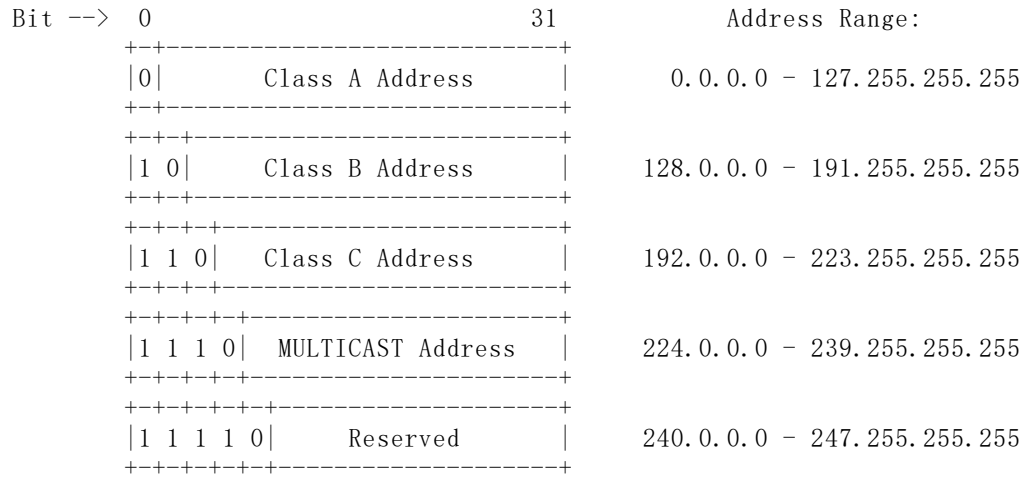
Next Previous Contents

2. Multicast Explained.

2.1 Multicast addresses.

As you probably know, the range of IP addresses is divided into "classes" based on the high order bits of a 32 bits IP address:



The one which concerns us is the "Class D Address". Every IP datagram whose destination address starts with "1110" is an IP Multicast datagram.

The remaining 28 bits identify the multicast "*group*" the datagram is sent to. Following with the previous analogy, you have to tune your radio to hear a program that is transmitted at some specific frequency, in the same way you have to "tune" your kernel to receive packets sent to an specific multicast group. When you do that, it's said that the host has *joined* that group in the interface you specified. More on this later.

There are some special multicast groups, say "well known multicast groups", you should not use in your particular applications due the special purpose they are destined to:

- 224.0.0.1 is the *all-hosts* group. If you ping that group, all multicast capable hosts on the network should answer, as every multicast capable host *must* join that group at start-up on all it's multicast capable interfaces.
- 224.0.0.2 is the *all-routers* group. All multicast routers must join that group on all it's multicast capable interfaces.
- 224.0.0.4 is the *all DVMRP routers*, 224.0.0.5 the *all OSPF routers*, 224.0.013 the *all PIM routers*, etc.

All this special multicast groups are regularly published in the "Assigned Numbers" RFC.

In any case, range 224.0.0.0 through 224.0.0.255 is reserved for local purposes (as administrative and maintenance tasks) and datagrams destined to them are never forwarded by multicast routers. Similarly, the range 239.0.0.0 to 239.255.255.255 has been

reserved for "administrative scoping" (see section 2.3.1 for information on administrative scoping).

2.2 Levels of conformance.

Hosts can be in three different levels of conformance with the Multicast specification, according to the requirements they meet.

Level 0 is the "no support for IP Multicasting" level. Lots of hosts and routers in the Internet are in this state, as multicast support is not mandatory in IPv4 (it is, however, in IPv6). Not too much explanation is needed here: hosts in this level can neither send nor receive multicast packets. They must ignore the ones sent by other multicast capable hosts.

Level 1 is the "support for sending but not receiving multicast IP datagrams" level. Thus, note that it is not necessary to join a multicast group to be able to send datagrams to it. Very few additions are needed in the IP module to make a "Level 0" host "Level 1-compliant", as shown in section 2.3.

Level 2 is the "full support for IP multicasting" level. Level 2 hosts must be able to both send and receive multicast traffic. They must know the way to join and leave multicast groups and to propagate this information to multicast routers. Thus, they must include an Internet Group Management Protocol (IGMP) implementation in their TCP/IP stack.

2.3 Sending Multicast Datagrams.

By now, it should be obvious that multicast traffic is handled at the transport layer with UDP, as TCP provides point-to-point connections, not feasibles for multicast traffic. (Heavy research is taking place to define and implement new multicast-oriented transport protocols. See section Multicast Transport Protocols for details).

In principle, an application just needs to open a UDP socket and fill with a class D multicast address the destination address where it wants to send data to. However, there are some operations that a sending process must be able to control.

TTL.

The TTL (Time To Live) field in the IP header has a double significance in multicast. As always, it controls the live time of the datagram to avoid it being looped forever due to routing errors. Routers decrement the TTL of every datagram as it traverses from one network to another and when its value reaches 0 the packet is dropped.

The TTL in IPv4 multicasting has also the meaning of "threshold". Its use becomes evident with an example: suppose you set a long, bandwidth consuming, video conference between all the hosts belonging to your department. You want that huge amount of traffic to remain in your LAN. Perhaps your department is big enough to have various LANs. In that case you want those hosts belonging to each of *your* LANs to attend the conference, but in any case you want to collapse the entire Internet with your multicast traffic. There is a need to limit how "long" multicast traffic will expand across routers. That's what the TTL is used for. Routers have a TTL threshold assigned to each of its interfaces, and only datagrams with a TTL greater than the interface's threshold are forwarded. Note that when

a datagram traverses a router with a certain threshold assigned, the datagram's TTL is *not* decremented by the value of the threshold. Only a comparison is made. (As before, the TTL is decremented by 1 each time a datagram passes across a router).

A list of TTL thresholds and their associated scope follows:

```
TTL     Scope
-----------------------------------------------------------------------
   0    Restricted to the same host. Won't be output by any interface.
   1    Restricted to the same subnet. Won't be forwarded by a router.
 <32         Restricted to the same site, organization or department.
 <64 Restricted to the same region.
<128 Restricted to the same continent.
<255 Unrestricted in scope. Global.
```

Nobody knows what "site" or "region" mean exactly. It is up to the administrators to decide what this limits apply to.

The TTL-trick is not always flexible enough for all needs, specially when dealing with overlapping regions or trying to establish geographic, topologic and bandwidth limits simultaneously. To solve this problems, administratively scoped IPv4 multicast regions were established in 1994. (see D. Meyer's "*Administratively Scoped IP Multicast*" Internet draft). It does scoping based on multicast addresses rather than on TTLs. The range 239.0.0.0 to 239.255.255.255 is reserved for this administrative scoping.

Loopback.

When the sending host is Level 2 conformant and is also a member of the group datagrams are being sent to, a copy is looped back by default. This does not mean that the interface card reads its own transmission, recognizes it as belonging to a group the interface belongs to, and reads it from the network. On the contrary, is the IP layer which, by default, recognizes the to-be-sent datagram and copies and queues it on the IP input queue before sending it.

This feature is desirable in some cases, but not in others. So the sending process can turn it on and off at wish.

Interface selection.

Hosts attached to more than one network should provide a way for applications to decide which network interface will be used to output the transmissions. If not specified, the kernel chooses a default one based on system administrator's configuration.

2.4 Receiving Multicast Datagrams.

Joining a Multicast Group.

Broadcast is (in comparison) easier to implement than multicast. It doesn't require processes to give the kernel some rules regarding what to do with broadcast packets. The kernel just knows what to do: read and deliver *all* of them to the proper applications.

With multicast, however, it is necessary to advise the kernel which multicast groups we are interested in. That is, we have to ask the kernel to "join" those multicast groups. Depending on the underlying hardware, multicast datagrams are filtered by the hardware or by the IP layer (and, in some cases, by both). Only those with a destination group previously registered via a join are accepted.

Essentially, when we join a group we are telling the kernel: "OK. I know that, by default, you ignore multicast datagrams, but remember that I am interested in *this* multicast group. So, do read and deliver (to any process interested in them, not only to me) any datagram that you see in this network interface with this multicast group in its destination field".

Some considerations: first, note that you don't just join a group. You join a group *on* a particular network interface. Of course, it is possible to join the same group on more than one interface. If you don't specify a concrete interface, then the kernel will choose it based on its routing tables when datagrams are to be sent. It is also possible that more than one process joins the same multicast group on the same interface. They will all receive the datagrams sent to that group via that interface.

As said before, any multicast-capable hosts join the *all-hosts* group at start-up , so "pinging" 224.0.0.1 returns all hosts in the network that have multicast enabled.

Finally, consider that for a process to receive multicast datagrams it has to ask the kernel to join the group *and* bind the port those datagrams were being sent to. The UDP layer uses both the destination address and port to demultiplex the packets and decide which socket(s) deliver them to.

Leaving a Multicast Group.

When a process is no longer interested in a multicast group, it informs the kernel that *it* wants to leave that group. It is important to understand that this doesn't mean that the kernel will no longer accept multicast datagrams destined to that multicast group. It will still do so if there are more precesses who issued a "multicast join" petition for that group and are still interested. In that case *the host* remains member of the group, until all the processes decide to leave the group.

Even more: if you leave the group, but remain bound to the port you were receiving the multicast traffic on, and there are more processes that joined the group, you will still receive the multicast transmissions.

The idea is that joining a multicast group only tells the IP and data link layer (which in some cases explicitly tells the hardware) to accept multicast datagrams destined to that group. It is not a per-process membership, but a per-host membership.

Mapping of IP Multicast Addresses to Ethernet/FDDI addresses.

Both Ethernet and FDDI frames have a 48 bit destination address field. In order to avoid a kind of multicast ARP to map multicast IP addresses to ethernet/FDDI ones, the IANA reserved a range of addresses for multicast: every ethernet/FDDI frame with its destination in the range 01-00-5e-00-00-00 to 01-00-5e-ff-ff-ff (hex) contains data for a multicast group. The prefix 01-00-5e identifies the frame as multicast, the next bit is always 0 and so only 23 bits are left to the multicast address. As IP multicast groups are 28 bits long, the

mapping can not be one-to-one. Only the 23 least significant bits of the IP multicast group are placed in the frame. The remaining 5 high-order bits are ignored, resulting in 32 different multicast groups being mapped to the same ethernet/FDDI address. This means that the ethernet layer acts as an imperfect filter, and the IP layer will have to decide whether to accept the datagrams the data-link layer passed to it. The IP layer acts as a definitive perfect filter.

Full details on IP Multicasting over FDDI are given in RFC 1390: "*Transmission of IP and ARP over FDDI Networks*". For more information on mapping IP Multicast addresses to ethernet ones, you may consult `draft-ietf-mboned-intro-multicast-03.txt`: "*Introduction to IP Multicast Routing*".

If you are interested in IP Multicasting over Token-Ring Local Area Networks, see RFC 1469 for details.

Next Previous Contents