Chapter 12 – Broadcast and Multicast

Three types of IP address

- **Unicast** – A single interface on a single host
- **Broadcast** – All hosts on a single network (or networks)
- **Multicast** – Some group of hosts

Packet filtering

<table>
<thead>
<tr>
<th>LAN</th>
<th>Address mismatch and not (broadcast or multicast)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multicast ethernet mask</td>
<td>01:00:00:00:00:00</td>
</tr>
<tr>
<td>Link Driver</td>
<td>Network layer Protocol unknown</td>
</tr>
<tr>
<td>IP</td>
<td>Transport layer protocol unknown</td>
</tr>
<tr>
<td>UDP</td>
<td>Port not usable</td>
</tr>
</tbody>
</table>

Principle objective in broadcasting and multicasting

Force the discard to take place as low as possible if no one is interested

IP Broadcast

Limited broadcast – 255.255.255.255
  Used by a host that doesn’t (yet) know its IP address and subnet mask
  Datagram sent this way are *never* forwarded by any router
  Should multi–homed hosts send on all interfaces or not?
  For BSD no.

Net directed broadcast – netid.255.255.255
  Router must be able to forward net directed broadcasts *and*
  to disable such forwarding

Subnet directed broadcast – netid.subnetid.255

All subnets directed broadcast
  Obsolete

Broadcasting examples:
- Ping (OS/2) 130.127.48.255 –
  Works as expected: all hosts respond
- Ping (OS/2) 130.127.255.255 –
  Response only from 130.127.44.1 (router on the back side of ours.)
Disadvantages of net and subnet directed broadcasts

- Creates excessive traffic
- Enables "Smurf" denial of service attacks when source address spoofed on broadcast ping.

Recommended solutions for router managers:
- Don’t forward broadcasts period.
- Don’t forward packets carrying source addresses you don’t own.

Broadcasting with UDP

- You may have to set the SO_BROADCAST socket option before attempting a broadcast
- On some systems root privilege is required.

Summary

- Broadcast should be used with care.
- Your broadcast mileage may vary!
- Subnet directed Ping provides a good way for a hacker to easily retrieve all your IP addresses
- Multicast is probably a better plan.

Multicasting on the Internet

Good starting-point references
- Van Jacobsen Tutorial – ACM SigComm ’94
- Comm. of the ACM – Aug 1994 – Special issue on Internet Technology

Primary target application
- Multimedia conferencing on the internet
- Available tools:
  - sd, ivs, vat, nv (MBONE, CACM Aug 1994)

Historical perspective and distortions (Jacobson’s view)
- Myths:
  - Real–time traffic like audio and video requires a connection oriented virtual circuit network (preferably unreliable like ATM–SONET)
  - Datagram networks won’t work because:
    - Lack of state information precludes resource reservation necessary to meet real–time demands
    - IP transit times are best effort and vary wildly

Proof of mythical nature of above statements:
- Existence counterexample:
  - The MBONE 10000+ users on 1500 networks in 30 countries
My view: the real issue is not real-time traffic, per se, but the scope of the conference

2 – person conference – Unreliable connection oriented service works just fine.. consider your telephone
n – person conference
Problems with the connection oriented approach
– To set it up all end users all have to know how to connect to each other
– N – participants requires N^2 connections... bad scalability
– Reliability problems.. conference may fail if any of the connections do
– Difficult to joint in progress
– Have you ever tried to make a conference call?

Keys to IP’s suitability

It’s the one address of the conference itself that must be considered in routing
No need to worry about the N^2 connections

‘‘Minor’’ problem
Privacy of conversations
Can be addressed by encryption

IP multicast architecture

Process joins a multicast group by specifying (group_address, local_interface).. The host might have multiple interfaces group membership is interface specific. Any datagram sent using the multicast address goes to all members of the group

Class D addresses used
1 1 1 0 ------------- 28 bit multicast group ID -------------

Group addresses live in the range
224.0.0.0 – 239.255.255.255

Reserved addresses
224.0.0.0
224.0.0.1 – All hosts participating in IP multicast (Comer)
224.0.0.2 – All routers participating in IP multicast

To receive a multicast message you must belong to a multicast host group.
Membership is dynamic... hosts may enter and exit
Membership may cross networks
Membership is managed by IGMP
Some host group numbers are permanently assigned by IANA (see assigned # RFC)
Mapping of Multicast to Ethernet addresses

Recall ethernet multicast mask 01:00:00:00:00:00

IANA reserved ethernet addresses in range
 01:00:5e:00:00:00
 01:00:5e:ff:ff:ff

IANA allocated half of these for multicast
 01:00:5e:00:00:00
 01:00:5e:7f:ff:ff

Mapping is:
  ENet address = 01:00:5e:low order 23 bits of IP

The low order bits of an IP address specify the multicast group thus,
since there are $2^{28}$ multicast addresses and $2^{23}$ E-net mcast addresses,
32 distinct multicast addresses map to a single E-net address.

Hardware considerations:

There are $2^{23}$ different multicast hardware addresses

How does an adapter determine filter out unwanted ones:
  Via a table lookup until table overflows
  Multicast promiscuous mode thereafter.
  (Other possibilities exist).
### INTERNET MULTICAST ADDRESSES

Host Extensions for IP Multicasting [RFC1112] specifies the extensions required of a host implementation of the Internet Protocol (IP) to support multicasting. Current addresses are listed below.

<table>
<thead>
<tr>
<th>Address</th>
<th>Description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>224.0.0.0</td>
<td>Base Address (Reserved)</td>
<td>[RFC1112, JBP]</td>
</tr>
<tr>
<td>224.0.0.1</td>
<td>All Systems on this Subnet</td>
<td>[RFC1112, JBP]</td>
</tr>
<tr>
<td>224.0.0.2</td>
<td>All Routers on this Subnet</td>
<td>[JBP]</td>
</tr>
<tr>
<td>224.0.0.3</td>
<td>Unassigned</td>
<td>[JBP]</td>
</tr>
<tr>
<td>224.0.0.4</td>
<td>DVMRP Routers</td>
<td>[RFC1075, JBP]</td>
</tr>
<tr>
<td>224.0.0.5</td>
<td>OSPFIGP OSPFIGP All Routers</td>
<td>[RFC1583, JXM1]</td>
</tr>
<tr>
<td>224.0.0.6</td>
<td>OSPFIGP OSPFIGP Designated Routers</td>
<td>[RFC1583, JXM1]</td>
</tr>
<tr>
<td>224.0.0.7</td>
<td>ST Routers</td>
<td>[RFC1190, KS14]</td>
</tr>
<tr>
<td>224.0.0.8</td>
<td>ST Hosts</td>
<td>[RFC1190, KS14]</td>
</tr>
<tr>
<td>224.0.0.9</td>
<td>RIP2 Routers</td>
<td>[RFC1583, JXM1]</td>
</tr>
<tr>
<td>224.0.0.10</td>
<td>IGRP Routers</td>
<td></td>
</tr>
<tr>
<td>224.0.0.11</td>
<td>Mobile-Agents</td>
<td>[JBP]</td>
</tr>
<tr>
<td>224.0.0.12-224.0.0.255</td>
<td>Unassigned</td>
<td>[JBP]</td>
</tr>
<tr>
<td>224.0.1.0</td>
<td>VMTP Managers Group</td>
<td>[RFC1045, DRC3]</td>
</tr>
<tr>
<td>224.0.1.1</td>
<td>NTP Network Time Protocol</td>
<td>[RFC1119, DLM1]</td>
</tr>
<tr>
<td>224.0.1.2</td>
<td>SGI-Dogfight</td>
<td>[AXC]</td>
</tr>
<tr>
<td>224.0.1.3</td>
<td>Rwhod</td>
<td>[SXD]</td>
</tr>
<tr>
<td>224.0.1.4</td>
<td>VNP</td>
<td>[DRC3]</td>
</tr>
<tr>
<td>224.0.1.5</td>
<td>Artificial Horizons - Aviator</td>
<td>[BXF]</td>
</tr>
<tr>
<td>224.0.1.6</td>
<td>NSS - Name Service Server</td>
<td>[BXS2]</td>
</tr>
<tr>
<td>224.0.1.7</td>
<td>AUDIONEWS - Audio News Multicast</td>
<td>[MXF2]</td>
</tr>
<tr>
<td>224.0.1.8</td>
<td>SUN NIS+ Information Service</td>
<td>[CXM3]</td>
</tr>
<tr>
<td>224.0.1.9</td>
<td>MTP Multicast Transport Protocol</td>
<td>[SXA]</td>
</tr>
<tr>
<td>224.0.1.10</td>
<td>IETF-1-LOW-AUDIO</td>
<td>[SC3]</td>
</tr>
<tr>
<td>224.0.1.11</td>
<td>IETF-1-AUDIO</td>
<td>[SC3]</td>
</tr>
<tr>
<td>224.0.1.12</td>
<td>IETF-1-VIDEO</td>
<td>[SC3]</td>
</tr>
<tr>
<td>224.0.1.13</td>
<td>IETF-2-LOW-AUDIO</td>
<td>[SC3]</td>
</tr>
<tr>
<td>224.0.1.14</td>
<td>IETF-2-AUDIO</td>
<td>[SC3]</td>
</tr>
<tr>
<td>224.0.1.15</td>
<td>IETF-2-VIDEO</td>
<td>[SC3]</td>
</tr>
<tr>
<td>224.0.1.16</td>
<td>MUSIC-SERVICE</td>
<td>[Guido van Rossum]</td>
</tr>
<tr>
<td>224.0.1.17</td>
<td>SEANET-TELEMETRY</td>
<td>[Andrew Maffei]</td>
</tr>
<tr>
<td>224.0.1.18</td>
<td>SEANET-IMAGE</td>
<td>[Andrew Maffei]</td>
</tr>
<tr>
<td>224.0.1.19</td>
<td>MLOADD</td>
<td>[Braden]</td>
</tr>
<tr>
<td>224.0.1.20</td>
<td>any private experiment</td>
<td>[JBP]</td>
</tr>
<tr>
<td>224.0.1.21</td>
<td>DVMRP on MOSPF</td>
<td>[John Moy]</td>
</tr>
<tr>
<td>224.0.1.22</td>
<td>SVRLOC</td>
<td><a href="mailto:veizades@ftp.com">veizades@ftp.com</a></td>
</tr>
<tr>
<td>224.0.1.23</td>
<td>XINGTV</td>
<td><a href="mailto:hgxing@aol.com">hgxing@aol.com</a></td>
</tr>
<tr>
<td>224.0.1.24</td>
<td>microsoft-ds</td>
<td><a href="mailto:arnoldm@microsoft.com">arnoldm@microsoft.com</a></td>
</tr>
<tr>
<td>224.0.1.25</td>
<td>nbc-pro</td>
<td><a href="mailto:bloomer@birch.crd.ge.com">bloomer@birch.crd.ge.com</a></td>
</tr>
<tr>
<td>224.0.1.26</td>
<td>nbc-pfn</td>
<td><a href="mailto:bloomer@birch.crd.ge.com">bloomer@birch.crd.ge.com</a></td>
</tr>
<tr>
<td>224.0.1.27-224.0.1.255</td>
<td>Unassigned</td>
<td>[JBP]</td>
</tr>
</tbody>
</table>
Example multicast program:

```c
struct ip_mreq mreq;
unsigned char mgroup[4] = {0xe0, 0x00, 0x00, 0xe0};
int on = 1;

sock = socket(PF_INET, SOCK_DGRAM, 0);

/* Get host network address from command line parm */

hp = gethostbyname(argv[1]);
if (hp == 0)
{
    printf("Host %s not found\n", argv[1]);
    exit(1);
}

A multicast address — like any other IP address — is a (host, port) pair. Thus, if multiple
processes on the same system want to all listen in, they must explicitly share the port.

status = setsockopt(sock, SOL_SOCKET, SO_REUSEADDR,
                     (char *)&on, sizeof(on));
if (status < 0)
    printf("Set reuse failed with %d \n", status);

bcopy((char *)hp->h_addr, &mreq.imr_interface, 4);
bcopy((char *)mgroup, &mreq.imr_multiaddr, 4);

The setsockopt function call is required for the socket to be mulitcast usable and generates an igmp
add membership notification. For solaris systems the socket level should be set to IPPROTO_IP
(both are #defined as 0)

status = setsockopt(sock, SOL_IP, IP_ADD_MEMBERSHIP,
                     (char *)&mreq, sizeof(mreq));
```

6
The remainder of the example is just like a non multicast program.

/* Fill in protocol family and port # from command line then */
/* bind the socket to the specified address. */

    name.sin_family = AF_INET;
    name.sin_port = htons(atoi(argv[2]));
    bcopy((char *)mgroup,
          (char *)&name.sin_addr, 4);

    status = bind(sock, (struct sockaddr *)&name, sizeof(name));
    printf("Bind status = %d \n", status);
    if (status < 0)
        exit(1);

    printf("Dest net address = ");
    for (i = 0; i < hp->h_length; i++)
        { c = *((char *)&name.sin_addr + i);
          printf(" %2x", c);
        }

    status = sendto(sock, "This is a multicast\n", 20, 0,
                     &name, sizeof(name));
    printf("%d bytes sent \n", status);
    exit(1);

/* Wait for someone to send me a datagram */

    namelen = sizeof(struct sockaddr_in);
    status = recvfrom(sock, buf, sizeof(buf), 0,
                       (struct sockaddr *)&sname, &namelen);

TCP Dump output:

14:38:56.214849 jmw3 > 224.0.0.224: igmp nreport 224.0.0.224 [ttl 1]
14:38:56.221381 jmw3.55555 > 224.0.0.224.55555: udp 20 [ttl 1]
14:38:56.367139 jmw3 > 224.0.0.224: igmp leave 224.0.0.224 [ttl 1]