



The IPv4 Multicast Service Model - and - How Best To Support It

Or: What The Heck
Are The Network Folks
Trying To Do, Anyway?

Bill Nickless
nickless@mcs.anl.gov
<http://www.mcs.anl.gov/home/nickless>



Overview

- The Multicast Service Model
- Multicast Application Operations
- Current Implementations
- Towards a User-Level Multicast API and Library for Access Grid



The Multicast Service Model

- Application transmits UDP packets to a virtual Group Address
- Applications register their interest in those UDP packets (by Group Address [And Source?])
- The operating system, networks, and routers conspire to deliver the UDP packets to interested receivers



Group Address

- IPv4 address in the 224.0.0.0/4 range (224.0.0.0 to 239.255.255.255)
- IPv6 addresses start with the octet 0xFF
- No host or router is EVER assigned addresses within these ranges



Internetwork Multicast Routing

- A distribution tree is created:
 - Rooted at the transmitter
 - Follows routing tables away from the transmitter
- Interested receivers are grafted onto distribution tree
- Distribution tree \sim network flow
 - Stateful forwarding
 - Takes time to create



Internetwork Multicast Routing

- Dense Mode:
"Hurt Me Until I Tell You To Stop."
- Sparse Mode
"Hurt Me Only When and How I Tell You To."
- Tunnels
"You're probably hurting someone else, but I can't tell and I don't care."



Multicast Application Operations

- GroupInterfaceDiscovery

What IP address should the application use when transmitting packets to a Group address?

Needed to ensure that the transmission goes out the way that the routing protocols expect.



Multicast Application Operations

- GroupTransmit

Send a packet to a group address for interested receivers. (UDP)



Multicast Application Operations

- GroupJoinAny
“Hurt me—I want traffic on this group”
Register interest in all packets sent to a particular group address.
- GroupJoinOnly
“Only let that source hurt me”
Mostly the same as GroupJoinAny, but only interested in packets sent by a particular transmitter.



Multicast Application Operations

- GroupReceive

“Ouch!”

Get a group-addressed packet from the network and put it in a buffer for processing.



Multicast Application Operations

- GroupLeave

“Stop It Already—I’ve had enough!”

Tell the network to stop delivering packets addressed to a particular group address

(or maybe just from a given source?)



Multicast Application Operations (Wouldn't it be nice if....)

- GroupSize

“Does anyone care?”

- Application generates data only when there are interested listeners.
- In the AG Context: trans-code video to lower- or higher-bandwidth streams based on receiver interest.



Multicast Application Operations (Wouldn't it be nice if....)

- SecureGroupJoinAny, SecureGroupJoinOnly, SecureGroupTransmit
 - Cut back on Denial-of-Service attacks
 - Foil Those Evil Eavesdroppers
 - “Abandoned Node” problem



Current Implementations

- IGMPv2
 - The Original, Traditional, Standard
 - Implemented in many operating systems
 - Socket calls for group operations
 - Requires Layer 2 deployment to the edge
 - Does not support GroupJoinOnly
 - Implicit support for GroupInterfaceDiscovery



Current Implementations

- IGMPv3
 - Relatively recent IETF extension
 - Implemented in a few operating systems
 - Socket calls for group operations
 - Requires Layer 2 deployment to the edge
 - Supports GroupJoinOnly and GroupJoinAny
 - Implicit support for GroupInterfaceDiscovery



Current Implementations

- Multi-session Bridge, UMTP, Castgate
 - Intent: workaround for broken edge Layer-2 IGMP deployment User-level forwarding
 - Supports GroupJoinAny, not GroupJoinOnly
 - Requires bridge host: unicast encapsulation to routable multicast
 - Performs Network Address Translation rather than implement GroupInterfaceDiscovery



Possible Future Implementations

- Extend UMTP/CastGate
 - support GroupInterfaceDiscovery rather than performing Network Address Translation. Think DHCP, not NAT.
 - Make the bridge host speak PIM, so that a yes/no variant of GroupSize can be supported all the way to the client.



Possible Future Implementations

- Host-based PIM
 - Probably best for point-to-point network links
 - No IGMP required
 - Supports all core operations
 - GroupInterfaceDiscovery implicit
 - Yes/No variant of GroupSize
 - No changes required on the network



Towards A User-Level Multicast API

- What do we want in a User-Level Multicast API?
 - Thread-safety
 - Select()-ability?
 - Group-joined file handles?
 - Registration of callbacks for packet delivery?



Towards A User-Level Multicast API

- Scope of a User-Level Multicast API
 - Does it need to pass anything other than UDP packets?
 - How best to interface with....
 - Real Time Transport (RTP)?
 - Reliable Multicast?
 - Rate-limits? (*“Only Hurt Me So Much And No More”*)
 - Statistics?
 - Emulate features that aren't supported by the underlying network, like GroupJoinOnly?



Towards A User-Level Multicast API

- APIs are often implemented in libraries.
 - Required languages (C, C++, C#, Perl, Java, ...?)
- System administrator control of transport used:
 - UMTP/CastGate/MSB server vs. IGMPv2 or IGMPv3.
 - Application transparent?
 - User transparent?
 - Application override for embedded applications?