A Network-Based Approach to Reliable Multicast

Arturo Azcorra <azcorra@dit.upm.es>
María Calderón <mcalderon@fi.upm.es>
Technical University of Madrid (Spain)

ABSTRACT

There are many distributed applications which require a reliable multicast service, we could point out among others the following ones: whiteboards, distributed interactive simulation, distributed data base systems and software distribution.

The majority of solutions given in order to provide reliable multicast service consist on extending to the multicast environment the solutions used in the unicast environment, this is, to use end-to-end protocols. Most end-to-end protocols have problems of scalability, performing poorly in terms of throughput, network resources and distribution mean delay.

In large internets transit delay and loss probability increase, making end-to-end protocols not suitable. For large multicast groups sender-based protocols cannot be used, and as loss probability increases with the size of the group, receiver-based protocols cannot be used either.

In this paper we present a novel reliable multicast protocol that follows a network-based approach to avoid the problems of end-to-end protocols. RMNP (Reliable Multicast Network Protocol) provides a reliable multicast service on top of an unreliable multicast datagram service. Our proposal is based on recovering lost packets from intermediate routers which are always closer to the member(s) than the source. This has two important consequences, distribution mean delay and retransmission overhead are reduced.

Furthermore, the network-based approach of RMNP allows that the routers perform an aggregation of concast packets, mainly acknowledgments, that flow from the members towards the source. Because of this, large groups are better supported by saving bandwidth and avoiding the implosion problems found on sender-approach protocols.

Keywords: Reliable Multicast, Scalability, Distribution Delay, Multimedia, Internetworking, Protocol Design.

1. Introduction

Many distributed applications require a reliable multicast service: whiteboards [1], distributed data base systems [2], distributed interactive simulation and software distribution. Some of these applications require in addition to reliability, other services such as atomicity and/or some type of ordering (total, causal, etc.).

The most frequent approach to provide reliability has been the extension of unicast end-to-end protocols to the multicast environment.

In the literature, many end-to-end protocols which provide a reliable multicast delivery and different types of ordering [3,4,5,6,7,8,9] have been proposed. Most of these protocols have problems of scalability, i.e., they are not suitable to wide internets and/or large groups. Besides, distribution mean delay is high in the face of packet loss. By distribution mean delay we understand the mean delay since the packets are released by the source until they are received by every member.

We propose a new approach. First, we judge better to separate the reliability from other kind of services, and these can be added if desired on top of the reliable multicast delivery service. Second, it is more efficient in the multicast environment to solve the reliability problem at the network layer. Here, reliability means that in the absence of network partitions, any member of a group will receive all multicast packets sent to that group without message duplication or loss, despite transmission errors, buffer overflow, router and link faults and in general, changes in the network topology which cause a change in the multicast delivery tree.

We describe a new reliable multicast protocol at the network layer. RMNP does not incur in a great control and processing overhead and avoids the implosion problem at the source. Because of this, it scales well and it is suitable to large groups. This is possible thanks to an aggregation process carried out with the acknowledgments at the routers belonging to the delivery tree. And also very important, RMNP imposes a lower distribution mean delay because lost packet recovery is performed from a router close to the member(s) which requested the retransmission.

This way of focusing the problem was already proposed by Rajagopalan in [10], but his approach differs on the following ways: in our scheme the packet retransmission is done from a point nearer to the place where the problem started, instead of from the source. Another difference is that RMNP uses a dynamic window mechanism in the source (instead of regular checkpoints as proposed by him) thus increasing the throughput. Finally, RMNP is not bound to a particular multicast routing architecture. The only requirement on the routing architecture is that each router on the tree has to record its parent router (upstream neighbor) and child routers (downstream neighbors) with respect to a each particular tree. RMNP uses, but not modifies, this information.

The network model that we use in this paper is an internet that works in datagram mode, composed of a collection of local area networks interconnected by routers, links and subnetworks.

The rest of the document is organized as follows. In the next section we analyze the main weaknesses of the current approaches to the problem. Section 3 describes the RMNP scheme and its use on top of multicast routing architectures based on source trees. Section 4 explains the use of RMNP on top of multicast routing architectures based on shared trees. Section 5 comments possible heuristics for the choice of storing routers. Section 6 shows the advantages and disadvantages of RMNP. Finally, section 7 offers some thoughts on future work and some conclusions.

2. Current Approaches to the Problem

There are many proposals of end-to-end protocols that offer a reliable multicast delivery service. These can be classified by the technique used to provide reliability in two categories: sender-approaches and receiver-approaches.

When a sender-approach is used [4,9] it is responsibility of the sender to check that all the receivers get a copy of each packet sent. In this case, reliability is ensured through the return by the receivers of positive acknowledgments (ACKs) and the use of timers at the sender for the purpose of detecting packet loss. Most early work focused on sender-approach protocols. The main problems of this type of solutions are:

- An implosion effect occurs at the source when the receivers attempt to return a reply (e.g. acknowledgments). An analysis of this effect appears in [11].
- Furthermore, as a result of the acknowledgments sent almost simultaneously by the receivers towards the source, the traffic on the routers and links leading to the source is increased. This could even cause a congestion problem in this part of the network.
- The source has to know the identity of all members in order to maintain state information regarding them.

The main consequence of the above problems is that end-to-end protocols that provide a reliable multicast service following a sender-approach scale poorly and are not suitable to large groups.

When a receiver-approach is used [5,6,7,8] it is responsibility of each receiver to detect lost packets and inform the sender via negative acknowledgments (NAKs) when it requires the retransmission of a packet. No positive acknowledgments are sent. Recent studies and proposals [8,12] conclude that to provide a reliable multicast service is most appropriate to use a receiver-approach. However, under this approach some problems arise, most of them due to a lack of feedback from the receivers under normal operation:

- The sender does not ever know for certain that every destination has received a given packet. Because of this, the sender has to indefinitely keep a copy of each packet sent, if the protocol is to be considered truly reliable.

- A lost packet will not be detected until another packet is received successfully. The consequence is that the receivers may have to await an unbounded time to detect the loss of the last packet in a burst. This increases the mean delivery delay because if a member does not detect the loss of a packet it cannot request its retransmission.

- Receiver failure cannot be detected. When the sender does not receive any negative acknowledgment it assumes that everything is working correctly, but the reality could be that a receiver(s) is faulty.

- It is complex to incorporate flow and congestion control mechanisms.

To solve these problems, many of the protocols which follow a receiver-approach introduce different types of mechanisms such as timers, random delays and combinations of positive and negative acknowledgments. This leads to protocols that are not very efficient, that introduce a considerable control overhead and impose a high mean delivery delay. Moreover, in some cases they also suffer from implosion effects.

3. RMNP Description

RMNP is a network layer protocol that offers a reliable multicast delivery service. In order to provide reliability it uses a positive acknowledgment mechanism with retransmissions. RMNP is built on the service of a multicast routing architecture based on delivery trees. These could be source-based trees [13] or center-based trees [14], also named shared trees, or a combination of both [15]. The multicast delivery tree is used to distribute to the members the data released by the source and also to perform an aggregation process with the reply packets which are sent by the members to the source.

There are four relevant aspects in the design of RMNP. First, in addition to positive acknowledgments it uses negative acknowledgments. Second, it performs an aggregation of reply packets (e.g. acknowledgments). Third, retransmissions are sent from intermediate routers (whenever it is possible) instead of only from the source. Finally, a dynamic window is used.

The advantage of using negative acknowledgments is that the mean throughput is increased because a router or member can request the retransmission of a packet when a gap is detected, without having to wait the corresponding time-out.

The benefits of reply aggregation are that it avoids an implosion effect at the source, and that it reduces traffic with the corresponding saving in network throughput. As the number of members can be large (even in the order of thousands), the routers send only one upstream reply instead of the *k* received replies. In this way the process performs an exponential reduction of traffic towards the source. The source receives only one reply per each router that is one hop away[1], as well as another reply from each member which is attached to its same subnetwork.

Intermediate packet retransmission has two advantages: that the delay caused by retransmission is lower, and that bandwidth is saved because the retransmitted packets need not be distributed through the whole tree, but just through the subtree with root at the nearest retransmission point.

The dynamic window scheme of RMNP allows the implementation of flow and congestion control mechanisms. The source varies dynamically the window size according to the feedback that it receives from the network and from the members.

The following subsections provide a more detailed analysis of RMNP, in particular, showing its relation to the source-trees built by the underlying unreliable multicast protocol. The behavior of RMNP regarding each source and its associated distribution trees is completely independent. Consequently, the protocol description will be performed regarding a single source and its associated tree. Section 4 describes the differences in procedures when centered or shared trees are used.

[1] If the source is on a subnetwork, instead of on a link to a router, there could be several routers belonging to the delivery tree that are one hop away from the source.

3.1 Network Architecture

In Figure 1, it is shown a typical of source-based delivery tree labeling the different types of routers defined in RMNP. The systems are classified by the function they fulfill:

Source (S). One of the various systems *sending* packets to the group. It has to know the identity of all the routers belonging to the delivery tree which are one hop away. It also has to know the identity of the members which are attached to its same subnetwork.

Basic Functionality Router (BFR). A router which performs sequence number control and acknowledgment aggregation. It does not store unacknowledged packets.

Storing Router (SR). A router which stores the packets pending acknowledgment, and other functions including that of a BFR.

Last Hop Router (LHR). A router which has directly attached members. This means that either they are in the same subnetwork (i.e. a LAN), and/or the members are connected to router ports. A LHR has to know the identity of each and every of the said members.

Member (m). An end system that belongs to the group, i.e., that *receives* the packets sent to this group. The source does not necessarily have to be a member.

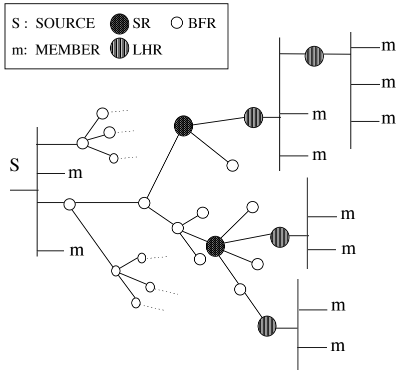

Figure 1: Example of source tree

It must be noted that a given router can fulfill several functions on the same tree. For example, one router could be SR and LHR. Obviously, a given router can fulfill different, and fully independent, functions on different trees.

3.2 Downstream Data Forwarding (normal state)

The source sends packets in sequence using a sliding window (subnetwork multicast service is used if available). For each packet sent, a timer is started. If the timer expires, the source will send again the packet followed by the remaining unacknowledged packets (simple reject). The window size is dynamically adjusted following a procedure described in section 3.7. The maximum window size, MW, is half the sequence number space to avoid packet duplication because routers and members are not aware of the window size. It seems reasonable that RMNP uses a sequence number field length in the range 10 to 16 bits.

Each system records in variable NSN the sequence number of the next packet expected in the normal state. For routers, this variable minus one also has the semantics of the sequence number of the highest packet sent to *all* downstream neighbors. Each system records in variable HAU the highest packet acknowledged to its upstream neighbor, this is, the sequence number contained in the last ACK sent.

When a system receives the expected packet *accepts* it and actualize their variables. If the system is a members it simply acknowledges the packet. If the system is a BFR, it forwards the packet to all its downstream neighbors (subnetwork multicast service is used if available). If it is an SR, in addition to forwarding, it stores a copy and starts a timer after having sent the packet to its downstream neighbors. If the timer expires, the SR will send again the packet followed by the remaining unacknowledged packets (simple reject).

When a router or member receives a packet with a sequence number in the range [NSN+1, HAU+MW-1], it sets a timer (the out-of-sequence timer) and waits to receive the missing packet(s), in which case it continues normal procedure by sequential downstream forwarding. But if the timer expires, it requests upstream a retransmission from NSN. Packet retransmission is requested using a NAK packet. A NAK_n requests the retransmission from packet *n,* but it does not imply any acknowledgment information.

One of the reasons why a system has to wait a certain time before sending a NAK after receiving an out of sequence packet is that there could be subnetworks which do not guarantee sequential delivery.

When a router or member receives a packet in the range [NSN-MW, HAU-1] it is a duplicate that has already been acknowledged, and therefore it will discard the packet and send an ACK with sequence number HAU.

When a router receives a packet with a sequence number in the range [HAU+MW, NSN-MW-1] it is a protocol error.

When a BFR receives a packet with a sequence number in the range [HAU, NSN-1] it is a duplicate unacknowledged packet, and therefore it will forward the packet to all downstream neighbors that have not previously confirmed this packet. This packets are typically caused by time-outs. When a SR receives a duplicate unacknowledged packet, it simply discards it because SRs handle their own timers.

3.3 Packet Acknowledgment

The acknowledgment process is initiated by the group members and it propagates through the tree towards the source, using an aggregation process at the routers.

To perform the aggregation process a router uses variable HAU, which contains the highest packet acknowledged to its upstream neighbor, this is, the sequence number contained in the last ACK sent. The router also records in variables HAD_i the highest packet confirmed by its downstream neighbors, this is, the sequence number contained in the last ACK received from each of them. Whenever an ACK is received or sent, these variables are updated. An incoming ACK_n acknowledges all packets sent up to, but not including, sequence number *n* (accumulative acknowledgment).

Whenever an incoming ACK is equal, or greater than, a sequence number that has already been acknowledged by all the remaining downstream neighbors, then the router will send upstream an ACK with the largest sequence number acknowledged by all downstream neighbors.

When a SR sends an acknowledgment to its upstream neighbor it frees up the buffer space, and stops the timers associated to, the packets being confirmed.

By means of this process the acknowledgments are propagated towards the source. When the source receives an acknowledgment from all its downstream neighbors, it actualizes its variables, frees up the associated buffers and advances the window.

3.4 Loss Recovery by Negative Acknowledgment (nak state)

Each BFR maintains for each downstream neighbor a variable LND_i containing the lowest sequence number among the pending NAKs received from it. Each BFR also records in variable LNU what is the lowest sequence number among the pending NAKs that it has sent to its upstream neighbor.

When a BFR in normal state receives a NAK_n packet it enters nak state. It forwards the NAK_n upstream and sets the corresponding LND_i variable and the LNU variable to *n*.

The following assertions in this subsection apply when in nak state.

When a BFR receives a NAK_k packet from a downstream neighbor (notice that a NAK packet could or could have not been previously received from this particular neighbor), the LND_i variable of this neighbor is set to *k*. If (*k*<LNU), then LNU is set to *k* and a NAK_k is sent.

When a packet with sequence number in the range [NSN, HAU+MW-1] is received, it is considered that if follows the interrupted sequence and it is ignored.

When a BFR receives a packet with sequence number LNU it will forward the packet to all neighbors with LND_i equal to LNU. Also, the corresponding variables LND_i, and LNU, are incremented. This process is iterated until packet NSN-1 is reached, in which moment nak state is exited to enter normal state and variables LNU and all LND_i are cleared.

When a router receives packet(s) with sequence number(s) in the range [LNU+1, NSN-1] they are considered out of sequence packet(s). If the out-of-sequence timer expires without receiving all the intermediate packets, a NAK with sequence number LNU is sent upstream.

When a router receives a packet in the range [NSN-MW, HAU-1] it is considered a retransmission caused by a time out of an already acknowledged packet. Then, it discards the packet and sends and ACK with sequence number HAU.

When a router receives packet *k,* with *k* in the range [HAU, LNU-1], it is considered a retransmission caused by a time out of an unacknowledged packet. This packet is forwarded to all downstream neighbors with variable HAD_i ≤ *k*. Also, the variables are actualized.

When a router receives a packet in the range [HAU+MW, NSN-MW-1] it is considered a protocol error.

3.5 Retransmission Timer Adjustment Mechanism

RMNP uses a generalization of the timers adjustment mechanism frequently used for TCP implementations. This mechanism is based on the followings algorithms:

- Round-Trip-Time Variance Estimation.
- Karn’s Algorithm
- Exponential Retransmit Timer Backoff

The difference is that TCP uses unicast RTT and RMNP uses instead the Subtree Round Trip Time (SRTT). The SRTT of a packet at an SR or the source is defined as the addition of the packet distribution time and the acknowledgment aggregation time (see Figure 2). The packet distribution time is the time a packet needs to be distributed downstream to all the subtree members (or tree members in the case of the source). The acknowledgment aggregation time is the time that the acknowledgments spend being aggregated upstream until they arrive to the subtree root. In this way, the SRTT will be determined by the farthest member in time.

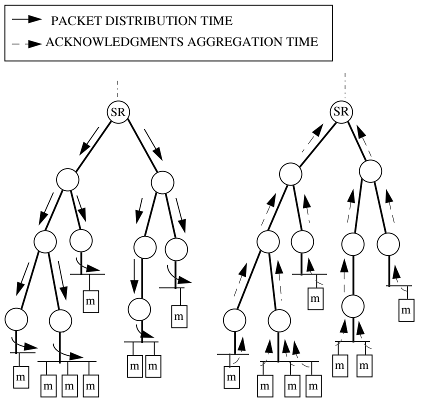

Figure 2: Example of SRTT

3.6 Reset procedure

A Reset procedure has to be performed when a change in the underlying delivery tree takes place. This is necessary because the information at the routers could be inconsistent, for example, a router could have acknowledged packet *n* to its upstream neighbor and in the new tree it could have downstream members which lack packets previous to *n* (see Figure 3). As a router does not distribute to its downstream members a previously acknowledged packet, the new members would never receive the packet(s) they lack.

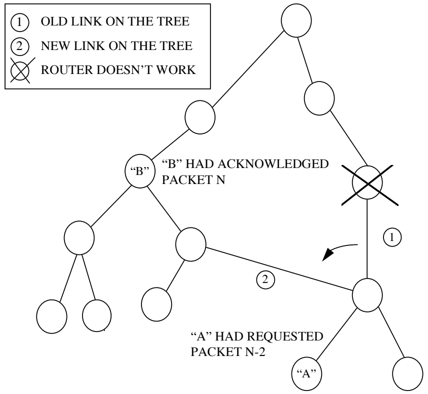

Figure 3: Reconfiguration of a delivery tree

An instance of a reset procedure is identified by a sequence number and it is started by the source at the request of the routers. A router detects a change in the delivery tree when it observes that at least one of its parent or child interfaces has changed, for example a new child interface is added or released.

The warning to the source is made by the routers in the new tree using of two parallel and independent procedures:

- Each SR which detects a change in the tree sends a reset request to the source as a unicast message. It is hardly likely that this behavior causes an implosion problem in the source because the number of SRs that detect a change in the delivery tree will not be very high.

- Each BFR that detects a change in the tree sends a reset request to the source using the new tree. These messages are propagated towards the source, using a aggregation process similar to the one used for NAKs but forwarding the highest reset sequence number instead of the lowest.

The source starts the reset procedure by retransmitting all pending packets. These packets bear information to perform the reset procedure.

The behavior of a router will depend on its state:

- If it was and it still is in the delivery tree, it will clear its buffers, return to the initial state, and set all the state variables to their initial value.

- If it was not but it is now in the delivery tree, it will create the necessary state information and reserve resources.

- Routers which have abandoned the tree erase all the associated state information and free all the associated resources.

The members do not participate in the reset procedure. They will just receive the retransmitted packets, acknowledging them following the normal procedures.

The reset procedure finishes when the source receives an acknowledgment, marked also with the reset sequence number, corresponding to one of the retransmitted packets.

3.7 Flow and Congestion Control

RMNP uses a dynamic window procedure to perform end-to-end flow control and to control network congestion.

When a member sends an ACK to the source, it includes the size of the source window that it desires according to its available resources. At selected instants the source uses the last received values of the desired window to calculate the new size. When a router sends an ACK to its upstream neighbor the desired window field is calculated as the average of the values of desired window received by the router from its downstream neighbors.

As a congestion control mechanism, RMNP uses an adaptation of a Binary Feedback Scheme proposed by Ramakrishnan and Jain in [16]. By means of this scheme the source reacts to the information received from the network increasing or decreasing its window. This feedback from the network is explicit using a bit included in the ACK header.

These mechanisms cause that the size of the window, and therefore throughput, are limited by the slowest member or by the path belonging to the tree that is capable to transport fewer packets per second. For example, as it is shown in Figure 4, if a link on the path to any destination has low capacity, the upstream router directly connected becomes congested and consequently the throughput to *all* destinations is reduced. This happens when the capacity to process packets of each member is widely different or if the capacity to transport packets of each path belonging to the tree is very disparate.

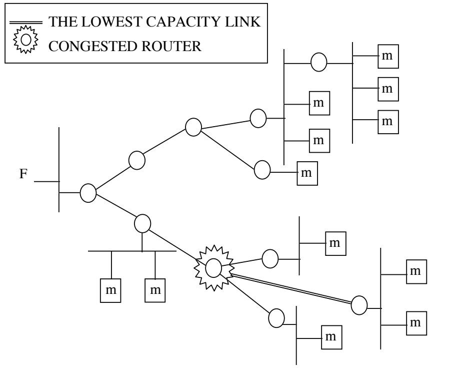

Figure 4: Congested path on a delivery tree

RMNP allows that the service user specify a minimum throughput. The members which do not meet the minimum required will be ejected. This facility will be used if the application which is using RMNP requires a minimum throughput to work correctly. In the case that the minimum specified is zero all members adapt themselves to the rate of the slowest one.

4. RMNP and Shared Trees

When this type of multicast routing strategy is used [14], there is a delivery tree per group that is shared by all the sources of the group. The root of this tree is called center or core.

The approach usually followed to deliver multicast data is shown in Figure 5. The packets are sent by the source towards a node belonging to the tree (i.e., towards the core). When it arrives to any node in the tree (“hits” the tree), the packet is distributed through it. The router in which this distribution starts will be called *access router*. The Access Router for each source and group will be determined by the underlying unicast routing level.

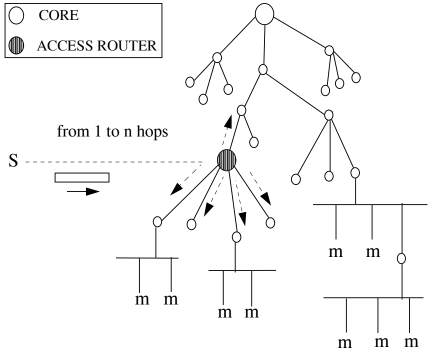

Figure 5: Example of shared tree.

In order to make aggregation process possible, acknowledgments and retransmission requests must follow the same delivery tree used for data but from the members towards the access router. This problem imposes that every router belonging to the tree stores for each source (S) the interface leading to the access router at RMNP layer. This information is learned by the routers when they initially receive data coming from S.

5. Selection of Storing Routers

The selection of routers which will have storing functionality is a relevant aspect because it will directly affect the mean delivery delay and the network resources, i.e., bandwidth and processing spent to get a reliable distribution.

If there are many SRs the mean delivery delay will decrease, but that imposes a memory overhead in each SR. In the opposite case, if there are only a few SRs, the memory overhead will be lower, but the mean delivery delay will not be much lower than the case in which the retransmissions are done only from the source. Therefore, it is necessary a trade-off between mean delivery delay and memory overhead.

A first possibility is to let the network administrator decide which routers should perform this function and configure them for that purpose. For example, it would be desirable having each first router for a domain configured to behave as an SR.

A second possibility, much more attractive, being researched is to make each router itself decide. In this case each router belonging to the delivery tree decides, according to certain parameters, if it is going to be SR for each group and source. Several heuristics are being evaluated:

- Based on network distance metrics.
- Based on the number of downstream neighbors.
- A combination of both.

6. Properties

The main advantages of the RMNP approach are the following:

- It scales well, and therefore it is suitable to large groups and/or large internets.
- As packet loss probability increases, RMNP decreases distribution mean delay, as well as NAK and retransmission bandwidth when compared to a end-to-end packet recovery protocols.
- RMNP is not bound to a particular underlying unreliable network multicast protocol, although it imposes some requirements on it.
- The source does not need to know the identity of any of the group members (save for the trivial case of the ones on its same subnetwork) and therefore our solution follows the model of group proposed by Deering [17].
- It does not introduce an excessive control and processing overhead.

On the other hand, the following weaknesses can be identified:

- It is necessary to introduce a new protocol in the routers of the network.
- There is a memory overhead, but all solutions that decrease distribution mean delay impose memory overhead.
- The aggregation and filtering procedures introduces a processing overhead.
- When there is a change in the delivery tree a reset procedure is started, and this imposes a bandwidth and processing overhead. However, delivery trees are usually stable.

After seeing the main properties of RMNP, we can summarize that RMNP does not have the problems of the end-to-end protocols that follow a receiver approach and it has the advantages of the end-to-end protocols that follow a sender approach without having their drawbacks.

7. Conclusions and future work

Despite many distributed applications can benefit greatly from a reliable multicast service and many proposals have been made, this area needs to be investigated more. Many of these proposals do not scale well and impose a mean delivery delay that increases sharply with loss probability, because they use a end-to-end recovery. These restrictions are intolerable for some applications.

RMNP does not suffer from implosion problems and therefore it is scaleable. This means, that it is suitable for large groups. Besides, the mean delivery delay is lower when compared to an end-to-end packet recovery scheme. This is so because loss recovery is performed from a router close the member(s) where the loss is detected.

Our proposal has a main drawback, it is necessary to introduce a new protocol in the routers of the internet.

Some areas on which further research could be carried out are:

- To experiment with heuristics to choose the SRs.
- To fine-tune the flow and congestion control mechanisms suggested.
- To study the feasibility of installing RMNP only at some of the routers in a distribution tree.
- The reset procedure affects all the tree and in some situations it would be possible to enter in undesirable status. Besides, partial reset mechanisms could be investigated.

8. Acknowledgments

We would like to thank Ana María Santos for her collaboration along the production of this paper.

REFERENCES

[1] Jacobson, V. and McCanne, S., “vat”, Manual Pages, Lawrence Berkeley Laboratory, Berkeley, CA.

[2] Chang, J.M., “Simplifying Distributed Database Systems Design by Using a Broadcast Network”, Proceedings of SIGMOD’ 84 pp.223-233, Jun. 1984.

[3] Chang, J-M. and Maxemchuk, N.F., “Reliable Broadcast Protocols”, ACM Transactions on Computer Systems 2(3) pp. 251-273, Aug. 1984.

[4] Crowcroft, J. and Paliwoda, K. “A Multicast Transport Protocol”. Proceedings of ACM SIGCOMM’ 88 pp. 247-256, Aug. 1988.

[5] Melliar-Smith, P.M., Moser, L.E. and Agrawala, V., “Broadcast Protocols for Distributed Systems”, IEEE Transactions on Parallel and Distributed Systems 1(1) pp.17-25, Jan. 1990.

[6] Armstrong, S., Freier, A. and Marzullo, K., “Multicast Transport Protocol”, RFC1301, Feb. 1992.

[7] Holbrook, H.W., Singhal, S.K. and Cheriton, D.R., “Log-Based Receiver-Reliable Multicast for Distributed Interactive Simulation”, to appear in Proceedings of ACM SIGCOMM’ 95, Aug. 1995.

[8] Floyd, S., Jacobson, V., McCanne, S., Liu, C-G. and Zhang, L., “A Reliable Multicast Framework for Lightweight Sessions and Application Level Framing”, to appear in Proceedings of ACM SIGCOMM’ 95, Aug. 1995.

[9] Birman, K., “The Process Group Approach to Reliable Distributed Computing”, Communications of the ACM 36(12) pp. 37-53, Dec. 1993

[10] Rajagopalan, B., “Reliability and Scaling Issues in Multicast Communication”. Proceedings of the ACM SIGCOMM’ 92 pp. 188-198, Aug. 1992.

[11] Danzing, P.B., “Finite Buffers and Fast Multicast”, Performance Evaluation Review 17(1) pp.108-117, May 1989

[12] Pingali, S., Towsley, D. and Kurose, J., “A Comparison of Sender-Initiated and Receiver-Initiated Reliable Multicast Protocols”, Tech-Rep Dpt of Computer Science University of Massachusetts, Amherst, Oct. 1993.

[13] Moy, J., “Multicast Extensions to OSPF”, RFC1584, Mar. 1994

[14] Ballardie, T., Francis, P. and Crowcroft, J., “Core Based Tree (CBT). And Architecture for Scalable Inter-Domain Multicast Routing”, Proceedings of ACM SIGCOMM’ 93 pp 85-95, Sep. 1993.

[15] Deering, S., Estrin, D., Farinacci, D. and Jacobson, V., “An Architecture for Wide Area Multicast Routing”, Proceedings of the ACM SIGCOMM’ 94 pp. 126-135, Oct. 1994.

[16] Ramakrishnan, K.K. and Jain, R., “A Binary Feedback Scheme for Congestion Avoidance in Computer Networks”, ACM Transactions on Computer Systems 8(2) pp. 159-181, May 1990.

[17] Deering, S., “Multicast Routing in a Datagram Internetwork”, PhD Thesis, Stanford Univertsity, Palo Alto, California, Dec. 1991.