Internet Engineering Task Force
INTERNET-DRAFT

Talpade and Ammar
Georgia Institute of Technology

March 20, 1996

<draft-talpade-ipatm-marsmcs-01.txt>

Multicast Server Architectures for MARS-based ATM multicasting.

Status of this Memo

Abstract

A mechanism to support the multicast needs of layer 3 protocols in general, and IP in particular, over UNI 3.0/3.1 based ATM networks has been described in draft-ietf-ipatm-ipmc-12.txt. Two basic approaches exist for the intra-subnet (intra-cluster) multicasting of IP packets. One makes use of a mesh of point to multipoint VCs (the 'VC Mesh' approach), while the other uses a shared point to multipoint tree rooted on a Multicast Server (MCS). This memo provides details on the design and implementation of an MCS, building on the core mechanisms defined in draft-ietf-ipatm-ipmc-12.txt.

1 Introduction

A solution to the problem of mapping layer 3 multicast service over the connection-oriented ATM service provided by UNI 3.0/3.1, has been presented in [GA96]. A Multicast Address Resolution Server (MARS) is used to maintain a mapping of layer 3 group addresses to ATM addresses in that architecture. It can be considered to be an extended analog of the ATM ARP Server introduced in RFC 1577 ([ML93]). Hosts in the ATM network use the MARS to resolve layer 3 multicast addresses into corresponding lists of ATM addresses of group members. Hosts keep the MARS informed when they need to join or leave a particular layer 3 group.

The MARS manages a "cluster" of ATM-attached endpoints. A "cluster" is defined as

"The set of ATM interfaces chosing to participate in direct ATM connections to achieve multicasting of AAL_SDUs between themselves."

In practice, a cluster is the set of endpoints that choose to use the same MARS to register their memberships and receive their updates from.

A sender in the cluster has two options for multicasting data to the group members. It can either get the list of ATM addresses constituting the group from the MARS, set up a point-to-multipoint virtual circuit (VC) with the group members as leaves, and then proceed to send data out on it. Alternatively, the source can make use of a proxy Multicast Server (MCS). The source transmits data to such an MCS, which in turn uses a point-to-multipoint VC to get the data to the group members.

The MCS approach has been briefly introduced in [GA96]. This memo presents a detailed description of MCS architecture and protocols. We assume an understanding of the IP multicasting over UNI 3.0/3.1 ATM network concepts described in [GA96], and access to it. This document is organized as follows. Section 2 presents interactions with the local UNI 3.0/3.1 signalling entity that are used later in the document and have been originally described in [GA96]. Section 3 presents an MCS architecture, along with a description of its interactions with the MARS. Section 4 describes the working of an MCS. The possibility of using multiple MCSs for the same layer 3 group, and the mechanism needed to support such usage, is described in section 5. Section 6 examines some unresolved issues and summarizes the document. A comparison of the VC Mesh approach and the MCS approach is presented in Appendix A.

2 Interaction with the local UNI 3.0/3.1 signalling entity

The following generic signalling functions are presumed to be available to local AAL Users:

L_CALL_RQ - Establish a unicast VC to a specific endpoint.
L_MULTI_RQ - Establish multicast VC to a specific endpoint.
L_MULTI_ADD - Add new leaf node to previously established VC.
L_MULTI_DROP - Remove specific leaf node from established VC.
L_RELEASE - Release unicast VC, or all Leaves of a multicast VC.

The following indications are assumed to be available to AAL Users, generated by by the local UNI 3.0/3.1 signalling entity:

L_ACK - Succesful completion of a local request.
L_REMOTE_CALL - A new VC has been established to the AAL User.
ERR_L_RQFAILED - A remote ATM endpoint rejected an L_CALL_RQ,
L_MULTI_RQ, or L_MULTI_ADD.
ERR_L_DROP - A remote ATM endpoint dropped off an existing VC.
ERR_L_RELEASE - An existing VC was terminated.

3 MCS Architecture

The MCS acts as a proxy server which multicasts data received from a source to the group members in the cluster. All multicast sources transmitting to an MCS-based group send the data to the specified MCS. The MCS then forwards the data over a point to multipoint VC that it maintains to group members in the cluster. Each multicast source thus maintains a single point-to-point VC to the designated MCS for the group. The designated MCS terminates one point-to-point VC from each cluster member that is multicasting to the layer 3 group. Each group member is the leaf of the point-to-multipoint VC originating from the MCS.

A brief introduction to possible MCS architectures has been presented in [GA96]. The main contribution of that document concerning the MCS approach is the specification of the MARS interaction with the MCS. The next section lists control messages exchanged by the MARS and MCS.

3.1 Control Messages exchanged by the MCS and the MARS

The following control messages are exchanged by the MARS and the MCS.

operation code	Control Message
1	MARS_REQUEST
2	MARS_MULTI
3	MARS_MSERV
6	MARS_NAK
7	MARS_UNSERV
8	MARS_SJOIN
9	MARS_SLEAVE
12	MARS_REDIRECT_MAP

MARS_MSERV and MARS_UNSERV are identical in format to the MARS_JOIN message. MARS_SJOIN and MARS_SLEAVE are also identical in format to MARS_JOIN. As such, their formats and those of MARS_REQUEST, MARS_MULTI, MARS_NAK and MARS_REDIRECT_MAP are described in [GA96]. We describe their usage in section 4. All control messages are LLC/SNAP encapsulated as described in section 4.2 of [GA96]. (The "mar$" notation used in this document is borrowed from [GA96], and indicates a specific field in the control message.) Data messages are reflected without any modification by the MCS.

3.2 Association with a layer 3 group

The simplest MCS architecture involves taking incoming AAL_SDUs from the multicast sources and sending them out over the point-to-multipoint VC to the group members. The MCS can service just one layer 3 group using this design, as it has no way of distinguishing between traffic destined for different groups. So each layer 3 MCS-supported group will have its own designated MCS.

However it is desirable in terms of saving resources to utilize the same MCS to support multiple groups. This can be done by adding minimal layer 3 specific processing into the MCS. The MCS can now look inside the received AAL_SDUs and determine which layer 3 group they are destined for. A single instance of such an MCS could register its ATM address with the MARS for multiple layer 3 groups, and manage multiple point-to-multipoint VCs, one for each group. We include this capability in our MCS architecture. We also include the capability of having multiple MCSs per group (section 5).

4 Working of MCS

An MCS MUST NOT share its ATM address with any other cluster member (MARS or otherwise). However, it may share the same physical ATM interface (even with other MCSs or the MARS), provided that each logical entity has a different ATM address. This section describes the working of MCS and its interactions with the MARS and other cluster members.

4.1 Usage of MARS_MSERV and MARS_UNSERV

4.1.1 Registration (and deregistration) with the MARS

The ATM address of the MARS MUST be known to the MCS by out-of-band means at startup. One possible way to do this is for the network administrator to specify the MARS address at command line while invoking the MCS. On startup, the MCS MUST open a point-to-point control VC (MARS_VC) with the MARS. All traffic from the MCS to the MARS MUST be carried over the MARS_VC. The MCS MUST register with the MARS using the MARS_MSERV message on startup. To register, a MARS_MSERV MUST be sent by the MCS to the MARS over the MARS_VC. On receiving this MARS_MSERV, the MARS adds the MCS to the ServerControlVC. The ServerControlVC is maintained by the MARS with all MCSs as leaves, and is used to disseminate general control messages to all the MCSs. The MCS MUST terminate this VC, and MUST expect a copy of the MCS registration MARS_MSERV on the MARS_VC from the MARS.

An MCS can deregister by sending a MARS_UNSERV to the MARS. A copy of this MARS_UNSERV MUST be expected back from the MARS. The MCS will then be dropped from the ServerControlVC.

No protocol specific group addresses are included in MCS registration MARS_MSERV and MARS_UNSERV. The mar$flags.register bit MUST be set, the mar$cmi field MUST be set to zero, the mar$flags.sequence field MUST be set to zero, the source ATM address MUST be included and a null source protocol address MAY be specified in these MARS_MSERV and MARS_UNSERV. All other fields are set as described in section 5.2.1 of [GA96] (the MCS can be considered to be a cluster member while reading that section). It MUST keep retransmitting (section 4.1.3) the MARS_MSERV/MARS_UNSERV over the MARS_VC until it receives a copy back.

In case of failure to open the MARS_VC, or error on it, the reconnection procedure outlined in section 4.5.2 is to be followed.

4.1.2 Registration (and deregistration) of layer 3 groups

The MCS can register with the MARS to support particular group(s). To register a group X, a MARS_MSERV with a <min, max> pair of <X, X> MUST be sent to the MARS. The MCS MUST expect a copy of the MARS_MSERV back from the MARS. The retransmission strategy outlined in section 4.1.3 is to be followed if no copy is received. Multiple groups can be supported by sending a separate MARS_MSERV for each group.

The MCS MUST similarly use MARS_UNSERV if it wants to withdraw support for a specific layer 3 group. A copy of the group MARS_UNSERV MUST be received, failing which the retransmission strategy in section 4.1.3 is to be followed.

The mar$flags.register bit MUST be reset and the mar$flags.sequence field MUST be set to zero in the group MARS_MSERV and MARS_UNSERV. All other fields are set as described in section 5.2.1 of [GA96] (the MCS can be considered to be a cluster member when reading that section).

4.1.3 Retransmission of MARS_MSERV and MARS_UNSERV

Transient problems may cause loss of control messages. The MCS needs to retransmit MARS_MSERV/MARS_UNSERV at regular intervals when it does not receive a copy back from the MARS. This interval should be no shorter than 5 seconds, and a default value of 10 seconds is recommended. A maximum of 5 retransmissions are permitted before a failure is logged. This MUST be considered a MARS failure, which SHOULD result in the MARS reconnection mechanism described in section 4.5.2.

A "copy" is defined as a received message with the following fields matching the previously transmitted MARS_MSERV/MARS_UNSERV:

- mar$op
- mar$flags.register
- mar$pnum
- Source ATM address
- first <min, max> pair

In addition, a valid copy MUST have the following field values:

- mar$flags.punched = 0
- mar$flags.copy = 1

There MUST be only one MARS_MSERV or MARS_UNSERV outstanding at a time.

4.1.4 Processing of MARS_MSERV and MARS_UNSERV

The MARS transmits copies of group MARS_MSERV and MARS_UNSERV on the ServerControlVC. So they are also received by MCSs other than the originating one. This section discusses the processing of these messages by the other MCSs.

If a MARS_MSERV is seen that refers to a layer 3 group not supported by the MCS, it MUST be used to track the Server Sequence Number (section 4.5.1) and then silently dropped.

If a MARS_MSERV is seen that refers to a layer 3 group supported by the MCS, the MCS learns of the existence of another MCS supporting the same group. We incorporate this possibility (of multiple MCSs per group) in this version of the MCS approach and discuss it in section 5.

4.2 Usage of MARS_REQUEST and MARS_MULTI

After the MCS registers to support a layer 3 group, it uses MARS_REQUEST and MARS_MULTI to obtain information about group membership from the MARS. These messages are also used during the revalidation phase (section 4.5) and when no outgoing VC exists for a received layer 3 packet (section 4.3).

On registering to support a particular layer 3 group, the MCS MUST send a MARS_REQUEST to the MARS. The mechanism to retrieve group membership and the format of MARS_REQUEST and MARS_MULTI is described in section 5.1.1 and 5.1.2 of [GA96] respectively. The MCS MUST use this mechanism for sending (and retransmitting) the MARS_REQUEST and processing the returned MARS_MULTI(/s). The MARS_MULTI MUST be received correctly, and the MCS MUST use it to initialize its knowledge of group membership.

As is described in section 5.1.1, the MCS is informed at startup whether it is a primary or secondary MCS, and whether the MCSs share senders or receivers of the group. A primary MCS can also be informed at startup of secondary MCSs which will be invoked along with it. An MCS which is designated as the primary, MUST continue with the VC establishment procedure as described later in this subsection. The next paragraph does not apply to it.

If receivers are being shared (section 5.1.1), a new secondary MCS

MUST NOT continue with the point-to-multipoint VC establishment. The new secondary MCS proceeds as described in section 5.4. If the MCSs share the senders to the group, the new secondary MCS opens the point-to-multipoint VC as described below. However, it MUST NOT forward traffic received from any senders onto this VC until it receives the list of senders allocated to it (see section 5.4 for more detail).

On successful reception of a MARS_MULTI, the MCS MUST attempt to open the outgoing point-to-multipoint VC using the mechanism described in section 5.1.3 of [GA96], if any group members exist. The MCS however MUST start transmitting data on this VC after it has opened it successfully with at least one of the group members as a leaf, and after it has attempted to add all the group members at least once.

4.3 Usage of outgoing point-to-multipoint VC

Cluster members which are sources for MCS-supported layer 3 groups send (encapsulated) layer 3 packets to the designated MCSs. An MCS, on receiving them from cluster members, has to send them out over the specific point-to-multipoint VC for that layer 3 group. This VC is setup as described in the previous section. However, it is possible that no group members currently exist, thus causing no VC to be setup. So an MCS may have no outgoing VC to forward received layer 3 packets on, in which case it MUST initiate the MARS_REQUEST and MARS_MULTI sequence described in the previous section. This new MARS_MULTI could contain new members, whose MARS_SJOINs may have been not received by the MCS (and the loss not detected due to absence of traffic on the ServerControlVC).

If an MCS learns that there are no group members (MARS_NAK received from MARS), it MUST delay sending out a new MARS_REQUEST for that group for a period no less than 5 seconds and no more than 10 seconds.

Layer 3 packets received from cluster members, while no outgoing point-to-multipoint VC exists for that group, MUST be silently dropped after following the guidelines in the previous paragraphs. This might result in some layer 3 packets being lost until the VC is setup.

Each outgoing point-to-multipoint has a revalidate flag associated with it. This flag MUST be checked whenever a layer 3 packet is sent out on that VC. No action is taken if it is not set. If it is set, the packet is sent out, the revalidation procedure (section 4.5.3) MUST be initiated for this group, and the flag MUST be reset.

In case of error on a point-to-multipoint VC, the MCS MUST initiate revalidation procedures for that VC as described in section 4.5.3.

Once a point-to-multipoint VC has been setup for a particular layer 3 group, the MCS MUST hold the VC open and mark it as the outgoing path for any subsequent layer 3 packets being sent for that group address. A point-to-multipoint VC MUST NOT have an activity timer associated with it. It is to remain up at all times, unless the MCS explicitly stops supporting that layer 3 group, or no more leaves exist on the VC which causes it to be shut down. The VC is kept up inspite of non-existent traffic to reduce the delay suffered by MCS supported groups. If the VC were to be shut down on absence of traffic, the VC reestablishment procedure (needed when new traffic for the layer 3 group appears) would further increase the end-to-end latency, which can be potentially higher than the VC mesh approach anyway as two VCs need to be setup in the MCS case (one from source to MCS, second from MCS to group) as opposed to only one (from source to group) in the VC Mesh approach. This approach of keeping the VC from the MCS open even in the absense of traffic is experimental. A decision either way can only be made after gaining experience (either through implementation or simulation) about the implications of keeping the VC open.

If the MCS supports multiple layer 3 groups, each data AAL_SDU MUST be examined for determining its recipient group, before being forwarded onto the appropriate outgoing point-to-multipoint VC.

4.3.1 Group member dropping off a point-to-multipoint VC

AN ERR_L_DROP may be received during the lifetime of a point-to-multipoint VC indicating that a leaf node has terminated its participation at the ATM level. The ATM endpoint associated with the ERR_L_DROP MUST be removed from the locally held set associated with the VC. The revalidate flag on the VC MUST be set after a random interval of 1 through 10 seconds.

If an ERR_L_RELEASE is received for a VC, then the entire set is cleared and the VC considered to be completely shutdown. A new VC for this layer 3 group will be established only on reception of new traffic for the group (as described in section 4.3).

4.4 Processing of MARS_SJOIN and MARS_SLEAVE

The MARS transmits equivalent MARS_SJOIN/MARS_SLEAVE on the ServerControlVC when it receives MARS_JOIN/MARS_LEAVE from cluster members. The MCSs keep track of group membership updates through these messages. The format of these messages are identical to MARS_JOIN and MARS_LEAVE, which are described in section 5.2.1 of [GA96]. It is sufficient to note here that these messages carry the ATM address of the node joining/leaving the group(/s), the group(/s) being joined or left, and a Server Sequence Number from MARS.

When a MARS_SJOIN is seen which refers to (or encompasses) a layer 3 group (or groups) supported by the MCS, the following action MUST be taken. The new member's ATM address is extracted from the MARS_SJOIN. An L_MULTI_ADD is issued for the new member for each of those referred groups which have an outgoing point-to-multipoint VC. An L_MULTI_RQ is issued for the new member for each of those refered groups which have no outgoing VCs.

When a MARS_SLEAVE is seen that refers to (or encompasses) a layer 3 group (or groups) supported by the MCS, the following action MUST be taken. The leaving member's ATM address is extracted. An L_MULTI_DROP is issued for the member for each of the refered groups which have an outgoing point-to-multipoint VC.

There is a possibility of the above requests (L_MULTI_RQ or L_MULTI_ADD or L_MULTI_DROP) failing. The UNI 3.0/3.1 failure cause must be returned in the ERR_L_RQFAILED signal from the local signalling entity to the AAL User. If the failure cause is not 49 (Quality of Service unavailable), 51 (user cell rate not available - UNI 3.0), 37 (user cell rate not available - UNI 3.1), or 41 (Temporary failure), the endpoint's ATM address is dropped from the locally held view of the group by the MCS. Otherwise, the request MUST be re-attempted with increasing delay (initial value between 5 to 10 seconds, with delay value doubling after each attempt) until it either succeeds or the multipoint VC is released or a MARS_SLEAVE is received for that group member. If the VC is open, traffic on the VC MUST continue during these attempts.

MARS_SJOIN and MARS_SLEAVE are processed differently if multiple MCSs share the members of the same layer 3 group (section 5.4). MARS_SJOIN and MARS_SLEAVE that do not refer to (or encompass) supported groups MUST be used to track the Server Sequence Number (section 4.5.1), but are otherwise ignored.

4.5 Revalidation Procedures

The MCS has to initiate revalidation procedures in case of certain failures or errors.

4.5.1 Server Sequence Number

The MCS needs to track the Server Sequence Number (SSN) in the messages received on the ServerControlVC from the MARS. It is carried in the mar$msn of all messages (except MARS_NAK) sent by the MARS to MCSs. A jump in SSN implies that the MCS missed the previous message(/s) sent by the MARS. The MCS then sets the revalidate flag on all outgoing point-to-multipoint VCs after a random delay of between 1 and 10 seconds, to avoid all MCSs inundating the MARS simultaneously in case of a more general failure.

The only exception to the rule is if a sequence number is detected during the establishment of a new group's VC (i.e. a MARS_MULTI was correctly received, but its mar$msn indicated that some previous MARS traffic had been missed on ClusterControlVC). In this case every open VC, EXCEPT the one just being established, MUST have its revalidate flag set at some random interval between 1 and 10 seconds from the time the jump in SSN was detected. (The VC being established is considered already validated in this case).

Each MCS keeps its own 32 bit MCS Sequence Number (MSN) to track the SSN. Whenever a message is received that carries a mar$msn field, the following processing is performed:

```
Seq.diff = mar$msn - MSN

mar$msn -> MSN

(.... process MARS message ....)

if ((Seq.diff != 1) && (Seq.diff != 0))
      then (.... revalidate group membership information ....)
```

The mar$msn value in an individual MARS_MULTI is not used to update the MSN until all parts of the MARS_MULTI (if > 1) have arrived. (If the mar$msn changes during reception of a MARS_MULTI series, the MARS_MULTI is discarded as described in section 5.1.1 of [GA96]).

The MCS sets its MSN to zero on startup. It gets the current value of SSN when it receives the copy of the registration MARS_MSERV back from the MARS.

4.5.2 Reconnecting to the MARS

The MCSs are assumed to have been configured with the ATM address of at least one MARS at startup. MCSs MAY choose to maintain a table of ATM addresses, each address representing alternative MARS which will be contacted in case of failure of the previous one. This table is assumed to be ordered in descending order of preference.

An MCS will decide that it has problems communicating with a MARS if:

- It fails to establish a point-to-point VC with the MARS.
- MARS_REQUEST generates no response (no MARS_MULTI or MARS_NAK returned).
- ServerControlVC fails.
- MARS_MSERV or MARS_UNSERV do not result in their respective copies being received.

(reconnection as in section 5.4 in [GA96]).

4.5.3 Revalidating a point-to-multipoint VC

The revalidation flag associated with a point-to-multipoint VC is checked when a layer 3 packet is to be sent out on the VC. Revalidation procedures MUST be initiated for a point-to-multipoint VC that has its revalidate flag set when a layer 3 packet is being sent out on it. Thus more active groups get revalidated faster than less active ones. The revalidation process MUST NOT result in disruption of normal traffic on the VC being revalidated.

The revalidation procedure is as follows. The MCS reissues a MARS_REQUEST for the VC being revalidated. The returned set of members is compared with the locally held set; L_MULTI_ADDs MUST be issued for new members, and L_MULTI_DROPs MUST be issued for non-existent ones. The revalidate flag MUST be reset for the VC.

5 Multiple MCSs for a layer 3 group

Having a single MCS for a layer 3 group can cause it to become a single point of failure and a bottleneck for groups with large numbers of senders. It is thus desirable to introduce a level of fault tolerance by having multiple MCS per group. These MCSs also share the group load amongst themselves.

5.1 Outline

The MCSs need to share the group load amongst themselves without involving either the MARS or the other cluster members in the allocation mechanism. In other words, use of multiple MCSs is transparent to both the MARS and the cluster members.

There are three possible approaches for multiple MCSs to share the group load. They can share either the senders (to the group) or the group members (receivers) or both. We do not consider sharing of both senders and receivers simultaneously as it becomes increasingly complex for the MCSs to ensure that all receivers get the data from all the senders. This is a topic for further research.

The bottleneck effect caused by the MCS approach is influenced significantly by the multiplexer effect of the MCS (it receives data from many sender VCs, but transmits it out on just one outgoing point-to-multipoint VC). Multiplexing performance is not influenced by the number of receivers, which are only leaves of the point-to-multipoint VC. Thus sharing of senders is more effective than sharing of receivers from the data latency perspective.

Each MCS needs to terminate a VC from each sender irrespective of the option selected. This is because the MARS sends the complete list of MCSs to the senders (in MARS_MULTI), causing the senders to connect to each MCS through a point-to-multipoint VC. Thus VC usage is better when sharing receivers as each receiver terminates a VC only from that MCS which serves it. This is not true when senders are shared since each receiver terminates a VC from all the MCSs, with each MCS actively dropping data received from senders that are not supported by it.

In the VC Mesh approach, the only data ordering is that enforced by each sender on its data, which is received in the same order at all group members. Data ordering is not maintained for all data received by the group members. However, use of one MCS enforces the same ordering on all data received at all the group members. This ordering again disappears when multiple MCSs are used, irrespective of the choice of sharing senders or receivers. Thus the data ordering obtained when using multiple MCSs is similar to that obtained in the VC Mesh case (i.e., data from a sender received in same order at all group members).

5.1.1 Guidelines for using multiple MCSs per group

To reduce the complexity of the allocation mechamism, the following guidelines need to be followed. These guidelines need to be enforced by out-of-band means which are not specified in this document and can be implementation dependent.

- The first MCS to be started up for the group is designated as the primary MCS. It is informed of this fact at startup. Other MCSs are considered to be secondary MCSs, and are informed of this at startup. The primary MCS MUST be the first MCS to register to support the group with the MARS.

- The choice of sharing senders or receivers is made by out-of-band means (the MCSs do not communicate with each other to arrive at this decision). All MCSs are informed of this choice when they are started up. Once a choice is made, it is not changed until the last MCS supporting this choice ceases its support of the group. The new MCSs that are brought up to support the group can now be configured to support a different choice if desired.

5.2 Discussion of Multiple MCSs in operation

The allocation mechanism is used by MCSs supporting the same group to keep themselves updated about the specific senders/receivers to be supported by each MCS. It is also used by the MCSs to learn the identity of other MCSs supporting the same group. This section gives an overview of the operation of this mechanism.

An MCS registers to support a group with the MARS on startup. The MCS is told whether it is a primary or secondary MCS at startup. A primary MCS supports the entire group by itself when no other MCS exists. The MARS may have to transition the group (using MARS_MIGRATE messages) from being VC Mesh supported to being MCS supported, if the group is already active when the primary MCS goes online.

The primary MCS communicates with the secondary MCSs through point-to-point VCs. It first communicates with a secondary MCS when the secondary MCS starts up. If the secondary MCS does not agree with the primary MCS about sharing senders or receivers on startup, the secondary MCS aborts after deregistering from the MARS. Otherwise, it is informed by the primary MCS about the senders/receivers that are to be supported by it. If senders are being shared, the MCS forwards data from only those senders supported by it to all receivers. Data received from other senders is dropped. If receivers are being shared, the MCS opens a point-to-multipoint VC to only those receivers supported by it, and forwards data from all senders on the VC.

The primary MCS also provides the secondary MCSs with a list of other MCSs supporting the same group. This list is consistent across all the MCSs supporting the same group, and is updated by the primary MCS at a regular interval. Secondary MCSs use this list to select a new primary MCS in case of failure of the existing one. The newly elected primary MCS reallocates the senders/receivers amongst the existing MCSs. The allocation mechanism also provides for reallocation of senders/receivers that were being supported by a failed secondary MCS amongst the remaining MCSs. This reallocation is done by the current primary MCS. The primary MCS also allocates any new group members to an existing MCS. Thus the primary MCS is the centralized decision maker, while information is distributed across all the MCSs supporting the group.

5.3 Inter-MCS control messages

MCSs supporting the same layer 3 group exchange the following control messages in the allocation mechanism.

1. MCS_MULTI: This is identical to the MARS_MULTI in format. It is used by the primary MCS to indicate the set of senders/receivers that are to be supported by a secondary MCS. It also contains a flag which indicates whether senders or receivers are being shared by the MCSs.

2. MCS_LIST: This is identical to the MARS_MULTI in format. This is sent by the primary MCS to each of the secondary MCSs at 30 second intervals. It indicates the current set of MCSs supporting the group. The order of MCSs listed is the same in the MCS_LIST sent to each secondary MCS. Each secondary MCS uses the MCS_LIST messages to maintain a local list of all the MCSs supporting the group. This list is identical on all MCSs.

3. MCS_REQUEST: This is identical to the MARS_REQUEST in format. It is sent by a secondary MCS to the primary MCS to request the set of receivers/senders supported by the secondary MCS.

4. MCS_ALIVE: Secondary MCSs send this to the primary MCS at 30 second intervals.

5. MCS_MULTI_ACK: A secondary MCS sends this to the primary MCS to acknowledge error-free reception of an MCS_MULTI.

6. MCS_SELECT: Existing MCSs send this to another MCS, if that MCS registers to support their group during the primary MCS selection procedure. i.e. the existing MCSs receive a copy of the new MCS's group registration MARS_MSERV from the MARS.

5.4 The Allocation Mechanism

The primary MCS learns of the existence of other MCSs supporting the same group on receiving copies of group registration MARS_MSERV on the Server Control VC from the MARS. If such a MARS_MSERV is received which has originated from another cluster member, the primary MCS MUST open a point-to-point VC (MCS VC) to this new MCS. The new MCS is now one of the secondary MCSs for the group. All traffic between the primary MCS and the secondary MCS MUST traverse this VC. The primary MCS MUST send an MCS_MULTI indicating the set of senders/receivers to be supported by the secondary MCS. This MCS_MULTI also indicates whether senders or receivers are being shared. It MUST send an MCS_MULTI to a secondary MCS whenever there is a change in the set being supported by that secondary MCS. If no MCS_MULTI_ACK is received from the secondary MCS within 5 seconds of sending it an MCS_MULTI, the primary MCS MUST retransmit the MCS_MULTI at 5 second intervals (section 5.5.3). The primary MCS MUST send MCS_LIST messages to each secondary MCS at 30 second intervals. A secondary MCS maintains its knowledge of all the MCSs supporting the group using the MCS_LIST messages.

The secondary MCS MUST terminate the MCS VC. It MUST send an MCS_ALIVE message to the primary MCS at 30 second intervals. On receiving the MCS_MULTI from the primary MCS, it first checks to see if the choice of sharing senders or receivers matches that of the primary MCS (as indicated in the MCS_MULTI). If not, the secondary MCS MUST abort with an appropriate error message, after degregistering from the MARS. Otherwise, it MUST update the set of senders/receivers supported by it. This set is initialized on the basis of the first MCS_MULTI received by it. The MCS_MULTI are processed similar to MARS_MULTI messages, with errored MCS_MULTI being discarded, and a MCS_REQUEST being sent to the primary MCS in that case. The MCS_REQUEST is retransmitted at 30 second intervals until either an error-free MCS_MULTI is received, or the secondary MCS declares the primary MCS as failed (section 5.5.1). The secondary MCS MUST send an MCS_MULTI_ACK to the primary MCS on error-free reception of an MCS_MULTI.

On receiving an MCS_MULTI as indicated above, if receivers are being shared, L_MULTI_ADDs MUST be issued for receivers being supported by that MCS, and L_MULTI_DROPs MUST be issued for receivers no longer supported by the MCS. If there is no outgoing point-to-multipoint VC from the secondary MCS, then an L_MULTI_REQUEST is issued before issuing L_MULTI_ADDs for the remaining supported receivers. A secondary MCS MUST NOT use the MARS_SJOIN and MARS_SLEAVE messages received from the MARS to update the point-to-multipoint VC. These messages are used to update its view of the group only. The setting of the revalidate flag associated with the point-to-multipoint VC causes a secondary MCS to send a MCS_REQUEST to the primary MCS. The secondary MCS MUST update its outgoing point-to-multipoint VC depending on the MCS_MULTI returned by the primary MCS. In case of error in receiving MCS_MULTI, the MCS_REQUEST MUST be retransmitted as described in previous paragraph.

If the MCSs share the senders to the group, the secondary MCS MUST NOT forward data traffic received from any of the senders until it receives the senders allocated to it in an error-free MCS_MULTI. It then sets a flag (mcs$support) for VCs from all senders supported by it (and resets it on VCs from other unsupported senders). The MCS MUST forward data received only on VCs with their mcs$support flag set onto the outgoing point-to-multipoint VC to the group members. Both the primary and secondary MCSs MUST use the MARS_SJOIN and MARS_SLEAVE to update their respective outgoing point-to-multipoint VCs.

The primary MCS learns of a secondary MCS ceasing support for the group when it receives a MARS_UNSERV on the ServerControlVC from the MARS. The primary MCS MUST reallocate the senders/receivers supported by the leaving secondary MCS. The primary MCS MUST initiate revalidation procedures as described in section 4.5 if the revalidate flag is set on the outgoing point-to-multipoint data VC. It MUST use the MARS_SJOIN and MARS_SLEAVE messages to update its view of the group, and reallocate/deallocate the receivers between the secondary MCSs.

The primary MCS also needs to deallocate and reallocate senders/receivers to achieve load balancing effects between different MCSs. The algorithms to make load balancing decisions are not specified further and can be implementation dependent.

5.4.1 Deallocation and Reallocation of senders/receivers

Deallocation and reallocation of current assignments of senders/receivers is needed for reasons noted in previous paragraph. For both deallocation and reallocation, the primary MCS MUST send an MCS_MULTI to a secondary MCS. This MCS_MULTI lists the new set of senders/receivers to be supported by that secondary MCS. The secondary MCS performs L_MULTI_DROPs for cluster members that are no longer supported by it, and L_MULTI_ADDs or L_MULTI_REQUEST for new group members, if receivers are being shared. Otherwise it sets/resets the mcs$support flag on incoming VCs if sharing senders, as described earlier. The primary MCS MUST expect an MCS_MULTI_ACK in response, failing which it MUST retransmit the MCS_MULTI at 5 second intervals (section 5.5.3).

Note that the primary MCS considers itself as an active MCS while performing the reallocation of senders/receivers. Hence it MUST make additions/deletions to its point-to-multipoint VC as needed.

5.5 Failure handling

5.5.1 Failure of Primary MCS

A secondary MCS detects the failure of the primary MCS in either of the following conditions:

- when it fails to receive three consecutive MCS_LIST messages from the primary MCS.
- when three successive transmissions of MCS_REQUESTs do not generate an error-free MCS_MULTI.
- when the MCS VC fails, and three attempts to reopen it (at 30 second intervals) fail.

On deciding that the primary MCS has failed, a secondary MCS MUST attempt to open a point-to-point VC to the first MCS (the candidate primary MCS) in the MCS list maintained by it locally. This list is identical on all the MCSs, hence each secondary MCS will attempt to open a VC to the same candidate MCS. Each of these VCs will be treated as the MCS VC to the originating secondary MCS. The candidate MCS MUST NOT attempt to open any VC. If there is only secondary MCS in the cluster when the primary MCS fails, and consequently, only one MCS on the MCS list, the secondary MCS will consider itself to be the primary MCS. The secondary MCSs MUST NOT interrupt forwarding of data traffic onto their point-to-multipoint VC during the primary MCS selection process.

On successful termination of a VC originated from a secondary MCS, the new primary MCS MUST start behaving as described in section 5.4 (send MCS_MULTI and MCS_LIST). Each secondary MCS, on getting MCS_MULTIs and MCS_LISTs, will process them as described in section 5.4. The primary MCS selection process is considered successful ONLY if MCS_LIST are received on the newly established MCS VC from the primary MCS candidate.

Failure of three attempts (at 30 second intervals) to open a VC to the first MCS on the list will cause a secondary MCS to attempt a VC to the next MCS on the list. If the VC is successfully established, but no MCS_LIST is received in 90 seconds (from point of opening of VC), it is treated as a primary MCS failure, and a VC is attempted to the next MCS on the list. If end of list is reached, an external error MUST be generated by each MCS, which can cause intervention by the network administrator.

An MCS MUST NOT send any control messages to any other MCSs until it considers itself as a primary MCS candidate. i.e., it finds itself next on the locally held MCS list. This ensures that all the MCS are synchronized with regards to selecting the current primary MCS candidate.

All MCSs MUST keep track of group membership (using MARS_SJOIN and MARS_SLEAVE) during the primary MCS selection process. They are processed as usual (section 5.4) by the MCSs.

If a new MCS registers to support the group (MARS_MSERV received) during the selection process, all existing MCSs MUST open a point-to-point VC to it and send an MCS_SELECT to it. This indicates to the new MCS that the primary MCS selection procedure is in progress. On reception of one (or more) such MCS_SELECTs during the 5 second wait period, the new MCS MUST deregister for the group (using MARS_UNSERV), wait for a period of 150 seconds (for the new primary MCS to be selected), and then restart the group registration process (by sending a group registration MARS_MSERV to the MARS).

5.5.2 A discussion of the primary MCS selection procedure

If the primary MCS fails, the above procedure is sufficient to elect a single new primary MCS. This is because each secondary MCS will learn of the primary MCS failing at approximately the same time. So each MCS will initiate the selection procedure almost simultaneously. So they will elect the same new primary MCS, assuming all links are up.

However if links between the primary MCS and some secondary MCSs fail, then scenarios can be constructed where all secondary MCSs do not consider the same MCS as the primary one. This is true for the multiple MARS case also, where cluster members might use different MARS after link failures. The problem here is compounded however as the MCSs not only maintain the database, but also actively participate in data transfer. This is not true in the MARS case, where each MARS may have the same database, and cluster members can use different MARSs simultaneously as long as the MARSs maintain consistency of their copies of the database.

The above discussion leads one to believe that only a best effort solution to the problem of bad links is possible. Manual intervention may be necessary to reset the whole system if too many links go down. However the mechanism does take into account transient failures of links by allowing repeated attempts of VC reestablishment on a failure, through a 90 second period.

A much simpler failure handling mechanism would be to have the group revert to being VC Mesh supported in case of failure of the primary

MCS. All secondary MCSs would deregister from supporting the group if the primary MCS failed. This mechanism can be provided as an option for the user, and can be indicated to the MCSs at startup.

5.5.3 Failure of Secondary MCS

A primary MCS detects that a secondary MCS has failed when either of the following occur:

1. when 3 successive MCS_ALIVEs are not received.
2. when the MCS VC gets disconnected, and 3 attempts to re-open it at 5 second intervals fail.
3. when 3 retransmissions of MCS_MULTI (at 5 second intervals) do not result in an MCS_MULTI_ACK being received.

On determining that a secondary MCS has failed, the primary MCS will close the MCS VC to it (if open). It will then reallocate (section 5.4.1) the senders/receivers being supported by that MCS amongst the remaining MCSs (including itself).

6 Conclusions

This document presents a Multicast Server Architecture for MARS-based ATM multicasting. The architecture includes an allocation mechanism which permits the use of multiple MCSs per layer 3 group. MCSs can also support multiple groups simultaneously. There remain some unresolved issues which are summarized below.

- The exact format of the control messages used for the allocation mechanism need to be specified.
- As MCSs are expected to handle data from several senders, it may be necessary to provide the point-to-multipoint data VC originating from an MCS with better QOS, as compared to that originating from a sender. The document does not address this issue.
- The VC Mesh and the MCS approaches can be considered to be two extremes of a more general approach wherein the same sender could multicast directly to group members as well as MCSs. This model

may be needed by applications with large numbers of senders and receivers in the cluster. It may also be desired for applications wherein senders generate traffic with different QOS requirements for the same layer 3 group, causing MCSs to be used for high bandwidth traffic, and VC Meshes to be used for low bandwidth traffic. A more complex protocol (based on the one proposed in this document), which may involve active use of the MARS, may be needed to support such a model. A similar model has been presented in [BK96].

- Usage of MCSs to enforce security mechanisms have not been addressed.

- The increased VC usage for the VC Mesh case leads to a decrease in the maximum permissible size of the LIS. Thus more LISs will be needed for supporting the same number of hosts in the VC Mesh case. Inter-LIS devices (IP routers) will need to be used for communicating between hosts on different LISs. It remains to be seen if the increased use of routers is more detrimental to the data throughput, as opposed to use of MCSs with larger LISs.

7 Acknowledgements

We would like to acknowledge Grenville Armitage (Bellcore) for reviewing the draft and suggesting improvements towards simplifying the allocation mechanism. Discussion with Joel Halpern (Newbridge) helped clarify the multiple MCS problem. Anthony Gallo (IBM RTP) pointed out security issues that are not adequately addressed in the current draft.

8 Authors' Address

Rajesh Talpade - taddy@cc.gatech.edu - (404)-894-9910
Mostafa H. Ammar - ammar@cc.gatech.edu - (404)-894-3292

College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280

References

[GA96] Armitage, G. J., "Support for Multicast over UNI 3.0/3.1 based ATM networks", Internet-Draft, draft-ietf-ipatm-ipmc-12.txt, Feb 1996.

[BK96] Birman, A., Kandlur, D., Rubas, J., "An extension to the MARS model", Internet Draft, draft-kandlur-ipatm-mars-directvc-00.txt, Nov 1995.

[ML93] Laubach, M., "Classical IP and ARP over ATM", RFC1577, Dec 1993.

Appendix A

A Comparison

The table below compares some quantitative parameters needed for supporting a single group while using the MCS and the VC mesh approaches (ignoring the control VCs maintained between the MARS and the cluster members).

Number of multicast sources multicasting to group G = n
Number of group members in G = m

	MCS	VC mesh
total VCs terminated at cluster members	n+m	n*m
point-to-multipoint VCs	1	n
point-to-point VCs	n	0
VCs terminated at each group member	1	n
signalling requests generated due to a single membership change	1	n

Advantages

- As can be seen above, VC usage is much better in the MCS case. The increased VC usage for the VC mesh leads to greater consumption of resources like memory for maintaining state, buffer allocation per VC, and the VCs themselves, which may be a scarce and/or expensive resource.

- Group membership changes also cause a decreased level of signalling load to be generated at the switch (and the senders to the group) in the MCS approach. This is because only the MCS has to add/delete a cluster member from the point-to-multipoint VC. The other VCs are not affected. Thus signalling requests only occur at the UNI between the MCS and the switch, as opposed to occuring at the UNIs between all the sources and the switch in the VC mesh case. This is especially beneficial when the group is highly dynamic, or when the links between the switch and the cluster members are error-prone, which may cause group members to be temporarily dropped from the multicast group, thus making the group more dynamic than it actually is.

- The MCS approach provides a centralized control of multicast bandwidth usage over the ATM network. This is useful in cases where policy demands a limit on the share of bandwidth available for multicast purposes. In such a case, the administrator who sets up a cluster member as the designated MCS for a group, can control the rate at which the MCS multicasts data. An additional level of security can also be maintained for sensitive multicasts, as all group members will have to be authorized by the centralized MCS only before they can receive the multicast data. The VC mesh approach would need such security control to be enforced at each multicast source to prevent unauthorized cluster members from receiving the multicast data.

Disadvantages

- Data throughput and end-to-end latency may be adversely affected due to the additional level of indirection introduced by the MCS. The MCS can potentially become a bottleneck and a central point of failure. We address this issue and suggest possible solutions in a later section.

- Each group member needs to terminate one point-to-multipoint VC originating from the MCS. So if a multicast source happens to be a member of the group it is transmitting to, it will receive a copy of the data back over the VC from the MCS. Thus additional header identification is needed for a source to discard such "bounced-back" data. This mechanism has been defined in [GA96] to be a 16 bit cluster member identifier.

- Each multicast source in the cluster may desire to use differing QOS parameters for outgoing traffic. Use of the MCS implies that all group members will receive data with QOS as determined by the MCS, irrespective of the QOS used to get them from the source to the MCS.

As was explained in section 6, it remains to be seen if the increased use of IP routers in the VC Mesh approach is more detrimental to data throughput, as opposed to use of MCSs with larger LISs.