NAME

pim - Protocol Independent Multicast

SYNOPSIS

options MROUTING

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netinet/ip_mroute.h>
#include <netinet/pim.h>

int
getsockopt(*int s, IPPROTO_IP, MRT_PIM, void *optval, socklen_t *optlen*);

int
setsockopt(*int s, IPPROTO_IP, MRT_PIM, const void *optval, socklen_t optlen*);

int
getsockopt(*int s, IPPROTO_IPV6, MRT6_PIM, void *optval, socklen_t *optlen*);

int
setsockopt(*int s, IPPROTO_IPV6, MRT6_PIM, const void *optval, socklen_t optlen*);

DESCRIPTION

PIM is the common name for two multicast routing protocols: Protocol Independent Multicast - Sparse Mode (PIM-SM) and Protocol Independent Multicast - Dense Mode (PIM-DM).

PIM-SM is a multicast routing protocol that can use the underlying unicast routing information base or a separate multicast-capable routing information base. It builds unidirectional shared trees rooted at a Rendezvous Point (RP) per group, and optionally creates shortest-path trees per source.

PIM-DM is a multicast routing protocol that uses the underlying unicast routing information base to flood multicast datagrams to all multicast routers. Prune messages are used to prevent future datagrams from propagating to routers with no group membership information.

Both PIM-SM and PIM-DM are fairly complex protocols, though PIM-SM is much more complex. To enable PIM-SM or PIM-DM multicast routing in a router, the user must enable multicast routing and PIM processing in the kernel (see *SYNOPSIS* about the kernel configuration options), and must run a PIM-SM or PIM-DM capable user-level process. From developer’s point of view, the programming

guide described in the *Programming Guide* section should be used to control the PIM processing in the kernel.

Programming Guide

After a multicast routing socket is open and multicast forwarding is enabled in the kernel (see multicast(4)), one of the following socket options should be used to enable or disable PIM processing in the kernel. Note that those options require certain privilege (i.e., root privilege):

```
/* IPv4 */
int v = 1;       /* 1 to enable, or 0 to disable */
setsockopt(mrouter_s4, IPPROTO_IP, MRT_PIM, (void *)&v, sizeof(v));

/* IPv6 */
int v = 1;       /* 1 to enable, or 0 to disable */
setsockopt(mrouter_s6, IPPROTO_IPV6, MRT6_PIM, (void *)&v, sizeof(v));
```

After PIM processing is enabled, the multicast-capable interfaces should be added (see multicast(4)). In case of PIM-SM, the PIM-Register virtual interface must be added as well. This can be accomplished by using the following options:

```
/* IPv4 */
struct vifctl vc;
memset(&vc, 0, sizeof(vc));
/* Assign all vifctl fields as appropriate */
...
if (is_pim_register_vif)
   vc.vifc_flags |= VIFF_REGISTER;
setsockopt(mrouter_s4, IPPROTO_IP, MRT_ADD_VIF, (void *)&vc,
         sizeof(vc));

/* IPv6 */
struct mif6ctl mc;
memset(&mc, 0, sizeof(mc));
/* Assign all mif6ctl fields as appropriate */
...
if (is_pim_register_vif)
   mc.mif6c_flags |= MIFF_REGISTER;
setsockopt(mrouter_s6, IPPROTO_IPV6, MRT6_ADD_MIF, (void *)&mc,
         sizeof(mc));
```

Sending or receiving of PIM packets can be accomplished by opening first a "raw socket" (see socket(2)), with protocol value of IPPROTO_PIM:

```
/* IPv4 */
int pim_s4;
pim_s4 = socket(AF_INET, SOCK_RAW, IPPROTO_PIM);

/* IPv6 */
int pim_s6;
pim_s6 = socket(AF_INET6, SOCK_RAW, IPPROTO_PIM);
```

Then, the following system calls can be used to send or receive PIM packets: sendto(2), sendmsg(2), recvfrom(2), recvmsg(2).

SEE ALSO

getsockopt(2), recvfrom(2), recvmsg(2), sendmsg(2), sendto(2), setsockopt(2), socket(2), inet(4), intro(4), ip(4), multicast(4)

STANDARDS

The PIM-SM protocol is specified in RFC 2362 (to be replaced by *draft-ietf-pim-sm-v2-new-*).* The PIM-DM protocol is specified in *draft-ietf-pim-dm-new-v2-*).*

AUTHORS

The original IPv4 PIM kernel support for IRIX and SunOS-4.x was implemented by Ahmed Helmy (USC and SGI). Later the code was ported to various BSD flavors and modified by George Edmond Eddy (Rusty) (ISI), Hitoshi Asaeda (WIDE Project), and Pavlin Radoslavov (USC/ISI and ICSI). The IPv6 PIM kernel support was implemented by the KAME project (*https://www.kame.net*), and was based on the IPv4 PIM kernel support.

This manual page was written by Pavlin Radoslavov (ICSI).