VERIFICATION OF A NEW SCALABLE IP/ATM MULTICAST ROUTING PROTOCOL

Krisztián Kiss, Gábor Rétvári

High Speed Networks Laboratory
Department of Telecommunications and Telematics
Technical University of Budapest
H-1117 Budapest Pázmány P. sétány 1/D.
Telephone: +36-20-9849-480, +36-30-9764-886
E-mail: {kiss, retvari}@ttt-atm.ttt.bme.hu

Abstract

This paper summarizes the work for defining and starting to verify a new protocol. The goal was to create a multicast routing protocol, which works in an IP over ATM network and – unlike the existing ones – is truly scalable. After studying the literature, we realized that no existing protocol is fully suitable for these requirements, so modifying an existing one we created a new routing protocol, and started to verify it using a formal description tool. In the first chapter we discuss the necessity of developing such a protocol. Next we provide the goals for the new protocol and the results of the studies of the existing proposals moreover some words are told about the essentials of our new protocol. In the third chapter we summarize the pros and the contras of the most scalable proposal: SEAM. In the fourth part of this documentation we show, how to manage the multicast tree in order to reach the best efficiency of network resource utilization. In the fifth chapter we provide a brief summary of the formal description of the new protocol. Finally in the sixth chapter we summarize the results of our verification studies.

Keywords: Multicast, routing protocol, IP over ATM, CBT, scalable, verification

1. INTRODUCTION

With the increasing need of transmitting multimedia communication, such as a videoconference, through local area networks and also wide area networks, like the Internet, multicasting became a key topic by now. It saves bandwidth if the same data has to be transferred simultaneously to several destinations by sending only one copy of the data stream from the source, and duplicating it only at the nodes of the network, where it is really necessary: where paths to different destinations fork.

Multicasting in IP (the network layer of the protocol stack used in the Internet) is already in an experimental phase: there are different working multicast routing protocols, however, commercial usage of them is just about to begin [1]. The problem with these protocols is that they suppose, that the underlying data link layer supports broadcast data distribution. This is true for medium like Ethernet, but is not for ATM [2].

2. THE NEW PROTOCOL

Our aim was to work out a protocol, which supports multicast in an IP over ATM environment. We wanted to have a truly *scalable* multicast protocol, which *establishes multicast tree at ATM level* independently from the network layer, but provides *support for* special needs of *IP*. There have already been different proposals for this problem in existence (e.g. MARS [3], [4], [5], VENUS [6], CONGRESS [7], IP–SENATE [8]). Let us see some aspects of one of these solutions: MARS (Multicast Address Resolution Server). MARS works in a limited cluster, so it is not really scalable for wide area. This is a problem, because if one wants to have IP over ATM multicasting in a large ATM network using this solution, he/she must divide the ATM network into small clusters, and use a conventional IP multicast router between the clusters. This results in the segmentation/re-assembly of the IP packets in every router, which means unnecessary additional cost and delay. VENUS deals with the problem of establishing *shortcut* connections between clusters, which implies the avoiding of segmentation/re-assembly method at the borders.

Our solution has two main parts. One is based on an existing proposal, called SEAM. SEAM is basically a proposal for multipoint-to-multipoint connections in ATM network, and will be detailed below. The other part of our solution is called MNS, which is responsible for making connection between the ATM layer (enhanced with SEAM) and the IP layer, and will be detailed in this documentation as

well.

SEAM [9] stands for Scalable and Efficient ATM Multicast. The aim of SEAM is to achieve truly scalable multicasting over ATM networks. Unlike other proposals here it is done by modifying the underlying ATM structure: in fact SEAM is a proposal for realizing multipoint-to-multipoint connections in ATM. This is a new approach, since other proposals, like MARS for example, use only the given ATM services, that is point-to-point and point-to-multipoint connections. A side effect of this new approach is that it does not require the above layer to be IP, any similar layer can take advantage of the functionality of SEAM.

Most IP/ATM multicast protocol proposals (e. g. MARS) use the concept of „source initiated tree". This means, that a multipoint-to-multipoint tree is separated into distinct point-to-multipoint trees, which are managed by their sources. It implies, that adding a new member to the multicast group is a process with increased network resource consumption: join message has to be sent to all sources and these sources add the new member to their trees. Disconnecting from a multicast group or changing the state of an endpoint (becoming sender from a receiver or vica versa) means a process with similar resource consumption to the join process's one, which acts against scalability. In SEAM multipoint-to-multipoint connections are realized with a so-called Core Based Tree [10], which is a special spanning tree of all the senders and receivers of a connection. Unlike „source initiated tree", CBT is special, because it is organized around a central node, called the „core". Data communication takes place within the spanning tree; data are forwarded from senders to receivers over the links, which are branches of the tree.

The concept of CBT is now presumed known by the reader. To increase the efficiency of the CBT SEAM uses a special kind of traffic control. It is accomplished by associating a bit with every link in the multicast tree. This bit is called Receivers Downstream (RD) bit and its role is to show if there are any receivers downstream from the core on a certain branch of the tree. If there is any (RD = 1), then traffic is passed through in that direction, but if there is not any receivers (RD = 0) that way (i. e. only senders take place on that branch), then multicast traffic is not routed in that direction. This enhancement of CBT means great deal of bandwidth save in comparison with „simple" CBT. On the other hand this solution leads to the disadvantage, that all traffic is routed to the core of the tree – even if there is not really need for it.

Using CBT concept for multicasting has many advantages, which are necessary for scalability: on one hand this way there is no need for a membership repository, which would be a bottleneck. Member initiated joins and leaves are also possible, which is necessary for the scalable membership management. Moreover this solution would also minimize the state information the network should store: every node has to know only about its neighboring nodes. On the other hand CBT may cause additional delays in data forwarding because of the suboptimal routing (data packets are not routed at the shortest path).

So what consequences does it have to implement the Core Based Tree algorithm on ATM level? ATM signalling certainly has to be modified to realize the new functionality. New state information is needed in the switches to store the topology of the tree: switching tables will be more complex and other information has to be stored as well, such as the address of the multicast group (to make joins possible). Data forwarding will be changed heavily: current switches do not support merging traffic from multiple incoming VCs into one (or more) outgoing VC.

There are many important issues that are not addressed by SEAM, or mentioned only at a very shallow level. These include questions like: processes for multicast tree management, algorithm for core selection, core management (relocation of the core, replication of the core for fault tolerance reasons), considerations of multiple active cores, and finally, the distribution of the address of the core. This last item means, that an algorithm should be worked out, which enables the possible group members (or their switches) the map from the IP address of the multicast group to the ATM address of the core.

We have made many enhancements to SEAM. Let us emphasize here our solution for the last problem: the core address distribution. Our proposed protocol for this purpose is called Multicast Network Service or MNS for short. Other aim for MNS is to advance the class D IP address resolution – when creating a new multicast group how to find a group address, which is free in the very moment? (This is a hot topic at IETF – some protocol proposals have already been done, but CBT based multicast protocols, such as PIM-SM or SEAM do not mention this question).

MNS can be thought of like a multicast pair of the well known Domain Name Service (DNS): a hierarchy of Multicast Name Service servers (MNS servers) will be responsible for answering the queries about core addresses of IP multicast groups. Just as with DNS, MNS servers will work by passing queries between each other, however unlike the DNS, this system is dynamic. The core point for a multicast group must be registered with the MNS server responsible for that group. As the first approach, if a query arrives to an MNS server about a group, that has no core specified, the switch that sent the query would be elected as the core. This switch may or may

not accept this. In the latter case the tree will not be set up and no communication will be available until a switch accepts the core role.

Certainly the SEAM/MNS system must be able to co-exist with traditional IP multicast protocols, such as DVMRP or PIM. It should be done by using border routers, which know both of the protocols and can translate between them: such a router should behave like a SEAM/MNS host on one side, and like an IP multicast router on the other side.

3. PROS AND CONTRAS OF SEAM

Most of the advantages of SEAM are originated of the scalable nature of CBT. These advantages are the following:

- there is no need for aggregated membership repository,
- reduced network resource consumption,
- stores state information only for multicast groups and not for every source,
- independence from the network layer.

Disadvantages:

- traffic concentration,
- increased delay because of sub-optimal routing,
- need for modification of ATM signalling
- requires VC merging (proposed solution: „cut-through forwarding"),
- lack of core management,
- insufficient documentation.

Taking these advantages and disadvantages into consideration we decided to apply the basics of SEAM with some kind of enhancements added to it in order to have an IP/ATM multicast routing protocol, which is really scalable.

Let us see a list of the enhancements developed SEAM with:

- re-interpretation of Receivers Downstream (RD) bit's role: RD is associated with ports, not with links,
- multicast tree management algorithms,
- protocol messages were defined,
- Multicast Name Service (MNS).

4. CBT MANAGEMENT

As it was mentioned earlier, SEAM's proposal for multicast traffic control results in the disadvantage, that all traffic is routed through the core – therefore core switch may act as a bottleneck. Figure 1. shows a scenario with the actual value of RD bits (which are associated with links). The dashed arrows show the suboptimal data forwarding scheme towards the core.

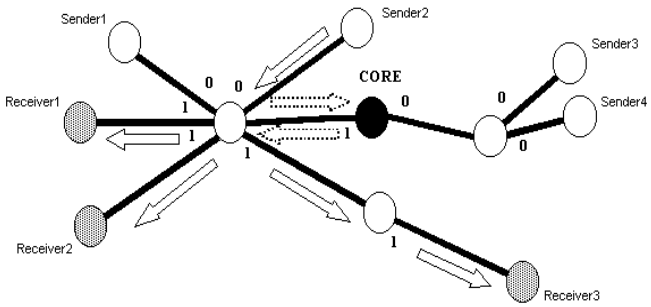

Figure 1.: Multicast routing by associating RD bits to links (only Sender_2 is active)

Our proposal is to have RD bit associated with all ports of the ATM switches (i. e. two RD bits per links). In this case multicast traffic can be passed through the network independently from the core, which means increased scalability. Management of the CBT means the setting of RD bits according to the actual state of the multicast group. Our algorithms detailed below carry this out. Figure 2. shows the same scenario as Figure 1. with the actual value of RD bits.

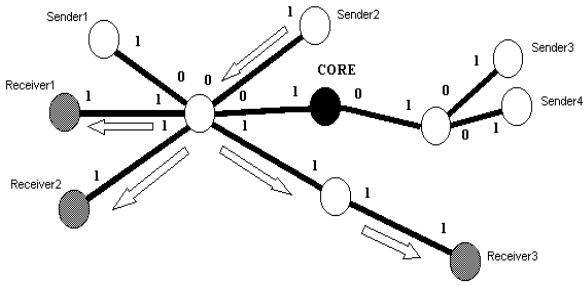

Figure 2.: Optimal multicast routing by associating RD bits to ports (only Sender_2 is active)

Management of the CBT can be achieved by invoking the four tree management algorithms, which makes the protocol able to track group membership changes. These membership changes can be as follows: change of an endpoint's state, adding a new branch (i. e. a new group member) to the tree and removing a branch from the tree.

Figure 3. shows the situation, when an endpoint, that was formerly a sender, becomes a receiver:

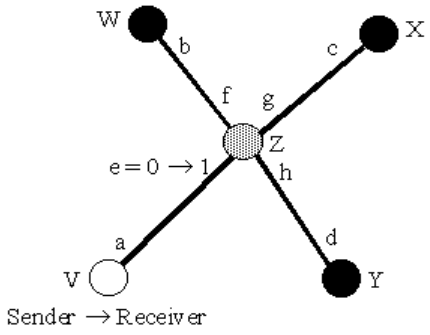

Figure 3.: Sender becomes a Receiver

Since the endpoint *V* is now a receiver, the value of *e* must be changed from *0* to *1*. Because of it, *b*, *c* and *d* must be set to *1* as well. If some of them were not *1* before, then this algorithm must be recursively re-run in the corresponding (*W*, *X*, *Y*) nodes. It means, for example when the algorithm must be re-run in *W*, that we forget about *V*, *X*, *Y* for a while, and run the algorithm of *Z* (a former sender) becomes a receiver with respect to *W*. Hopefully this simple algorithm does not require further explanation.

Now let us see the opposite situation, when a receiver becomes a sender (Figure 4)!

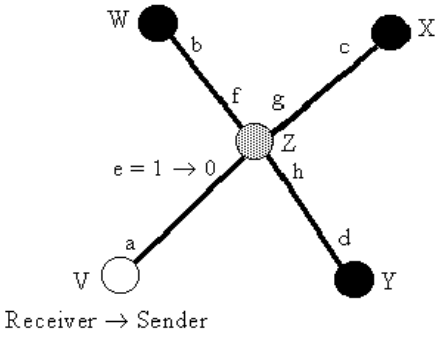

Figure 4.: Receiver becomes a Sender

In this case since *V* is a sender now, *e* turns from one to *0*. Before the change $a = f$ OR g OR h, where „OR" means the binary „OR" function. After the change, the value of *a* naturally remains the same, as *f, g, h* are unchanged. Before the change the values of *b, c* and *d* were all *1*, which maybe now changed, according to the following:

- If there are at least two *1*'s amongst *f, g* and *h*, then nothing more has to be changed, since all traffic has to go through the node *Z*.
- If there is exactly one *1* amongst *f, g* and *h*, say $f = 1$, $g = h = 0$, then all traffic coming from the nodes *V*, *X* and *Y* has to go through *Z*, so the value of *a, c* and *d* is not changed. The value of *b*, however changes to *0* form *1*, since data coming from *W* no more has to travel towards *Z*. Because the value of *b* has been changed, this algorithm has to be re-run on node *W* as well, as described in the previous subsection.
- If $f = g = h = 0$ was the situation before the change, then – because of reasons detailed in the previous point – *b, c* and *d* becomes *0*, and the algorithm must be re-run in nodes *W, X, Y*.

So far we have discussed, what happens if a receiver becomes a sender or vice versa. Now we will see, what happens, if a new member joins the tree (see Figure 5).

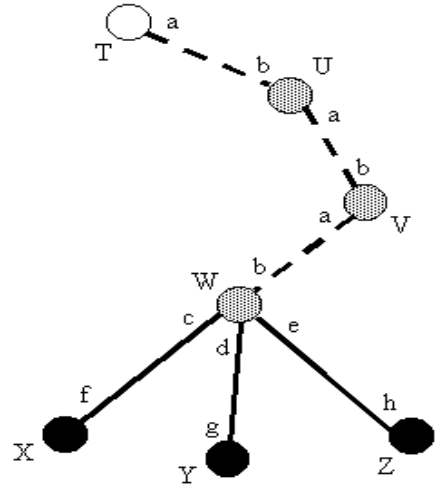

Figure 5.: Adding a new branch to the tree

If the endpoint *T* wishes to join the tree, it must send a join message towards the core. This message travels hop-by-hop until it hits a branch from the tree. In the example shown in Figure 5. we have examined quite a general case: when the join message hits the existing tree in the middle of it: node *W* has three outgoing links in the tree. Let us see how to modify the state bits in this case! First of all, it should be intuitively clear, that the values of *a*'s shown on the figure must be the same, and the same holds for the values of *b*'s. It is also easy to prove, that $a = c$ OR d OR e. Furthermore, if the endpoint *T* joins as a receiver, then $b = 1$, otherwise $b = 0$. All that remains is that if *T* is a receiver, then the „sender becomes a receiver" algorithm must be run on node *W*, as if *W* were part of the tree before, but it now have changed from being a sender to be a receiver. If *T* is connecting to the multicast group as receiver recursive refresh of network state may be necessary in node *X*, *Y* and *Z*.

When a member decides to leave the group, it must send a release message in the tree towards the core. The message travels till the first junction of the tree and that part of the tree will be removed. This scenario can also be discussed on Figure 5. If *T* wishes to leave the tree, then the *T–U–V* branch will be removed. After this if *T* was a receiver, then the

algorithm „receiver becomes a sender" must be run on *W*, as if *W* became a sender from a receiver.

5. FORMAL DESCRIPTION

After defining the new protocol the next step was to describe it in a formal language and verify it formally. We decided to use Telelogic's SDT software, a graphical representation of SDL (Specification and Description Language) as the tool of the formal description [11].

The first steps we have made in order to achieve the formal verification were the following:

- checking on the basic principles of the protocol: is the protocol suitable for establishing and managing multicast connections?
- informal verification of the protocol (i. e. examining the behavior of the protocol in a model network),
- trying out the features of the protocol for later studies.

First we had to find a good way of modelling the system. The aspects of determining the efficiency of a concrete model were the following:

- are the model results in a description, which is complex enough to examine the behavior of the protocol on?
- is the model simple enough so that the additional functionality (like routing) can be implemented in such an easy way, which does not divert our attention of the essential multicast functionalities?
- the tree management algorithms must be flowcharts, which are easy to understand,
- in order to achieve our third goal – trying out the features of the protocol for later studies – we must have a model, on which the working of the protocol, the passing of subsequent messages between network nodes can be examined seamlessly,
- the model must match to the SDL/SDT system's requirements and restrictions.

After taking more possibilities into consideration we have chosen the model of a *concrete topology network with fix, central routing*. This way of modelling the system results in the simplicity and the perspiciousness of the description. Tracking the process of exchanging messages between network nodes can be accomplished in an easy way as well. Other advantage is that we do not need to apply complicated routing mechanisms, central, fix routing is sufficient. There is a trade-off between simplicity and complexity: having a simple description means that we will not be able to examine the protocol under any possible circumstances. So this model brings up the disadvantage of studying the working of the protocol in a concrete network, but not in general case[1].

Some methods were introduced in order to make the description simpler. We have not dealt with the problems of data transmission, the model describes only the behavior of the signalling protocol. In spite of the fact, that SDL/SDT supports some kind of restricted simulation functionalities (e. g. delay can be associated with network links) we did not apply these, because defining timing functionalities is for further study.

The only way of examining a protocol managing multipoint-to-multipoint connections is to use a network topology, which is simple enough to maintain the comprehensivity but complex enough to be a model of a real structure. It means that the optimal manner is to have relatively few network nodes with a lot of connections between them. Therefore the network is quite simple, it includes 12 ATM switches – each one has got four ports – and 3 ATM terminals.

Four-four switches constitute two rings, each ring has got a central switch. This structure can be thought as two high capacity backbone networks connected to each other. Two terminals are connected through an access switch, the third endpoint is connected directly to this structure.

6. VERIFICATION STUDIES

After realizing the formal description we undertook to the verification of the protocol. The tool of the verification method was the Simulator User Interface of the SDT software environment, which helped this process with a lot of useful services: e.g. several means of running the simulation in step-by-step mode, direct access to all inner variables and bittables and the ability of sending any protocol messages to any parts of the model network while running the simulation. Some essential simulation studies should be achieved (e. g. signalling load, call setup time etc.) yet in such an initial state of the protocol, unfortunately SDT does not provide tools to make this kind of measurements.

In the verification phase first we gained experience by establishing and tearing down multicast connections in the model network. We could check if

---

[1] Note that the „model of the structure of one switch communicating with its environment" has not been definitively refused yet, because it seems to be the most generic manner of the description in order to validate the protocol. While our model is not a general model, the algorithms were composed to work in an optional environment, so the formal description can be used at the validation phase as well.

the protocol worked correctly by following the effects of the message sequences on the inner bittables. Processes of the protocol are initiated by sending TRIGGER messages from the environment to one of the terminals. These processes are: joining to a multicast group or establishing a new one, leaving a group and changing the state of the endpoint. A lot of scenarios were examined: all endpoints are members of the same group, simultaneous existence of more groups etc. In all configurations the working of the MNS was studied as well.

In the course of the verification it was ascertained that the protocol works properly, the fundamentals are correct: the signalling protocol is suitable for establishing, managing and tearing down multicast connections. Thanks to the Multicast Network Service the assignment of the multicast addresses and the core ATM address resolution is working seamlessly.

REFERENCES:

[1] C. Semeria, T. Maufer, *Introduction to IP Multicast Routing*, 3com White Paper, May 1996, http://www.3com.com/nsc/501303.html

[2] Laubach, M., *Classical IP and ARP over ATM*, RFC 1577, Hewlett-Packard Laboratories, December 1993 http://ds.internic.net/rfc/rfc1577.txt

[3] Armitage, G., *Support for Multicast over UNI 3.0 / 3.1 based ATM Networks (MARS)*, RFC 2022, November 1996 http://ds.internic.net/rfc/rfc2022.txt

[4] Armitage, G., *Redundant MARS architectures and SCSP*, Bellcore, Work in Progress, November 1996

[5] Armitage, G., *Issues affecting MARS Cluster Size*, Bellcore, RFC 2121, March 1997 http://ds.internic.net/rfc/rfc2121.txt

[6] Armitage, G., *VENUS - Very Extensive Non-Unicast Service*, RFC 2191, September 1997 http://ds.internic.net/rfc/rfc2191.txt

[7] T. Anker, D. Breitgand, D. Dolev, Z. Levy, *CONGRESS: CONnection-oriented Group address RESolution Service*, Institute of Computer Science, The Hebrew University of Jerusalem, Jerusalem, Israel, December 1996

[8] T. Anker, D. Breitgand, D. Dolev, Z. Levy, *IP-Senate: IP multicast SErvice for Non-broadcast Access networking TEchnology*, Institute of Computer Science, The Hebrew University of Jerusalem, Jerusalem, Israel, December 1996

[9] M. Grossglauser, K.K. Ramakrishnan, *SEAM: Scalable and Efficient ATM Multicast*, Published in the Proceedings of INFOCOM'97, April 7-11, 1997 in Kobe, Japan

[10] A.Ballardie, *Core Based Trees (CBT version 2) Multicast Routing* , RFC 2189, September 1997 http://ds.internic.net/rfc/rfc2189.txt

[11] Jan Ellsberger, Dieter Hogrete, Amardeo Sarma, *SDL Formal Object-oriented Language for Communicating Systems*, Prentice Hall Europe, 1997