An Architecture for Integrated Policy-Based Management of QoS and Multicast-enabled Networks

Leandro Vaguetti, Ricardo Neisse, Lisandro Zambenedetti Granville, Maria Janilce Bosquiroli Almeida, Liane Margarida Rockenbach Tarouco

Federal University of Rio Grande do Sul - Institute of Informatics
Av. Bento Gonçalves, 9500 - Bloco IV - Porto Alegre, RS - Brazil
{vaguetti, neisse, granville, janilce, liane}@inf.ufrgs.br

Abstract. *QoS and multicast are facilities that several modern applications require from networks. However, the management of such facilities is complex and not integrated, when based on traditional management architectures. In this paper we present a policy-based management architecture and system for the integrated management of QoS and multicast-enabled networks. The definition of policies for this architecture is also presented showing how a network administrator can use such policies in order to manage the QoS and multicast facilities. The proposed architecture is based on the IETF approach. However, we discuss how such approach had to be adapted in order to support not only QoS management, but also multicast management, in an integrated fashion. Finally, we also present the implementation of our proposal providing some management example scenarios.*

1. Introduction

Several modern networked applications, like distance learning and remote collaboration, require two related but different facilities from networks: QoS and multicast. QoS is needed in order to allow a proper transmission of critical flows when available network resources (e.g. bandwidth) are limited. Multicast, on its turn, is needed to save network resources through the avoidance of multiple copies of packets directed to several receivers. The management of multicast finds important support in the Simple Network Management Protocol (SNMP), since several Management Information Bases (MIBs) have been defined by the Internet Engineering Task Force (IETF) and device manufacturers [Al-Shaer et al.,2002]. QoS management, on the other hand, has evolved from simple SNMP-based solutions to the recent sophisticated policy-based management approach [Sloman 1994].

Although multicast and QoS are facilities jointly required by several applications, their management are executed from separated perspectives. For example, in a multiconference that needs multicast to distribute participants' voice and video, and QoS to guarantee the proper delivery of such information in congested links, the network administrator manages the network using different tools. A multicast management tool can be used, for instance, to select the appropriate multicast routing protocol and to monitor the multicast traffic forwarded by routers. At the same time, another management tool is used to deploy QoS policies in the network devices in order to reserve bandwidth for the multicast traffic. Thus, in this context, network administrators are forced to use two different tools in order to manage related facilities (QoS and multicast) required by the applications.

In this paper we introduce a Policy-Based Network Management (PBNM) architecture that allows the integration of the management of QoS and multicast in the same management system. Although there is no standardized PBNM architecture, several researchers agree with a set of common elements needed in PBNM [Mahon et al.,2000], such as Policy Enforcement Points (PEPs), Policy Decision Points (PDPs) and Policy Repositories [Westerinen et al.,2001]. From these mostly accepted elements, we show how policies for our architecture and system are created and deployed, and, most important, we highlight how "standard" PDPs have been modified to support an integrated configuration of QoS and multicast-enabled networks.

Policies have been acclaimed as an effective mechanism to orchestrate the behavior of distributed systems accordingly to the business goals [Sloman 1994]. Policy languages are used to express such goals, but an architecture is needed to translate the expressed goals into device-specific actions. In this context, the main contribution of our work comes from the fact that the presented architecture is not only able to translate business goals into QoS parameters, but it is also able to translate such goals into parameters related to the multicast infrastructure provided by the managed network. Mostly important, the translations from business goals into QoS and multicast parameters are expressed through new integrated QoS and multicast policies.

The remaining of this paper is organized as follows. Section 2 presents a review of "classic" PBNM. Section 3 discusses about management of multicast networks, while section 4 introduces our proposal for the definition of integrated QoS and multicast policies. The management architecture that supports such policies is presented in section 5, and section 6 provides some examples and results taken from the management (through the proposed architecture) of a test network. Finally, section 7 concludes this paper presenting final remarks and future work.

2. Policy-Based Network Management Review

Policy-Based Network Management (PBNM) used in the QoS management found its first days within the Integrated Services (IntServ) architecture [Braden et al.,1994], where policies were used to coordinate the admission control functions in IntServ-enabled routers. Soon, it was realized that policies could ease the management of Differentiated Services (DiffServ) [Blake et al.,1998] networks too [Ponnappan et al.,2002]. The management of other QoS-related technologies (e.g. MPLS [Brunner et al.,2001]) and fields (e.g. security management [Damianou et al.,2002]) can also gain advantages from PBNM.

In the specific case of QoS management, there is no standardized architecture, but some functional elements are recognized as needed. Figure 1, shows a commonly referenced policy-based architecture for QoS management. In this architecture, network administrators define and coordinate the use of policies throughout the policy management tool. Once policies are created in the management tool, they are stored in the policy repository. Policies from the repository can be also edited or removed by actions executed throughout the management tool. Stored policies can be deployed in order to lead the network behavior to states compatible with the business goals expressed in the policies. To do so, the network administrator informs to the management tool which policies from the policy repository must be deployed. The administrator must also in-

form which Policy Decision Points (PEPs) are going to be used in the policy deployment. A PEP is an active element within a network device that influence the final QoS observed in the network. Each device can have several PEPs. For example, in a router each interface's queuing discipline is a PEP. Thus, PEPs are the final elements that effectively implement a QoS provisioning architecture.

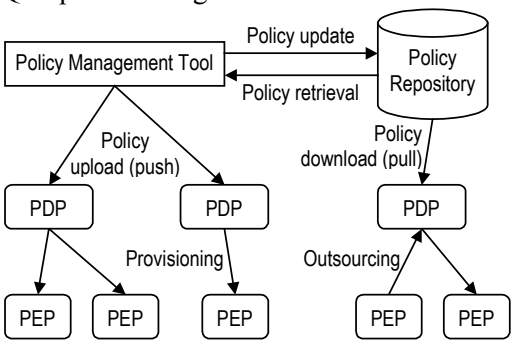

Figure 1. Traditional policy-based management architecture

Although the administrator defines the policies to be deployed in the PEPs, the management tool, on its turn, also determines, based on the informed PEPs, the associated Policy Decision Points (PDPs) that have to be contacted in order to translate the policies. The PDPs are the elements responsible for the deployment of management policies into PEPs, translating the business goal expressed in the policies to device specific-actions in the PEPs. From a certain point-of-view, PDPs are drivers used to configure PEPs using a standard interface: the policies.

There are two approaches used to transfer policies from the policy repository to PDPs. In the first one, once contacted, the PDP downloads policies from the policy repository. In this case, the policy is transferred in a pull operation. The IETF suggests that such transfer should be based on LDAP [Hodges et al.,2002] protocol, when the policy repository would be implemented as LDAP service directory. The second option is to let the management tool retrieve policies from the repository and actively upload them to the PDPs. In this case, the transfer is executed through a push operation.

There are also two approaches in the PDP and PEP communication. In the policy provisioning approach the PDP typically initiates the communication and configures a PEP according to a policy. In the policy outsourcing approach, on the other hand, the PEP normally initiates the communication asking for the PDP which actions the PEP should take accordingly to the current installed policies. The IETF has defined the COPS [Durham et al.,2000] protocol to be used in the outsourcing approach. An adaptation of COPS (COPS-PR [Chan et al.,2001]) has also been defined to support the provisioning approach. In the case of provisioning, however, others protocols besides COPS-PR could also be used (e.g. SNMP, TELNET and SSH), to configuration PEPs from devices that do not support COPS.

3. Multicast Management Issues

The most common used multicast solution is the one in which a host (send or receiver) can join a multicast group without requesting its inclusion to an admission controller. Also, whenever a host leaves a group, no notification is trigged in order to let the other

hosts to know that. Thus, with these features, the participants of multicast groups are quite dynamic because in two different moments the set of hosts of a group can be totally different. The majority of TCP/IP multicast networks is based on this approach. This can be observed by the set of multicast supporting protocols defined to help the maintenance of multicast in TCP/IP network (e.g. IGMP [Fenner 1997], DVMRP [Waitzman et al.,1988], PIM-DM [Adams et al.,1998] and MOSPF [Moy 1994]). Obviously, such multicast support has to be managed.

3.1. SNMP for Multicast Management

Although the dynamic nature of multicast networks is an advantage, the management of such networks is a complex task. The IETF have tried to support management of these networks defining MIBs to manage the supporting protocols via SNMP. For example, the IGMP-MIB [McCloghrie et al.,2000] defines objects that allow a network administrator to discover, inside a multicast-enabled device, the multicast groups that generated traffic that have passed through the device. Other MIBs defined for the management of multicast supporting protocols are, for example, IPMRoute-MIB [McCloghrie et al.,2000], DVMRP-MIB [Thaler 2000] and PIM-MIB [Nicholas 2002].

Although the MIBs for multicast management allow the access of important information via SNMP, they do not define the actions to be executed by a management system. In this context, we used the following set of basic management operations for a multicast network based on the TCP/IP suite:

- Enable/disable multicast support for each interface in each multicast-enabled device (typically, routers);
- Define the polling interval each multicast-enabled device should use to query for new multicast nodes;
- Discover the hosts and network nodes that participate in each multicast group;
- Configure the multicast routing protocol used by each node of a multicast network.

These basic tasks can be executed through SNMP using the multicast MIBs pointed before. However, a pure SNMP-based management of multicast networks is more complex than the management of ordinary networks because the amount of available management information is greater in multicast networks. This complexity could be reduced using a policy-based management solution for multicast management. The idea of using policies to manage multicast network, in this context, seams to be adequate, but as we are going to seen in the next subsection, it is not trivial.

3.2. PBNM and Multicast Management

An immediate approach for the management of multicast using policies would be adaptation of the “standard” policy-based architecture originally defined for the management of QoS. However, a carefully look at the architecture shows important drawbacks that can prevent a proper multicast management.

There are two different and important steps in using policies in the QoS policy-based management architecture. The first step is quite obvious that is the policy definition. The second step is also obvious but its operation is not that one. It consists on the selection of PEPs where the policy is going to be deployed. To do that, the PEPs of a

network have to be previously saved in a storage (e.g. the policy repository or an alternative database system) and associated with corresponding PDPs. This allows the PBNM system to determine which PDPs contact to deploy a policy in a PEP. If a new PEP is included in the network, this PEP has to be registered in the system in order to let policies to be deployed on it. All these operations indicate that the managed PEPs have to be mapped to a storage before its use.

Differently from a "classic" QoS network, multicast networks have more dynamically PEPs. Every time a new host joins a multicast group, several different PEPs within multicast devices in the networks may have to be activated to support the new member. The same way, every time a host leaves a group, several PEPs may have to be deactivated because multicast support is no longer needed. This dynamic nature of active PEPs does not allow such PEPs to be constantly mapped to storage as required by standard PBNM systems, since the set of PEPs can quickly change, leaving the storage out of date. Thus, the operations on the policy-based management system have to be now based on a quite more dynamic environment.

Until now, we have considered the management of multicast using standard PBNM systems. Actually, a single adaptation in the standard approach is not enough, because we are not interested in isolated multicast management, but we aim at an integrated QoS and multicast management. Thus, the changes to be executed in the standard system and in the standard way of defining policies have to consider both QoS and multicast.

3.3. QoS and Multicast Management Tasks

In a simple QoS management environment, the following parameters should be defined in policies and enforced by the management system:

- The schedule for the policy activation;
- The flows to grant network resources. Typically, flows are defined by IP source and destination addresses, transport source and destination ports and transport protocol used;
- The network resources to grant to the flow. Related to QoS, such resources are normally throughput, delay, jitter and loss.

An example of a policy for QoS management is presented in figure 2. Policy from figure 2 is composed by only one policy rule that defines the schedule, traffic and networks resources allocated for the backup of a database.

For an integrated management of QoS and multicast, however, we believe that a different set of management actions are:

- Define the multicast groups to grant network resources (e.g. which multicast groups will receive more bandwidth);
- Define parameters, according to the multicast group participants, to consider when granting network resources (e.g. how many hosts in a group are required to trigger a bandwidth allocation);
- Define the QoS parameters associated to the network group according to the set of its participants (e.g. how much bandwidth will be allocated to a multicast conference);

- Create QoS and multicast policies for the above parameters, also including the policy schedule;
- Select the network critical PEPs where the multicast group traffic are supposed to pass through;
- Deploy the QoS and multicast policies in the critical PEPs.

```
Policy: Database backup priorities
Rule: Backup schedule and traffic
    if (timeOfDay >= 9pm) and  (timeOfDay =<  11pm)
      and (SourceAddr == 143.54.47.2) // Database source
      and (DestAddr == 143.54.6.240)  // Backup destination
      and (DestPort == FTP) // Protocol used to transfer backup
      and (TranspProtocol == TCP)
    then
       Bandwidth = 300 Kbps
       LossPriority = 0
```

Figure 2. An example policy for database backup

To allow all this operations we need: a way to define policies for QoS and multicast, and a policy-based system that supports such policies. The next section presents a proposal for the definition of QoS and multicast policies, while section 5 shows the PBNM system we have developed to support the integrated QoS and multicast policies.

4. Integrated Policies for QoS and Multicast Networks

In order to integrate QoS and multicast policies, first we need to check how QoS policies and multicast policies are defined separately. In this section we verify how to define such "standard" policies and then proposed an approach for the integrated definition of policies that express QoS and multicast requirements.

4.1. Definition of Standard QoS and Multicast Policies

The main goal of the multicast management is to make the underlying network able to transport data using multicast facilities. To allow that, some elements in the network must be enabled (e.g. the multicast support in routers) and other elements must be configured (e.g. the query interval on each interface of an IGMP-enabled router). An immediate approach to define policies for multicast management is to select multicast parameters that required customization and choose values for them in a multicast policy definition. Figure 3 suggests a multicast policy based on such approach.

This policy is composed by tree rules: the first rule enables multicast using DVMRP as routing protocol between 8am and 11am. The second rule defines the querying interval used to poll hosts about joining/leaving multicast groups. Finally, the last rule disables multicast support in two different periods: between 12am and 2pm, and between 9pm and 8am of the next day.

The management of QoS, on its turn, has to coordinate network resources in order to maximize performance of critical flows, in detriment of less important ones. The management of QoS normally deals with the throughput, delay, jitter and loss of critical flows. In this context, the standard QoS policies usually define the expected values for such parameters. Added to figure 2, figure 4 presents an example of a QoS policy.

```
Policy: Multicast example
Rule: Multicast using DVMRP as routing protocol
    if (timeOfDay >= 8am) and  (timeOfDay =<  11am)
    then
      MroutedEnable = true
      MRoutingProtocol = DVMRP
Rule:Configure time interval for multicast groups polling
    if (timeOfDay >= 2pm) and  (timeOfDay =<  4pm)
    then
      QueryInterval = 1000 ms
Rule:Disable multicast
    if ((timeOfDay >= 12am) and  (timeOfDay =<  2pm))
    or ((timeOfDay >= 9pm) and  (timeOfDay =<  8am))
    then
      MroutedEnable = false
```

Figure 3. Standard Multicast policies

```
Policy: QoS policy
Rule: Configure traffic using for a specific host
    if (timeOfDay >= 8am) and  (timeOfDay =<  11am)
    and (DestAddr == 143.54.6.245)
    then
      Bandwidth = 300 Kbps
      MaxDelay = 100 ms
      MaxJitter = 10 %
Rule:   Configure QoS for FTP traffic between two hosts
    if (timeOfDay >= 2pm) and  (timeOfDay =<  4pm)
    and (SourceAddr == 143.54.47.47)
    and (DestAddr == 143.54.47.12)
    and (DestPort == FTP) and (TraspProtocol == TCP)
    then
      Bandwidth = 700 Kbps
      MaxLoss = 20%
```

Figure 4. Standard QoS policies

The QoS policy from figure 4 is composed by two rules used to configure QoS parameters. The first rule defines that all traffic directed to host whose IP address is 143.54.6.254 should receive network resources that provide a max delay up to 100 ms and a max jitter up to 10%. The last rule defines that the FTP traffic between two 143.54.47.47 and 143.54.81.12 must receive 700 Kbps of bandwidth and loss must be under 20%.

Taking these two kinds of policies (multicast policies and QoS policies) separately, they can efficiently operate from different tools. However, as observed before, integration is required to allow the management of QoS and multicast from a single management environment. Thus, a first step is to define how QoS and multicast parameters could be defined in the same policies. However, this integration of policies definition is not easy because some "interesting" situations come to life from the integration of QoS and multicast. To exemplify one of these situation let's observe QoS and multicast policy from figure 5.

```
Policy: Policy for QoS and multicast
Rule:  EnableMulticastSupport
        if (timeOfDay >= 4pm) and (timeOfDay <= 6pm)
        then
          MroutedEnabled = true
          MRoutingProtocol = PIM
Rule:  QoSforMulticastGroup
        if (timeOfDay >= 4pm) and (timeOfDay <= 6pm)
          and (DestAddr == 224.0.0.214)
        then
          Bandwidth = 300 Kbps
          MaxJitter = 10%
          MaxLoss = 30%
```

Figure 5. A QoS and Multicast policy

This QoS and multicast policy is defined to enable multicast support in routers everyday between 4pm and 6pm, using PIM as the multicast routing protocol. Also, it will allocate resources in order to support the multicast flow 224.0.0.214 with 300 Kbps, jitter up to 10% and loss up to 30%. Although this policy seems to be properly defined, it does not work fine for the following situation. Suppose that the network administrator does not exactly know which network segments will experience the multicast flow. In this case, the administrator deploys the policy from figure 5 in all routers, expecting to

enable multicast in all of them. The problem in this case is that the second rule requires resource allocation, which means that resources will be allocated in every router, even though some of them will not transport the multicast flow of the group 224.0.0.214. For those routers that will not experience the multicast flow to resources will be allocated as well, wasting them because no flow will consume such resources.

A more proper approach would be to enable multicast support in every router, but allocate resources only on those routers where the multicast flow will pass through. In this case, the first rule should be activated for every router, but the second rule should be only activated on those routers where the group 224.0.0.214 is also active. An event driven policy approach could be used here, where a key event on the network would be the observation that a router experienced a packet directed to 224.0.0.214. When such event happens the second rule should be then evaluated, and the network resources would be reserved on such router.

The IETF approach, however, is not event driven, and the events are said to be implicit. In our opinion, a more explicit way to indicate key events easies the definition of integrated policies for QoS and multicast. Since we based our policy system on the IETF definitions, we had to extend such definition to accommodate situations like the one above explained. Thus, we proposed an enhancement that uses a finite state machine that helps the definition of integrated QoS and multicast policies. Actually, this proposal is an improvement of a previous work developed by our research group [Granville et al.,2002].

4.2. QoS and Multicast Policies using Finite State Machines

We proposed that integrated policies for QoS and multicast, in this work, are defined as a set of the following elements:

- One single rule for the policy schedule information;
- A set of rules related to QoS and multicast conditions;
- Scripts that enforce/remove the policy rules for QoS and multicast;
- A finite state machine that coordinates when the scripts should be executed.

We explain these elements through the policy from figure 6, that presents the proposed definition. The Policy clause, as on the other examples, is used to name the policy being defined, while the ScheduleRule clause is used to define the scheduling rule of the policy. Differently from the other examples, a single rule for schedule is used to define the moments in which all rules are valid or not. This remove possible conflicting situation related to schedule between rules. At the bottom, one can verify the set of rules for QoS and multicast parameters. These rules can be numbered (rule1, rule2, etc.) for further references. Rule1 activates multicasting support with a polling interval of 1000 ms and PIM as the multicast routing protocol. Rule2, on its turn, allocates network resources for the multicast group 224.0.0.214.

Although the rules define the values for the QoS and multicast parameters, the activation of the rules is coordinated by the finite state machine (FSM) and the associated scripts. In the FSM, each node represents a policy state, and the edges represent the transitions, from one node to other, triggered by events. When a node is reached, an associated script for that node is executed. The first node is always S1, and then the first script executed is the script S1. If an event is triggered in there is a transition from the

current node to another one related to that event, then the current node is left, the next node is reached and the associated script executed. If there is no transition from the current node corresponding to the triggered event, the current node remains the same and the event is discarded.

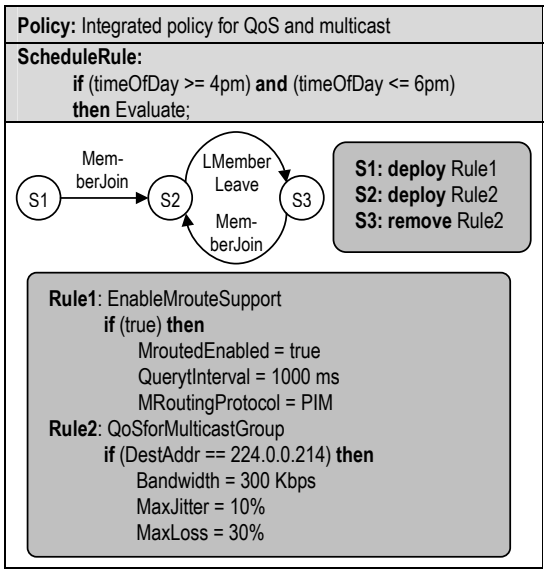

Figure 6. An integrated policy for QoS and multicast

Looking back at figure 6, the FSM would be evaluated as follows. Initially, the node S1 is reached and the script S1 is executed. Script S1 enforces rule number 1 through the clause *deploy Rule1*. In this state, the multicasting support is activated and the FSM waits for a *MemberJoin* event. If a packet directed to the group 224.0.0.214 is forwarded by the multicast device, the *MemberJoin* event is triggered and a transition from state S1 to S2 happens. Since S2 is reached, the script S2 is executed, deploying rule number 2 through the clause *deploy Rule2*. While in S2, the FSM will only change state if a *LMemberLeave* (last member leave) event is triggered. In this case the state S3 is reached and the script S3 will remove the rule number 2 through *remove Rule2*. While in S3, if another member joins the group again, another *MemberJoin* event will be triggered and the network resources will be again allocated when script S2 is once more executed.

In our approach there are also two implicit events: the first one is the event that enables the policy as a whole and is triggered the first time the schedule rule evaluates to true. This activates the policy, the state S1 is reached and the corresponding script is executed. The second implicit event is triggered the first time the schedule rule evaluates to false after being evaluated to true. It indicates that the policy is no longer active and the policy rules previously deployed should be removed. In this case an implicit script SRemove: remove all is executed. If, after a policy deactivation, the schedule rule evaluates to true again, the policy returns to be valid and the S1 state is reached.

We believe that with this mechanism, the integrated policies for QoS and multicast can be more properly defined. However, the definition of policies is not enough to provide a real integrated management of QoS and multicast: a PBNM system that is able

to handle this kind of policies is also required. The next section presents the PBNM architecture and system we have developed to support such policies.

5. QoS and Multicast PBNM Architecture

Comparing with the traditional PBNM architecture, the main difference in our proposal is that we have defined a PDP that is able to actively decide when the policy rules must be deployed and when the associated script should be executed based on the status of the policy FSM. The internal architecture of our PDP is an evolution of a previous PBNM system we have defined for the automation of policies replacement [Granville et al.,2002].

5.1. General Architecture

Figure 7 presents our proposed architecture. The policy management tool is responsible, like in the "standard" PBNM architecture, for the creation, edition and removal of policies stored in the policy repository (PR). Differently from the common solution, however, the user interface of the management tool should be able not only to define QoS policies, but also to allow the definition of the integrated QoS and multicast policies proposed in the section 4.

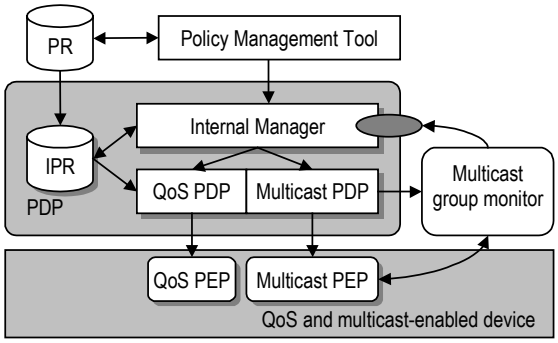

Figure 7. QoS and multicast PBNM architecture

If a policy in the repository needs to be deployed, it is transferred (via pull or push) to the *Internal Policy Repository* (IPR) of one of our PDPs, and the *Internal Manager* (IM) begins controlling deployment of the rules of the transferred policy.

The IM is the most important element of the architecture because it is responsible for the whole coordination of the other elements. The first IM task is to verify when a policy should be activated, based on the policy scheduling data, and when a policy rule should be deployed, based on the policy FSM and associated scripts.

If a rule has to be deployed, the IM contacts internal PDPs. Since we want to support QoS and multicast parameters, we have two internal PDPs: on for QoS support and the other for multicast support. Each contacted PDP is responsible to configure a corresponding external PEP in order to deploy the rule being activated. The multicast PEP should be configured, for example, to enable multicast support in a router interface. In the specific case of the multicast PDP, it is also responsible to contact a multicast group monitor that is the element able to verify if hosts join or leave multicast groups. Although the multicast group monitor and the multicast PEP are functional different

elements, they can be implemented in the same physical device.

Every time the multicast group monitor detects a new group member, the monitor notifies the IM that can, based on the current policy, contacts an internal PDP asking for configuration based on the activated rules. The same way, every time the multicast group monitor detects that a member leaved a group, the monitor notifies the IM that can now remove previously deployed configuration if that is expressed in the policy scripts. Finally, if the policy is no longer active, due to its schedule, the IM contacts the internal PDPs in order to remove the previously deployed rules.

5.2. Example Management Operation

Here we will check how the proposed architecture can be used to ease the management of a QoS and multicast-enabled network. Figure 8 presents an example network.

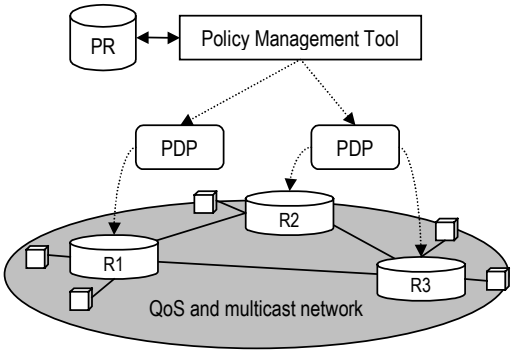

Figure 8. Example QoS and multicast network

In this network, two PDPs operate over three QoS and multicast-enabled routers (R1, R2, R3). The boxes denote the network hosts. First, let's suppose that the routers have disabled multicast traffic on their interfaces. Let's also suppose that a critical multicast session (e.g. a directors' remote videoconferencing) is supposed to start at 4pm and finish at 6pm.For this situation we use the policy previously presented at figure 6.

When ordered by the network administrator, the policy is transferred from the PR to the PDPs in order to be deployed. Each PDP controls the time when the policy have to be activated (in the case of this example, at 4pm). Once the activation time is reached, each PDP, according to the rule number 1 from figure 6, contacts a router internal multicast PEP to enable the multicast support on the router interfaces and configures the multicast group monitor (that can be located internally to the routers) in order to receive monitor notifications.

Once the multicast traffic is started, the proper multicast protocol is used by the routers to announce the groups. Once new members join the groups, the multicast monitor sends a notification to the associated PDP. Accordingly to the policy rule number two, the PDPs have to configure the QoS PEPs to reserve network resources for the multicast group. This on demand configuration saves network resources because they are only reserved when needed. For example, if a host connected to the R1 router sends multicast packets, the R1 QoS PEP will be contacted only if another host from the other routers joins the group. If the host from R2 router joins the group, both routers R1 and R2 will have resources allocated, leaving the R3 router untouched, although its PDP is

ready to configure it.

5.3. Implementation

We have implemented the proposed solution and applied it to a test network that supports QoS through CBQ [Risso 2001] and multicast through the IGMP, DVMRP, MOSPF, MBGP and PIM. The management tool [Granville et al.,2001] was developed as a set of PHP4 scripts that access a LDAP server using the LDAP support provided by PHP4. Also, the policies are defined in the user interface and stored in the LDAP server using an LDAP schema based on the IETF schemas for policy-based management [Strassner et al.,2002]. A complementary MySQL base was used to store standard management information, such as the network map and SNMP community string for each network device (figure 9).

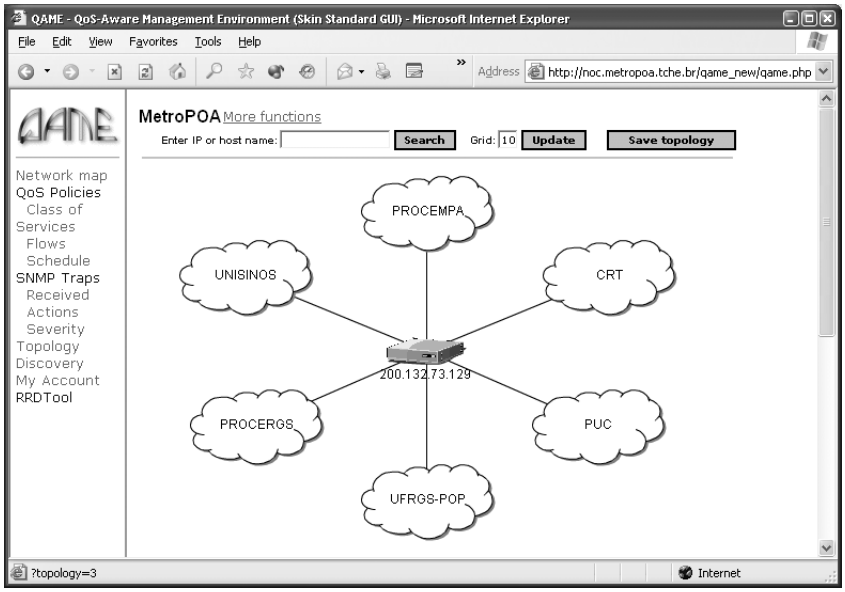

Figure 9. Snapshoot of the management tool

The PDPs were implemented in Java, for portability, and the policy transfer model adopted was the pull approach. The management tool informs a PDP through SNMP that a new policy is available on the LDAP server. The PDP then downloads the new policy and deploys it when necessary. Thus, PDPs and policy repository communication is based on LDAP, while the management tool and PDP communication is based on SNMP (figure 10).

To allow such communication we have implemented the IM as an SNMP agent that supports a policy transferring MIB that we have defined (figure 11). The management tool creates new entries in a policy table to inform the new policy to be used by the PDP. One data on such entry is the URL, in the LDAP, that the PDP have to access in order to download the policy.

In our test network, network routers are linux routers with CBQ, IGMP, PIM, MBGP and DVMRP or MOSPF support. The CBQ turns to be the QoS PEP from the general architecture, while IGMP and routing modules are the multicast PEP. The CBQ is configured through telnet sessions, and the IGMP and routing protocols are accessed

through SNMP. Two different MIB were used to monitor the contents of multicast group: IGMP-MIB and IPMroute-MIB. One important observation is the fact that the multicast monitoring agent is a process inside the multicast router (actually, it is a software that supports a subset of the RMON2 MIB), that observes objects from both MIBs and notifies the associated PDP via SNMP trap messages every time a new member joins or the last member leaves a multicast group. In this case, we have an example where the multicast group monitor is located within the managed device.

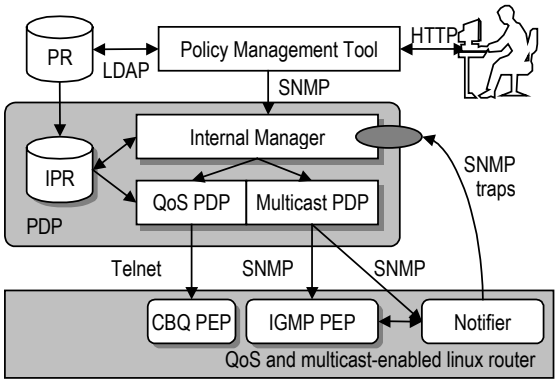

Figure 10. Implemented architecture

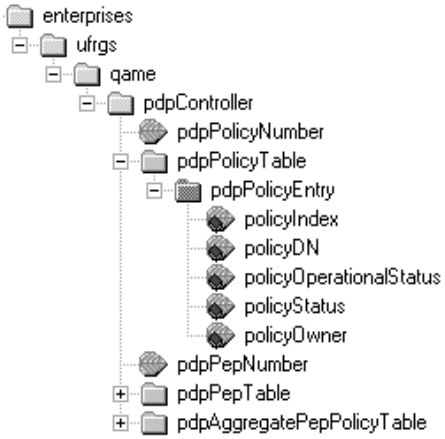

Figure 11. MIB for policy transfer

6. Some Experiences in a Multicast Test Network

We have applied our solution in the management of a test network installed in three of our university campus. In this section we present the test network, the experiences carried on it, to and some preliminary results. Figure 12 presents the test network.

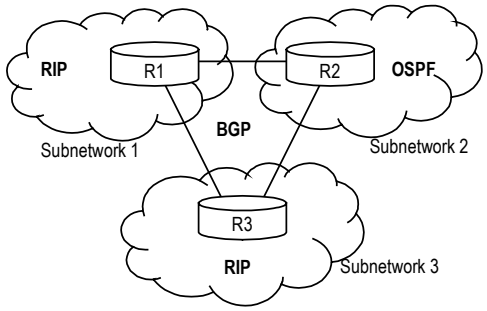

Figure 12. Test network

The test network is composed of three subnetworks connected to each other via three border routers. Route R1 uses BGP as external gateway protocol, and RIP as internal gateway protocol. Routers R2 and R3 also uses BGP for external routing, and OSPF and RIP for internal routing, respectively. As a consequence of the routing protocols, the network can use MBGP, MOSPF, DVMRP and PIM as multicast routing protocol. The multicast protocol to be used have to be defined, as presented in the previous examples, in the QoS and multicast policies.

Figure 13, presents the first policy we have defined and deployed in the test net-

work. It uses DVMRP and MBGP as multicast routing protocols and allocates resources for the 224.0.0.214 multicast group. We have generated multicast traffic for this group from a host on subnetwork 1 at 8:45am. At the same time we ordered the deployment of the policy on the three border routers. Two PDPs, not depicted in figure 10, were used. One PDP was able to configure routers R1 and R3 (BGP and RIP), and the second PDP was able to configure router R2 (BGP and OSPF). Although the PDPs downloaded the policy at 8:45am, the policy was not activated because its start time is 9am. From 8:45am until 9am, only receivers located at subnetwork 1 were able to access the multicast traffic, since the border routers were not yet supporting multicast.

At 9am, when the policy became active, the three routers were configured to support multicast using DVMRP and MBGP. Since router R2 does not support DVMRP as internal multicast routing protocol, its internal network keep isolated from receiving the multicast traffic. Receivers from subnetwork 3, on the other hand, started receiving the multicast traffic because the R3 border router now support multicast and can handle the traffic using DVMRP too. It is important to notice that all routers support IGMP, and, according to the policy, the polling interval inside each subnetwork was 1000 ms.

At 11am, when the policy became invalid, all routers configurations were undone in order to remove the allocated resource. It is also important to notice that, although the subnetwork 2 was kept isolated due to the lack of DVMRP support, no resources were allocated at R2 because no multicast traffic was experienced.

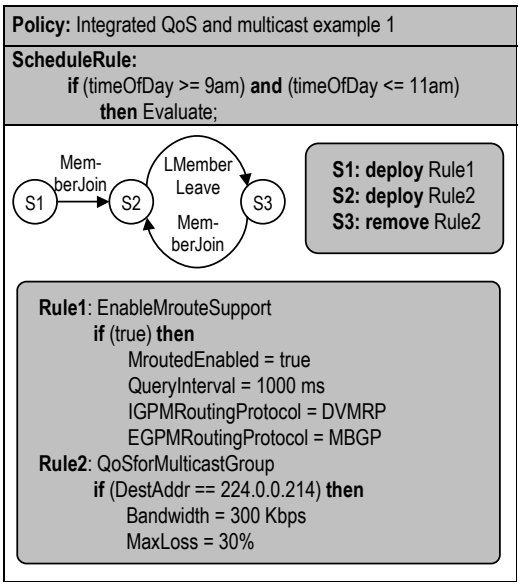

Figure 13. Policy example 1

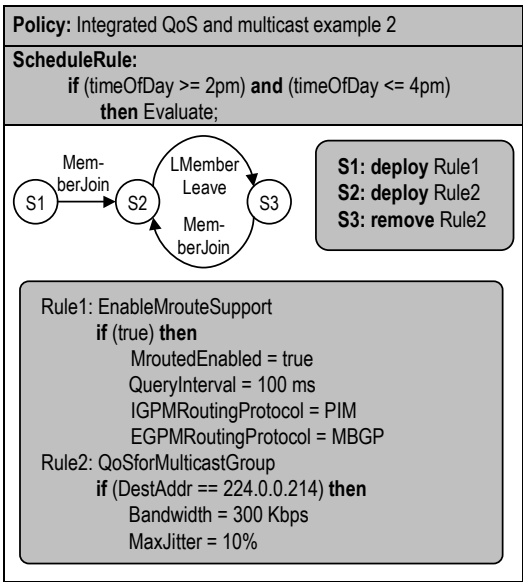

Figure 14. Policy example 2

In order to enable multicast on the whole network, the policy from figure 14 was defined and deployed in the test network. This second policy is activated at 2pm and remains valid until 4pm. Different from the first example policy, this one uses PIM as internal multicast routing protocol, that is more "generic" than the specifics MOSPF and DVMRP. Also, another difference is that, in order to detected new members faster, the QueryInternal was reduced to 100ms. For this policy, we have enabled receivers at the

three subnetworks at 1:45pm, and only one sender from subnetwork 1 at the same time. At 2pm the policy turn to active and the receivers from all the subnetworks were able to receive the multicast flow. At 2:15pm we made all receivers from subnetwork 3 leave the group, and soon the bandwidth allocated in the router R3 (300 Kbps) were released.

7. Conclusions and Future Work

In this paper we have presented an architecture and a system for the integrated management of QoS and multicast-enabled network based on policy-base network management (PBNM). The architecture implements special Policy Decision Points (PDPs) that are able to deploy policies for QoS and multicast, and react upon the triggering of multicast events in the network (e.g. a host joining a multicast group, or the last hosts leaving another group).

From the definition of policies for our proposed environment we have learned that expressing requirements for QoS and multicast support is not an easy task, but we believe that our approach for integrated policies easies the maintenance of networks whose applications require both QoS and multicast facilities.

From the implementation of the PBNM system we noticed that, although policies for QoS are already mature, policies for multicast requires more investigation. In our approach, for example, some detail of the underlying multicast technology are still "seen" by the network administrator. For example, the administrator must choose a multicast routing protocol when enabling multicast support in a router.

We have implemented our proposed system in a test network, from where we can conclude, with the first results, that the approach adopted is effective an allows a proper management of QoS and multicast, although further investigation is required, for example, to determine the network load when such policies are used.

Future work still needs to be executed. First, we have to collect more data from the test network to evaluate the network load. Further investigation on the policy definition is required in order to verify a more proper way to define such policies (maybe throughout graphical interfaces). Finally, we still have to evaluate the performance of the implemented PDPs when different multicast routing protocols are used.

References

Adams, A., Nicholas, J. and Siadak, W. (2002) "Protocol Independent Multicast - Dense Mode (PIM-DM): Protocol Specification (Revised)", <draft-ietf-pim-dm-new-v2-02.txt>, IETF.

Al-Shaer, E. and Tang, Y. (2002) "SMRM: SNMP-based Multicast Reachability Monitoring", IEEE/IFIP NOMS 2002.

Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., Weiss, W. (1998) "An Architecture for Differentiated Services", RFC 2475, IETF.

Braden, R., Clark, D. and Shenker, S. (1994) "Integrated Services in the Internet Architecture: an Overview", RFC 1633, IETF.

Brunner, M. and Quittek, J. (2001) "MPLS Management using Policies", IEEE/IFIP IM 2001.

Chan, K., Seligson, J., Durham, D., Gai, S., McCloghrie, K., Herzog, S., Reichmeyer, F., Yavatkar, R., and Smith, A. (2001) "COPS Usage for Policy Provisioning (COPS-PR)", RFC 3084, IETF.

Damianou, N., Dulay, N., Lupu, E., Sloman, M. and Tonouchi, T. (2002) "Tools for Domain-Based Policy Management of Distributed Systems", IEEE NOMS 2002.

Durham, D., Boyle, J., Cohen, R., Herzog, S., Rajan, R. and Sastry, A. (2000) "The COPS (Common Open Policy Service) Protocol", RFC 2748, IETF.

Fenner, W. (1997) "Internet Group Management Protocol, Version 2", RFC 2236, IETF.

Flegkas, P., Trimintzios, P. and Pavlou, G. (2002) "A Policy-Based Quality of Service Management System for IP DiffServ Networks", IEEE Network, v. 16.

Granville, L., Ceccon, M., Tarouco, L., Almeida, M. and Carissimi, A. (2001) "An Approach for Integrated Management of Networks with Quality of Service Support Using QAME", IEEE/IFIP DSOM 2001.

Granville, L., Coelho, G., Almeida, M. and Tarouco, L. (2002) "PoP - An Automated Policy Replacement Architecture for PBNM", IEEE POLICY 2002.

Hodges, J. and Morgan, R. (2002) "Lightweight Directory Access Protocol (v3): Technical Specification", RFC 3377, IETF.

Mahon, H., Bernet, Y., Herzog, S. and Schnizlein, J. (2000) "Requiriments for a Policy Management System", <draft-ietf-policy-req-02.txt>, IETF (expired draft).

McCloghrie, K., Farinacci, D. and Thaler, D. (2000) "IPv4 Multicast Routing MIB", RFC 2932, IETF.

McCloghrie, K., Farinacci, D. and Thaler, D. (2000) "Internet Group Management Protocol MIB", RFC 2933, IETF.

Moy, J. (1994) "Multicast Extensions to OSPF", RFC 1584, IETF.

Nicholas, J. (2002) "Protocol Independent Multicast MIB", <draft-ietf-pim-mib-v2-01.txt>, IETF.

Ponnappan, A., Yang, L., Pillai, R. and Braun, P. (2002) "A Policy Based QoS Management System for the IntServ/DiffServ Based Internet", IEEE POLICY 2002.

Risso, F. (2001) "Decoupling bandwidth and delay properties in class based queuing", IEEE ISCC 2001.

Sloman, M. (1994) "Policy Driven Management for Distributed Systems", Journal of Network and Systems Management, vol. 2.

Strassner, Moore, B. and Ellesson, E. (2002) "Policy Core LDAP Schema", <draft-ietf-policy-core-schema-16.txt>, IETF.

Thaler, D. (2001) "Distance-Vector Multicast Routing Protocol MIB", <draft-ietf-idmr-dvmrp-mib-11.txt>, IETF.

Waitzman, D., Partridge, C. and Deering, S. (1988) "Distance Vector Multicast Routing Protocol", RFC 1075, IETF.

Westerinen, A. et al. (2001) "Terminology for Policy-Based Management", RFC 3198, IETF.