Reliable Multicast Transport Protocol: RMTP

Pradip M. Jawandhiya
Research Scholar, Jawaharlal Darda Inst.of Enggneering & Technology,
MIDC,Lohara, Yavatmal – 445001(M.S.), INDIA
+91 9822726385
pmjawandhiya@rediffmail.com

Dr. M. S. Ali
Principal, Prof. Ram Meghe Inst.of Technology & Management,
Bypass highway road, Badnera,
Dist. Amravati (M.S.), INDIA
+91 9370155150
softalis@hotmail.com

Sakina F.Husain & M.R.Parate
Faculty,Jawaharlal Darda Institute of Engineering & Technology,
MIDC,Lohara, Yavatmal – 445001(M.S.), INDIA
+91 9890140576
Skap_88_e@yahoo.com
Mayur_parate@yahoo.in

Prof. J.S.Deshpande
Pro Vice-Chancellor Sant Gadge Baba Amravati University, Amravati, INDIA
+91 9422881986

***Abstract*— This paper presents the design, implementation, and performance of a reliable multicast transport protocol (RMTP). RMTP is based on a hierarchical structure in which receivers are grouped into local regions or domains and in each domain there is a special receiver called a designated receiver (DR) which is responsible for sending acknowledgments periodically to the sender, for processing acknowledgment from receivers in its domain, and for retransmitting lost packets to the corresponding receivers. Since lost packets are recovered by local retransmissions as opposed to retransmissions from the original sender, end-to-end latency is significantly reduced, and the overall throughput is improved as well. Also, since only the DR's send their acknowledgments to the sender, instead of all receivers sending their acknowledgments to the sender, a single acknowledgment is generated per local region, and this prevents *acknowledgment implosio*n. Receivers in RMTP send their acknowledgments to the DR's periodically, thereby simplifying error recovery. In addition, lost packets are recovered by selective repeat retransmissions, leading to improved throughput at the cost of minimal additional buffering at the receivers. This paper also describes the implementation of RMTP and its performance on the Internet.**

Keywords- multicast routing, MANET, acknowledgement implosion, designated receiver

I. INTRODUCTION

MULTICASTING provides an efficient way of disseminating data from a sender to a group of receivers. Instead of sending a separate copy of the data to each individual receiver, the sender just sends a single copy to all the receivers. A multicast tree is set up in the network with the sender as the root node and the receivers as the leaf nodes. Data generated by the sender goes through the multicast tree, traversing each tree edge exactly once. However, distribution of data using the multicast tree in an unreliable network does not guarantee reliable delivery, which is the prime requirement for several important applications, such as distribution of software, financial information, electronic newspapers, billing records, and medical images. Reliable multicast is also necessary in distributed interactive simulation (DIS) environment, and in collaborative applications. Therefore, reliable multicasting is an important problem which needs to be addressed. Several papers have addressed the issue of multicast routing [1], but the design of a reliable multicast transport protocol in broadband packet-switched networks has only recently received attention [2].Reliable multicast protocols are not new in the area of distributed and satellite broadcast systems [3]. However, most of these protocols apply to local area networks and do not scale well in wide area networks, mainly because the entities involved in the protocol need to exchange several control messages for coordination purposes. In addition, they do not address fundamental issues of flow control, congestion avoidance, end-to-end latency, and propagation delays which play a critical role in wide area networks. Several new distributed systems have been built for group communication recently, namely, Totem [10] and Transis [7]. Totem [10] provides reliable totally ordered multicasting of messages based on which more complex distributed applications can be built. Transis [7] builds the framework for fault tolerant distributed systems by providing mechanisms for merging components of a partitioned network that operate autonomously and later become reconnected.

Both these systems assume the existence of multiple senders and try to impose a total ordering on delivery of packets. However, the reliable multicast transport protocol in this paper has been designed to operate at a more fundamental level where the objective is to deliver packets in ordered lossless manner from a single sender to all receivers. In other words, our protocol can potentially be used by Totem to provide reliable total ordering in a wide area packet-switched network. Other transaction-based group communication semantics like atomic multicast, permanence, and serializability can also be built using our reliable multicast transport protocol. *Multicasting* is a very

broad term and different multicasting applications have, in general, different requirements. For example, a real-time multipoint-to-multipoint multimedia multicasting application, such as, nationwide video conferencing, has very different requirements from a point-to-multipoint reliable data transfer multicasting application, such as, the distribution of software. Recently, researchers have demonstrated multicasting real-time data, such as real-time audio and video, over the Internet using the multicast backbone *(MBon*e) [8]. Since most real-time applications can tolerate some data loss but cannot tolerate the delay associated with retransmissions, they either accept some loss of data or use forward error correction for minimizing such loss. Multicasting of multimedia information has been recently receiving a great deal of attention [4].However, the main objective of these multicast protocols is to guarantee quality of service by reducing end-to-end delay at the cost of reliability. In contrast, the objective of our protocol in this paper is to guarantee *reliability* achieving high throughput, maintaining low end-to-end delay. This is achieved by reducing unnecessary retransmissions by the sender. In addition, we adopt a novel technique of grouping receivers into local regions and generating a single acknowledgment per local region to avoid the acknowledgment implosion problem [12] inherent in any reliable multicasting scheme.

We also use the principle of periodic sending of state information from the receivers to the transmitter to avoid complex error-recovery procedures. Finally we use a selective repeat retransmission scheme to achieve high throughput. In this paper, we describe our detailed experience with the design and implementation of reliable multicast transport protocol (RMTP).

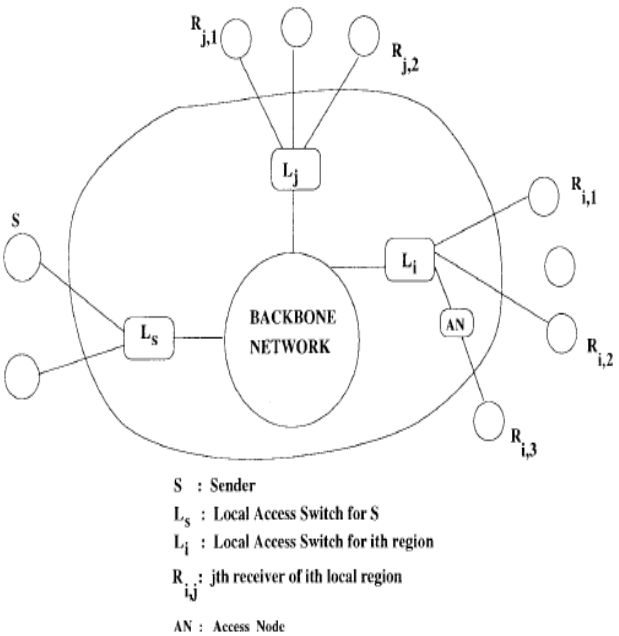


Fig 1: Model of the network

/*The original work consisted of proposing three different multicast transport protocols, comparing them using simulation, and recommending one for reliable multicasting. In fact,*/ the notion of local recovery using a designated receiver (DR) was proposed for the first time in the literature in [11]. The recommended protocol was implemented and its performance, measured on the Internet, was reported in [9]. In this paper, we have combined the ideas and results from [9] and [11] to present a comprehensive picture of our efforts in designing RMTP. RMTP is very general in the sense that it can be built on top of either virtual-circuit networks or datagram networks. The only service expected by the protocol from the underlying network is the establishment of a multicast tree from the sender to the receivers. However, resource reservation is not really necessary for the proper functioning of RMTP. The function of RMTP is to deliver packets from the sender to the receivers in sequence along the multicast tree, independent of how the tree is created and resources are allocated. For example, RMTP can be implemented over available bit rate (ABR) type service in ATM networks for reliable multicasting applications. In this paper, we have addressed the design issues for RMTP in the Internet environment. In particular, the notion of multilevel hierarchy using an internet-like advertisement mechanism is described, and issues related to row control and late-joining receivers in an ongoing multicast session are dealt with extensively. In addition, a detailed description of the implementation using MBone [19] technology in the Internet is also presented and performance measurements are included as well. Most of these ideas and results are taken from [27]. Rest of the paper is organized as follows. Section II discusses the network architecture and the assumptions made in the design of RMTP. Implementation of RMTP is presented in Section III, and its performance evaluation parameters on the Internet and its measurements are presented in Section IV followed by some conclusions.

II. RMTP

2.1 Network Architecture and Assumptions

Let the sender and receiver be connected to the backbone network through local access switches either directly or indirectly through access nodes (fig 2.1) The following are some assumptions made in the protocol design.

a) The receivers can be grouped into local regions [6] based on their proximity in the network. For example, if a hierarchical addressing scheme like E.164 (which is very similar to the current telephone numbering system) is assumed, and then receivers can be grouped into local regions based on area code in an internet protocol.

Network, receivers can be grouped into local regions by using the time-to-live (TTL) field of IP packets. More details on how the TTL field can be used are given in the next section.

b) A multicast tree rooted at the sender S and spanning all the receivers, is set up at the network layer (ATM layer in the context of ATM networks). This is referred to as the global multicast tree in several parts of the paper to distinguish it from the local multicast tree which is a part of the global multicast tree. The global multicast tree is shown by solid lines in Fig 2. Receivers in the local region served by L_i are denoted by Ri,j .Note that L_i denotes the local access switch for the ith region and is not a receiver.

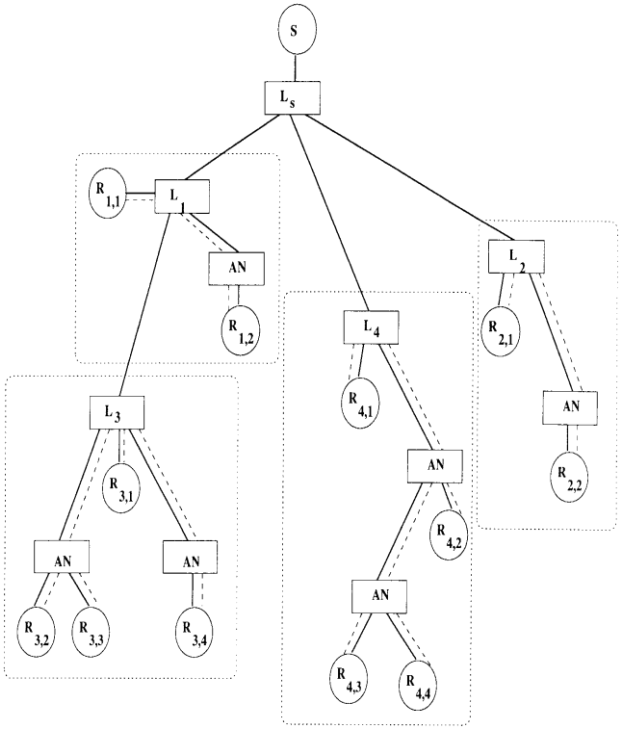


Fig 2:Global multicast tree rooted at S and local multicast trees rooted at $R_{i,j}$'s

c) RMTP is described in this paper as a protocol for point-to-multipoint reliable multicast. Multipoint-to-multipoint reliable multicast is possible if multicast trees are set up for each sender.

2.2 Design Overview

RMTP provides sequenced, lossless delivery of bulk data from one sender to a group of receivers. The sender ensures reliable delivery by selectively retransmitting lost packets in response to the retransmission request of the receivers[9]. If each receiver sends its status (ACK/NACK) all the way to the sender, it results in the throttling of the sender which is the well-known ACK-implosion problem. In addition, if some receivers are located far away from the sender and the sender retransmits lost packets to these distant receivers, the end-to-end delay is significantly increased, and throughput is considerably reduced.

RMTP has been designed to alleviate the ack-implosion problem by using a tree-based hierarchical approach. The key idea in RMTP is to group receivers into local regions and to use a DR as a representative of the local region. Although the sender multicasts every packet to all receivers using the global multicast tree, only the DR's send their own status to the sender indicating which packets they have received and which packets they have not received. The receivers in a local region send their status to the corresponding DR. Note that a DR does not consolidate status messages of the receivers in its local region., but uses these status messages to perform local retransmissions to the receivers, reducing end-to-end delay significantly. Thus the sender sees only the DR's and DR sees only the receivers in its local region. Processing of status messages is distributed among the sender and the DR's, thereby avoiding the ack-implosion problem.

In Fig. 2, receiver Ri,1 is chosen as the DR for the group of Ri,j's, in the local region served by Li. A local multicast tree, rooted at Ri,1, is defined as the portion of the global multicast tree spanning the Ri,j's in the local region served by Li. Local multicast trees are indicated by dashed lines in Fig. 2.

III. IMPLEMENTATION

3.1 RMTP Packet

The sender in RMTP divides the data to be transmitted into fixed-size data packets, with the exception of the last one. A data packet is identified by packet type DATA, while type DATA EOF identifies the last data packet.

Packet Types	
ACK	ACK packet
ACK_TXNOW	ACK - immediate transmission req.
DATA	Data packet
DATA_EOF	Last data packet
RESET	Packet to terminate a connection
RTT_MEASURE	Packet to measure round-trip time
RTT_ACK	ACK to RTT_MEASURE packet
SND_ACK_TOME	Packet for selecting an AP

Table 1: RMTP Packet Types

The sender assigns each data packet a sequence number, starting from zero [5]. A receiver periodically sends ACK packets to the sender/DR. An ACK packet contains the lower end of receive window (L) and a fixed-length bit vector of receive window size indicating which packets are received and which packets are lost. Table 4.1 lists the packet types used in RMTP. Each of their functions will be described in the following subsections

3.2 RMTP Connection

An RMTP connection is identified by a pair of endpoints: a source endpoint and a destination endpoint. The source endpoint consists of the sender's network address and port number; the destination endpoint consists of the multicast group address and a port number. Each RMTP connection has a set of associated connection parameters (see Table 1). RMTP assumes that there is a Session Manager who is responsible for providing the sender and the receiver(s) with the associated connection parameters. RMTP uses default values for any connection parameter that is not explicitly given.

Once the Session Manager has provided the sender and receivers with the session information, receivers initialize the connection control block and remain in an unconnected state; the sender meanwhile starts transmitting data. On receiving a data packet from the sender, a receiver goes from the unconnected state to the connected state. In the

connected state, receivers emit ACK's periodically, keeping the connection alive.

Connection Parameters	
W_r	receive window size in packets
W_s	send window size in packets
T_{dally}	delay after sending the last packet
T_{retx}	time interval to process retx requests
T_{rtt}	time interval to measure RTT
T_{sap}	time interval to send SND_ACK_TOME
T_{send}	time interval to send data packets
T_{ack}	time interval to send status packets
$Packet\_Size$	data packet size in octets
$Cache\_Size$	sender's in-memory data cache size
$CONG_{thresh}$	congestion avoidance threshold
$MCAST_{thresh}$	multicast retransmission threshold

Table3.2: RMTP connection parameters

RMTP is designed based on the IP-multicast philosophy in which the sender does not explicitly know who the receivers are. Receivers may join or leave a multicast session without informing the sender [6]. Therefore the goal in RMTP is to provide reliable delivery to the current members of the multicast session. Since the sender does not keep an explicit list of receivers, termination of RMTP session is timer based. After the sender transmits the last data packet, it starts a timer that expires after Tdally. (ADR also starts the timer when t has correctly received all the data packets.) When the timer expires, the sender deletes all state information associated with the connection (i.e., it deletes the connection's control block). Time interval Tdally is at least twice the lifetime of a packet n an internet. Any ACK from a receiver resets the timer to its initial value. A normal receiver deletes its connection control block and stops emitting ACK's when it has correctly received all data packets. A DR behaves like a normal receiver except that it deletes its connection control block only after the Tdally timer expires.

Since the time period between the transmission of consecutive ACK's from a receiver is much smaller than Tdally , the session Manager is not a part of RMTP transport protocol, but is used at the session layer to manage a given RMTP session.

Sender assumes that either all receivers have received every packet or something "exceptional" has happened. Possible exceptional situations include: network partition and receivers voluntarily or involuntarily leaving the multicast group [8]. RMTP assumes that the Session Manager is responsible for detecting such situations and taking necessary actions.

In addition to normal connection termination, RESET packets can be used to terminate connections. For example, when RMTP detects that the sending application has aborted before data transfer is complete, it uses RESET to inform all the receivers to close the connection.

3.3 RMTP Entities

RMTP has three main entities: 1) Sender, 2) Receiver, and 3) DR. A block diagram description of each of these entities is given in Fig. 3 we describe the major components of these entities below [6].

The Sender entity has a controller component called CONTROLLER, which decides whether the sender should be transmitting new packets(using the Tx component), retransmitting lost packets (using the RTx component), or sending messages advertising itself as an ACK Processor (AP) (using the AP A component and SEND ACK TOME message). There is another component called STATUS PROCESSOR, which processes ACK's (status) from receivers and updates relevant data structures.

Also, note that there are several timer components: TSend, TRetx, and T Sap in the Sender entity, to inform the controller about whether the Tx component, the RTx component or the AP A component should be activated. Timer T Dally is used for terminating a connection.

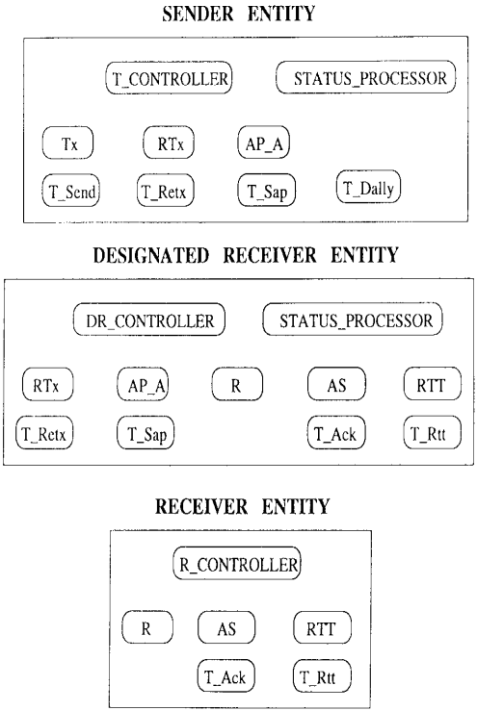


Fig 3: Block diagram of RMTP.

The Receiver entity also has a controller component called RCONTROLLER which decides whether the receiver should be delivering data to the receiving application (using the R component), sending ACK messages (using the AS component), or sending RTT measure packets (using the RTT component) to dynamically compute the round-trip time (RTT) between itself and its corresponding ACK Processor. Note that there are two timer components: 1) T Ack and 2) T Rtt to inform the controller as to whether the AS or the RTT component should be activated. The component R is not timer driven. It is activated asynchronously whenever the receiving application asks for packets.

The DR entity is, in fact, a combination of the Sender entity and the Receiver entity. Key functions

performed by the components of each entity are described next.

3.4. Transmission

RMTP sender (in particular, the Tx component of sender entity in Fig. 4) multicasts data packets at regular intervals defined by a configuration parameter Tsend. The number of packets transmitted during each interval normally depends on the space available in send window. The sender can at most transmit one full window of packets(Ws) during Tsend, thereby limiting the sender's maximum transmission rate to Ws * packet_size/Tsend . To set a multicast session's maximum data transmission rate, the Session Manager simply sets the parameters Ws, packet_size/Tsend , and Tsend accordingly. However, during network congestion, the sender is further limited by the congestion window during the same Tsend interval.

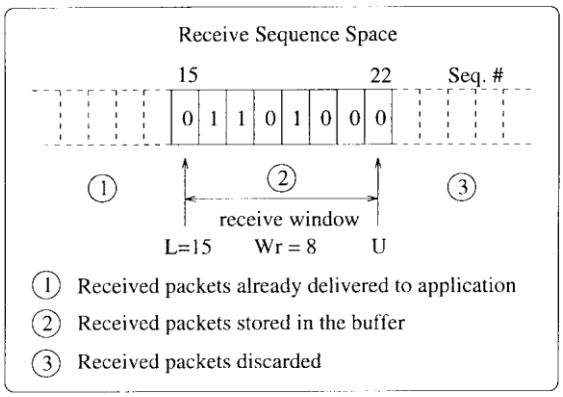


Fig 4: A receiver's receive window and related variables.

3.5 Acknowledgments

RMTP receivers (in particular, the AS component of the receiver/DR entity in Fig. 4.2) send ACK packets periodically, indicating the status of receive window. Receivers use a bit vector of Wr bits (size of receive window) to record the existence of correctly received packets stored in the buffer. As fig. 4 illustrates, each bit corresponds to one packet slot in the receive buffer. Bit 1 indicates a packet slot contains a valid data packet. For example, Fig. 4 shows a receive window of eight packets; packets 16, 17, and 19 are received correctly and stored in the buffer [6]. When a receiver sends an ACK to its AP, it includes the left edge of the receive window and the bit vector. Note that the receiver delivers packets to the application in sequence. For example, if the receiver receives packet 15 from the sender and does not receive packet 18, it can deliver packets 15–17 to the application and advance L to 18.

3.6 Late Joining Receivers

Since RMTP allows receivers to join any time during an ongoing session, a receiver joining late will need to catch up with the rest. In addition, some receivers may temporarily fall behind because of various reasons such as network congestion or even network partition. There are two features in RMTP which together provide the functionality of allowing lagging receivers to catch up with the rest:

1) Immediate transmission request and

2) Data cache in the sender and the DR's.

3.7 Immediate Transmission Request

When a receiver joins late, it receives packets being multicast by the sender at that time, and by looking at the sequence number of those packets, it can immediately find out that it has missed earlier packets.

At that instant, it uses an ACK TXNOW packet to request its AP for immediate transmission of earlier packets. An ACK TXNOW packet differs from an ACK packet only in the packet type field. When an AP receives an ACK TXNOW packet from a receiver R, it checks bit vector V and immediately transmits the missed packet(s) to R using unicast.

3.8 Data Cache

RMTP allows receivers to join an ongoing session at any time and still receive the entire data reliably. However, this flexibility does not come without a price. In order to provide this feature, the senders and the DR's in RMTP need to buffer the entire file during the session. This allows receivers to request for the retransmission of any transmitted data from the corresponding AP. A two-level caching mechanism is used in RMTP. The most recent packets of data are cached in memory, and the rest are stored in disk.

3.9 Flow Control

A simple window-based flow control mechanism is not adequate in a reliable multicast transport protocol in the Internet environment. The main reason is that in the Internet multicast model, receivers can join or leave a multicast session without informing the sender. Thus a sender does not know who the receivers are at any instant during the lifetime of a multicast session [5].

Therefore if we want to design a transport-level protocol to ensure guaranteed delivery of data packets to all the current members of a multicast session, without explicitly knowing the members, a different technique for flow control is needed. Note that if RMTP used a simple window-based flow control mechanism, then the sender would have to know if all the DR's in level 1 have received the packets before the window is advanced. However, the sender may not know how many level 1 DR's are there, because the underlying multicast tree can change and either new DR's may be added to the multicast tree dynamically or old DR's may leave! Fail.

In order to deal with this situation, the sender operates in a cycle. The sender transmits a window full of new packets in the first cycle and in the beginning of the next cycle, it updates the send window and transmits as many new packets as there is room for in its send window. The window update is done as follows. Instead of making sure that each level 1 DR has received the packets, the sender makes sure that all the DR's, that have sent status messages within a given interval of time, have successfully received the relevant packets before advancing the lower end of its send window. Note that

the advancement of send window does not mean that the sender discards the packets outside the window. The packets are still kept in a cache to respond to retransmission requests. In addition, note that the sender never transmits more than a full window of packets during a fixed interval, thereby limiting the maximum transmission rate to Ws * Packet_Size/Tsend. This scheme of flow control can thus be referred to as rate-based windowed flow control.

3.10 Congestion Avoidance

RMTP provides mechanisms to avoid flooding an already congested network with new packets, without making the situation even worse. The scheme used in RMTP for detecting congestion is described below.

RMTP uses retransmission requests from receivers as an indication of possible network congestion [5]. The sender uses a congestion window cong_win to reduce data transmission rate when experiencing congestion. During Tsend, the sender computes the number of ACK's, N, with retransmission request. If exceeds a threshold, CONGtresh, it sets cong_win to one. Since the sender always computes a usable send window as Min(avail_win, cong_win), setting to one reduces data transmission rate to at most one data packet per Tsend if avail_win is nonzero. If N does not exceed CONGthres during Tsend, the sender increases cong_win by one until cong_win reaches Ws. The procedure of setting cong_win to one and linearly increasing cong_win is referred to as slow-start and is used in TCP implementation. The sender begins with a slow-start to wait for the ACK's from far away receivers to arrive.

3.11 Choice Of Dr's And Formation Of Local Regions

RMTP assumes that there is some information about the approximate location of receivers and based on that information, either some receivers or some servers are chosen as DR's. Although specific machines are chosen to act as DR's, the choice of an AP for a given local region is done dynamically. The basic idea is outlined below.

Each DR as well as the sender periodically sends a special packet, called the SEND ACK TOME packet, in which the time-to-live (TTL) field is set to a predetermined value (say 64), using the multicast tree down to each receiver. Thus, if there are several DR's along a given path from the sender to a given receiver, the receiver will receive several SEND ACK TOME packets, one from each DR. However, since the TTL value of an IP datagram gets decremented by one at each hop of the network, the closer a DR is to a given receiver, the higher is the TTL value in the corresponding SEND ACK TOME packet. Therefore, if each receiver chooses the DR, whose SEND ACK TOME packet has the largest TTL value, it will have chosen the DR nearest to it in terms of number of hops. Effectively, a local region will be defined around each DR.

This approach gives us several benefits in terms of robustness and multiple levels of hierarchy. First of all, if the DR, selected by a set of receivers as their AP, fails, then the same set of receivers will choose the DR least upstream from the failed DR, as their new AP [6]. This is because SEND ACK TOME packets from the failed DR will no longer arrive at the receivers and the SEND ACK TOME packet from the DR least upstream from the failed DR will have the largest TTL value. This leads to the dynamic selection of AP for a given set of receivers.

3.12 Multilevel Hierarchy in RMTP

RMTP has been described earlier as a two-tier system in which the sender multicasts to all receivers and DR's; and DR's retransmit lost packets to the receivers in their respective local regions. However, the limitations of a two-level hierarchy are obvious in terms of scalability and a multilevel hierarchy is desirable. The objective of this section is to describe how a multilevel hierarchy is obtained in RMTP with the help of the DR's sending SEND ACK TOME packets.

Recall that each DR periodically sends SEND ACK TOME packets along the multicast tree, and each receiver chooses the DR whose SEND ACK TOME packet has the largest TTL value. Moreover, note that each DR is also a receiver [5]. Therefore, if each DR ignores its own SEND ACK TOME packets, it will choose the DR least upstream from itself as its DR and will send its status messages to that DR during the multicast session. Fig. 5 illustrates the idea.

Effectively, if there are n DR's along a path from the sender to a group of receivers, and these DR's are different hop counts away from the receivers in question, there will be n local regions in an n-level hierarchy, such that the DR of the n th level will send its status to the DR in level n-1, a DR of level n-1 will send its status to the DR in level n-2, and so on, until the DR in level 1 sends its status to the sender (DR at level 0). That is, a DR at the ith level acts as a receiver for the i-1th level for all i, i=n,…., 1 where the zero level refers to the global multicast tree rooted at the sender.

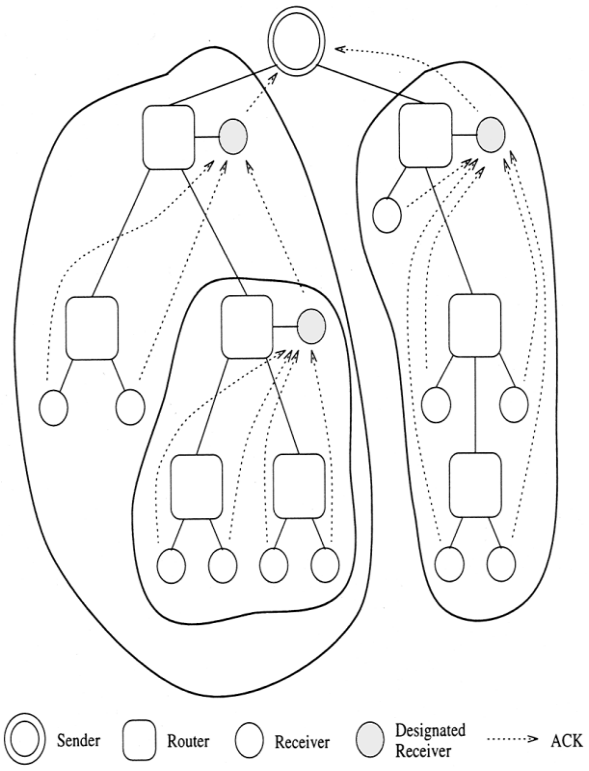


Fig 5: Multilevel Hierarchy of DR's.

IV. Performance Evaluation Routing Metrics

Routing table information used by switching software to select the best route. But how, specifically, are routing table built? What is the specific nature of the information that they content? How routing algorithms determine that one route is preferable to others?

Routing algorithms can use many different metrics to determine the best route. Sophisticated routing algorithms can base route selection on multiple metrics, combining them in single (hybrid) metric. All of the following metric has been used:

Route Acquisition Time

It is the time required to establish route(s) when requested and therefore is of good importance to on demand routing protocols.

Packet Delivery Ratio

The ratio of the data delivered to destination (i.e. throughput) to the data send out by the sources.

Average End-To-End Delay

The average time it takes for packet to reach the destination. It includes all possible delays in the source and each intermediate host, caused by routing discovery, queuing at the interface queue, transmission at the MAC layer, etc. Only successfully delivered packets are counted.

Power Consumption

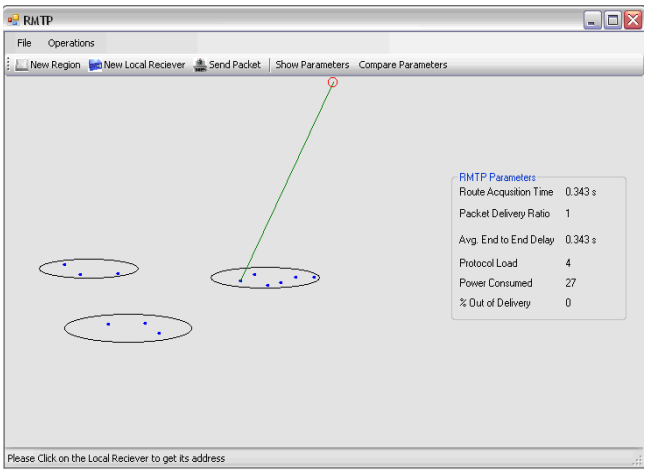
It is the total consumed energy divided by the number of delivered packets. We measure the power consumption because it is one of the precious commodities in mobile communication.

Protocol Load

The routing load per unit data successfully delivered to the destination. The routing load is measure as the number of protocol messages transmitted hop wise (i.e. the transmission on each hop is counted once). A unit data can be a byte or packet.

Percentage Out Of Order Delivery

An external measure of connectionless routing performance of particular interest to transport your protocol such as TCP, which prefer in order delivery.



V. Conclusion

We have presented in this paper the complete design and implementation of RMTP and also provided performance measurements of the actual implementation on the internet. The main contribution of the design include reducing the acknowledgement traffic. The design also include extension of two-level hierarchy to multilevel hierarchy of DR's in the Internet environment. It also includes the use of periodic status messages and the use of selective repeat retransmission mechanism to improve throughput.

The performance figure of RMTP implementation for the data transmission and calculation of parameters is also given in the paper.

References

[1] L.Aguilar, "Datagram routing for Internet multicasting," ACM Comp. Comm. Rev. vol. 14, no 2, pp. 58-63, 1984.

[2] S. Armstron, A. Freier, and K. Marzullo, "RFC-1301 Multicast transport protocol," Feb. 1992.

[3] R. Aiello, E. Pagani, and G. P. Rossi, "Design of a reliable multicast protocol," in proc. IEEE. INFOCOM '93, Mar. 1992, pp. 75-81.

[4] F. Adelstein and M. Singhal, "Real-time causal message ordering in multimedia systems," in Proc. 15th ICDCS '95, June 1995.

[5] J-M. Chang and N. F. Maxemchuk, "Reliable broadcast protocols," ACM Trans. Comp. Syst., vol. 2, no. 3, pp. 251-173, Aug. 1984.

[6] K. P. Birman and T. A. Joseph, "Reliable communication in the presence of failures," ACM Trans. Comp. Syst., vol 5, no. 1, Feb. 1987.

[7] D. Dolev and D. Malki,"The transis approach to high availability clster communication," Comm. ACM, vol. 39, no. 4, pp. 64-70, Apr. 1996.

[8] H. Eriksson, "MBONE: The multicast backbone," Comm. ACM, vol. 37, no. 8, pp. 54-60, Aug. 1994.

[9] J. C. Lin and S. Paul, "RMTP: A reliable multicast transport protocol," in Proc. IEEE INFOCOM '96, Mar. 1996, pp. 1414-1424.

[10] L. E. Moser, P. M. Melliar-Smith, D. A. Agarwal, R. K. Budhia, and C. C. Lingley-Papadopoulos, "Totem: Afault-tolerant multicast group communication system,"Comm. ACM, vol- 39, no. 4, pp. 54-63, Apr. 1996.

[11] S. Paul, K. K. Sabnani, and D. M. Kristol, "Multicast Transport protocols for high speed networks," in Proc. Int. Conf. Network Protocols, 1994, pp. 4-14.

[12] B. Rajagopalan, "Reliability and scaling issues in multicast communication," in Proc. ACM SIGCOMM '92, Sept. 1992, pp. 188-1989.