join list. Or,
   (b) If a (S,G)RPT-bit or (*,G) entry already exists, the interface included in the IGMP membership indication is added to the oif list (if it was not included already).
2. Receipt of a Join/Prune message for (S,G), (*,G) or (*,*,RP) will cause building or modifying corresponding state, and subsequent triggering of upstream Join/Prune messages, in the following cases:
   (a) When there is no current route entry, the RP address included in the Join/Prune message is checked against the local RP-Set information. If it matches, an entry will be created and the new entry will in turn trigger an upstream Join/Prune message. If the router has no RP-Set information it may discard the message, or optionally use the RP address included in the message.
   (b) When the outgoing interface list of an (S,G)RPT-bit entry becomes null, the triggered Join/Prune message will contain S in the prune list.
   (c) When there exists a (S,G)RPT-bit with null oif list, and an (*,G) Join/Prune message is received, the arriving interface is added to the oif list and a (*,G) Join/Prune message is triggered upstream.
   (d) When there exists a (*,G) with null oif list, and a (*,*,RP) Join/Prune message is received, the receiving interface is added to the oif list and a (*,*,RP) Join/Prune message is triggered upstream.
3. Receipt of a packet that matches on a (S,G) entry whose SPT-bit is cleared triggers the following if the packet arrived on the correct incoming interface and there is a (*,G) or (*,*,RP) entry with a different incoming interface: a) the router sets the SPT-bit on the (S,G) entry, and b) the router sends a Join/Prune message towards the RP with S in the prune list and the RPT-bit set.
4. Receipt of a packet at the DR from a directly connected source S, on the subnet containing the address S, triggers a Join/Prune message towards the RP with S in the prune list and the RPT-bit set under the following conditions: a) there is no matching (S,G) state, and b) there exists a (*,G) or (*,*,RP) for which the DR is not the RP.
5. When a Join/Prune message is received for a group G, the prune list is checked. If the prune list contains a source or RP for which the receiving router has a corresponding active (S,G), (*,G) or (*,*,RP) entry, and whose *iif* is that on which the Join/Prune was received, then a join for (S,G), (*,G) or (*,*,RP) is triggered to override the prune, respectively. (This is necessary in the case of parallel downstream routers connected to a multi-access network.)

6. When the RP fails, the RP will not be included in the Bootstrap messages sent to all routers in that domain. This triggers the DRs to send (*,G) Join/Prune messages towards the new RP for the group, as determined by the RP-Set and the hash function. As described earlier, PMBRs trigger (*,*,RP) joins towards each RP in the RP-Set.

7. When an entry's Join/Prune-Suppression timer expires, a Join/Prune message is triggered upstream corresponding to that entry, even if the outgoing interface has not transitioned between null and non-null states.

8. When the RPF neighbor changes (whether due to an Assert or changes in unicast routing), the router sets a random delay timer (the Random-Delay-Join-Timer) whose expiration triggers sending of a Join/Prune message for the asserted route entry to the Assert winner (if the Join/Prune Suppression timer has expired.)

We do not trigger prunes onto interfaces based on data packets. Data packets that arrive on the wrong incoming interface are silently dropped. However, on point-to-point interfaces triggered prunes may be sent as an optimization.

3.2.1.3 Fragmentation It is possible that a Join/Prune message constructed according to the preceding rules could exceed the MTU of a network. In this case, the message can undergo semantic fragmentation whereby information corresponding to different groups can be sent in different messages. However, if a Join/Prune message must be fragmented the complete prune list corresponding to a group G must be included in the same Join/Prune message as the associated RP-tree Join for G. If such semantic fragmentation is not possible, IP fragmentation should be used between the two neighboring hops.

3.2.2 Receiving Join/Prune Messages

When a router receives a Join/Prune message, it processes it as follows.

The receiver of the Join/Prune notes the interface on which the PIM message arrived, call it I. The receiver then checks to see if the Join/Prune message was addressed to the receiving router itself (i.e., the router's address appears in the Unicast Upstream Neighbor Router field of the Join/Prune message). (If the router is connected to a multiaccess LAN, the message could be intended for a different router.) If the Join/Prune is for this router the following actions are taken.

For each group address G, in the Join/Prune message, the associated join list is processed as follows. We refer to each address in the join list as Sj; Sj refers to the RP if the RPT-bit and WC-bit are both set. For each Sj in the join list of the Join/Prune message:

1. If an address, Sj, in the join list of the Join/Prune message has the RPT-bit and WC-bit set, then Sj is the RP address used by the downstream router(s) and the following actions are taken:

   (a) If Sj is not the same as the receiving router's RP mapping for G, the receiving router may ignore the Join/Prune message with respect to that group entry. If the router does not have any RP-Set information, it *may* use the address Sj included in the Join/Prune message as the RP for the group.

   (b) If Sj is the same as the receiving router's RP mapping for G, the receiving router adds I to the outgoing interface list of the (*,G) route entry (if there is no (*,G) entry, the router creates one first) and sets the Oif-timer for that interface to the Holdtime specified in the Join/Prune message. In addition, the Oif-Deletion-Delay for that interface is set to 1/3rd the Holdtime specified in the Join/Prune message. If a (*,*,RP) entry exists, for the RP associated with G, then the oif list of the newly created (*,G) entry is copied from that (*,*,RP) entry.

(c) For each (Si,G) entry associated with group G: i) if Si is not included in the prune list, ii) if I is not on the same subnet as the address Si, and iii) if I is not the iif, then interface I is added to the *oif* list and the Oif-timer for that interface in each affected entry is increased (never decreased) to the Holdtime included in the Join/Prune message. In addition, if the Oif-timer for that interface is increased, the Oif-Deletion-Delay for that interface is set to 1/3rd the Holdtime specified in the Join/Prune message.

If the group address in the Join/Prune message is '*' then every (*,G) and (S,G) entry, whose group address hashes to the RP indicated in the (*,*,RP) Join/Prune message, is updated accordingly. A '*' in the group field of the Join/Prune is represented by a group address 224.0.0.0 and a group mask length of 4, indicating a (*,*,RP) Join.

(d) If the (Si,G) entry has its RPT-bit flag set to 1, and its *oif* list is the same as the (*,G) *oif* list, then the (Si,G)RPT-bit entry is deleted,

(e) The incoming interface is set to the interface used to send unicast packets to the RP in the (*,G) route entry, i.e., RPF interface toward the RP.

2. For each address, Sj, in the join list whose RPT-bit and WC-bit are *not* set, and for which there is no existing (Sj,G) route entry, the router initiates one. The router creates a (S,G) entry and copies all outgoing interfaces from the (S,G)RPT-bit entry, if it exists. If there is no (S,G) entry, the oif list is copied from the (*,G) entry; and if there is no (*,G) entry, the oif list is copied from the (*,*,RP) entry, if it exists. In all cases, the iif of the (S,G) entry is always excluded from the oif list.

   (a) The outgoing interface for (Sj,G) is set to I. The incoming interface for (Sj,G) is set to the interface used to send unicast packets to Sj (i.e., the RPF neighbor).

   (b) If the interface used to reach Sj, is the same as I, this represents an error (or a unicast routing change) and the Join/Prune must not be processed.

3. For each address, Sj, in the join list of the Join/Prune message, for which there is an existing (Sj,G) route entry,

   (a) If the RPT-bit is not set for Sj listed in the Join/Prune message, but the RPT-bit flag is set on the existing (Sj,G) entry, the router clears the RPT-bit flag on the (Sj,G) entry, sets the incoming interface to point towards Sj for that (Sj,G) entry, and sends a Join/Prune message corresponding to that entry through the new incoming interface; and

   (b) If I is not the same as the existing incoming interface, the router adds I to the list of outgoing interfaces.

   (c) The Oif-timer for I is increased (never decreased) to the Holdtime included in the Join/Prune message. In addition, if the Oif-timer for that interface is increased, the Oif-Deletion-Delay for that interface is set to 1/3rd the Holdtime specified in the Join/Prune message.

   (d) The (Sj,G) entry's SPT bit is cleared until data comes down the shortest path tree.

For each group address G, in the Join/Prune message, the associated prune list is processed as follows. We refer to each address in the prune list as Sp; Sp refers to the RP if the RPT-bit and WC-bit are both set. For each Sp in the prune list of the Join/Prune message:

1. For each address, Sp, in the prune list whose RPT-bit and WC-bit are cleared:

(a) If there is an existing (Sp,G) route entry, the router lowers the entry's Oif-timer for I to its Oif-Deletion-Delay, allowing for other downstream routers on a multi-access LAN to override the prune. However, on point-to-point links, the oif-timer is expired immediately.

(b) If the router has a current (*,G), or (*,*,RP), route entry, and if the existing (Sp,G) entry has its RPT-bit flag set to 1, then this (Sp,G)RPT-bit entry is maintained (not deleted) even if its outgoing interface list is null.

2. For each address, Sp, in the prune list whose RPT-bit is set and whose WC-bit cleared:

   (a) If there is an existing (Sp,G) route entry, the router lowers the entry's Oif-timer for I to its Oif-Deletion-Delay, allowing for other downstream routers on a multi-access LAN to override the prune. However, on point-to-point links, the oif-timer is expired immediately.

   (b) If the router has a current (*,G), or (*,*,RP), route entry, and if the existing (Sp,G) entry has its RPT-bit flag set to 1, then this (Sp,G)RPT-bit entry is not deleted, and the Entry-timer is restarted, even if its outgoing interface list is null.

   (c) If (*,G), or corresponding (*,*,RP), state exists, but there is no (Sp,G) entry, an (Sp,G)RPT-bit entry is created . The outgoing interface list is copied from the (*,G), or (*,*,RP), entry, with the interface, I, on which the prune was received, is deleted. Packets from the pruned source, Sp, match on this state and are not forwarded toward the pruned receivers.

   (d) If there exists a (Sp,G) entry, with or without the RPT-bit set, the oif-timer for I is expired, and the Entry-timer is restarted.

3. For each address, Sp, in the prune list whose RPT-bit and WC-bit are both set:

   (a) If there is an existing (*,G) entry, with Sp as the RP for G, the router lowers the entry's Oif-timer for I to its Oif-Deletion-Delay, allowing for other downstream routers on a multi-access LAN to override the prune. However, on point-to-point links, the oif-timer is expired immediately.

   (b) If the corresponding (*,*,RP) state exists, but there is no (*,G) entry, a (*,G) entry is created. The outgoing interface list is copied from (*,*,RP) entry, with the interface, I, on which the prune was received, deleted.

For any new (S,G), (*,G) or (*,*,RP) entry created by an incoming Join/Prune message, the SPT-bit is cleared (and if a Join/Prune-Suppression timer is used, it is left off.)

If the entry has a Join/Prune-Suppression timer associated with it, and if the received Join/Prune does not indicate the router as its target, then the receiving router examines the join and prune lists to see if any addresses in the list 'completely-match' existing (S,G), (*,G), or (*,*,RP) state for which the receiving router currently schedules Join/Prune messages. An element on the join or prune list 'completely-matches' a route entry only if both the addresses and RPT-bit flag are the same. If the incoming Join/Prune message completely matches an existing (S,G), (*,G), or (*,*,RP) entry and the Join/Prune arrived on the *iif* for that entry, then the router compares the Holdtime included in the Join/Prune message, to its own [Join/Prune-Holdtime]. If its own [Join/Prune-Holdtime] is lower, the Join/Prune-Suppression-timer is started at the [Join/Prune-Suppression-Timeout]. If the [Join/Prune-Holdtime] is equal, the tie is resolved in favor of the Join/Prune Message originator that has the higher network layer address. When the Join/Prune timer expires, the router triggers a Join/Prune message for the corresponding entry(ies).

3.3 Register and Register-Stop

When a source first starts sending to a group its packets are encapsulated in Register messages and sent to the RP. If the data rate warrants source-specific paths, the RP sets up source specific state and starts sending (S,G) Join/Prune messages toward the source, with S in the join list.

3.3.1 Sending Registers and Receiving Register-Stops

Register messages are sent as follows:

1. When a DR receives a packet from a directly connected source, S, on the subnet containing the address S,

   (a) If there is no corresponding (S,G) entry, and the router has RP-Set information, and the DR is not the RP for G, the DR creates an (S,G) entry with the Register-Suppression-timer turned off and the RP address set according to the hash function mapping for the corresponding group. The oif list is copied from existing (*,G) or (*,*,RP) entries, if they exist. The iif of the (S,G) entry is always excluded from the oif list. If there exists a (*,G) or (*,*,RP) entry, the DR sends a Join/Prune message towards the RP with S in the prune list and the RPT-bit set.

   (b) If there is a (S,G) entry in existence, the DR simply restarts the corresponding Entry-timer.

   When a PMBR (e.g., a router that connects the PIM-SM region to a dense mode region running DVMRP or PIM-DM) receives a packet from a source in the dense mode region, the router treats the packet as if it were from a directly connected source. A separate document will describe the details of interoperability.

2. If the new or previously-existing (S,G) entry's Register-Suppression-timer is not running, the data packet is encapsulated in a Register message and unicast to the RP for that group. The data packet is also forwarded according to (S,G) state in the DR if the oif list is not null; since a receiver may join the SP-tree while the DR is still registering to the RP.

3. If the (S,G) entry's Register-Suppression-timer is running, the data packet is not sent in a Register message, it is just forwarded according to the (S,G) oif list.

When the DR receives a Register-Stop message, it restarts the Register-Suppression-timer in the corresponding (S,G) entry(ies) at [Register-Suppression-Timeout] seconds. If there is data to be registered, the DR may send a null Register (a Register message with a zero-length data portion in the inner packet) to the RP, [Probe-Time] seconds before the Register-Suppression-timer expires, to avoid sending occasional bursts of traffic to an RP unnecessarily.

3.3.2 Receiving Register Messages and Sending Register-Stops

When a router (i.e., the RP) receives a Register message, the router does the following:

1. Decapsulates the data packet, and checks for a corresponding (S,G) entry.

   (a) If a (S,G) entry with cleared (0) SPT bit exists, and the received Register does not have the Null-Register-Bit set to 1, the packet is forwarded; and the SPT bit is left cleared (0). If the SPT bit is 1, the packet is dropped, and Register-Stop messages are triggered. Register-Stops should be rate-limited (in an implementation-specific manner) so that no more than a few are sent per round trip time. This prevents a high datarate stream of packets from triggering a large number of Register-Stop messages between the time that the first packet is received and the time when the source receives the first Register-Stop.

(b) If there is no (S,G) entry, but there is a (*,G) entry, and the received Register does not have the Null-Register-Bit set to 1, the packet is forwarded according to the (*,G) entry.

(c) If there is a (*,*,RP) entry but no (*,G) entry, and the Register received does not have the Null-Register-Bit set to 1, a (*,G) or (S,G) entry is created and the oif list is copied from the (*,*,RP) entry to the new entry. The packet is forwarded according to the created entry.

(d) If there is no G or (*,*,RP) entry corresponding to G, the packet is dropped, and a Register-Stop is triggered.

(e) A "Border bit" bit is added to the Register message, to facilitate interoperability mechanisms. PMBRs set this bit when registering for external sources (see Section 2.7). If the "Border bit" is set in the Register, the RP does the following:

   i. If there is no matching (S,G) state, but there exists (*,G) or (*,*,RP) entry, the RP creates a (S,G) entry, with a 'PMBR' field. This field holds the source of the Register (i.e. the outer network layer address of the register packet). The RP triggers a (S,G) join towards the source of the data packet, and clears the SPT bit for the (S,G) entry. If the received Register is not a 'null Register' the packet is forwarded according to the created state. Else,

   ii. If the 'PMBR' field for the corresponding (S,G) entry matches the source of the Register packet, and the received Register is not a 'null Register', the decapsulated packet is forwarded to the oif list of that entry. Else,

   iii. If the 'PMBR' field for the corresponding (S,G) entry matches the source of the Register packet, the decapsulated packet is forwarded to the oif list of that entry, else

   iv. The packet is dropped, and a Register-stop is triggered towards the source of the Register.

The (S,G) Entry-timer is restarted by Registers arriving from that source to that group.

2. If the matching (S,G) or (*,G) state contains a null oif list, the RP unicasts a Register-Stop message to the source of the Register message; in the latter case, the source-address field, within the Register-Stop message, is set to the wildcard value (all 0's). This message is not processed by intermediate routers, hence no (S,G) state is constructed between the RP and the source.

3. If the Register message arrival rate warrants it and there is no existing (S,G) entry, the RP sets up a (S,G) route entry with the outgoing interface list, excluding iif(S,G), copied from the (*,G) outgoing interface list, its SPT-bit is initialized to 0. If a (*,G) entry does not exist, but there exists a (*,*,RP) entry with the RP corresponding to G , the oif list for (S,G) is copied -excluding the iif- from that (*,*,RP) entry.

   A timer (Entry-timer) is set for the (S,G) entry and this timer is restarted by receipt of data packets for (S,G). The (S,G) entry causes the RP to send a Join/Prune message for the indicated group towards the source of the register message.

   If the (S,G) oif list becomes null, Join/Prune messages will not be sent towards the source, S.

3.4 Multicast Data Packet Forwarding

Processing a multicast data packet involves the following steps:

1. Lookup route state based on a longest match of the source address, and an exact match of the destination address in the data packet. If neither S, nor G, find a longest match entry, and the RP for the packet's destination group address has a corresponding (*,*,RP) entry, then the longest match does not require an exact match on the destination group address. In summary, the longest match is performed in the following order: (1) (S,G), (2) (*,G). If neither is matched, then a lookup is performed on (*,*,RP) entries.

2. If the packet arrived on the interface found in the matching-entry's *iif* field, and the *oif* list is not null:

   (a) Forward the packet to the *oif* list for that entry, excluding the subnet containing S, and restart the Entry-timer if the matching entry is (S,G). Optionally, the (S,G) Entry-timer may be restarted by periodic checking of the matching packet count.

   (b) If the entry is a (S,G) entry with a cleared SPT-bit, and a (*,G) or associated (*,*,RP) also exists whose incoming interface is different than that for (S,G), set the SPT-bit for the (S,G) entry and trigger an (S,G) RPT-bit prune towards the RP.

   (c) If the source of the packet is a directly-connected host and the router is the DR on the receiving interface, check the Register-Suppression-timer associated with the (S,G) entry. If it is not running, then the router encapsulates the data packet in a register message and sends it to the RP.

   This covers the common case of a packet arriving on the RPF interface to the source or RP and being forwarded to all joined branches. It also detects when packets arrive on the SP-tree, and triggers their pruning from the RP-tree. If it is the DR for the source, it sends data packets encapsulated in Registers to the RPs.

3. If the packet matches to an entry but did not arrive on the interface found in the entry's *iif* field, check the SPT-bit of the entry. If the SPT-bit is set, drop the packet. If the SPT-bit is cleared, then lookup the (*,G), or (*,*,RP), entry for G. If the packet arrived on the *iif* found in (*,G), or the corresponding (*,*,RP), forward the packet to the *oif* list of the matching entry. This covers the case when a data packet matches on a (S,G) entry for which the SP-tree has not yet been completely established upstream.

4. If the packet does not match any entry, but the source of the data packet is a local, directly-connected host, and the router is the DR on a multi-access LAN and has RP-Set information, the DR uses the hash function to determine the RP associated with the destination group, G. The DR creates a (S,G) entry, with the Register-Suppression-timer not running, encapsulates the data packet in a Register message and unicasts it to the RP.

5. If the packet does not match to any entry, and it is not a local host or the router is not the DR, drop the packet.

3.4.1 Data triggered switch to shortest path tree (SP-tree)

Different criteria can be applied to trigger switching over from the RP-based shared tree to source-specific, shortest path trees.

One proposed example is to do so based on data rate. For example, when a (*,G), or corresponding (*,*,RP), entry is created, a data rate counter may be initiated at the last-hop routers. The counter is incremented with every data packet received for directly connected members of an SM group, if the longest match is (*,G) or (*,*,RP). If and when the data rate for the group exceeds a certain configured threshold (t1), the router initiates 'source-specific' data rate counters for the following data packets. Then, each counter for a source, is incremented when packets matching on (*,G), or (*,*,RP), are received from that source. If the data rate from the particular source exceeds a configured threshold (t2), a (S,G) entry is created and a Join/Prune message is sent towards the source. If the RPF interface for (S,G) is *not* the same as that for (*,G) -or (*,*,RP), then the SPT-bit is cleared in the (S,G) entry.

Other configured rules may be enforced to cause or prevent establishment of (S,G) state.

3.5 Assert

Asserts are used to resolve which of the parallel routers connected to a multi-access LAN is responsible for forwarding packets onto the LAN.

3.5.1 Sending Asserts

The following Assert rules are provided when a multicast packet is received on an outgoing multi-access interface "I" of an existing active (S,G), (*,G) or (*,*,RP) entry:

1. Do unicast routing table lookup on source address from data packet, and send assert on interface "I" for source address in data packet; include metric preference of routing protocol and metric from routing table lookup.

2. If route is not found, use metric preference of 0x7fffffff and metric 0xffffffff.

When an assert is sent for a (*,G) entry, the first bit in the metric preference (the RPT-bit) is set to 1, indicating the data packet is routed down the RP-tree.

Asserts should be rate-limited in an implementation-specific manner.

3.5.2 Receiving Asserts

When an Assert is received the router performs a longest match on the source and group address in the Assert message, only active entries – that have packet forwarding state – are matched. The router checks the first bit of the metric preference (RPT-bit).

1. If the RPT-bit is set, the router first does a match on (*,G), or (*,*,RP), entries; if no matching entry is found, it ignores the Assert.

2. If the RPT-bit is not set in the Assert, the router first does a match on (S,G) entries; if no matching entry is found, the router matches (*,G) or (*,*,RP) entries.

3.5.2.1 Receiving Asserts on an entry's outgoing interface If the interface that received the Assert message is in the *oif* list of the matched entry, then this Assert is processed by this router as follows:

1. If the Assert's RPT-bit is set and the matching entry is (*,*,RP), the router creates a (*,G) entry. If the Assert's RPT-bit is cleared and the matching entry is (*,G), or (*,*,RP), the router creates a (S,G)RPT-bit entry. Otherwise, no new entry is created in response to the Assert.

2. The router then compares the metric values received in the Assert with the metric values associated with the matched entry. The RPT-bit and metric preference (in that order) are treated as the high-order part of an Assert metric comparison. If the value in the Assert is less than the router's value (with ties broken by the IP address, where higher network layer address wins), delete the interface

from the entry. When the deletion occurs for a (*,G) or (*,*,RP) entry , the interface is also deleted from any associated (S,G)RPT-bit or (*,G) entries, respectively. The Entry-timer for the affected entries is restarted.

3. If the router has won the election the router keeps the interface in its outgoing interface list. It acts as the forwarder for the LAN.

The winning router sends an Assert message containing its own metric to that outgoing interface. This will cause other routers on the LAN to prune that interface from their route entries. The winning router sets the RPT-bit in the Assert message if a (*,G) or (S,G)RPT-bit entry was matched.

3.5.2.2 Receiving Asserts on an entry's incoming interface If the Assert arrived on the incoming interface of an existing (S,G), (*,G), or (*,*,RP) entry, the Assert is processed as follows. If the Assert message does not match the entry, *exactly*, it is ignored; i.e, longest-match is not used in this case. If the Assert message does match exactly, then:

1. Downstream routers will select the upstream router with the smallest metric preference and metric as their RPF neighbor. If two metrics are the same, the highest network layer address is chosen to break the tie. This is important so that downstream routers send subsequent Joins/Prunes (in SM) to the correct neighbor. An Assert-timer is initiated when changing the RPF neighbor to the Assert winner. When the timer expires, the router resets its RPF neighbor according to its unicast routing tables to capture network dynamics and router failures.

2. If the downstream routers have downstream members, and if the Assert caused the RPF neighbor to change, the downstream routers must trigger a Join/Prune message to inform the upstream router that packets are to be forwarded on the multi-access network.

3.6 Candidate-RP-Advertisements and Bootstrap messages

Candidate-RP-Advertisements (C-RP-Advs) are periodic PIM messages unicast to the BSR by those routers that are configured as Candidate-RPs (C-RPs).

Bootstrap messages are periodic PIM messages originated by the Bootstrap router (BSR) within a domain, and forwarded hop-by-hop to distribute the current RP-set to all routers in that domain.

The Bootstrap messages also support a simple mechanism by which the Candidate BSR (C-BSR) with the highest BSR-priority and address (referred to as the preferred BSR) is elected as the BSR for the domain. We recommend that each router configured as a C-RP also be configured as a C-BSR. Sections 3.6.2 and 3.6.3 describe the combined function of Bootstrap messages as the vehicle for BSR election and RP-Set distribution.

A Finite State Machine description of the BSR election and RP-Set distribution mechanisms is included in Appendix II.

3.6.1 Sending Candidate-RP-Advertisements

C-RPs periodically unicast C-RP-Advs to the BSR for that domain. The interval for sending these messages is subject to local configuration at the C-RP.

Candidate-RP-Advertisements carry group address and group mask fields. This enables the advertising router to limit the advertisement to certain prefixes or scopes of groups. The advertising router may enforce this scope acceptance when receiving Registers or Join/Prune messages. C-RPs should send C-RP-Adv messages with the 'Priority' field set to '0'.

3.6.2 Receiving C-RP-Advs and Originating Bootstrap

Upon receiving a C-RP-Adv, a router does the following:

1. If the router is not the elected BSR, it ignores the message, else

2. The BSR adds the RP address to its local pool of candidate RPs, according to the associated group prefix(es) in the C-RP-Adv message. The Holdtime in the C-RP-Adv message is also stored with the corresponding RP, to be included later in the Bootstrap message. The BSR may apply a local policy to limit the number of Candidate RPs included in the Bootstrap message. The BSR may override the prefix indicated in a C-RP-Adv unless the 'Priority' field is not zero.

The BSR keeps an RP-timer per RP in its local RP-set. The RP-timer is initialized to the Holdtime in the RP's C-RP-Adv. When the timer expires, the corresponding RP is removed from the RP-set. The RP-timer is restarted by the C-RP-Advs from the corresponding RP.

The BSR also uses its Bootstrap-timer to periodically send Bootstrap messages. In particular, when the Bootstrap-timer expires, the BSR originates a Bootstrap message on each of its PIM interfaces. To reduce the bootstrap message overhead during partition healing, the BSR should set a random time (as a function of the priority and address) after which the Bootstrap message is originated only if no other preferred Bootstrap message is received. For details see appendix 6.2. The message is sent with a TTL of 1 to the 'ALL-PIM-ROUTERS' group. In steady state, the BSR originates Bootstrap messages periodically. At startup, the Bootstrap-timer is initialized to [Bootstrap-Timeout], causing the first Bootstrap message to be originated only when and if the timer expires. For timer details, see Section 3.6.3.

A DR unicasts a Bootstrap message to each new PIM neighbor, i.e., after the DR receives the neighbor's Hello message (it does so even if the new neighbor becomes the DR).

The Bootstrap message is subdivided into sets of {group-prefix,RP-Count,RP-addresses}. For each RP-address, the corresponding Holdtime is included in the "RP-Holdtime" field. The format of the Bootstrap message allows 'semantic fragmentation', if the length of the original Bootstrap message exceeds the packet maximum boundaries (see Section 4). However, we recommend against configuring a large number of routers as C-RPs, to reduce the semantic fragmentation required.

3.6.3 Receiving and Forwarding Bootstrap

Each router keeps a Bootstrap-timer, initialized to [Bootstrap-Timeout] at startup.

When a router receives Bootstrap message sent to 'ALL-PIM-ROUTERS' group, it performs the following:

1. If the message was not sent by the RPF neighbor towards the BSR address included, the message is dropped. Else

2. If the included BSR is **not** preferred over, and not equal to, the currently active BSR:

   (a) If the Bootstrap-timer has **not** yet expired, or if the receiving router is a C-BSR, then the Bootstrap message is dropped. Else

   (b) If the Bootstrap-timer **has** expired and the receiving router is not a C-BSR, the receiving router stores the RP-Set and BSR address and priority found in the message, and restarts the timer by setting it to [Bootstrap-Timeout]. The Bootstrap message is then forwarded out all PIM interfaces, excluding the one over which the message arrived, to 'ALL-PIM-ROUTERS' group, with a TTL of 1.

3. If the Bootstrap message includes a BSR address that **is** preferred over, or equal to, the currently active BSR, the router restarts its Bootstrap-timer at [Bootstrap-Timeout] seconds. and stores the BSR address and RP-Set information.

   The Bootstrap message is then forwarded out all PIM interfaces, excluding the one over which the message arrived, to 'ALL-PIM-ROUTERS' group, with a TTL of 1.

4. If the receiving router has no current RP set information and the Bootstrap was unicast to it from a directly connected neighbor, the router stores the information as its new RP-set. This covers the startup condition when a newly booted router obtains the RP-Set and BSR address from its DR.

When a router receives a new RP-Set, it checks if each of the RPs referred to by existing state (i.e., by (*,G), (*,*,RP), or (S,G)RPT-bit entries) is in the new RP-Set. If an RP is not in the new RP-set, that RP is considered unreachable and the hash algorithm (see below) is re-performed for each group with locally active state that previously hashed to that RP. This will cause those groups to be distributed among the remaining RPs. When the new RP-Set contains a new RP, the value of the new RP is calculated for each group covered by that C-RP's Group-prefix. Any group for which the new RP's value is greater than the previously active RP's value is switched over to the new RP.

3.7 Hash Function

The hash function is used by all routers within a domain, to map a group to one of the C-RPs from the RP-Set. For a particular group, G, the hash function uses only those C-RPs whose Group-prefix covers G. The algorithm takes as input the group address, and the addresses of the Candidate RPs, and gives as output one RP address to be used.

The protocol requires that all routers hash to the same RP within a domain (except for transients). The following hash function must be used in each router:

1. For RP addresses in the RP-Set, whose Group-prefix covers G, select the RPs with the highest priority (i.e. lowest 'Priority' value), and compute a value:

   $$Value(G, M, C_i) = (1103515245 \cdot ((1103515245 \cdot (G\&M) + 12345) \text{ XOR } C_i) + 12345) \quad \bmod 2^{31}$$

   where C_i is the RP address and M is a hash-mask included in Bootstrap messages. The hash-mask allows a small number of consecutive groups (e.g., 4) to always hash to the same RP. For instance, hierarchically-encoded data can be sent on consecutive group addresses to get the same delay and fate-sharing characteristics.

   For address families other than IPv4, a 32-bit digest to be used as C_i must first be derived from the actual RP address. Such a digest method must be used consistently throughout the PIM domain. For IPv6 addresses, we recommend using the equivalent IPv4 address for an IPv4-compatible address, and the CRC-32 checksum [7] of all other IPv6 addresses.

2. From the RPs with the highest priority (i.e. lowest 'Priority' value), the candidate with the highest resulting value is then chosen as the RP for that group, and its identity and hash value are stored with the entry created.

   Ties between RPs having the same hash value and priority, are broken in advantage of the highest address.

The hash function algorithm is invoked by a DR, upon reception of a packet, or IGMP membership indication, for a group, for which the DR has no entry. It is invoked by any router that has (*,*,RP) state when a packet is received for which there is no corresponding (S,G) or (*,G) entry. Furthermore, the hash function is invoked by all routers upon receiving a (*,G) or (*,*,RP) Join/Prune message.

3.8 Processing Timer Events

In this subsection, we enumerate all timers that have been discussed or implied. Since some critical timer events are not associated with the receipt or sending of messages, they are not fully covered by earlier subsections.

Timers are implemented in an implementation-specific manner. For example, a timer may count up or down, or may simply expire at a specific time. Setting a timer to a value T means that it will expire after T seconds.

3.8.1 Timers related to tree maintenance

Each (S,G), (*,G), and (*,*,RP) route entry has multiple timers associated with it: one for each interface in the outgoing interface list, one for the multicast routing entry itself, and one optional Join/Prune-Suppression-Timer. Each (S,G) and (*,G) entry also has an Assert-timer and a Random-Delay-Join-Timer for use with Asserts. In addition, DR's have a Register-Suppression-timer for each (S,G) entry and every router has a single Join/Prune-timer. (A router may optionally keep separate Join/Prune-timers for different interfaces or route entries if different Join/Prune periods are desired.)

Join/Prune-Timer This timer is used for periodically sending aggregate Join/Prune messages. To avoid synchronization among routers booting simultaneously, it is initially set to a random value between 1 and [Join/Prune-Period]. When it expires, the timer is immediately restarted to [Join/Prune-Period]. A Join/Prune message is then sent out each interface. This timer should not be restarted by other events.

Join/Prune-Suppression-Timer (kept per route entry) A route entry's (optional) Join/Prune-Suppression-Timer may be used to suppress duplicate joins from multiple downstream routers on the same LAN. When a Join message is received from a neighbor on the entry's incoming interface in which the included Holdtime is higher than the router's own [Join/Prune-Holdtime] (with ties broken by higher network layer address), the timer is set to [Join/Prune-Suppression-Timeout], with some random jitter introduced to avoid synchronization of triggered Join/Prune messages on expiration. (The random timeout value must be $<$ 1.5 * [Join/Prune-Period] to prevent losing data after 2 dropped Join/Prunes.) The timer is restarted every time a subsequent Join/Prune message (with higher Holdtime/IP address) for the entry is received on its incoming interface. While the timer is running, Join/Prune messages for the entry are not sent. This timer is idle (not running) for point-to-point links.

Oif-Timer (kept per oif for each route entry) A timer for each oif of a route entry is used to time out that oif. Because some of the outgoing interfaces in an (S,G) entry are copied from the (*,G) outgoing interface list, they may not have explicit (S,G) join messages from some of the downstream routers (i.e., where members are joining to the (*,G) tree only). Thus, when an Oif-timer is restarted in a (*,G) entry, the Oif-timer is restarted for that interface in each existing (S,G) entry whose oif list contains that interface. The same rule applies to (*,G) and (S,G) entries when restarting an Oif-timer on a (*,*,RP) entry.

The following table shows its usage when first adding the oif to the entry's oiflist, when it should be restarted (unless it is already higher), and when it should be decreased (unless it is already lower).

Set to	**When**	**Applies to**
included Holdtime	adding oif off Join/Prune	(S,G) (*,G) (*,*,RP)
Increased (only) to	**When**	**Applies to**
included Holdtime	received Join/Prune	(S,G) (*,G) (*,*,RP)
(*,*,RP) oif-timer value	(*,*,RP) oif-timer restarted	(S,G) (*,G)
(*,G) oif-timer value	(*,G) oif-timer restarted	(S,G)
Decreased (only) to	**When**	**Applies to**
Oif-Deletion-Delay	prune received	(S,G) (*,G)

When the timer expires, the oif is removed from the oiflist if there are no directly-connected members. When deleted, the oif is also removed in any associated (S,G) or (*,G) entries.

Entry-Timer (kept per route entry) A timer for each route entry is used to time out that entry. The following table summarizes its usage when first adding the oif to the entry's oiflist, and when it should be restarted (unless it is already higher).

Set to	**When**	**Applies to**		
[Data-Timeout]	created off data packet	(S,G)		
included Holdtime	created off Join/Prune	(S,G)	(*,G)	(*,*,RP)
Increased (only) to	**When**	**Applies to**		
[Data-Timeout]	receiving data packets	(S,G)no RPT-bit		
oif-timer value	any oif-timer restarted	(S,G)RPT-bit	(*,G)	(*,*,RP)
[Assert-Timeout]	assert received	(S,G)RPT-bit	(*,G) w/null oif	

When the timer expires, the route entry is deleted; if the entry is a (*,G) or (*,*,RP) entry, all associated (S,G)RPT-bit entries are also deleted.

Register-Suppression-Timer (kept per (S,G) route entry) An (S,G) route entry's Register-Suppression-Timer is used to suppress registers when the RP is receiving data packets natively. When a Register-Stop message for the entry is received from the RP, the timer is set to a random value in the range 0.5 * [Register-Suppression-Timeout] to 1.5 * [Register-Suppression-Timeout]. While the timer is running, Registers for that entry will be suppressed. If null registers are used, a null register is sent [Probe-Time] seconds before the timer expires.

Assert-Timer (per (S,G) or (*,G) route entry) The Assert-Timer for an (S,G) or (*,G) route entry is used for timing out Asserts received. When an Assert is received and the RPF neighbor is changed to the Assert winner, the Assert-Timer is set to [Assert-Timeout], and is restarted to this value every time a subsequent Assert for the entry is received on its incoming interface. When the timer expires, the router resets its RPF neighbor according to its unicast routing table.

Random-Delay-Join-Timer (per (S,G) or (*,G) route entry) The Random-Delay-Join-Timer for an (S,G) or (*,G) route entry is used to prevent synchronization among downstream routers on a LAN when their RPF neighbor changes. When the RPF neighbor changes, this timer is set to a random value between 0 and [Random-Delay-Join-Timeout] seconds. When the timer expires, a triggered Join/Prune message is sent for the entry unless its Join/Prune-Suppression-Timer is running.

3.8.2 Timers relating to neighbor discovery

Hello-Timer This timer is used to periodically send Hello messages. To avoid synchronization among routers booting simultaneously, it is initially set to a random value between 1 and [Hello-Period]. When it expires, the timer is immediately restarted to [Hello-Period]. A Hello message is then sent out each interface. This timer should not be restarted by other events.

Neighbor-Timer (kept per neighbor) A Neighbor-Timer for each neighbor is used to time out the neighbor state. When a Hello message is received from a new neighbor, the timer is initially set to the Holdtime included in the Hello message (which is equal to the neighbor's value of [Hello-Holdtime]). Every time a subsequent Hello is received from that neighbor, the timer is restarted to the Holdtime in the Hello. When the timer expires, the neighbor state is removed.

3.8.3 Timers relating to RP information

C-RP-Adv-Timer (C-RP's only) Routers configured as candidate RP's use this timer to periodically send C-RP-Adv messages. To avoid synchronization among routers booting simultaneously, the timer is initially set to a random value between 1 and [C-RP-Adv-Period]. When it expires, the timer is immediately restarted to [C-RP-Adv-Period]. A C-RP-Adv message is then sent to the elected BSR. This timer should not be restarted by other events.

RP-Timer (BSR only, kept per RP in RP-Set) The BSR uses a timer per RP in the RP-Set to monitor liveness. When a C-RP is added to the RP-Set, its timer is set to the Holdtime included in the C-RP-Adv message from that C-RP (which is equal to the C-RP's value of [RP-Holdtime]). Every time a subsequent C-RP-Adv is received from that RP, its timer is restarted to the Holdtime in the C-RP-Adv. When the timer expires, the RP is removed from the RP-Set included in Bootstrap messages.

Bootstrap-Timer This timer is used by the BSR to periodically originate Bootstrap messages, and by other routers to time out the BSR (see 3.6.3). This timer is initially set to [Bootstrap-Timeout]. A C-BSR restarts this timer to [Bootstrap-Timeout] upon receiving a Bootstrap message from a preferred router, and originates a Bootstrap message and restarts the timer to [Bootstrap-Period] when it expires. Routers not configured as C-BSR's restart this timer to [Bootstrap-Timeout] upon receiving a Bootstrap message from the elected or a more preferred BSR, and ignore Bootstrap messages from non-preferred C-BSRs while it is running.

3.8.4 Default timer values

Most of the default timeout values for state information are 3.5 times the refresh period. For example, Hellos refresh Neighbor state and the default Hello-timer period is 30 seconds, so a default Neighbor-timer duration of 105 seconds is included in the Holdtime field of the Hellos. In order to improve convergence, however, the default timeout value for information related to RP liveness and Bootstrap messages is 2.5 times the refresh period.

In this version of the spec, we suggest particular numerical timer settings. A future version of the specification will specify a mechanism for timer values to be scaled based upon observed network parameters.

[Join/Prune-Period] This is the interval between sending Join/Prune messages. **Default: 60 seconds.** This value may be set to take into account such things as the configured bandwidth and expected average number of multicast route entries for the attached network or link (e.g., the period would be longer for lower-speed links, or for routers in the center of the network that expect to have a larger number of entries). In addition, a router could modify this value (and corresponding Join/Prune-Holdtime value) if the number of route entries changes significantly (e.g., by an order of magnitude). For example, given a default minimum Join/Prune-Period value, if the number of route entries with a particular iif increases from N to N*100, the router could increase its Join/Prune-Period (and Join/Prune-Holdtime), for that interface, by a factor of 10; and if/when the number of entries decreases back to N, the Join/Prune-Period (and Join/Prune-Holdtime) could be decreased to its previous value. If the Join/Prune-Period is modified, these changes should be made relatively infrequently and the router should continue to refresh at its previous Join/Prune-Period for at least Join/Prune-Holdtime, in order to allow the upstream router to adapt.

[**Join-Prune Holdtime**] This is the Holdtime specified in Join/Prune messages, and is used to time out oifs. This should be set to 3.5 * [Join/Prune-Period]. **Default: 210 seconds.**

[**Join/Prune-Suppression-Timeout**] This is the mean interval between receiving a Join/Prune with a higher Holdtime (with ties broken by higher network layer address) and allowing duplicate Join/Prunes to be sent again. This should be set to approximately 1.25 * [Join/Prune-Period]. **Default: 75 seconds.**

[**Data-Timeout**] This is the time after which (S,G) state for a silent source will be deleted. **Default: 210 seconds.**

[**Register-Suppression-Timeout**] This is the mean interval between receiving a Register-Stop and allowing Registers to be sent again. A lower value means more frequent register bursts at RP, while a higher value means longer join latency for new receivers. **Default: 60 seconds.** (Note that if null Registers are sent [Probe-Time] seconds before the timeout, register bursts are prevents, and [Register-Suppression-Timeout] may be lowered to decrease join latency.)

[**Probe-Time**] When null Registers are used, this is the time between sending a null Register and the Register-Suppression-Timer expiring unless it is restarted by receiving a Register-Stop. Thus, a null Register would be sent when the Register-Suppression-Timer reaches this value. **Default: 5 seconds.**

[**Assert-Timeout**] This is the interval between the last time an Assert is received, and the time at which the assert is timed out. **Default: 180 seconds.**

[**Random-Delay-Join-Timeout**] This is the maximum interval between the time when the RPF neighbor changes, and the time at which a triggered Join/Prune message is sent. **Default: 4.5 seconds.**

[**Hello-Period**] This is the interval between sending Hello messages. **Default: 30 seconds.**

[**Hello-Holdtime**] This is the Holdtime specified in Hello messages, after which neighbors will time out their neighbor entries for the router. This should be set to 3.5 * [Hello-Period]. **Default: 105 seconds.**

[**C-RP-Adv-Period**] For C-RPs, this is the interval between sending C-RP-Adv messages. **Default: 60 seconds.**

[**RP-Holdtime**] For C-RPs, this is the Holdtime specified in C-RP-Adv messages, and is used by the BSR to time out RPs. This should be set to 2.5 * [C-RP-Adv-Period]. **Default: 150 seconds.**

[**Bootstrap-Period**] At the elected BSR, this is the interval between originating Bootstrap messages, and should be equal to 60 seconds.

[**Bootstrap-Timeout**] This is the time after which the elected BSR will be assumed unreachable when Bootstrap messages are not received from it. This should be set to '2 * [Bootstrap-Period] + 10'. **Default: 130 seconds.**

3.9 Summary of flags used

Following is a summary of all the flags used in our scheme.

Bit	Used in	Definition
Border	Register	Register for external sources is coming from a PIM multicast border router
Null	Register	Register sent as Probe of RP, the encapsulated data packet should not be forwarded
RPT	Route entry	Entry represents state on the RP-tree
RPT	Join/Prune	Join is associated with the shared tree and therefore the Join/Prune message is propagated along the RP-tree (source encoded is an RP address)
RPT	Assert	The data packet was routed down the shared tree; thus, the path indicated corresponds to the RP tree
SPT	(S,G) entry	Packets have arrived on the iif towards S, and the iif is different from the (*,G) iif
WC	Join	The receiver expects to receive packets from all sources via this (shared tree) path. Thus, the Join/Prune applies to a (*,G) entry
WC	Route entry	Wildcard entry; if there is no more specific match for a particular source, packets will be forwarded according to this entry

3.10 Security

All PIM control messages may use IPsec [6] to address security concerns.

4 Packet Formats

This section describes the details of the packet formats for PIM control messages.

All PIM control messages have protocol number 103.

Basically, PIM messages are either unicast (e.g. Registers and Register-Stop), or multicast hop-by-hop to 'ALL-PIM-ROUTERS' group '224.0.0.13' (e.g. Join/Prune, Asserts, etc.).

```
 0                   1                   2                   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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PIM Ver| Type  | Reserved      |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

PIM Ver PIM Version number is 2.

Type Types for specific PIM messages. PIM Types are:

0 = Hello
1 = Register
2 = Register-Stop
3 = Join/Prune
4 = Bootstrap
5 = Assert
6 = Graft (used in PIM-DM only)
7 = Graft-Ack (used in PIM-DM only)
8 = Candidate-RP-Advertisement

Reserved set to zero. Ignored upon receipt.

Checksum The checksum is the 16-bit one's complement of the one's complement sum of the entire PIM message, (excluding the data portion in the Register message). For computing the checksum, the checksum field is zeroed.

4.1 Encoded Source and Group Address formats

1. Encoded-Unicast-address: Takes the following format:

```
 0                   1                   2                   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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Addr Family   | Encoding Type |     Unicast Address           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+~+~+~+~+~+~+
```

Addr Family The address family of the 'Unicast Address' field of this address. Here is the address family numbers assigned by IANA:

```
Number    Description
- ------    ---------------------------------------------------------
     0    Reserved
     1    IP (IP version 4)
     2    IP6 (IP version 6)
     3    NSAP
     4    HDLC (8-bit multidrop)
     5    BBN 1822
     6    802 (includes all 802 media plus Ethernet "canonical format")
     7    E.163
     8    E.164 (SMDS, Frame Relay, ATM)
     9    F.69 (Telex)
    10    X.121 (X.25, Frame Relay)
    11    IPX
    12    Appletalk
    13    Decnet IV
    14    Banyan Vines
    15    E.164 with NSAP format subaddress
```

Encoding Type The type of encoding used within a specific Address Family. The value '0' is reserved for this field, and represents the native encoding of the Address Family.

Unicast Address The unicast address as represented by the given Address Family and Encoding Type.

2. Encoded-Group-Address: Takes the following format:

```
 0                   1                   2                   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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Addr Family   | Encoding Type |   Reserved    |  Mask Len     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Group multicast Address                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Addr Family described above.

Encoding Type described above.

Reserved Transmitted as zero. Ignored upon receipt.

Mask Len The Mask length is 8 bits. The value is the number of contiguous bits left justified used as a mask which describes the address. It is less than or equal to the address length in bits for the given Address Family and Encoding Type. If the message is sent for a single group then the Mask length must equal the address length in bits for the given Address Family and Encoding Type. (e.g. 32 for IPv4 native encoding and 128 for IPv6 native encoding).

Group multicast Address contains the group address.

3. Encoded-Source-Address: Takes the following format:

```
 0                   1                   2                   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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Addr Family   | Encoding Type | Rsrvd   |S|W|R|  Mask Len     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Source Address                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Addr Family described above.

Encoding Type described above.

Reserved Transmitted as zero, ignored on receipt.

S,W,R See Section 4.5 for details.

Mask Length Mask length is 8 bits. The value is the number of contiguous bits left justified used as a mask which describes the address. The mask length must be less than or equal to the address length in bits for the given Address Family and Encoding Type. If the message is sent for a single group then the Mask length must equal the address length in bits for the given Address Family and Encoding Type. In version 2 of PIM, it is strongly recommended that this field be set to 32 for IPv4 native encoding.

Source Address The source address.

4.2 Hello Message

It is sent periodically by routers on all interfaces.

```
 0                   1                   2                   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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PIM Ver| Type  | Reserved      |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       OptionType              |         OptionLength          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          OptionValue                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+~+~+
|                               .                               |
|                               .                               |
|                               .                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       OptionType              |         OptionLength          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          OptionValue                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+~+~+
```

PIM Version, Type, Reserved, Checksum Described above.

OptionType The type of the option given in the following OptionValue field.

OptionLength The length of the OptionValue field in bytes.

OptionValue A variable length field, carrying the value of the option.

The Option fields may contain the following values:

- OptionType = 1; OptionLength = 2; OptionValue = Holdtime; where Holdtime is the amount of time a receiver must keep the neighbor reachable, in seconds. If the Holdtime is set to '0xffff', the receiver of this message never times out the neighbor. This may be used with ISDN lines, to avoid keeping the link up with periodic Hello messages. Furthermore, if the Holdtime is set to '0', the information is timed out immediately.

- OptionType 2 to 16: reserved

- The rest of the OptionTypes are defined in another document.

In general, options may be ignored; but a router must not ignore the 'Holdtime' OptionType."

4.3 Register Message

A Register message is sent by the DR or a PMBR to the RP when a multicast packet needs to be transmitted on the RP-tree. Source address is set to the address of the DR, destination address is to the RP's address.

```
 0                   1                   2                   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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PIM Ver| Type  | Reserved      |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|B|N|                        Reserved                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                       Multicast data packet                   ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

PIM Version, Type, Reserved, Checksum Described above. *Note that the checksum for Registers is done only on the PIM header, excluding the data packet portion.*

B The Border bit. If the router is a DR for a source that it is directly connected to, it sets the B bit to 0. If the router is a PMBR for a source in a directly connected cloud, it sets the B bit to 1.

N The Null-Register bit. Set to 1 by a DR that is probing the RP before expiring its local Register-Suppression timer. Set to 0 otherwise.

Multicast data packet The original packet sent by the source.

For (S,G) null Registers, the Multicast data packet portion contains only a dummy header with S as the source address, G as the destination address, and a data length of zero.

4.4 Register-Stop Message

A Register-Stop is unicast from the RP to the sender of the Register message. Source address is the address to which the register was addressed. Destination address is the source address of the register message.

```
 0                   1                   2                   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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PIM Ver| Type  | Reserved      |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Encoded-Group Address                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Encoded-Unicast-Source Address             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

PIM Version, Type, Reserved, Checksum Described above.

Encoded-Group Address Format described above. Note that for Register-Stops the Mask Len field contains full address length * 8 (e.g. 32 for IPv4 native encoding), if the message is sent for a single group.

Encoded-Unicast-Source Address host address of source from multicast data packet in register. The format for this address is given in the Encoded-Unicast-Address in 4.1. A special wild card value (0's), can be used to indicate any source.

4.5 Join/Prune Message

A Join/Prune message is sent by routers towards upstream sources and RPs. Joins are sent to build shared trees (RP trees) or source trees (SPT). Prunes are sent to prune source trees when members leave groups as well as sources that do not use the shared tree.

```
 0                   1                   2                   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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PIM Ver| Type  | Reserved      |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Encoded-Unicast-Upstream Neighbor Address         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Reserved     | Num groups    |          Holdtime             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Encoded-Multicast Group Address-1                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Number of Joined  Sources   |   Number of Pruned Sources    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Encoded-Joined Source Address-1                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             .                                 |
|                             .                                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Encoded-Joined Source Address-n                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Encoded-Pruned Source Address-1                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             .                                 |
|                             .                                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Encoded-Pruned Source Address-n                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           .                                   |
|                           .                                   |
|                           .                                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Encoded-Multicast Group Address-n              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Number of Joined  Sources   |   Number of Pruned Sources    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Encoded-Joined Source Address-1                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             .                                 |
|                             .                                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Encoded-Joined Source Address-n                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Encoded-Pruned Source Address-1                 |
```

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               .                               |
|                               .                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Encoded-Pruned Source Address-n                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

PIM Version, Type, Reserved, Checksum Described above.

Encoded-Unicast Upstream Neighbor Address The address of the RPF or upstream neighbor. The format for this address is given in the Encoded-Unicast-Address in 4.1.

Reserved Transmitted as zero, ignored on receipt.

Holdtime The amount of time a receiver must keep the Join/Prune state alive, in seconds. If the Holdtime is set to '0xffff', the receiver of this message never times out the oif. This may be used with ISDN lines, to avoid keeping the link up with periodical Join/Prune messages. Furthermore, if the Holdtime is set to '0', the information is timed out immediately.

Number of Groups The number of multicast group sets contained in the message.

Encoded-Multicast group address For format description see Section 4.1. A wild card group in the (*,*,RP) join is represented by a 224.0.0.0 in the group address field and '4' in the mask length field. A (*,*,RP) join also has the WC-bit and the RPT-bit set.

Number of Joined Sources Number of join source addresses listed for a given group.

Join Source Address-1 .. n This list contains the sources that the sending router will forward multicast datagrams for if received on the interface this message is sent on.

See format section 4.1. The fields explanation for the Encoded-Source-Address format follows:

Reserved Described above.

S The Sparse bit is a 1 bit value, set to 1 for PIM-SM. It is used for PIM v.1 compatibility.

W The WC bit is a 1 bit value. If 1, the join or prune applies to the (*,G) or (*,*,RP) entry. If 0, the join or prune applies to the (S,G) entry where S is Source Address. Joins and prunes sent towards the RP must have this bit set.

R The RPT-bit is a 1 bit value. If 1, the information about (S,G) is sent towards the RP. If 0, the information must be sent toward S, where S is the Source Address.

Mask Length, Source Address Described above.

Represented in the form of $< WC - bit >< RPT - bit >< Mask\ length >< Source\ address >$:

A source address could be a host IPv4 native encoding address :

$$< 0 >< 0 >< 32 >< 192.1.1.17 >$$

A source address could be the RP's IP address :

$$< 1 >< 1 >< 32 >< 131.108.13.111 >$$

A source address could be a subnet address to prune from the RP-tree :

$$< 0 >< 1 >< 28 >< 192.1.1.16 >$$

A source address could be a general aggregate :

$$< 0 >< 0 >< 16 >< 192.1.0.0 >$$

Number of Pruned Sources Number of prune source addresses listed for a group.

Prune Source Address-1 .. n This list contains the sources that the sending router does not want to forward multicast datagrams for when received on the interface this message is sent on. If the Join/Prune message boundary exceeds the maximum packet size, then the join and prune lists for the same group must be included in the same packet.

4.6 Bootstrap Message

The Bootstrap messages are multicast to 'ALL-PIM-ROUTERS' group, out all interfaces having PIM neighbors (excluding the one over which the message was received). Bootstrap messages are sent with TTL value of 1. Bootstrap messages originate at the BSR, and are forwarded by intermediate routers.

Bootstrap message is divided up into 'semantic fragments', if the original message exceeds the maximum packet size boundaries.

The semantics of a single 'fragment' is given below:

```
 0                   1                   2                   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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PIM Ver| Type  | Reserved      |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Fragment Tag          | Hash Mask len | BSR-priority  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Encoded-Unicast-BSR-Address                   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Encoded-Group Address-1               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| RP-Count-1    | Frag RP-Cnt-1 |         Reserved              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Encoded-Unicast-RP-Address-1                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          RP1-Holdtime         | RP1-Priority  |   Reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Encoded-Unicast-RP-Address-2                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          RP2-Holdtime         | RP2-Priority  |   Reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               .                               |
|                               .                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Encoded-Unicast-RP-Address-m                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          RPm-Holdtime         | RPm-Priority  |   Reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Encoded-Group Address-2               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               .                               |
|                               .                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Encoded-Group Address-n               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| RP-Count-n    | Frag RP-Cnt-n |          Reserved             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Encoded-Unicast-RP-Address-1                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
|          RP1-Holdtime         | RP1-Priority  |   Reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Encoded-Unicast-RP-Address-2                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          RP2-Holdtime         | RP2-Priority  |   Reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               .                               |
|                               .                               |
|                               .                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Encoded-Unicast-RP-Address-m                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          RPm-Holdtime         | RPm-Priority  |   Reserved    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

PIM Version, Type, Reserved, Checksum Described above.

Fragment Tag A randomly generated number, acts to distinguish the fragments belonging to different Bootstrap messages; fragments belonging to same Bootstrap message carry the same ‘Fragment Tag’.

Hash Mask len The length (in bits) of the mask to use in the hash function. For IPv4 we recommend a value of 30. For IPv6 we recommend a value of 126.

BSR-priority Contains the BSR priority value of the included BSR. This field is considered as a high order byte when comparing BSR addresses.

Encoded-Unicast-BSR-Address The address of the bootstrap router for the domain. The format for this address is given in the Encoded-Unicast-Address in 4.1.

Encoded-Group Address-1..n The group prefix (address and mask) with which the Candidate RPs are associated. Format previously described.

RP-Count-1..n The number of Candidate RP addresses included in the whole Bootstrap message for the corresponding group prefix. A router does *not* replace its old RP-Set for a given group prefix until/unless it receives ‘RP-Count’ addresses for that prefix; the addresses could be carried over several fragments. If only part of the RP-Set for a given group prefix was received, the router discards it, without updating that specific group prefix’s RP-Set.

Frag RP-Cnt-1..m The number of Candidate RP addresses included in this fragment of the Bootstrap message, for the corresponding group prefix. The ‘Frag RP-Cnt’ field facilitates parsing of the RP-Set for a given group prefix, when carried over more than one fragment.

Encoded-Unicast-RP-address-1..m The address of the Candidate RPs, for the corresponding group prefix. The format for this address is given in the Encoded-Unicast-Address in 4.1.

RP1..m-Holdtime The Holdtime for the corresponding RP. This field is copied from the ‘Holdtime’ field of the associated RP stored at the BSR.

RP1..m-Priority The ‘Priority’ of the corresponding RP and Encoded-Group Address. This field is copied from the ‘Priority’ field stored at the BSR when receiving a Candidate-RP-Advertisement. The highest priority is ‘0’ (i.e. the lower the value of the ‘Priority’ field, the higher). Note that the priority is per RP per Encoded-Group Address.

4.7 Assert Message

The Assert message is sent when a multicast data packet is received on an outgoing interface corresponding to the (S,G) or (*,G) associated with the source.

```
 0                   1                   2                   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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PIM Ver| Type  | Reserved      |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Encoded-Group Address                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             Encoded-Unicast-Source Address                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|R|                       Metric Preference                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Metric                                |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

PIM Version, Type, Reserved, Checksum Described above.

Encoded-Group Address The group address to which the data packet was addressed, and which triggered the Assert. Format previously described.

Encoded-Unicast-Source Address Source address from multicast datagram that triggered the Assert packet to be sent. The format for this address is given in the Encoded-Unicast-Address in 4.1.

R RPT-bit is a 1 bit value. If the multicast datagram that triggered the Assert packet is routed down the RP tree, then the RPT-bit is 1; if the multicast datagram is routed down the SPT, it is 0.

Metric Preference Preference value assigned to the unicast routing protocol that provided the route to Host address.

Metric The unicast routing table metric. The metric is in units applicable to the unicast routing protocol used.

4.8 Graft Message

Used in dense-mode. Refer to PIM dense mode specification.

4.9 Graft-Ack Message

Used in dense-mode. Refer to PIM dense mode specification.

4.10 Candidate-RP-Advertisement

Candidate-RP-Advertisements are periodically unicast from the C-RPs to the BSR.

```
 0                   1                   2                   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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|PIM Ver| Type  | Reserved      |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Prefix-Cnt    |   Priority    |             Holdtime          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                 Encoded-Unicast-RP-Address                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Encoded-Group Address-1               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                               .                               |
|                               .                               |
|                               .                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Encoded-Group Address-n               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

PIM Version, Type, Reserved, Checksum Described above.

Prefix-Cnt The number of encoded group addresses included in the message; indicating the group prefixes for which the C-RP is advertising. A Prefix-Cnt of '0' implies a prefix of 224.0.0.0 with mask length of 4; i.e. all multicast groups. If the C-RP is not configured with Group-prefix information, the C-RP puts a default value of '0' in this field.

Priority The 'Priority' of the included RP, for the corresponding Encoded-Group Address (if any). highest priority is '0' (i.e. the lower the value of the 'Priority' field, the higher the priority). This field is stored at the BSR upon receipt along with the RP address and corresponding Encoded-Group Address.

Holdtime The amount of time the advertisement is valid. This field allows advertisements to be aged out.

Encoded-Unicast-RP-Address The address of the interface to advertise as a Candidate RP. The format for this address is given in the Encoded-Unicast-Address in 4.1.

Encoded-Group Address-1..n The group prefixes for which the C-RP is advertising. Format previously described.

5 Acknowledgments

Tony Ballardie, Scott Brim, Jon Crowcroft, Bill Fenner, Paul Francis, Joel Halpern, Horst Hodel, Polly Huang, Stephen Ostrowski, Lixia Zhang and Girish Chandranmenon provided detailed comments on previous drafts. The authors of CBT [8] and membership of the IDMR WG provided many of the motivating ideas for this work and useful feedback on design details.

This work was supported by the National Science Foundation, ARPA, cisco Systems and Sun Microsystems.

6 Appendices

6.1 Appendix I: Major Changes and Updates to the Spec

This appendix populates the major changes in the specification document as compared to 'draft-ietf-idmr-pim-spec-01.{ps,txt}'.

Major Changes

List of changes since March '96 IETF:

1. (*,*,RP) Joins state and data forwarding check; replaces (*,G-Prefix) Joins state for interoperability. (*,G) negative cache introduced for the (*,*,RP) state supporting mechanisms.
2. Semantic fragmentation for the Bootstrap message.
3. Refinement of Assert details.
4. Addition and refinement of Join/Prune suppression and Register suppression (introduction of null Registers).
5. Editorial changes and clarifications to the timers section.
6. Addition of Appendix II (BSR Election and RP-Set Distribution), and Appendix III (Glossary of Terms).
7. Addition of table of contents.

List of changes incurred since version 1 of the spec.:

1. Proposal and refinement of bootstrap router (BSR) election mechanisms
2. Introduction of hash functions for Group to RP mapping
3. New RP-liveness indication mechanisms based upon the the Bootstrap Router (BSR) and the Bootstrap messages.
4. Removal of reachability messages, RP reports and multiple RPs per group.

Packet Format Changes

Packet Format incurred updates to accommodate different address lengths, and address aggregation.

1. The 'Addr Family' and 'Encoding Type' fields were added to the packet formats.
2. The Encoded source and group address formats were introduced, with the use of a 'Mask length' field to allow aggregation, section 4.1.
3. Packet formats are no longer IGMP messages; rather PIM messages.

PIM message types and formats were also modified:
[*Note: most changes were made to the May 95 version, unless otherwise specified*].

1. Obsolete messages:

   (a) Register-Ack [Feb. 96]
   (b) Poll and Poll Response [Feb. 96]
   (c) RP-Reachability [Feb. 96]
   (d) RPlist-Mapping [Feb. 96]

2. New messages:
   (a) Candidate-RP-Advertisement [change made in October 95]
   (b) RP-Set [Feb. 96]

3. Modified messages:
   (a) Join/Prune [Feb. 96]
   (b) Register [Feb. 96]
   (c) Register-Stop [Feb. 96]
   (d) Hello (addition of OptionTypes) [Aug 96]

4. Renamed messages:
   (a) Query messages are renamed as Hello messages [Aug. 96]
   (b) RP-Set messages are renamed as Bootstrap messages [Aug. 96]

6.2 Appendix II: BSR Election and RP-Set Distribution

For simplicity, the bootstrap message is used in both the BSR election and the RP-Set distribution mechanisms. These mechanisms are described by the following state machine, illustrated in figure 4. The protocol transitions for a Candidate-BSR are given in state diagram (a). For routers not configured as Candidate-BSRs, the protocol transitions are given in state diagram (b).

State variables

LclBSR = Local concatenated BSR priority and BSR IP address
LclRP-Set = Local RP-Set
RxdBSR = Received concatenated BSR priority and BSR IP address
RxdRP-Set = Received RP-Set
Bootstrap-Period = 60 seconds
Bootstrap-Timeout = 2.5 x Bootstrap-Period = 150 seconds

Predicates

Name	*Meaning*
P	$RxdBSR \geq LclBSR$

Incoming events

Name	*Interface*	*Meaning*
PrefBSMRxd	RPF nbr toward included BSR	Bootstrap msg rcvd satisfying **P**
BSMRxd	RPF nbr toward included BSR	Bootstrap message received
TExp	Timer provider machinery	Bootstrap timer expired

States

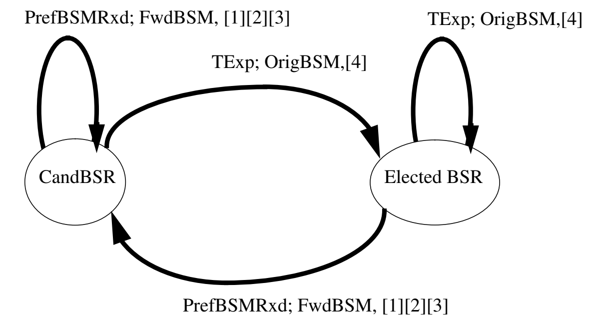
Name	*Meaning*
AxptPref	Accept only Bootstrap messages from preferred or equal BSR
AxptAny	Accept any RP-Set messages coming thru the right interface
CandBSR	Candidate bootstrap router
ElectedBSR	Elected bootstrap router

Outgoing events

Name	*Interface*	*Meaning*
FwdBSM	All interfaces except receiving interface	Forward Bootstrap message
OrigBSM	All interfaces	Originate Bootstrap message

Specific actions

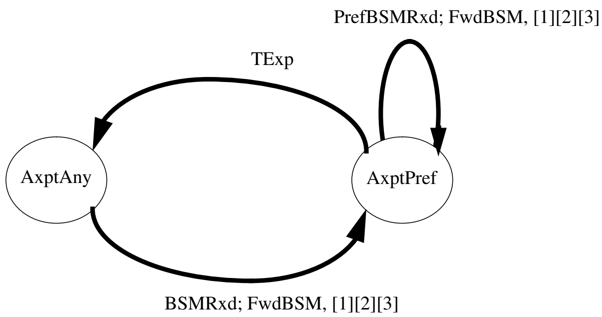
[1] = Restart Bootstrap timer at *Bootstrap-Timeout*
[2] = (*LclBSR = RxdBSR*)
[3] = (*LclRP-Set = RxdRP-Set*)
[4] = Restart Bootstrap timer at *Bootstrap-Period*



Initial state: CandBSR; LclBSR = Local address, LclRP-Set = empty

State transition diagram for a Candidate BSR

(a)



Initial state: AxptAny; LclBSR = 0, LclRP-Set = empty

State transition diagram for a router not configured as C-BSR

(b)

Figure 4: State Diagram for the BSR election and RP-Set distribution mechanisms

Each PIM router keeps a bootstrap-timer, initialized to [Bootstrap-Timeout], in addition to a local BSR field *'LclBSR'* (initialized to a local address if Candidate-BSR, or to 0 otherwise), and a local RP-Set *'LclRP-Set'* (initially empty). The main stimuli to the state machine are timer events and arrival of bootstrap messages:

Initial States and Timer Events

1. If the router is a Candidate-BSR:

   (a) The router operates initially in the 'CandBSR' state, where it does not originate any bootstrap messages.

   (b) If the bootstrap-timer expires, and the current state is 'CandBSR', the router originates a bootstrap message carrying the local RP-Set and its own BSR priority and address, restarts the bootstrap-timer at [Bootstrap-Period] seconds, and transits into the 'ElectedBSR' state. Note that the actual sending of the bootstrap message may be delayed by a random value to reduce transient control overhead. To obtain best results, the random value is set such that the preferred BSR is the first to originate a bootstrap message. We propose the following as an efficient implementation of the random value delay (in seconds):

   $$Delay = 5 + 2 * log_2(1 + bestPriority - myPriority) + AddrDelay$$

   where *myPriority* is the Candidate-BSR's configured priority, and *bestPriority* equals:

   $$bestPriority = Max(storedPriority, myPriority)$$

   and *AddrDelay* is given by the following:

   i. if (*bestPriority* equals *myPriority*) then

   $$AddrDelay = log_2(bestAddr - myAddr)/16,$$

   ii. else

   $$AddrDelay = 2 - (myAddr/2^{31})$$

   where *myAddr* is the Candidate-BSR's address, and *bestAddr* is the stored BSR's address.

   (c) If the bootstrap-timer expires, and the current state is 'ElectedBSR', the router originates a bootstrap message, and restarts the RP-Set timer at [Bootstrap-Period]. No state transition is incurred.

   This way, the elected BSR originates periodic bootstrap messages every [Bootstrap-Period].

2. If a router is not a Candidate-BSR:

   (a) The router operates initially in the 'AxptAny' state. In such state, a router accepts the first bootstrap message from the The Reverse Path Forwarding (RPF) neighbor toward the included BSR. The RPF neighbor in this case is the next hop router en route to the included BSR.

   (b) If the bootstrap-timer expires, and the current state is 'AxptPref'– where the router accepts only preferred bootstrap messages (those that carry BSR-priority and address higher than, or equal to, *'LclBSR'*) from the RPF neighbor toward the included BSR– the router transits into the 'AxptAny' state.

   In this case, if an elected BSR becomes unreachable, the routers start accepting bootstrap messages from another Candidate-BSR after the bootstrap-timer expires. All PIM routers within a domain converge on the preferred reachable Candidate-BSR.

Receiving Bootstrap Message

To avoid loops, an RPF check is performed on the included BSR address. Upon receiving a bootstrap message from the RPF neighbor toward the included BSR, the following actions are taken:

1. If the router is not a Candidate-BSR:
   (a) If the current state is 'AxptAny', the router accepts the bootstrap message, and transits into the 'AxptPref' state.
   (b) If the current state is 'AxptPref', and the bootstrap message is preferred, the message is accepted. No state transition is incurred.
2. If the router is a Candidate-BSR, and the bootstrap message is preferred, the message is accepted. Further, if this happens when the current state is 'Elected BSR', the router transits into the 'CandBSR' state.

When a bootstrap message is accepted, the router restarts the bootstrap-timer at [Bootstrap-Timeout], stores the received BSR priority and address in *'LclBSR'*, and the received RP-Set in *'LclRP-Set'*, and forwards the bootstrap message out all interfaces except the receiving interface.

If a bootstrap message is rejected, no state transitions are triggered.

6.3 Appendix III: Glossary of Terms

Following is an alphabetized list of terms and definitions used throughout this specification.

- **Bootstrap router (BSR)**. A BSR is a dynamically elected router within a PIM domain. It is responsible for constructing the RP-Set and originating Bootstrap messages.

- **Candidate-BSR (C-BSR)**. A C-BSR is a router configured to participate in the BSR election and act as BSRs if elected.

- **Candidate RP (C-RP)**. A C-RP is a router configured to send periodic Candidate-RP-Advertisement messages to the BSR, and act as an RP when it receives Join/Prune or Register messages for the advertised group prefix.

- **Designated Router (DR)**. The DR sets up multicast route entries and sends corresponding Join/Prune and Register messages on behalf of directly-connected receivers and sources, respectively. The DR may or may not be the same router as the IGMP Querier. The DR may or may not be the long-term, last-hop router for the group; a router on the LAN that has a lower metric route to the data source, or to the group's RP, may take over the role of sending Join/Prune messages.

- **Incoming interface (iif)**. The iif of a multicast route entry indicates the interface from which multicast data packets are accepted for forwarding. The iif is initialized when the entry is created.

- **Join list**. The Join list is one of two lists of addresses that is included in a Join/Prune message; each address refers to a source or RP. It indicates those sources or RPs to which downstream receiver(s) wish to join.

- **Last-hop router**. The last-hop router is the last router to receive multicast data packets before they are delivered to directly-connected member hosts. In general the last-hop router is the DR for the LAN. However, under various conditions described in this document a parallel router connected to the same LAN may take over as the last-hop router in place of the DR.

- **Outgoing interface (oif) list**. Each multicast route entry has an oif list containing the outgoing interfaces to which multicast packets should be forwarded.

- **Prune List**. The Prune list is the second list of addresses that is included in a Join/Prune message. It indicates those sources or RPs from which downstream receiver(s) wish to prune.

- **PIM Multicast Border Router (PMBR)**. A PMBR connects a PIM domain to other multicast routing domain(s).

- **Rendezvous Point (RP)**. Each multicast group has a shared-tree via which receivers hear of new sources and new receivers hear of all sources. The RP is the root of this per-group shared tree, called the RP-Tree.

- **RP-Set**. The RP-Set is a set of RP addresses constructed by the BSR based on Candidate-RP advertisements received. The RP-Set information is distributed to all PIM routers in the BSR's PIM domain.

- **Reverse Path Forwarding (RPF)**. RPF is used to select the appropriate incoming interface for a multicast route entry . The RPF neighbor for an address X is the the next-hop router used to forward packets toward X. The RPF interface is the interface to that RPF neighbor. In the common case this is the next hop used by the unicast routing protocol for sending unicast packets toward X. For example, in cases where unicast and multicast routes are not congruent, it can be different.

- **Route entry.** A multicast route entry is state maintained in a router along the distribution tree and is created, and updated based on incoming control messages. The route entry may be different from the forwarding entry; the latter is used to forward data packets in real time. Typically a forwarding entry is not created until data packets arrive, the forwarding entry's iif and oif list are copied from the route entry, and the forwarding entry may be flushed and recreated at will.

- **Shortest path tree (SPT)**. The SPT is the multicast distribution tree created by the merger of all of the shortest paths that connect receivers to the source (as determined by unicast routing).

- **Sparse Mode (SM)**. SM is one mode of operation of a multicast protocol. PIM SM uses explicit Join/Prune messages and Rendezvous points in place of Dense Mode PIM's and DVMRP's broadcast and prune mechanism.

- **Wildcard (WC) multicast route entry**. Wildcard multicast route entries are those entries that may be used to forward packets for any source sending to the specified group. Wildcard bots in the join list of a Join/Prune message represent either a (*,G) or (*,*,RP) join; in the prune list they represent a (*,G) prune.

- **(S,G) route entry**. (S,G) is a source-specific route entry. It may be created in response to data packets, Join/Prune messages, or Asserts. The (S,G) state in routers creates a source-rooted, shortest path (or reverse shortest path) distribution tree. (S,G)RPT bit entries are source-specific entries on the shared RP-Tree; these entries are used to prune particular sources off of the shared tree.

- **(*,G) route entry**. Group members join the shared RP-Tree for a particular group. This tree is represented by (*,G) multicast route entries along the shortest path branches between the RP and the group members.

- **(*,*,RP) route entry**. (*,*,RP) refers to any source and any multicast group that maps to the RP included in the entry. The routers along the shortest path branches between a domain's RP(s) and its PMBRs keep (*,*,RP) state and use it to determine how to deliver packets toward the PMBRs if data packets arrive for which there is not a longer match. The wildcard group in the (*,*,RP) route entry is represented by a group address of 224.0.0.0 and a mask length of 4 bits.

References

[1] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C. Liu, L. Wei, P. Sharma, and A. Helmy. Protocol independent multicast (pim) : Motivation and architecture. *Internet Draft*, May 1995.

[2] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C. Liu, and L. Wei. The pim architecture for wide-area multicast routing. *ACM Transactions on Networks*, April 1996.

[3] D. Estrin, D. Farinacci, V. Jacobson, C. Liu, L. Wei, P. Sharma, and A. Helmy. Protocol independent multicast-dense mode (pim-dm) : Protocol specification. *Internet Draft*, November 1995.

[4] S. Deering. Host extensions for ip multicasting, aug 1989. RFC1112.

[5] W. Fenner. Internet group management protocol, version 2. *Internet Draft*, May 1996.

[6] R. Atkinson. Security architecture for the internet protocol, August 1995. RFC-1825.

[7] Mark R. Nelson. File verification using CRC. *Dr. Dobb's Journal*, May 1992.

[8] A. J. Ballardie, P. F. Francis, and J. Crowcroft. Core based trees. In *Proceedings of the ACM SIGCOMM*, San Francisco, 1993.