

Design and Verification of Multicast Protocol



EECS700 – System Requirement Modeling
Instructor: Dr Perry Alexander

Done by,

Shankar Palanivel – 706506
Vijayaraghavan Rajan – 705073
Kandaswamy Pazhanivelan Muthuvelan – 703847

1. Objective

The objective of the project is to specify and verify a simple multicast protocol. The project defines specifications for nodes, multicast routers and a network of such devices. The various features supported are creation/ deletion of multicast group and addition/deletion of a nodes in a multicast group.

2. Introduction to Multicast

A simple multicast protocol is designed for the project. The project assumes presence of two types of networks – local network & core network. The local network consists of nodes/hosts and a designated router. A local network forms a particular domain. There is only a single designated router for a local network. All the designated routers of each local network are connected to each other through the core network. The core network only consists of the designated routers. The purpose of the core network is to transfer packets destined to different domains (Fig 1).

Both the local networks and the core network is assumed to be broadcast networks(like Ethernet). All the devices in the network can see the packets being sent in the network. But only receive those addressed to them.

There are two types of packets that can originate from a node in a network – unicast & multicast packet.

1. If a unicast packet is to be delivered to a node in the same domain then it is broadcast in the local network. The corresponding node in the network receives it from the local network.

If a unicast packet is to be delivered to a node in a different domain then it is received by the designated router from the local network and rebroadcast into the core network. The designated routers listening to the core network receive only those

packets that are addressed to nodes in its local network (domain). The designated router then broadcasts the packet in its local network. The corresponding node in the destination domain then receives the packet.

2. If a multicast packet is sent by a node in a local network, then it is received by the designated router of the domain. The designated router first sees the multicast address in the packet. It then looks up its group membership table to find the members of the group. Each router maintains a group membership table that only lists the members that are in its domain. The designated router then sends unicast packets to the member nodes in its domain. It also has to send the multicast packets to other routers in the core network.. As a multicast group can have members belonging to different domains. When a router receives a multicast packet from the core network it looks up its group membership table to send unicast packets to corresponding member nodes.

There are also multicast group information to be distributed across all networks. The following are the other different types of packets that can be sent in a network

1. Add Group

The ADDGROUP packets are sent by the administrator for adding different multicast groups. The ADDGROUP packets are broadcast in the core network. All the routers then update their group membership table to include the new multicast group.

2. Delete Group

The DELETEGROUP packets are sent by the administrator for deleting different multicast groups. The DELETEGROUP packets are broadcast in the core network. All the routers then update their group membership table to delete the multicast group.

3. Add Member

The ADDMEMBER packets are generated by different nodes in different domains. Each node requests its designated router to include it in a particular multicast group. Hence the node sends the ADDMEMBER packet in its own network. The designated router then reads the packet to update the group membership table.

4. Delete Member

The DELETEMEMBER packets are generated by different nodes in different domains. Each node requests its designated router to delete it from a particular multicast group. Hence the node sends the DELETEMEMBER packet in its own network. The designated router then reads the packet to update the group membership table.

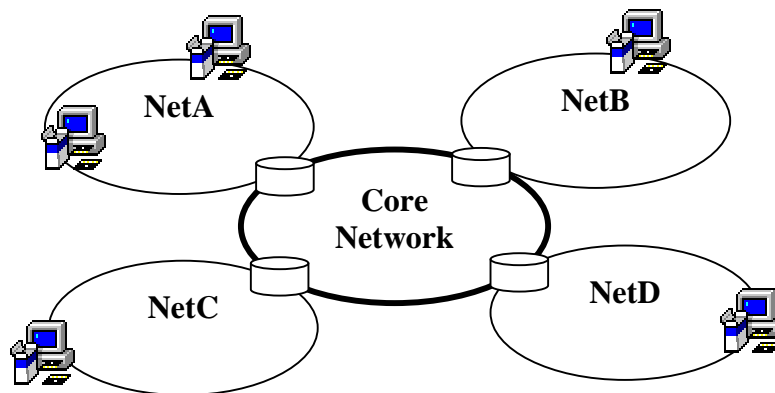


Figure 1: General Network Design

3. Design

The above protocol has to be modeled using the features provides by the SPIN. The various things that are to be modeled are as follows

1. Network

Since both the core network and all the local networks are considered as broadcast network, the network is model as a channel. The channel for the router and the network has two objects – packet type and packet. The packet type is can be any of the following - DATA, ADDGROUP, DELETEDGROUP, ADDMEMBER and DELETEDMEMBER. The packet is defined as below

```
typedef Packet
{
    int srcAddr;
    int srcDomain;
    int destAddr;
    int destDomain;
    int multicast;
    int data;
}
```

Depending upon the packet type the rest of the elements are processed.

2. Router

Each router is modeled as a process that listens both to its core network and its local network. Both these networks are modeled as channels. The router listens to the network channel. It takes multicast packets from the local network. It only takes the unicast packets from its network when its destination domain is not the same as the

local network. It is because the local nodes directly receive the unicast packet if it is originated in the same local network/domain.

The router reads packets from the core network if its destination domain is same as that of the router. The router should process the receiving packets and switch them to appropriate networks or performs management operations. This processing of packet is explained below.

3. Node

The node is represented by a process, which listens to the local channel and receives only unicast packets that are destined to it.

4. Processing of Packets

The processing of different packets are explained below

1. DATA

When a router sees a unicast packet in the core with the same destination domain, then it switches that packet to the local network. Similarly if it sees a unicast packet in the local network with different destination domain then it switches it into the core network

When it sees a multicast packet in the local network, it resends packets to the members in the local network as unicast packets. It also sends the multicast packets to other routers in the core network. When it sees a multicast packet in the core network, then it takes the packet as sends the unicast packet to its members in the local network.

2. ADDGROUP

When a router sees an ADDGROUP packet in the core network with the destination same as the routers domain then it adds the multicast group in the group membership table which is organized as below.

```
typedef RoutingTable
{
    int gname;
    bool gmembers[MAXNODES];
}
```

RoutingTable groups [MAXGROUPS];

The gname in the RoutingTable structure is NULL then it means the absence of a multicast address. Hence while processing the packet it check for the first instance where gname is NULL and assigns the multicast address. Each multicast addr is associated the array of bools indexed by the host name. Hence when a group is created all the nodes are initialized to false.

3. DELETEGROUP

When a router sees a DELETEGROUP packet in the core network with the destination same as the router domain then it deletes the multicast group in the group membership table. It searches for the appropriate multicast addr as sets it to NULL to indicate the deletion of the group.

4. ADDMEMBER

When a router sees a ADDMEMBER packet from the local network, then it changes the group membership table by setting the gmembers[srcAddr] to the value true for the multicast address mentioned in the packet.

5. DELETEMEMBER

When a router sees a DELETEMEMBER packet from the local network, then it changes the group membership table by setting the gmembers[srcAddr] to the value false for the multicast address mentioned in the packet.

5. Salient features of the Project

The salient features of the project are indicated as below

- Though the basic structure of having a core network and number of local networks is fixed. The main feature is to add any number of local networks to the core network. To each local network any number of nodes can be added.
- In order to add routers or nodes only the init process needs to be changed, i.e. it is general to certain extent.

6. Problems Encountered

The various problems encountered are listed below

- One of the main problems encountered is the race condition. When group management packet is sent before a data packet, the data packet may be received

before the group management packet. In order to avoid such problems semaphores are used.

- The channel used does not actually does not perform the function of a broadcast network. Hence, we had to duplicate packets for each router.

7. Verification

The simple multicast packet designed was verified for various properties using spin as a model checker. The various criteria that were checked are as follows

1. Deadlocks

In the system, the router process listens to the local and the core network channels for any packets. The node process also listens to the local network channel for any packets. Both these processes do their operations in a do..od loop. In spin whenever a process continuously executes the do.. od loop without doing any useful work then it is considered as a deadlock. But in the above case, there is no deadlock situation. Hence we indicate this to the spin model checker by adding an end label to the starting of both the do..od loops. Spin does not consider this as a deadlock situation.

2. Progress

whenever a node in the network receives a packet from the network then we can say that the system is making progress. This progress state is done in spin by putting a progress label to the statement where the node process reads the packet from the network.

3. Assertion

Assertion is statement which needs to be true in particular state. In the multicast cast protocol modeled, it is modeled such that a multicast group must be created before a multicast packet can be sent with that multicast address. Hence we write a assertion which says that the destination multicast address of a packet is already present in the group membership table.

4. Never Claims

Never claims are those which indicate the temporal ordering of events that has to be prevented for the system to be functioning properly. One of the important property of the system modeled is that whenever a packet is send by a sender it must be received by a node.

8. Testing

The properties that are to be supported by the system are transferring of unicast packets, multicast packets, addition of groups, deletion of groups, addition of nodes and deletion of nodes. The init process is written as below

```
init
{
chan network[4]      = [25] of { mtype, int, int , int , int, int , int };
chan core           = [30] of { mtype, int, int , int , int, int , int };

    domains[0]=NetA;
    domains[1]=NetB;
    domains[2]=NetC;
    domains[3]=NetD;
```

```
no_of_domains = 4;
sema =0;

run router(NetA, network[0], core);
run router(NetB, network[1], core);
run router(NetC, network[2], core);
run router(NetD, network[3], core);

node1 = run node( NetA, 1, network[0] );
node2 = run node( NetB, 1, network[1] );
node3 = run node( NetB, 2, network[1] );
node4 = run node( NetC, 1, network[2] );
node5 = run node( NetD, 1, network[3] );

run addgroup( core, 1000);

sema == 4;

run addgroup( core, 2000);

sema == 8;

run deletegroup( core, 1000);

sema == 12;

run addmember( network[0], 1, NetA, 2000);
run addmember( network[1], 1, NetB, 2000);
run addmember( network[1], 2, NetB, 2000);
run addmember( network[2], 1, NetC, 2000);
run addmember( network[3], 1, NetD, 2000);
```

```

sema == 17;

run deletemember( network[1],1, NetB, 2000);
run deletemember( network[3],1, NetD, 2000 );

sema == 18;

sendpid1 = run sendData( 2, NetA, 1 , NetB, UNICAST, 143, network[0] );
run sendData( 1, NetC, 1 , NetD, UNICAST, 341, network[2] );
run sendData( 2, NetB, 2000, MULTICAST, MULTICAST, 789, network[1] );

}

```

We can actually add any number of router to the core. Here we have added four routers associated with four local networks/domains. We have added some nodes to each of the domains. We added two multicast groups and deleted one of them. We then deleted some of the nodes in the local networks.

We then send some unicast packets across domains and sent multicast packets also across domains. The results are as follows

```

Node UP!!! addr: 1 domain: 131
Node UP!!! addr: 1 domain: 130
Node UP!!! addr: 1 domain: 129
Node UP!!! addr: 2 domain: 130
Node UP!!! addr: 1 domain: 132
Router: 129 Packet received from the core
Router: 130 Packet received from the core

```

Router: 129 Add Group Packet Received
Router: 129 Group Added = 1000
Router: 131 Packet received from the core
Router: 130 Add Group Packet Received
Router: 130 Group Added = 1000
Router: 132 Packet received from the core
Router: 131 Add Group Packet Received
Router: 131 Group Added = 1000
Router: 132 Add Group Packet Received
Router: 132 Group Added = 1000
Router: 129 Packet received from the core
Router: 129 Add Group Packet Received
Router: 129 existing group 1000
Router: 129 Group Added = 2000
Router: 130 Packet received from the core
Router: 130 Add Group Packet Received
Router: 130 existing group 1000
Router: 130 Group Added = 2000
Router: 131 Packet received from the core
Router: 131 Add Group Packet Received
Router: 131 existing group 1000
Router: 131 Group Added = 2000
Router: 132 Packet received from the core
Router: 132 Add Group Packet Received
Router: 132 existing group 1000
Router: 132 Group Added = 2000
Router: 129 Packet received from the core
Router: 129 Delete Group Packet Received
Router: 130 Packet received from the core
Router: 129 Group Deleted = 1000
Router: 130 Delete Group Packet Received

Router: 130 Group Deleted = 1000

Router: 131 Packet received from the core

Router: 131 Delete Group Packet Received

Router: 132 Packet received from the core

Router: 131 Group Deleted = 1000

Router: 132 Delete Group Packet Received

Router: 132 Group Deleted = 1000

Router: 130 Member = 1 added to Group = 2000

Router: 129 Member = 1 added to Group = 2000

Router: 131 Member = 1 added to Group = 2000

Router: 132 Member = 1 added to Group = 2000

Router: 130 Member = 2 added to Group = 2000

Router: 132 Member = 1 deleted from Group = 2000

Router: 130 Member = 1 deleted from Group = 2000

Router: 131 Unicast packet send to addr= 1 destination domain= 132.

Router: 130 Multicast packet destined to multicast addr= 2000 received from local network.

Router: 129 Unicast packet send to addr= 1 destination domain= 130.

Router: 130 Multicast packet resend to other routers in the core domain = 129.

Router: 130 Multicast packet resend to other routers in the core domain = 131.

Router: 130 Multicast packet resend to other routers in the core domain = 132.

Router: 132 Packet received from the core

Router: 130 Packet received from the core

Router: 132 Unicast Packet written to network srcAddr = 1 srcDomain = 131

Router: 129 Packet received from the core

Node Process.

Node : 1 , Domain : 132

Packet from srcAddr=1 srcDomain=131 destAddr=1 destDomain=132 Data=341

Router: 131 Packet received from the core

Router: 132 Packet received from the core

Router: 129 Multicast Packet with addr =2000 received from core

Router: 129 Multicast translated into unicast packet and sent to destaddr=1 destDomain=
129

Router: 132 Multicast Packet with addr =2000 received from core

Router: 130 Unicast Packet written to network srcAddr = 2 srcDomain = 129

Node Process.

Node : 1 , Domain : 129

Packet from srcAddr=2 srcDomain=130 destAddr=1 destDomain=129 Data=789

Node Process.

Node : 1 , Domain : 130

Packet from srcAddr=2 srcDomain=129 destAddr=1 destDomain=130 Data=143

Router: 131 Multicast Packet with addr =2000 received from core

Router: 131 Multicast translated into unicast packet and sent to destaddr=1 destDomain=
131

Node Process.

Node : 1 , Domain : 131

Packet from srcAddr=2 srcDomain=130 destAddr=1 destDomain=131 Data=789

9. References

- Gerard J. Holzmann, "Design and validation of computer protocols", Prentice Hall, 1991.
- Spin online documentation, <http://cm.bell-labs.com/cm/cs/what/spin>.