19.12 MCNA - A RELIABLE MULTICAST PROTOCOL FOR RADAR PRODUCT DISTRIBUTION

Zack Jing[1], Sallie Ahlert[1] and Gary Gookin[2]

[1]NOAA/NWS/Radar Operations Center (ROC), Norman, Oklahoma
[2]METI, 3200 Marshall Ave, Norman, Oklahoma

1. INTRODUCTION

The WSR88D products and base data, like many other operational meteorological data, are currently distributed on networks via "unicast". That is a copy of the data is sent to each receiver separately. If multiple receivers request the same data, multiple copies of the same data must be sent across the networks.

As the number of product users increases, the product distribution bandwidth must increase proportionally. The data transmission delay will also increase as more data have to be transmitted on the same network. Additional bandwidth will add operational cost and the increased data communications delay will reduce quality of operations.

IP multicast technology provides a possible solution to this problem. In a multicast network environment, data from a single source can be directed to multiple receivers without duplicated transmission on any of the physical lines in the network. The network uses minimum bandwidth to route the data to all registered receivers.

IP packet multicast capability is now available on many of the networks. The technology, however, has not yet often been used for operational data distribution. One of the reasons is because of reliability concerns. Data sent through IP multicast in the network are not guaranteed to reach receivers correctly and completely. Although reliable unicast protocol (e.g. TCP) has been available for many years, there is so far no standard protocol for reliable multicasting. Several experimental protocols have been proposed. These protocols are mostly designed for applications with large number of receivers and moderate reliability requirements.

Since the FAA is studying the possibility of receiving WSR-88D products via multicasting, the NWS Radar Operations Center (ROC) is currently designing and implementing a reliable multicasting protocol for distributing radar products. The protocol is specifically designed for distributing operational data with high reliability.

In this paper, we will briefly present the protocol, describe an implementation of the protocol and explain how it can be used in WSR88D product distribution. Preliminary test results indicate that reliable and efficient distribution of operational data via multicast may be both possible and practical.

* *Corresponding author address*: Zack Jing, Radar Operations Center, Engineering Branch, 3200 Marshall Avenue, Norman, OK 73072; e-mail: zack.z.jing@noaa.gov.

2. MULTICAST PROTOCOLS

Internet Protocol (IP) multicast is a bandwidth-conserving technology that reduces traffic by simultaneously delivering a single stream of information to many receivers. IP Multicast delivers source traffic to multiple receivers without adding any additional burden on the source or the receivers, while using the least network bandwidth of any competing technology. Multicast packets are replicated in the network by routers enabled with a multicast protocol and sent to registered receivers. Packets are transported in the network without duplication.

IP multicast protocols have been standardized for many years. They are supported by most modern computing equipment: Computers, Switches and Routers. Operational LANs and WANs can be configured to support IP multicast. The IP multicast protocols address how receivers register in the network and how data packets are routed to the destinations. IP Multicast technology, however, does not address the reliability issue. Data packets may be lost or become corrupted while being transported from the sender to the receiver. The IP Multicast protocols do not guarantee that a registered receiver always receives exactly the same data sent by the sender. Several experimental, reliable multicast protocols have been proposed to address the reliability issue.

Pragmatic General Multicast (PGM) is an experimental Protocol (RFC 3208, http://www.ietf.org/rfc.html) supported by Cisco. It is specifically intended as a workable solution for multicast applications with basic reliability requirements. A multicast source sends, via IP multicast, sequenced data packets and receives, via unicast, selective negative acknowledgments (NAKs) for data packets detected to be missing. The source then retransmits the missing packets. NAK provides the sole mechanism for reliability. The packet availability for retransmission depends on the transmit window size allocated on the sender side, as well as the data rate. PGM scales well in terms of number of receivers because it uses an efficient, built-in-router mechanism for transporting NAKs from the receivers to the sender. Thousands of receivers can be supported. PGM is targeted for applications such as stock and news updates, data conferencing and video transfer. PGM is best suited to those applications that are either insensitive to unrecoverable data packet loss or are prepared to resort to application recovery in such event.

Another NAK based protocol is NACK-Oriented Reliable

Multicast (NORM, INTERNET-DRAFT draft-ietf-rmt-pi-norm-10, http://www.ietf.org/html.charters/rmt-charter.html) proposed by researchers at the Naval Research Laboratory (NRL) and others. Unlike PGM, NORM does not require router-assisted NAK transportation which is a vendor-specific, non-standard router feature. Because of this, NORM is easier to deploy, but less scalable, in comparison with PGM. NORM, using IP multicast for NAKs, is an end-to-end approach. NORM offers basic reliability and ordered data delivery as PGM does. For more information about NORM, please refer to the following url http://pf.itd.nrl.navy.mil/projects.php?name=norm.

Instead of relying on NAK-based packet retransmission, Forward Error Correction (FEC) technology has also been proposed for increased reliability. Each packet is augmented with additional redundant data before it is transmitted to the network. On the receiving side, lost packets can be recovered from other received packets. FEC based technique offers the best scalability. It works well if the packet loss is random and of relatively low probability. An example of FEC based reliable multicast proposals is "Asynchronous Layered Coding (ALC) (RFC 3450, http://www.ietf.org/rfc)". Vincent Rosa et al (www.inrialpes.fr/planete/people/roca/mcl) have investigated the FEC technique called "Low Density Generator Matrix (LDGM)" which is more efficient than traditional Reed-Solomon erasure code for large data blocks. FEC based multicast typically adds about 50 percent overhead to the data rate. This bandwidth consumption overhead is present even if the network error rate is very low.

At this moment, there is no standard reliable multicast protocol. The proposed or experimental protocols are mostly targeted for distribution of data to a large number of receivers while achieving moderate reliability. Because the protocols are experimental, the sources of implementation and support are limited. Thus, most of the operational data (e.g. weather radar products) are transported through unicast, so far, although the multicast capability is often already available on many LANs, WANs and inter-networks.

3. MCNA (MULTICAST WITH NEGATIVE ACKNOWLEDGMENT - A MULTICAST PROTOCOL FOR RELIABLE DATA DISTRIBUTION

In the following we introduce a simple multicast protocol that is specifically designed for transporting data with high reliability. The design goals of the protocol are:

- Reliability. Mission or business critical data distribution often has more stringent reliability requirements than news and video. As long as bandwidth is available, data must be reliably received by all receivers, even if the network circuits are noisy or there is temporary network break down or congestion.

- Simplicity. A practical protocol must be simple to understand and implement such that it can be part of the application system and most of the application developers can develop and maintain it.

- Easy to deploy. An end-to-end protocol on top of standard IP multicast can be deployed on many existing LANs and WANs. The assumption about the network topology is minimal.

- Bandwidth efficiency. The protocol should add little overhead when the network is in normal condition, and traffic generated by NAKs and packet retransmission should be minimized.

- Scalability. Moderate scalability supporting hundreds of receivers is desirable. The number of direct receivers of operational weather radar data is typically less than that of news and publicly offered video. Note that, even if there are only two receivers, the network bandwidth savings can be substantial in many cases when compared to unicast. In the case of a large number of receivers, redistribution of the data at additional locations can often provide larger scale solution.

MCNA consists of the following.

- MCNA transports messages of arbitrary sizes. Each message is segmented to fit in IP multicast packets. A unique sequence number is associated with each packet. Packets are sent with IP multicast. On the receiving side, the IP packet payload is recombined to create the original messages. MCNA assumes that the packets received may not be in the order they have been sent. Because MCNA is on top of the UDP protocol, detecting corrupted packet is not necessary (UDP drops any corrupted data).

- A receiver sends NAK messages to the sender to request retransmission of missing packets. The receiver detects missing packets through the packet sequence numbers and through a timing scheme. The sender resends missing packets requested by one or more receivers through IP multicast. A receiver repeats NAKing until all missing packets are received.

- The NAK messages are sent from receivers to the sender through IP multicast. Each NAK message is sent with an exponentially distributed random time delay to implement "NAK suppression" (Nonnemacher and Biersack, 1998). That is, if a number of receivers request the same retransmission, only the earliest responding receiver actually sends the NAK message and all others will back off upon receiving the NAK message from the earliest responder. Each NAK message can contain NAKs of multiple missing packets.

- The sender does not resend a lost packet immediately after receiving a NAK from a receiver. Instead, it does so only after a specified delay. This reduces the probability of resending the same lost packet multiple times. The delay time increases as the number of retransmissions of the same packet increases.

- The sender sends a special "I-am-alive" packet periodically if there is no data packet to send. This helps the receivers detect any lost packet and a dead sender.

- The sender controls the data rate of IP multicast according to a specified maximum rate. This is designed to reduce the possibility of packet loss in the routers due to buffer overflow. It also reduces the likelihood of network congestion problems. If the sender needs to send both new data packets and retransmission packets, the allowable bandwidth is equally divided between them.

- The sender notifies the receivers if a requested retransmission packet is not available so no further NAK is generated. The sender also notifies the receivers before termination so no further NAK is generated. Three termination packets are sent in a row to increase the probability of reception.

- On the receiving side, only missing packets within a specified time period of interest (TPI) are NAKed. This eliminates NAKs and packet retransmission of old data in which receivers are no longer interested.

- A receiver times out a quiescent sender to suppress NAKs in case where either the sender fails to send termination packets or the receiver fails to receive any of termination packets.

- Messages are delivered to the application and discarded from the buffer immediately upon complete reception. The messages may not be delivered in the same sequence they are originally sent. This minimizes the message transport delay. Because the receiving buffer is only used for storing incomplete messages, the probability of buffer overflow is reduced and, consequently, the reliability is increased.

- The sender may use external storage for providing retransmission data older than the sending buffer can hold.

- MCNA has built-in data compression. The compression routine is user-provided.

- Multiple senders are supported in the same multicast group.

- A sender or a receiver can join and leave the group at any time without disturbing others. In order to do this, a sender is uniquely identified by its source IP address, a random number deduced from its start time and a user specified sender ID number. When a sender leaves, any retransmission from that sender becomes impossible.

The characteristics above are descriptive of the approach designed in MCNA for reliability and conservation of bandwidth resource. Because the NAK is repeated until a missing packet is received, no data will be lost if both sending and receiving buffers are sufficiently large and the bandwidth resource is available. If we choose to use externally stored (e.g. in data files or databases) data for retransmission, the data availability for retransmission can be practically unlimited. Because only partially received messages need to be buffered on the receiving side, the probability of receiving buffer overflow is reduced substantially. For example, the large number of missing packets in an extended network outage will not take any space from the receiving buffer.

In normal cases, the packet loss rate should be small and MCNA adds very little overhead to the bandwidth consumption. NAK messages and retransmissions happen infrequently. "I-am-alive" packets may add some load to the network, if the messages are sent intermittently. This load, however, is only present when there is no data to send and the network is in low-utilization. The substantial problem of packet loss happens in the following cases:

a. Part of the network is overloaded.

b. Some of the network devices or physical links are not in normal operating condition.

c. Some of the network devices or physical links are broken.

In such cases, the number of NAK messages and retransmitted data packets may become large, generating substantial load to the network. Several techniques have been introduced to MCNA to minimize NAK messages and redundant retransmissions. The more efficient use of the network, the better chance there is to survive network problems. Another issue is so-called “NAK implosion”. The NAK suppression technique in MCNA reduces NAK messages in case the packet loss happens close to the sender (all receivers lose the same packets). However, when the number of receivers increases, the number of NAK messages will increase in general. It is expected that MCNA will be able to support hundreds of receivers.

4. AN MCNA IMPLEMENTATION

An implementation of the MCNA has been completed at the ROC. We chose to implement the protocol in the form of an Application Programming Interface (API) and the associated software library. The choice of API over a background program (daemon process) is to offer flexibility for building applications for different systems and run-time environments. The design goal is a simple and easy to use API. Buffer sizes, IP multicast setups and MCNA parameters are all configurable. Multiple sender and receiver sessions can be opened in the same process. Status and statistics information is available through the API. The implementation must be efficient in terms of CPU and memory resource utilizations.

The API consists of the following functions.

int **MCNA_open** (char *ip, int port,
void (*receive_func)(int, char *, int, int));

```
int MCNA_send_message (int sd, char *msg, int msg_len);
int MCNA_set_options (int sd,
                      const char *optname, int value);
void MCNA_main_loop (void (*application_func)(void));
MCNA_statistics_t *MCNA_get_statistics (int sd);
MCNA_status_t *MCNA_get_status (int sd);
int MCNA_close (int sd);
int MCNA_set_external_buffer (int sd,
                      int (*get_data_func)(int, int, int, char *),
                      int *size, char **msg);
```

MCNA_open opens a multicast session for group IP address "ip" for sending and/or receiving messages. MCNA_open, if successful, returns a non-negative number called session ID which will be used in other MCNA function calls to identify this open session. The "port" parameter is the UDP port number used for the group. If the session is opened for receiving messages, one must supply a call-back function "receive_func", which will be called when a message is completely received. In the function, one can process the incoming message or simply save it in a file. The receiving function must have the following interface:

```
void receive_func (int sd, char *msg,
                      int msg_len, int sender_id);
```

When a message is delivered to the application through "receive_func", the session ID ("sd"), a pointer to the message ("msg"), the message length in number of bytes ("msg_len"), and the sender ID ("sender_id"), are passed to the function.

For multicasting a message to a group, one first opens a session by callling MCNA_open. Then, one calls MCNA_send_message to pass the message to MCNA for sending.

The application must pass program control to the MCNA library routines so the MCNA can perform real-time processing. This is done by calling MCNA_main_loop. Function MCNA_main_loop takes control of the program and never returns. In order for the application to do useful jobs such as processing incoming messages, sending outgoing messages, opening and closing sessions and checking status or statistics, the application passes an application function through argument "application_func" to MCNA. "application_func" is then called frequently from MCNA after the application passes control to MCNA. All of the above mentioned tasks can be performed in "application_func".

MCNA_set_options sets various options and MCNA parameters for an open session. One can use this function to customize the behaviors for each open multicast group. MCNA_get_status and MCNA_get_statistics retrieve, respectively, the current status and statistics data for a multicast session.

MCNA_close closes a multicast session if it is no longer needed. The application leaves the multicast group and all resources allocated for the session are freed.

MCNA_set_external_buffer can be used for setting up an interface to get retransmission data from a source external to MCNA.

Developing an application for sending or receiving multicast messages is simple. The following code segment shows how to write an application that sends multicast messages to one group (IP address "239.25.172.1") and receives messages from another group (IP address "239.25.172.2").

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <mcna.h>

static int Sd1, Sd2;
static void Application_func ();
static void Receive_func (int id, char *msg,
                          int msg_len, int sender_id);

int main (int argc, char *argv[]) {

   Sd1 = MCNA_open ("239.25.172.1", 15213, NULL);
   if (Sd1 < 0) {
      printf ("MCNA_open failed (error code %d)\n", Sd1);
      exit (1);
   }
   Sd2 = MCNA_open
            ("239.25.172.2", 15214, Receive_func);
   if (Sd2 < 0) {
      printf ("MCNA_open failed (error code %d)\n", Sd2);
      exit (1);
   }
   MCNA_main_loop (Application_func);

   exit (0);
}

static void Application_func () {
   int ret_value, msg_len;
   char *new_msg;

   ... Getting a new message "new_msg" for multicast ...
   ret_value = MCNA_send_message
                          (Sd1, new_msg, msg_len);
   if (ret_value < 0)
      printf ("MCNA_send_message failed (code %d)\n",
                                  ret_value);
}

static void Receive_func (int id, char *msg,
                          int msg_len, int sender_id) {
   if (msg_len < 0) {
      printf ("Exception code %d received (%s)\n",
                                  msg_len, msg);
      return;
   }
   ... process or save message "msg" of "msg_len" bytes ...
}
```

5. PRELIMINARY TEST RESULTS OF APPLYING MCNA TO THE WSR-88D PRODUCT DISTRIBUTION

The ROC is developing a prototype multicast-based product distribution support for the WSR-88D Radar Product Generator (RPG). Up to 10 product groups can be configured. Each group contains a set of selected products and uses a unique multicast group (IP address) for distributing these products. The RPG's product multicast application uses the MCNA API for sending products to the network. Radar product user receiver applications then can be developed with the MCNA API. A simple user application has been built for test and demonstration purposes. User applications can receive products from any particular radar by joining the appropriate multicast groups.

A simple test network has been set up in the ROC for testing the system. The test environment consists of one RPG host and two client hosts connected with routers and serial connections. One of the clients is a SPARC machine from SUN Microsystems and the other is a PC running Linux. Test user applications run on both client hosts to receive products from all 10 groups.

Preliminary test results have been encouraging. In one of the tests, we randomly dropped 10 percent of the data packets on both the sending and receiving sides to simulate a noisy network. In a continuous 30 hour operation, more than 30,000 products were generated and distributed to the clients. All products were successfully received by the two clients. In another test, to simulate a network failure, we turned off one of the routers for one hour while the RPG continued generating products. When the router was powered back up, the IP multicast service resumed automatically. Both users received all products generated during the network outage through packet retransmission.

6. ACKNOWLEDGMENTS

Jose Marcos, John Moyer, Courtenay Clifford, Bill Bumgarner and many others from the FAA made numerous contributions to both the MCNA protocol and the operational concepts of multicast based radar product distribution.

7. REFERENCES

Nonnemacher, J and Biersack, E.W., 1998: Optimal Multicast Feedback, Proc. IEEE INFOCOMM, 1998.

PGM, RFC 3208, http://www.ietf.org/rfc.html

ALC, RFC 3450, http://www.ietf.org/rfc

NORM, INTERNET-DRAFT draft-ietf-rmt-pi-norm-10, http://www.ietf.org/html.charters/rmt-charter.html