Network Working Group
Request for Comments: 3170
Category: Informational

B. Quinn
Celox Networks
K. Almeroth
UC-Santa Barbara

September 2001

IP Multicast Applications: Challenges and Solutions

Status of this Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (2001). All Rights Reserved.

Abstract

This document describes the challenges involved with designing and implementing multicast applications. It is an introductory guide for application developers that highlights the unique considerations of multicast applications as compared to unicast applications.

To this end, the document presents a taxonomy of multicast application I/O models and examples of the services they can support. It then describes the service requirements of these multicast applications, and the recent and ongoing efforts to build protocol solutions to support these services.

Table of Contents

1. Introduction. 2
  1.1 Motivation 2
  1.2 Focus and Scope. 3
2. IP Multicast-enabled Network. 3
  2.1 Essential Protocol Components. 4
    2.1.1 Expedient Joins and Leaves 5
    2.1.2 Send without a Join. 5
3. IP Multicast Application Taxonomy 6
  3.1 One-to-Many Applications 8
  3.2 Many-to-Many Applications. 9
  3.3 Many-to-One Applications 10
4. Common Multicast Service Requirements 13
  4.1 Bandwidth Requirements 13

4.2 Delay Requirements 13
5. Unique Multicast Service Requirements 14
5.1 Address Management 16
5.2 Session Management 16
5.3 Heterogeneous Receiver Support 18
5.4 Reliable Data Delivery 20
5.5 Security 21
5.6 Synchronized Play-Out. 23
6. Service APIs. 23
7. Security Considerations 24
8. Acknowledgements. 24
9. References. 24
10. Authors' Addresses 27
11. Full Copyright Statement 28

1. Introduction

IP Multicast will play a prominent role on the Internet in the coming years. It is a requirement, not an option, if the Internet is going to scale. Multicast allows application developers to add more functionality without significantly impacting the network.

Developing multicast-enabled applications is ostensibly simple. Having datagram access allows any application to send to a multicast address. A multicast application need only increase the Internet Protocol (IP) time-to-live (TTL) value to more than 1 (the default value) to allow outgoing datagrams to traverse routers. To receive a multicast datagram, applications join the multicast group, which transparently generates an [IGMPv2, IGMPv3] group membership report.

This apparent simplicity is deceptive, however. Enabling multicast support in applications and protocols that can scale well on a heterogeneous network is a significant challenge. Specifically, sending constant bit rate datastreams, reliable data delivery, security, and managing many-to-many communications all require special consideration. Some solutions are available, but many of these services are still active research areas.

1.1 Motivation

The purpose of this document is to provide a framework for understanding the challenges of designing and implementing multicast applications. In order to use multicast communications correctly, application developers must first understand the various I/O models and the network services (in addition to basic multicast communication) that are required. Secondly, application developers

need to be aware of efforts underway to provide these services. Such efforts range in maturity from deployed commercial products to basic research efforts to evaluate feasibility.

Multicast-based applications and services will play an important role in the future of the Internet as continued multicast deployment encourages their use and development. It is important that developers be aware of the issues and solutions available--and especially of their limitations--in order to avoid protocols that negatively impact networks (thereby counter-acting the benefits of multicast) or wasting their efforts "re-inventing the wheel".

The hope is that by raising developers' awareness, we can adjust their expectations of finding solutions and lead them to successful, scalable, and "network-friendly" development efforts.

1.2 Focus and Scope

Our initial premise is that the multicast infrastructure is transparent to applications, so it is not directly relevant to this discussion. Our focus here is on multicast application protocol services, so this document explicitly avoids any discussion of multicast routing issues. We identify and describe the multicast-specific issues involved with developing applications.

We assume the reader has a general understanding of the mechanics of multicast, and in this respect we intend to compliment other introductory documents [BeauW, Maufer, Miller]. Since this is an introductory survey rather than a comprehensive examination, we refer readers to other multicast application requirements descriptions [RM, LSMA, Miller] for more detail.

In the remainder of this document we first define the term "IP multicast enabled network", the multicast infrastructure and essential multicast services. Next we describe the types of new functionality that multicast applications can enable and their requirements. We then examine the services that satisfy these requirements, the challenges they present, and provide a brief survey of the solutions available or under development. We wrap up with a discussion of application programming interfaces (APIs) for multicast services.

2. IP Multicast Enabled Network

An "IP multicast-enabled network" provides end-to-end services in the IP network infrastructure to allow any IP host to send datagrams to an IP multicast address that any number of other IP hosts widely dispersed can receive.

There are two kinds of multicast-enabled networks available. The first is based on the original multicast service model as defined in RFC 1112 [Deering]. In this model, a receiver simply joins the group and does not need to know the identity of the source(s). This model is known by a number of names including Internet Standard Multicast (ISM), Internet Traditional Multicast (ITM), Deering multicast, etc. The second kind of multicast modifies the original service model such that in addition to knowing the group address, a receiver must know the set of relevant sources. This type of multicast is called Source Specific Multicast (SSM) [SSM]. It becomes the application's responsibility to know what kind of multicast capability the network provides. Currently, the only way for an application to know the type of multicast is based on the group address. If the group is in the 232/8 range, the application should assume SSM is the service model. Otherwise, the application should assume source-generic multicast is the service model.

At the time of this writing, end-to-end "global" multicast service is not yet available, but the size of the "multicast-enabled" Internet is growing. Recent development and deployment of interdomain multicast routing protocols and multicast-friendly Internet exchanges have enabled peering between major ISPs. This, along with the increasing availability of compelling content, is spurring deployment and availability of the IP Multicast Enabled Network.

In the remainder of this document we assume that the multicast-enabled network is already ubiquitous. Since such a large and growing portion of the global Internet is IP multicast-enabled now, and many enterprise networks (intranets) are also, this perspective is relevant today.

2.1 Essential Protocol Components

An IP multicast enabled network requires two essential protocol components:

1) An IP host-based protocol to allow a receiver application to notify a local router(s) that it has joined the group, and initiate the data flow from all sender(s) within the scope

2) An IP router-based protocol to allow any routers with multicast group members (receivers) on their local networks to communicate with other routers to ensure that all datagrams sent to the group address are forwarded to all receivers within the intended scope

Ideally, these protocol components are transparent to multicast applications. However, there are two aspects of their functionality requirements that are worth mentioning specifically, since they affect application performance and design. These are the multicast application requirements for:

- Expedient Joins and Leaves
- Sends without a Join

2.1.1 Expedient Joins and Leaves

Some applications require expedient group joins and leaves, as their usability or functionality are sensitive to the latency involved with joining and leaving a group.

> Join Latency: The time it takes for data to begin flowing after an application issues a command to join a multicast group
>
> Leave Latency: The time it takes for data to stop flowing after an application issues a command to leave a multicast group [IGMPv2,IGMPv3]

For example, using distributed a/v as a multicast-based "television" must allow users to "channel surf"--changing channels frequently. Each channel change generates a group leave and group join, so delays in either will affect usability. In a sense, this is more of a user requirement than an application requirement.

The functionality of distributed interactive simulations [DIS] is often sensitive to join/leave latency. This is particularly true when trying to exchange information to represent fast moving objects that quickly enter and exit the scope of a simulated environment (e.g., low-flying, fast-moving aircraft). If the join latency is too long, the information provided may be old by the time it is received.

A fast leave phase is highly desirable both for effective congestion control mechanisms, to stop undesirable flows quickly, and for the network in general, to better filter unwanted traffic [Rizzo]. Applications cannot affect control over either join or leave latency, but are dependent on the multicast infrastructure to enable expedient operations. This is a basic multicast service requirement.

2.1.2 Sends without a Join

Applications that send to a multicast address should be able to start sending immediately, without having to join the destination group first. This is important for embedded devices and bursty-source applications with low-delay delivery requirements.

The current IGMP-based multicast host model and all current implementations allow senders to send to a group without joining it as a standard feature.

3. IP Multicast Application Taxonomy

With an IP multicast-enabled network available, some unique and powerful applications and application services are possible. "Multicast enables coordination - it is well suited to loosely coupled distributed systems (of people, servers, databases, processes, devices...)" [Estrin].

A "multicast application" is simply defined as any application that sends to and/or receives from an IP multicast address. These may or may not also reference IP unicast addresses, as we describe later.

What differentiates IP multicast applications from one-to-one unicast applications are the other sender and receiver relationships multicast enables. There are three general categories of multicast applications:

One-to-Many (1toM): A single host sending to two or more (n) receivers

Many-to-Many (MtoM): Any number of hosts sending to the same multicast group address, as well as receiving from it

Many-to-One (Mto1): Any number of receivers sending data back to a (source) sender via unicast or multicast

			Host 2->n ("many")			
			One-Way	Two-Way		
			A B	C	D	E
		I/O Operations	S(m) R(m)	S(m)/ R(m)	S(u)/ R(m)	S(m)/ R(u)
Host	1	S(m)	1toM	MtoM		
	2	R(m)	Mto1	MtoM		
	3	S(m)/R(m)	Mto1 1toM	MtoM		
	4	S(m)/R(u)			Mto1	
("one")	5	S(u)/R(m)				Mto1

Legend: S: "Send" (u): "unicast"
------ R: "Receive" (m): "multicast"

Table 1: Application types characterized by I/O relationships and destination address types (multicast or unicast)

Table 1 defines these application types in terms of the I/O relationships they represent. These categories are defined in terms of the combination of communication mechanisms they use. At the IP level, all multicast I/O is only 1toM or MtoM and unicast is always one-to-one (1to1). The Mto1 category, for example, refers to several possible combinations of IP-level relationships, including unicast. We created the Mto1 category to help differentiate between the many applications and services that use multicast.

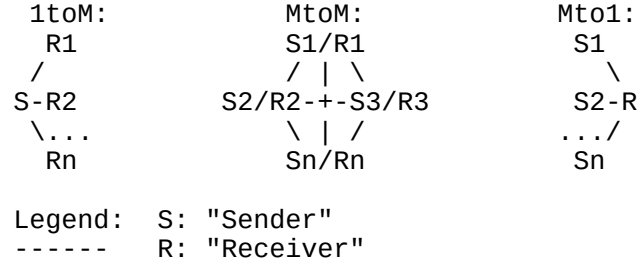

Figure 1: Generalization of the three application categories

Figure 1 illustrates the general model for each of the three multicast application categories. Again it is worth emphasizing that Mto1 is an artificial category defined by the application-level

relationship between sender(s) and receiver. At the IP-level, multicast does not provide an Mto1 I/O mechanism, since it does not allow senders to limit receivers, nor even know who they are. Receiver information and limitations are enabled at the application level, as required by the multicast application.

We describe each of these general application types in more detail and provide application examples of each in the sub-sections below. The list of examples is not comprehensive, but attempts to define the prominent multicast application and service types in each of the three general categories. We reference the items in these lists in the remainder of this document as we describe their specific service requirements, define the challenges they present, and reference solutions available or under development.

3.1 One-to-Many Applications

One-to-Many (1toM) applications have a single sender, and multiple simultaneous receivers. Entry B1 in Table 1 represents the classic 1toM relationship. Entry B3 differs only slightly, as the sender also acts as receiver (i.e., it has loopback enabled).

When people think of multicast, they most often think of broadcast-based multimedia applications: television (video) and radio (audio). This is a reasonable analogy and indeed these are significant multicast applications, but these are far from the extent of applications that multicast can enable. Audio/Video distribution represents a fraction of the multicast application possibilities, and most do not have analogs in today’s consumer broadcast industry.

a) Scheduled audio/video (a/v) distribution: Lectures, presentations, meetings, or any other type of scheduled event whose multimedia coverage could benefit an audience (i.e. television and radio "broadcasts"). One or more constant-bit-rate (CBR) datastreams and relatively high-bandwidth demands characterize these applications. When more than one datastream is present--as with an audio/video combination--the two are synchronized and one typically has a higher priority than the other(s). For example, in an a/v combination it is more important to ensure an intelligible audio stream, than perfect video.

b) Push media: News headlines, weather updates, sports scores, or other types of non-essential dynamic information. "Drip-feed", relatively low-bandwidth data characterize these applications.

c) File Distribution and Caching: Web site content, executable binaries, and other file-based updates sent to distributed end-user or replication/caching sites

d) Announcements: Network time, multicast session schedules, random numbers, keys, configuration updates, (scoped) network locality beacons, or other types of information that are commonly useful. Their bandwidth demands can vary, but generally they are very low bandwidth.

e) Monitoring: Stock prices, Sensor equipment (seismic activity, telemetry, meteorological or oceanic readings), security systems, manufacturing or other types of real-time information. Bandwidth demands vary with sample frequency and resolution, and may be either constant-bit-rate or bursty (if event-driven).

3.2 Many-to-Many Applications

In many-to-Many (MtoM) applications two or more of the receivers also act as senders. In other words, MtoM applications are characterized by two-way multicast communications.

The many-to-many capabilities of IP multicast enable the most unique and powerful applications. Each host running an MtoM application may receive data from multiple senders while it also sends data to all of them. As a result, many-to-many applications often present complex coordination and management challenges.

f) Multimedia Conferencing: Audio/Video and whiteboard comprise the classic conference application. Having multiple datastreams with different priorities characterizes this type of application. Co-ordination issues--such as determining who gets to talk when--complicate their development and usability. There are common heuristics and "rules of play", but no standards exist for managing conference group dynamics.

g) Synchronized Resources: Shared distributed databases of any type (schedules, directories, as well as traditional Information System databases).

h) Concurrent Processing: Distributed parallel processing.

i) Collaboration: Shared document editing.

j) Distance Learning: This is a one-to-many a/v distribution application with "upstream" capability that allows receivers to question the speaker(s).

k) Chat Groups: These are like text-based conferences, but may also provide simulated representations ("avatars") for each "speaker" in simulated environments.

l) Distributed Interactive Simulations [DIS]: Each object in a simulation multicasts descriptive information (e.g., telemetry) so all other objects can render the object, and interact as necessary. The bandwidth demands for these can be tremendous, as the number of objects and the resolution of descriptive information increases.

m) Multi-player Games: Many multi-player games are simply distributed interactive simulations, and may include chat group capabilities. Bandwidth usage can vary widely, although today's first-generation multi-player games attempt to minimize bandwidth usage to increase the target audience (many of whom still use dial-up modems).

n) Jam Sessions: Shared encoded audio (e.g., music). The bandwidth demands vary based on the encoding technique, sample rate, sample resolution, number of channels, etc.

3.3 Many-to-One Applications

Unlike the one-to-many and many-to-many application categories, the many-to-one (Mto1) category does not represent a communications mechanism at the IP layer. Mto1 applications have multiple senders and one (or a few) receiver(s), as defined by the application layer. Table 1 shows that Mto1 applications can be one-way or use a two-way request/response type protocol, where either senders or receiver(s) may generate the request. Figure 2 characterizes the I/O relationship possibilities in Mto1 applications:

Mto1 applications have many scaling issues. Too many simultaneous senders can potentially overwhelm receiver(s), a condition characterized as an "implosion problem". Another considerable scaling problem is the large amount of state in the network that having many multicast senders generates. This is largely transparent to applications and the effect may be diminished in the future with the use of bi-directional trees in multicast routing protocols, but nonetheless it should be considered by application designers.

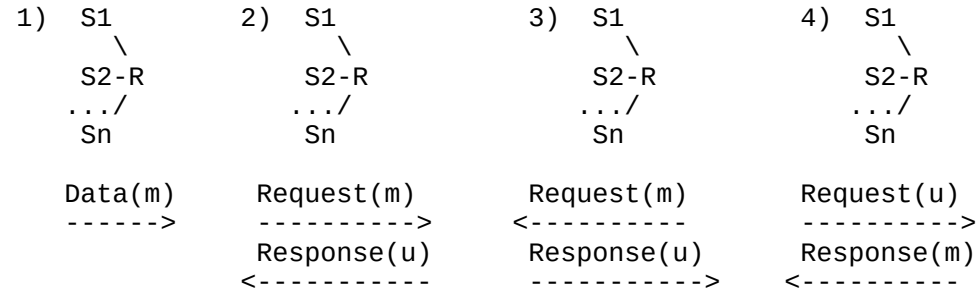

Figure 2: Characterization of Mto1 I/O possibilities

No standards yet exist for alternate and equivalent Mto1 application designs, but there are a number of possibilities to consider [HNRS]. Since the main advantage of using multicast in a Mto1 application is that senders need not know the receiver(s) unicast address(es), one alternative is for each receiver to advertise its unicast address via multicast. However, since this strategy still leaves the potential for implosion on each receiver, additional strategies may be needed to distribute the load. For example, it may be possible to increase the number of receivers (in a "flat" receiver topology) or establish localized receivers (in a "hierarchical" topology) as used in "local recovery" (section 5.3). Such methods have coordination issues, and since standard solutions have not yet been identified, Mto1 application developers should consider their alternatives carefully.

o) Resource Discovery: Service Location, for example, leverages IP Multicast to enable something like a "host anycasting service" capability [AnyCast]: A multicast receiver to send a query to a group address, to elicit responses from the closest host so they can satisfy the request. The responses might also contain information that allows the receiver to determine the most appropriate (e.g., closest) service provider to use.

   In Table 1, this application is entry D4. It is also illustrated in Figure 2 by possibility number 3. Alternately, the response could be to a multicast rather than unicast address, although technically that would make it an MtoM application type (this is how the Service Location Protocol [SLP] operates, when a user agent attempts to locate a directory agent).

p) Data Collection: This is the converse of a one-to-many "monitoring" application described earlier. In this case there may be any number of distributed "sensors" that send data to a data collection host. The sensors might send updates in response to a request from the data collector, or send

continuously at regular intervals, or send spontaneously when a pre-defined event occurs. Bandwidth demands can vary based on sample frequency and resolution.

This is illustrated in Table 1 by entries A1 and A3, the only difference being that A3 has a loopback interface. In Figure 2, this is possibility number 1. Since the number of receivers can easily be more than one, this is really an MtoM application.

q) Auctions: The "auctioneer" starts the bidding by describing whatever it is for sale (product or service or whatever), and receivers send their bids privately or publicly (i.e., to a unicast or multicast address).

This is possibility number 2 in Figure 2, and D5 in Table 1. The response could be sent to a multicast address, although technically that would make it an MtoM application.

r) Polling: The "pollster" sends out a question, and the "pollees" respond with answers. This is possibility number 2 in Figure 2, and could also be characterized as an MtoM application if the response is to a multicast address.

s) Jukebox: Allows near-on-demand a/v playback. Receivers use an "out-of-band" protocol mechanism (via web, email, unicast or multicast requests, etc.) to send their playback request into a scheduling queue [IMJ].

This is characterized by possibility number 4 in Figure 2, and entry D4 in Table 1. The initial unicast request is the only difference between this type of application and a typical 1toM. If that initial request were sent to a multicast address, this would effectively be an MtoM application.

t) Accounting: This is basically data collection but is worth separating since it is such an important application. In some multicast applications it is imperative to know information about each receiver, possibly in real-time. But such information can be overwhelming [MRM]. Current mechanisms, like RTCP (which is actually MtoM since it is multicast but could be made Mto1), use scaling techniques but trade-off information granularity. As a group grows the total amount of feedback is constant but each receiver sends less.

4. Common Multicast Service Requirements

Some multicast application service requirements are common to unicast network applications as well. We characterize two of them here--bandwidth and delay requirements.

4.1 Bandwidth Requirements

Figure 3 shows multicast applications approximate bandwidth requirements.

Unicast and multicast applications both need to design applications to adapt to the variability of network conditions. But as we describe in section 5.3, it is the need to accommodate multiple heterogeneous multicast receivers--with their diversity of bandwidth capacity and delivery delays--that presents the unique challenge for multicast applications to satisfy these requirements.

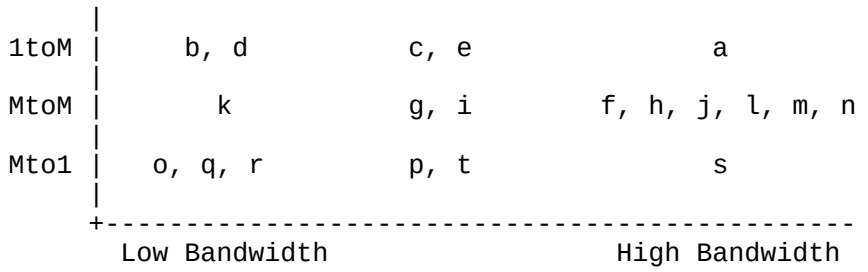

Figure 3: Bandwidth Requirements of applications

4.2 Delay Requirements

Aside from those with time-sensitive data (e.g., stock prices, and real-time monitoring information), most one-to-many applications have a high tolerance for delay and delay variance (jitter). Constant bit-rate (CBR) data--such as streaming media (audio/video)--are sensitive to jitter, but applications commonly counteract the effects by buffering data and delaying playback.

Most many-to-one and many-to-many multicast applications are intolerant of delays because they are bidirectional, interactive and request/response dependent. As a result, delays should be minimized, since they can adversely affect the application's usability.

This need to minimize delays is most evident in (two-way) conference applications, where users cannot converse effectively if the audio or video is delayed more than 500 milliseconds. For this and other

examples see Figure 4, which plots multicast applications on a (coarse) scale of sensitivity to delivery delays.

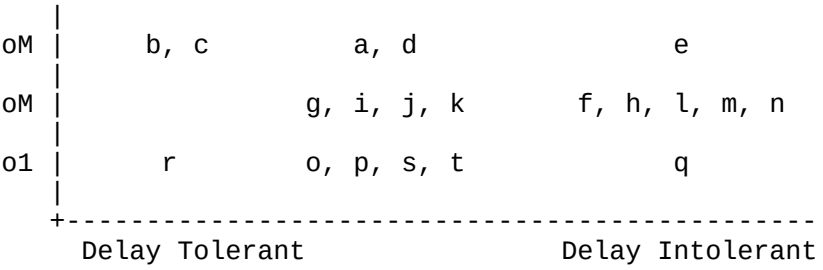

Figure 4: Delay tolerance of application types

For delay-intolerant multicast (or unicast) applications, quality of service (QoS) is the only option. IP networks currently provide only "best effort" delivery, so data are subject to variable router queuing delays and loss due to network congestion (router queue overflows). IP QoS standards do exist now [RSVP] and efforts to enable end-to-end QoS support in the Internet are underway [E2EQOS].

However, QoS support is an IP network infrastructure consideration. Although there are multicast-specific challenges for implementing QoS in the network for multicast flows, they are beyond the control of applications, so further discussion of the QoS protocols and services is beyond the scope of this document.

5. Unique Multicast Service Requirements

The three application categories described earlier are very general in nature. Within each category and even among each of the application types, specific application instances have a variety of application requirements. One-to-many application types are relatively simple to develop, but as we pointed out there are challenges involved with developing many-to-one and many-to-many applications. Some of these have requirements bandwidth and delay requirements, similar to unicast applications.

Multicast applications can be further differentiated from unicast applications and from each other by the services they require. In this section we provide a survey of the various services that have unique requirements for multicast applications.

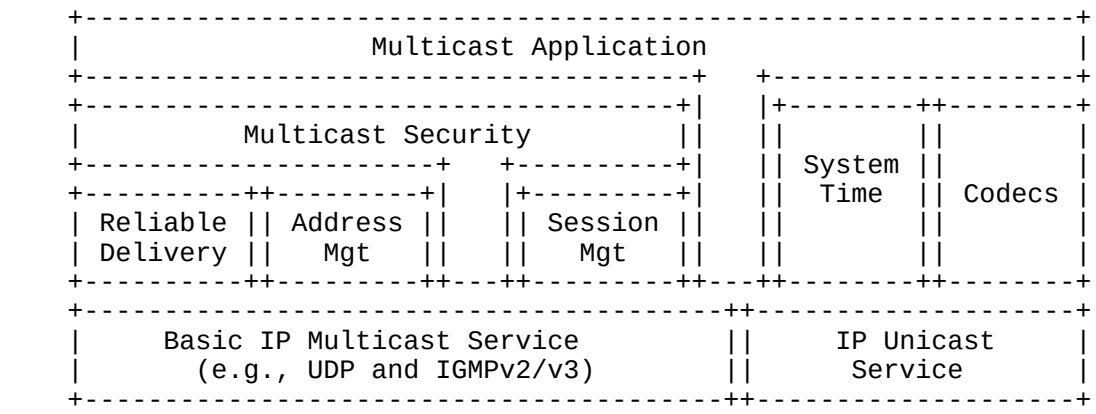

Figure 5: Multicast service requirements summary

Here's the list of multicast application service requirements:

Address Management - Selection and coordinated of address allocation. The need is to provide assurances against "address collision" and to provide address ownership.

Session Management - Perform application-layer services on top of multicast transport. These services depend heavily on the application but include functions like session advertisement, billing, group member monitoring, key distribution, etc.

Heterogeneous Receiver Support - Sending to receivers with a wide variety of bandwidth capacities, latency characteristics, and network congestion requires feedback to monitor receiver performance.

Reliable Data Delivery - Ensuring that all data sent is received by all receivers.

Security - Ensuring content privacy among dynamic multicast group memberships, and limiting senders.

Synchronized Play-Out - Allow multiple receivers to "replay" data received in synchronized fashion.

In the remainder of this section, we describe each of these application services in more detail, the challenges they present, and the status of standardized solutions.

5.1 Address Management

One of the first questions facing a multicast application developer is what multicast address to use. Multicast addresses are not assigned to individual hosts, assignments can change dynamically, and addresses sometimes have semantics of their own (e.g., Admin Scoping). Multicast applications require an address management service that provides address allocation or assignment queries. There are a number of ways for applications to learn about multicast addresses:

Hard-Coded: Software configuration, encoded in a binary executable, or burned into ROM in embedded devices. These applications typically reference IANA statically allocated multicast addresses (including relative addresses).

Advertised: Session announcements (as described in the next section), or via another "out-of-band" query or discovery protocol mechanism.

Algorithmically Derived: Using a programmatic algorithm to allocate a statistically random (unused) address.

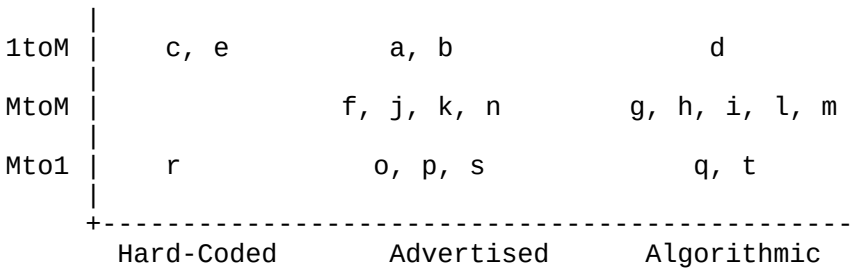

Figure 6: Multicast address usage for application types

In almost all cases, application designers should assume that multicast addresses are to be dynamic. Very little of the multicast address space is available for static assignment by IANA [MADDR]. Also, given the host-specific addressing available with SSM, Internet-wide, static address assignment is expected to be very rare.

5.2 Session Management

Session management is one of the most misunderstood services with respect to multicast. Most application developers assume that multicast will provide services like security, encryption, reliability, session advertisement, monitoring, billing, etc. In fact, multicast is simply a transport mechanism that provides end-

to-end delivery. All of the other services are application-layer services that must be provided by each particular application. Furthermore, in most cases there are not defined standards for how these functions should be provided. The particular functions are dependent on the particular needs of the application, and no single method (or standard) can be made to be sufficient for all cases.

While there are no generic solutions which provide all session management functions, there are some protocols and common techniques that provide support for some of the functions. Techniques for congestion control and heterogeneous receiver support are discussed in Section 5.3. Protocols for reliability are discussed in Section 5.4. Security considerations are discussed in Section 5.5.

With respect to session advertisement, there are a number of mechanisms for advertising sessions. One commonly used technique is to advertise sessions via the WWW. Users can join a group by clicking on URLs, and then having a response returned to the user that includes the group address and maybe information about group source(s). Another mechanism is the session description protocol [SDP]. It provides a format for representing information about sessions, but it does not provide the transport for dissemination of these session descriptions, nor does it provide address allocation and management. SDP only provides the syntax for describing session attributes.

SDP session descriptions may be conveyed publicly or privately by means of any number of transports including web (HTTP) and MIME encoded email. The session announcement protocol [SAP] is the de facto standard transport and many multicast-enabled applications currently use it. SAP limits distribution via multicast scoping, but the current protocol definition has scaling issues that need to be addressed. Specifically, the initialization latency for a session directory can be quite long, and it increases in proportion to the number of session announcements. This is to an extent a multicast infrastructure issue, however, as this level of protocol detail should be transparent to applications.

The session management service needs to:

- Advertise scheduled sessions
- Provide a query mechanism for retrieving information about session schedules

5.3 Heterogeneous Receiver Support

The Internet is a network of networks. IP's strength is its ability to enable seamless interoperability between hosts on disparate network media, the heterogeneous network.

When two hosts communicate via unicast--one-to-one--across an IP network, it is relatively easy for senders to adapt to varying network conditions. The Transmission Control Protocol (TCP) provides reliable data transport, and is the model of "network friendly" adaptability.

TCP receivers send acknowledgements back to the sender for data delivered. A TCP sender detects data loss from the data sent that is not acknowledged. When it detects data loss, TCP infers that there is network congestion or a low-bandwidth link, and adapts by throttling down its send rate [SlowStart].

User Datagram Protocol (UDP) does not enable a receiver feedback loop the way TCP does, since UDP does not provide reliable data delivery service. As a result, it also does not have a loss detection and adaptive congestion control mechanism as TCP does. However, it is possible for a unicast UDP application to enable similar adaptive algorithms to achieve the same result, or even improve on it.

A unicast UDP application that uses a feedback mechanism to detect data loss and adapt the send rate, can do so better than TCP. TCP automatically reduces the "congestion window" when data loss is detected, although the updated send rate may be slower than a CBR audio/video stream requires. When a UDP application detects loss, it can adapt the data itself to accommodate the lower send rate. For example, a UDP application can:

- Reduce the data resolution (e.g., send lower fidelity audio/video by reducing sample frequency or frame rate) to reduce data rate.

- Modify the data encoding to add redundant data (e.g., forward error correction) offset in time to avoid fate sharing. This could also be "layered", so a percentage of data loss will simply reduce fidelity rather than corrupt the data.

- Reduce the send rate of one datastream in order to favor another of higher priority (e.g., sacrifice video in order to ensure audio delivery).

- Send data at a lower rate (i.e., with a different encoding) on a separate multicast address and/or port number for high-loss receivers.

However, with multicast applications--one-to-many or many-to-many--which have multiple receivers, the feedback loop design needs modification. If all receivers return data loss reports simultaneously, the sender is easily overwhelmed in the storm of replies. This is known as the "implosion problem".

Another problem is that heterogeneous receiver capabilities can vary widely due to the wide range of (static) network media bandwidth capabilities and dynamically due to transient traffic conditions. If a sender adapts its send rate and data resolution based on the loss rate of its worst receiver(s), then it can only service the lowest common denominator. Hence, a single "crying baby" can spoil it for all other receivers.

Strategies exist for dealing with these heterogeneous receiver problems. Here are two examples:

Shared Learning - When loss is detected (i.e., a sequenced packet isn't received), a receiver starts a random timer. If it receives a data loss report sent by another receiver as it waits for the timer to expire, it stops the timer and does not send a report. Otherwise, it sends a report when the timer expires. The Real-Time Protocol and its feedback-loop counterpart Real-Time Control Protocol [RTP/RTCP] employ a strategy similar to this to keep feedback traffic to 5 percent or less than the overall session traffic. This technique was originally utilized in IGMP.

Local Recovery - Some receivers may be designated as local distribution points or "transcoders" that either re-send data locally (possibly via unicast) when loss is reported or they re-encode the data for lower bandwidth receivers before re-sending. No standards exist for these strategies, although "local recovery" is used by several reliable multicast protocols.

Adaptive multicast application design for heterogeneous receivers is still an active area of research. The fundamental requirements are to maximize application usability, while accommodating network conditions in a "network friendly" manner. In other words, congestion detection and avoidance are (at least) as important in protocol design as the user experience. The adaptive mechanisms must also be stable, so they do not adapt too quickly--changing encoding and rates based on too little information about what may be a transient condition--to avoid oscillation.

This "feedback loop" service necessary for support of heterogeneous receivers is not illustrated in the services summary in Figure 4, although it could be added alongside "Reliable Transport" and the others. This service could be implemented within an application or accessed externally, as provided by the operating system or a third party. See [HNRS] for a taxonomy of strategies for providing feedback for multicast, with the ultimate goal of developing a common multicast feedback protocol.

5.4 Reliable Data Delivery

Many of the multicast application examples in our list--like audio/video distribution--have loss-tolerant data content. In other words, the data content itself can remain useful even if some of it is lost. For example, audio might have a short gap or lower fidelity but will remain intelligible despite some data loss.

Other application examples--like caching and synchronized resources--require reliable data delivery. They deliver content that must be complete, unchanged, in sequence, and without duplicates. The "Loss Intolerant" column in Figure 7 shows a list of applications with this requirement, while the others can tolerate varying levels of data loss. The tolerance levels are typically determined by the nature of the data and the encoding in use.

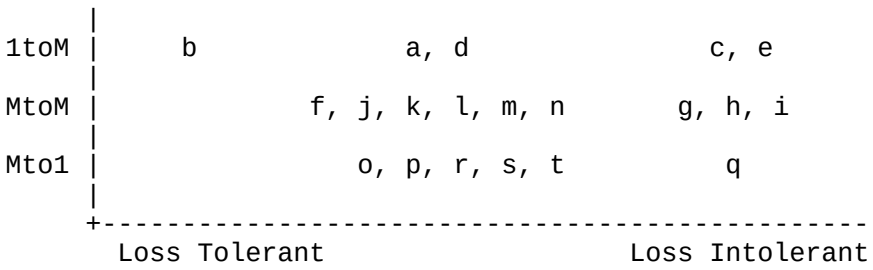

Figure 7: Reliability Requirements of Application types

Some of the challenges involved with enabling reliable multicast transport are the same as those of sending to heterogeneous receivers, and some solutions are similar also. For example, many reliable multicast transport protocols avoid the implosion problem by using negative acknowledgements (NAKs) from receivers to indicate what was lost. They also use "shared learning" whereby receivers listen to others' NAKs and then listen for the resulting retransmission of data, rather than requesting retransmission by sending a NAK themselves.

Although reliable delivery cannot change the data sent--except, perhaps, to use a loss-less data compression algorithm--they can use other adaptive techniques like sending redundant data, or adjusting the send rate.

Although many reliable multicast protocol implementations exist [Obraczka], and a few are already available in commercial products, none of them are standardized. Work is ongoing in the "Reliable Multicast" research group of the Internet Research Task Force [IRTF] to provide a better definition of the problem, the multicast transport requirements, and protocol mechanisms.

Scalability is the paramount concern, and it implies the general need for "network friendly" protocols that detect and avoid congestion as they provide reliable delivery. Other considerations are protocol robustness, support for "late joins", group management and security (which we discuss next).

The current consensus is that due to the wide variety of multicast application requirements--some of which are at odds--no single multicast transport will likely be appropriate for all applications. As a result, most believe that we will eventually standardize a number of reliable multicast protocols, rather than a single one [BULK, RMT].

5.5 Security

For any IP network application--unicast or multicast--security is necessary because networks comprise users with different levels of trust.

Network application security is challenging, even for unicast. And as the need for security increases--gauged by the risks of being without it--the challenges increase also. Security system complexity and overhead is commensurate with the protection it provides. "No one can guarantee 100% security. But we can work toward 100% risk acceptance...Strong cryptography can withstand targeted attacks up to a point--the point at which it becomes easier to get the information some other way...A good design starts with a threat model: what the system is designed to protect, from whom, and for how long." [Schneier]

Multicast applications are no different than unicast applications with respect to their need for security, and they require the same basic security services: user authentication, data integrity, data privacy and user privacy (anonymity). However, enabling security for

multicast applications is even more of a challenge than for unicast. Having multiple receivers makes a difference, as does their heterogeneity and the dynamic nature of multicast group memberships.

Multicast security requirements can include any combination of the following services:

Limiting Senders - Controlling who can send to group addresses

Limiting Receivers - Controlling who can receive

Limiting Access - Controlling who can "read" multicast content either by encrypting content or limiting receivers (which isn't possible yet)

Verifying Content - Ensuring that data originated from an authenticated sender and was not altered en route

Protecting Receiver Privacy - Controlling whether sender(s) or other receivers know receiver identity

Firewall Traversal - Proxying outgoing "join" requests through firewalls, allowing incoming or outgoing traffic through, and (possibly) authenticating receivers for filtering purposes and security [Finlayson].

This list is not comprehensive, but includes the most commonly needed security services. Different multicast applications and different application contexts can have very different needs with respect to these services, and others. Two main issues emerge, where the performance of current solutions leaves much to be desired [MSec].

Individual authentication - how is sender identity verified for each multicast datagram received?

Membership revocation - how is further group access disabled for group members that leave the group (e.g., encryption keys in their possession disabled)?

Performance is largely a factor when a user joins or leaves a group. For example, methods used to authenticate potential group members during joins or re-keying current members after a member leaves can involve significant processing and protocol overhead and result in significant delays that affect usability.

Like reliable multicast, secure multicast is also under investigation in the Internet Research Task Force [IRTF]. Protocol mechanisms for many of the most important of these services--such as limiting senders--have not yet been defined, let alone developed and deployed.

As is true for reliable multicast, the current consensus is that no single security protocol will satisfy the wide diversity of sometimes-contradictory requirements among multicast applications. Hence, multicast security will also likely require a number of different protocols.

5.6 Synchronized Play-Out

This refers to having all receivers simultaneously play-out the multicast data they received. This may be necessary for fairness--playing-out prices for auctions, or stock-prices--or to ensure synchronization with other receivers, such as when playing music.

Here is an analogy to illustrate: Imagine a multi-speaker stereo system that is wired throughout a home (via analog). With the stereo playing on all speaker sets, you will hear continuous music as you walk from room-to-room.

Now imagine a house full of multi-media and network enabled computer systems. Although they will all receive the same music datastream simultaneously via multicast, they will provide discontinuous music playback as you walk room-to-room.

To provide synchronized playback that would enable continuous music from room-to-room would require three things:

1) system clocks on all systems should be synchronized
2) datastreams must be framed with timestamps
3) you must know the playback latency of the multimedia hardware

The third of these is the most difficult to achieve at this time. Hardware and drivers don’t provide any mechanism for retrieving this information, although different audio and video devices have a wide-range of performance.

6. Service APIs

In some cases, the protocol services mentioned in this document can be enabled transparently by passive configuration mechanisms and "middleware". For example, it is conceivable that a UDP implementation could implicitly enable a reliable multicast protocol without the explicit interaction of the application.

Sometimes, however, applications need explicit access to these services for flexibility and control. For example, an adaptive application sending to a heterogeneous group of receivers using RTP may need to process RTCP reports from receivers in order to adapt accordingly (by throttling send rate or changing data encoders, for example) [RTP API]. Hence, there is often a need for service APIs that allow an application to qualify and initiate service requests, and receive event notifications. In Figure 5, the top edge of the box for each service effectively represents its API.

Network APIs generally reflect the protocols they support. Their functionality and argument values are a (varying) subset of protocol message types, header fields and values. Although some protocol details and actions may not be exposed in APIs--since many protocol mechanics need not be exposed--others are crucial to efficient and flexible application operation.

A more complete examination of the application services described in this document might also identify the protocol features that could be mapped to define a (generic) API definition for that service. APIs are often controversial, however. Not only are there many language differences, but it is also possible to create different APIs by exposing different levels of detail in trade-offs between flexibility and simplicity.

7. Security Considerations

See section 5.4

8. Acknowledgements

The authors would like to acknowledge and thank the following individuals for their helpful feedback: Ran Canetti, Brian Haberman, Eric A. Hall, Kenneth C. Miller, and Dave Thaler.

9. References

[AnyCast] Partridge, C., Mendez, T. and W. Milliken, "Host Anycasting Service", RFC 1546, November 1993.

[BeauW] B. Williamson, "Developing IP Multicast Networks, Volume I", (c) 2000 Cisco Press, Indianapolis IN, ISBN 1-57870-077-9.

[BULK] Whetten, B., Vicisano, L., Kermode, R., Handley, M., Floyd, S. and M. Luby, "Reliable Multicast Transport Building Blocks for One-to-Many Bulk-Data Transfer", RFC 3048, January 2001.

[Deering] Deering, S., "Host Extensions for IP Multicasting", STD 5, RFC 1112, August 1989.

[DIS] Pullen, J., Mytak, M. and C. Bouwens, "Limitations of Internet Protocol Suite for Distributed Simulation in the Large Multicast Environment", RFC 2502, February 1999.

[E2EQOS] Bernet, Y., Yavatkar, R., Ford, P., Baker, F., Zhang, L., Speer, M., Braden, R. and B. Davie, "Integrated Services Operation over Diffserv Networks", RFC 2998, November 2000.

[Estrin] D. Estrin, "Multicast: Enabler and Challenge", Caltech Earthlink Seminar Series, April 22, 1998.

[Finlayson] Finlayson, R., "IP Multicast and Firewalls", RFC 2588, May 1999.

[HNRS] Hofman, Nonnenmacher, Rosenberg, Schulzrinne, "A Taxonomy of Feedback for Multicast", June 1999, Work in Progress.

[IGMPv2] Fenner, B., "Internet Group Management Protocol, Version 2", RFC 2236, November 1997.

[IGMPv3] Cain, B., Deering, S., Kouvelas, I. and A. Thyagarajan, "Internet Group Management Protocol, Version 3", Work in Progress.

[IMJ] K. Almeroth and M. Ammar, "The Interactive Multimedia Jukebox (IMJ): A New Paradigm for the On-Demand Delivery of Audio/Video", Proceedings of the Seventh International World Wide Web Conference, Brisbane, AUSTRALIA, April 1998.

[IRTF] Weinrib, A. and J. Postel, "The IRTF Guidelines and Procedures", BCP 8, RFC 2014, January 1996.

[Kermode] Kermode, R., "MADCAP Multicast Scope Nesting State Option", RFC 2907, September 2000.

[LSMA] Bagnall, P., Briscoe, R. and A. Poppitt, "Taxonomy of Communication Requirements for Large-scale Multicast Applications", RFC 2729, December 1999.

[MADDR] Albanna, Z., Almeroth, K., Meyer, D. and M. Schipper, "IANA Guidelines for IPv4 Multicast Address Assignments", BCP 51, RFC 3171, August 2001.

[MASC] Estrin, D., Govindan, R., Handley, M., Kumar, S., Radoslavov, P. and D. Thaler, "The Multicast Address-Set Claim (MASC) Protocol", RFC 2909, September 2000.

[Maufer] T. Maufer, "Deploying IP Multicast in the Enterprise", (c) 1998 Prentice Hall, Upper Saddle River NJ ISBN 0-13-897687-2.

[Miller] C. K. Miller, "Multicast Networking and Applications", (c) 1999 Addison Wesley Longman, Reading MA ISBN 0-201-30979-3.

[MADCAP] Hanna, S., Patel, B. and M. Shah, "Multicast Address Dynamic Client Allocation Protocol (MADCAP)", RFC 2730, December 1999.

[MRM] K. Sarac, K. Almeroth, "Supporting Multicast Deployment Efforts: A Survey of Tools for Multicast Monitoring", Journal of High Speed Networking--Special Issue on Management of Multimedia Networking, March 2001

[MSec] Multicast Security (msec) IETF Working Group charter

[MZAP] Handley, M., Thaler, D. and R. Kermode, "Multicast-Scope Zone Announcement Protocol (MZAP)", RFC 2776, February 2000.

[Obraczka] K. Obraczka "Multicast Transport Mechanisms: A Survey and Taxonomy", IEEE Communications Magazine, Vol. 36 No. 1, January 1998.

[Rizzo] L. Rizzo, "Fast Group management in IGMP", HIPPARC 98 workshop, June 1998, UCL London http://www.iet.unipi.it/~luigi/hipparc98.ps.gz

[RM] Mankin, A., Romanow, A., Bradner, S. and V. Paxson, "IETF Criteria for Evaluating Reliable Multicast Transport and Application Protocols", RFC 2357, June 1998.

[RSVP] Wroclawski, J., "The Use of RSVP with IETF Integrated Services", RFC 2210, September 1997.

[RTP API] H. Schulzrinne, et al, "RTP Library API Specification," http://www.cs.columbia.edu/IRT/software/rtplib/rtplib-1.0a1/rtp_api.html

[RTP/RTCP] Schulzrinne, H., Casner, S., Frederick, R. and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 1889, January 1996.

[SAP] Handley, M., Perkins, C. and E. Whelan, "Session Announcement Protocol", RFC 2974, October 2000.

[SDP] Handley, M., and V. Jacobson, "SDP: Session Description Protocol", RFC 2327, April 1998.

[Schneier] B. Schneier, "Why Cryptography Is Harder Than It Looks", December 1996, http://www.counterpane.com/whycrypto.html

[SlowStart] Stevens, W., "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms", RFC 2001, January 1997.

[SLP] Veizades, J., Guttman, E., Perkins, C. and S. Kaplan, "Service Location Protocol", RFC 2165, June 1997.

[SSM] Holbrook, H. and B. Cain, "Specific Multicast for IP", Work in Progress.

10. Authors' Addresses

Bob Quinn
Celox Networks
2 Park Central Drive
Southborough, MA 01772

Phone: +1 508 305 7000
EMail: bquinn@celoxnetworks.com

Kevin Almeroth
Department of Computer Science
University of California
Santa Barbara, CA 93106-5110

Phone: +1 805 893 2777
EMail: almeroth@cs.ucsb.edu

11. Full Copyright Statement

Copyright (C) The Internet Society (2001). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.