How to measure reliability and quality of IP multicast services?

Falko Dressler
University of Erlangen-Nuremberg
Regional Computing Center (RRZE) /
Department of Computer Science IV (Operating Systems)
Martensstr. 1, 91058 Erlangen, Germany
falko.dressler@rrze.uni-erlangen.de

I Introduction

IP multicast is the most often used technique to transport multimedia data over the internet. Due to the low resource requirements when transmitting one multimedia stream to more than just one client, it is deployed in most backbone networks as well as in most campus networks. Typical services are video broadcasts or video conferences.

IP multicast like IP does not support to transport real-time data with a specified QoS. It is a best effort transport network, but there are some approaches to put some quality to IP to provide a better quality of service. Also, the network administrators have to find out information about the reliability of their network. For both measurements, there are a lot of tools available and the number of such tools is still growing.

The problem is, that you cannot measure the whole internet to apply these data to all possible services (applications). This paper describes an idea how to identify the components required for a particular service. Using this knowledge you can use the available tools to set up measurements for reliability and QoS of your network for this service.

II Typical services

The term service means for this work, which application the users do need, which multicast groups this application uses and which end systems do participate. The most interesting types of service are conferences and broadcasts.

A. Conferences

One example of a multicast conference is the TKBRZL [1]. About 8 to 10 participants from several universities in Bavaria talk about their current problems. They are using audio and video tools and a shared text editor to communicate. So every workstation in fact has up to 9 connections to other machines. If you want to make sure that the service is running fine, you have to check each of the 10 workstations against every other one for a proper multicast connectivity. If you also want to test the actual QoS, you have to check it the same way. Because such conferences appear to last only a short time, you need to do some measurement before they start. First, because the users want to know if the network has the capabilities to start the session and second, because the short time the communication takes does not allow some passive measurements while the service is running.

B. Broadcasts

The other common type of a multicast service is the broadcast. It looks like the conference, if you assume, that only one person is talking and all others are only listening. To verify the functionality and the QoS, you have to check the connectivity from the sender (broadcaster) to all the clients (participants, receiver). One example for such a broadcast are the channels from Uni-TV [2], which stream on a near-Video-on-Demand scheme recorded lectures to students of the University of Erlangen-Nuremberg. Since such broadcasts usually last for a longer time, also passive measurements are possible, based on the service itself.

III Modeling multicast networks

As shown in the previous chapter, you need to have the knowledge about 'important' services to improve you measurements. Also, you have to have knowledge about the network to do some real measurement. To combine both and, finally, to offer a mechanism to do some automatic tests, I want to introduce a model for multicast enabled IP networks and the overlying services. The model should be able to include important functions from OSI Layer 1/2 (Link), Layer 3 (IP) and Layer 7 (Application, the services). The primary result of such a model is to find out which parts of a network are required for a particular service. This can be done by attaching various routing algorithms.

For the specification, UML (Uniform Modeling Language) has been used which allows a straight forward implementation in object oriented programming languages such as JAVA. A first implementation called MRT (Multicast Routing Tool) has been done by Juan Meija, a student at the University of Erlangen-Nuremberg.

For starting with the modeling of the link layer, we must introduce at this point the basic objects that we want to model. Some candidate objects will now be presented: network, node, link and interface. This is also shown in Fig. 1. Examining more specialized objects let you come to the same three basic objects. Examples of node objects are a host or a router. An ethernet or a FDDI are typical links.

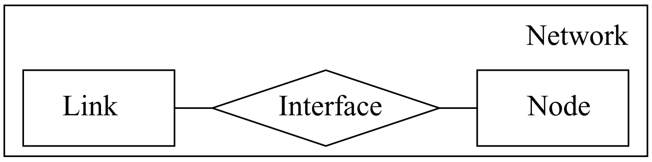

Fig. 1. Basic objects to model a network

A. Link Layer

The characteristics of the interface at the link layer can be described as follows. It has a MAC address assigned to it which identifies it uniquely. For QoS considerations, the most important characteristics are the capacity of the interface (bandwidth), the queuing delay (which is very hard to measure in real systems) and the packet loss ratio.

The object node summarizes objects like hosts, bridges or routers which look all the same at the link layer. Properties of nodes are accepted interfaces, CPU load or the amount of memory.

The object link itself has no specifications such as a bandwidth assigned to it because most layer-2 networks allow connections with different qualities. A typical example is an ethernet switch which allows connections from 10Mbps half duplex up to 1000Mbps full duplex. So these specifications are part of the Interface.

B. Network Layer

As for the link layer, one has to define all the important rules for the basic objects for the network layer. Basically, the Link object has no special rules or enhancements in the network layer. But we introduce a new type of link called 'cloud' or 'tunnel' which connects two nodes over an unknown transport network. This object has no link layer properties and exists as a higher level abstraction of a physical link.

The next object to model is the interface. At the network layer it gets an IP address assigned to it. Based on this address, a routing algorithm may find the optimum way between two different nodes.

The objects host and router which are examples of the more common object node have different properties in the network layer. A host allows only one interface attached to it but a router allows more than one. The network layer supports all the required information for a routing algorithm to work on.

Including the attached routing algorithm, the model can also be used to create and/or modify an IP network based on some information about typical (expected) communications within this network.

C. Application Layer

In our model, all the services are located at the application Layer. A service is defined as a number of hosts transmitting to a set of multicast groups and a number of hosts receiving this data. So only the hosts have representations within the application layer. The only property of a host at the application layer is its membership to one or more services.

D. Routing Algorithm

The model should allow to attach various routing algorithms. They are used to find the best (or optimum) path through a particular IP multicast network using the information out of the model of this network. The first algorithm which has been implemented is the Dijkstra algorithm which is also used in popular routing protocols such as OSPF. The routing algorithm finds only the best path between two hosts. So it must be run for each pair of hosts (or more correct between every sender and each of its receivers).

IV Modeling services

Besides the model of the network, the representations for the services have to be modeled. Each object of class service stands for one multicast transmission which may use more than one multicast group.

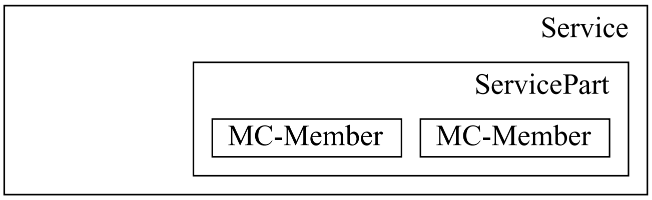

Fig. 2. Basic objects to model a service

This leads to the creation of the following objects: service, which defines a particular application, service part, which identifies used IP multicast groups and multicast members, which refer to particular hosts transmitting data to or receiving data from this group (Fig. 2.).

One example is a broadcast of a video from the project Uni-TV uses one group for video and another one for audio. The movie is streamed by a single video server which means there is only one sender but there is at least one receiver for the video data. Another example is the conference, where basically every hosts is transmitting and receiving data for each service part belonging to this conference (Fig. 3.).

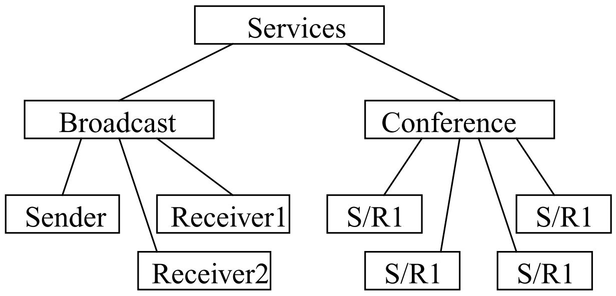

Fig. 3. Example of service objects

Based on the concept of analyzing the most important services within the network first and applying these information to a detailed model of the complete IP multicast network allows to extract the interesting parts of the network. This means, that it enables the network administrator to deploy measurement tools not only based on his wisdom but also on the requirements of the used applications.

V Object hierarchy

Summarizing the already presented objects and their capabilities, the following picture (Fig. 4.) shows to full class diagram which has been used to implement the model in JAVA. This implementation has been done by Juan Ceballos-Mejia at the University of Erlangen-Nuremberg [3] as a part of his masters thesis.

Of course, this is only a part of the whole class diagram but it already shows all the interesting parts of the object structure. Not shown are application specific parts such as an overlying project mechanism and parts of the routing algorithm.

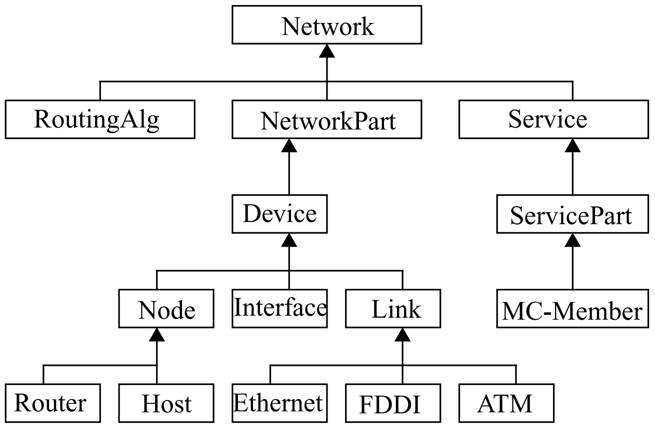

Fig. 4. Object hierarchy

VI Using the model for measurements

In the last chapters an overview of a method to model an IP multicast network including the network itself, the applications / services and the participating hosts has been presented. The final question is 'How do I use this model to measure reliability and quality of IP multicast services?'.

The current implementation of the model allows you to model your network and check for optimum paths for IP multicast transmissions using the attached routing algorithm. To find the best way, the algorithm uses the constants out of the modeled objects such as bandwidth of a particular interface or the hop count. The following figures show the mechanism. Using the information about the network and the service you can find out which parts of the network are used for this service.

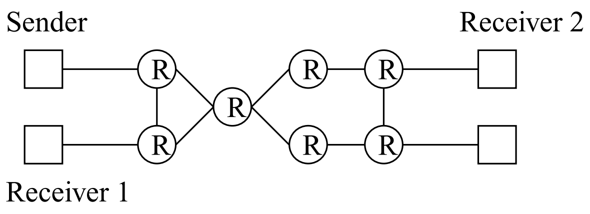

Fig. 5. physical network

Please note, this is only a first step to find all the required parts of your network for a particular service. Our implementation allows already to incorporate dynamic information about the current state of the network. The most interesting values are the state of a node, the state of an interface, the packet loss ratio from an interface and the load of a node. Also, the routing tables of your routers (IP unicast and IP multicast) have to be examined to get closer to the real behavior of your IP network.

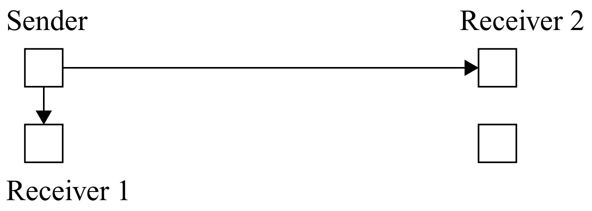

Fig. 6. logical data flow

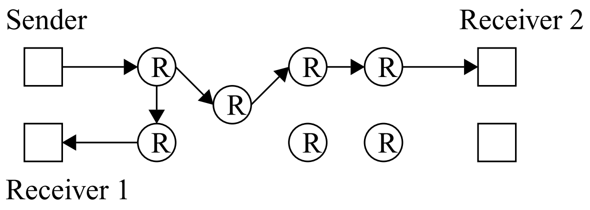

Fig. 7. real packet flow

Based on these dynamic data and the knowledge about the network, the tool allows to find the used components and paths for the current situation and a particular service (Fig. 7.). Using this wisdom, you can adapt your measurement tools to prevent unnecessary tests. This is very important for two reasons. First, you cannot measure the whole internet. You do not have access to all the routers and hosts and you cannot deploy probes nearly everywhere in the internet. Second, also for smaller networks, you need to deploy your measurement tools very carefully since most measures are active tests. This means that you need to create test packets and send them around the network. Using IP multicast, you need to test between every sender and all of its receivers. This does not scale very well, so you have to identify the parts of the network where the probes have to be placed.

VII References

[1] TKBRZL: http://tkbrzl.bhn.de/index.html

[2] Uni-TV: http://www.university-tv.de/index_en.html

[3] J.-F. Ceballos-Mejia, “Design and implementation of a modeling tool for multicast networks,” Master-Thesis, University of Erlangen-Nuremberg, 2000.

[4] K. Almeroth, L. Wei, D. Farinacci, “Multicast reachability monitor (MRM),” IETF draft, January 2001.

[5] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, “RTP: a transport protocol for real-time applications,” RFC 1889, January 1996.

[6] J. Postel, “User datagram protocol,” RFC 768, August 1980.