Chapter 17

INTEGRATING IP MULTICASTING IN STANDARD NETWORK MANAGEMENT PROTOCOLS

Ehab Al-Shaer Yongning Tang
Multimedia Networking Research Laboratory
School of Computer Science, Telecommunications and Information Systems
DePaul University
Chicago, IL 60604
(ehab,ytang)@cs.depaul.edu

Key words: Network Management, Group Communication, IP multicast, SNMP

Abstract: Supporting multi-point group communications in the network management platform is essential for improving scalability and responsiveness of management applications. With the deployment of IP multicasting as the standard infrastructure for multi-point group communications in the Internet, the integration of IP multicasting in SNMP becomes significantly important to achieve these goals. This paper presents a flexible, efficient and easy-to-integrate framework for integrating IP Multicast in standard SNMP agents. The proposed framework enables managers to re-configure the agents' group membership (e.g., join and leave) and the communication model (e.g., one-to-many, many-to-one and many-to-many) dynamically based on the application requirements. This framework exploits the advantages of IP multicasting without requiring any significant changes or performance overhead in the protocol or the agent architecture. Our ultimate goal is to promote the integration of IP multicasting as a standard service in SNMP agents.

1. INTRODUCTION

J. N. de Souza et al. (eds.), *Managing QoS in Multimedia Networks and Services*

The Internet network management is based on client-server architecture where the manager is the client that sends the requests (e.g., *get* and *set*) and the agent is server that collects and reports information about *managed* objects. In network management environment, the manager may contact a group of agents frequently in local or remote domains to perform, for example, fault, performance or security management tasks. SNMP agents may also send same information to one or more managers for event correlation [Al-Shaer, 2000, Al-Shaer et al., 1999] or for increasing the information availability by redundancy. Therefore, the multi-point group communication between agents and managers is important to provide a scalable network management infrastructure that can handle large number of managed objects and agents. Compared to point-to-point (unicast) service, the multi-point group communication such as IP multicasting also reduces the latency and the number of messages of management tasks.

IP multicast is the de facto standard of multi-point group communication in the Internet. IP multicasting enables sending packets to a group of receivers without duplicating them at the source. Multicast packets are, instead, duplicated automatically by multicast routers that are close to the receivers in the network. Multicast receivers can start and stop receiving multicast packets dynamically through *join* and *leave* multicast operations respectively.

Current implementations of SNMP agents use only unicast for communication. Many approaches were proposed to support group communication in network management platform as a separate layer or a broker service between the manager and the unicast SNMP agents. However, in these approaches, the communication between the brokers and SNMP agents is still unicast and causes unnecessary overhead. In addition, they lack the dynamic reconfigurability of group communication at run-time. This paper presents a framework that facilitates integrating IP multicasting in standard SNMP agents with minimal changes in the agent implementation. The proposed framework is a manager-centric control which allows managers to reconfigure the agents' group membership and communication dynamically at run-time and based on the application demands.

This paper is organised as follows: Section 1 describes the design principles for efficient integration of multicasting in SNMP; Section 2 presents our proposed framework for integrating IP multicast in SNMP agents; Section 3 describes the framework implementation and benchmarking results; Section 4 discusses related work; Section 5 presents the summary and concluding remarks.

2. FRAMEWORK DESIGN PRINCIPLES

The main two issues that distinguish our approach are (1) using IP multicasting for multi-point communication, and (2) offering a flexible framework that can be customized by the network management application to construct any group management scheme. In this section, we describe the design principles to develop an efficient and flexible framework for integrating group communications in SNMP.

IP-multicast-based group communication. The employment of IP multicast in this framework is due to its availability in almost every network infrastructure today, and its scalability of large groups as no group information is required in the manager.

Dynamic Manager-centric Group Management. This framework enables SNMP agents and managers to join and leave multicast groups at anytime. It also enables network management applications (or managers) to configure the groups structure at run-time. This provides high flexibility while retains the simplicity of SNMP agents.

Flexible Group Communication. The framework also allows management applications (managers) to select the appropriate communication model (i.e., one-to-one, one-to-many, many-to-one and many-to-many) between SNMP managers and agents. In addition, managers may request the SNMP agents to reply to a unicast address, or a multicast address which may be used by multiple managers. This configuration can be performed dynamically at run-time.

Minimize overhead. It is important to show that this new feature cause no considerable performance overhead in SNMP agent. We discuss this issue in Section 3 and show that our framework has a negligible overhead in an SNMP agent.

Easy-to-integrate Framework. Our ultimate goal is to facilitate integrating IP multicasting in standard SNMP agents. To achieve this goal, the framework must require minimal code changes in SNMP agent implementations and other protocols.

3. A FRAMEWORK FOR INTEGRATING IP MULTICAST IN SNMP

In this section, we describe our approach and framework for extending the standard SNMP agent to support IP multicasting. The framework is designed based on the principles described in Section 1. In this framework, the integrated IP multicast service of an SNMP agent is completely configurable. The group information is stored in the MIB and used by the agents subsequently to establish the requested communication model and group organization. Therefore, the manager can request the SNMP agents to implement and dynamically change whatever group architecture suitable for the network management applications at run-time. In the following, we describe the extensions of the standard SNMP agent in order to support the IP multicast framework.

3.1 EXTENDING THE MIB WITH THE MULTICAST INFORMATION GROUP

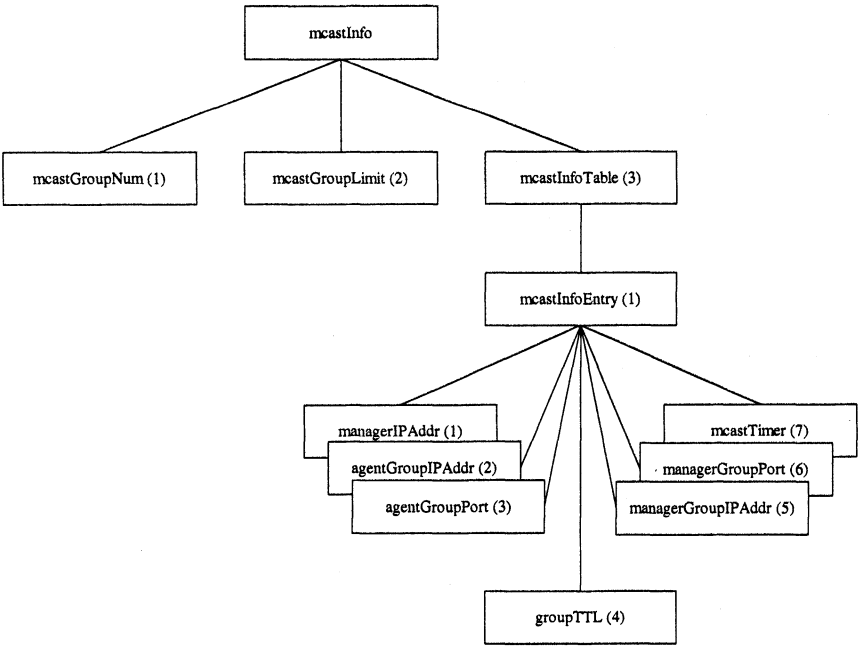

Figure 1. mcastInfo Group in Extended MIB II

A new MIB group, called Multicast Information Group (mcastInfo), was added to the standard MIB [McCloghrie and Rose, 1991] in order to store the multicast communication and group management information. Two classes of objects are supported in mcastInfo group:
(1) *Group Management Objects* that hold the address information. This class consists of **agentGroupIPAddr** and **agentGroupPort** to store the multicast IP address and port number, respectively, that the agent uses to join the multicast group. This class also contains **managerGroupIPAddr** and **managerGroupPort** that hold the IP address and port number of the manager multicast group respectively that the agent uses for the reply. If the **managerGroupIPAddr** or **managerGroupPort** is unspecified (i.e., NULL), then the agent replies to the unicast IP address of the requesting manager.
(2) *Group Communication Objects* that hold information about the communication parameters. It contains **groupTTL** object that specifies the TTL value in multicast packets, and **mcastTimer** object which indicates if the agent must use a randomized timer before sending the reply in order to avoid reply implosion in the manager [Floyd et al., 1997].

Because SNMP agent may interact with more than one manger that might request joining different multicast groups, the mcastInfo group is a table in which each entry contains the objects describe before and it is indexed by the IP address of the manager (**managerIPAddr**). Figure 1 shows the structure of the mcastInfo group table. It is important to notice that mcastInfo allows any manager to request the agent to join more than one multicast group.

Table 1. Example of Multicast Group Information in mcastInfo Table

MIP	AGIP	AGPort	MGIP	MGPort	TTL	Timer
128.39.2.2	224.4.4.4	5000	NULL	NULL	8	0
128.39.2.2	224.5.5.5	6000	224.33.2211	5555	8	0
128.39.5.5	224.8.8.8	7000	NULL	NULL	8	0

MIP: managerIPAddr AGIP: agentGroupIPAddr Timer: mcastTimer
MGIP: managerGroupIPAddr AGPort: agentGroupPort MGPort: managerGroupPort

Table 1 shows an example of mcastInfo table. In this table, the agent is requested by the manager of IP address 128.39.2.2 to join the following two multicast groups 224.4.4.4/5000 and 224.5.5.5/6000, and to send the replies to the manager IP address (i.e. using unicast because **managerGroupIPAddr** is NULL) for requests are received via 224.4.4.4, and to the multicast address 224.33.22.11 for requests received via 224.5.5.5. This tables also shows that the agent is requested by another manager of address 128.39.5.5 to join 224.8.8.8/7000 and to send the replies to the manager unicast address.

We propose inserting the mcastInfo objects in the SNMP Group of MIB II because this group extends the SNMP agent operations. However, for experimental purposes, we implemented mcastInfo group under *enterprise* of MIB II.

3.2 EXTENDING THE MASTER SNMP AGENT

The master SNMP agent (snmpd) must be extended in order to receive, process and respond to multicast set and get requests. We outline these extensions as follows:

(1) *Integrating the MIB Instrumentation Routines:* MIB II is extended to include mcastInfo group. This also involves implementing functions in the agent program that facilitate accessing the mcastInfo objects through *get* and *set* operations. This is a standard step in extending any MIB with new objects. As we describe later, the multicast operations (join and leave) are also invoked within these functions as a result of setting the mcastInfo objects.

(2) *Integrating Multicast Join and Leave Operations:* Multicast group

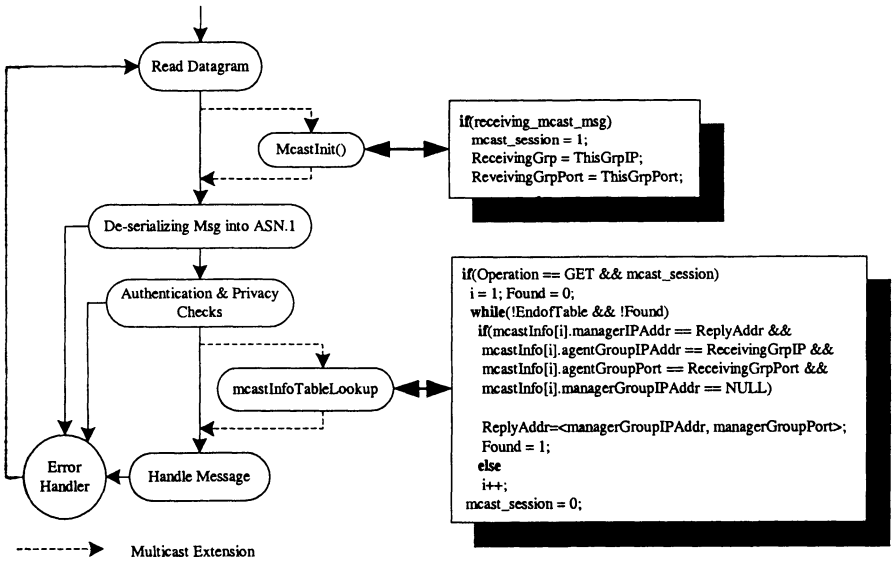

Figure 2. Extending the SNMP Agent Main Event Loop

membership is manipulated by agents through join and leave operations. The extended agent invokes these functions with the proper parameters obtained from the manager's *set* request. Another routine is also added to set the multicast TTL value in **groupTTL**. These functions (*joinGroup*() and *leave-*

Group()) are invoked by the agent as a result of setting the **agentGroupIPAddr** and **agentGroupPort** to valid values.

(3) *Extending the Agent Main Loop:* The main event loop of an SNMP agent basically consists of receiving, reading, authenticating and handling the incoming SNMP requests such as *get*, *set* requests. Figure 2 shows the main event loop of standard SNMP agents. The dashed lines in Figure 2 outline the multicast extension in the main event loop. As a result of reading the request, the IP address of the sending manager (*ReplyAddr*) is identified [Rose, 1994]. Because the proposed framework enables multicast managers to specify the type (unicast or multicast) and the multicast address of the response messages, the SNMP agent event loop was extended to search the mcastInfo table in order to find the reply address (*ReplyAddr*) before handling the message (Handle Messages in Figure 2). If the extended agent receives a multicast *get* or *get-next* request from a multicast group, then the address is stored in *ReceivingGrpIP* and *ReceivingPort* by *McastInit*() in Figure 2. The extended agent then uses *ReplyAddr*, *ReceivingGroupIP* and *ReceivingPort* to looks for a **managerIPAddr**, **agentGroupIPAddr** and **agentGroupPort** matching entry in the mcastInfo table.

The agent uses **managerGroupIPAddr** and **managerGroupPort** as the new end-point address for the reply message (*ReplyAddr*) if the corresponding entry is found. If **managerGroupIPAddr** or **managerGroupPort** is NULL (i.e., not specified by the multicast manager), then the unicast address of the sender is used for reply. This is performed by *mcastInfoTableLookup*() shown in Figure 2. It is important to notice that this function is invoked only if an SNMP request such as *get* or *get-next* is received via multicasting groups. This implies that this extension has no impact on the unicast-request path. In addition, later in Section 3, we measure and evaluate the overhead of these functions on SNMP agents.

(4) *Integrating Randomised Multicast Timer:* If the **mcastTimer** is *set* to a non-zero value by the manager, then the extended agent invokes the randomized timer function, *McastRandTimer*(), before sending the *get-reply* to the manger which puts the agent in waiting state for a very short random time. This disperses agents' replies over a short amount of time and avoid reply implosion at the manager host. We use the following exponentially weighted random timer as proposed [Handley, 1992] to estimate the random time value (t):

$$t = D_1 + RTT_{\max} * \log_2(2^{(D_2 / RTT_{\max})} * X + 1)$$

such that X is chosen from the uniform random interval [0:1], $RTT_{\max}$ is the estimated maximum round trip time, and D_1 and D_2 are the ms time interval boundaries for t.

3.3 MANAGER-AGENT INTERACTION

Similarly, SNMP managers that intend to use the multicast service must also be extended to incorporate the mcastInfo MIB objects and may be the multicast join/leave operations. Since IP multicasting allow sending to a multicast group without being a member, managers are required to incorporate join and leave functions if they intend to receive multicast reply from agents.

This section shows the interaction of a multicast-capable manager and agent using this framework. They both were extended as described in Section 2.2. One of the key attributes of such framework is that it does not require using any proprietary protocol or request type to integrate group communication in standard network management protocols. Instead, our framework utilizes the existing SNMP *get* and *set* operations to configure and control group communications in network management environments.

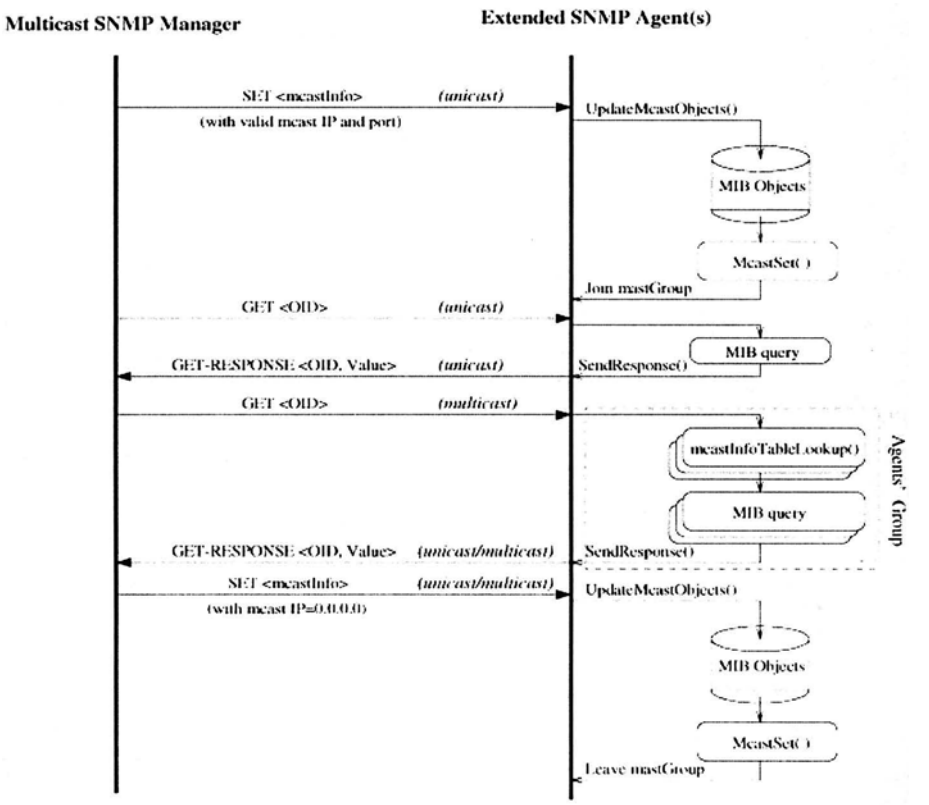

Figure 3. The Interaction of SNMP Manager and Agent Using the Multicasting Framework

Figure 3 shows the basic interaction between a manager and an agent using this framework. In this figure, the manager configures the group communication by setting the multicast information in mcastInfo objects of the agent. The manager sends this request as a unicast message to every agent to be involved in the group communication. Then, the agent reacts by updating the mcastInfo object in the MIB and joining the specified multicast group (*McastSet*() in Figure 3). On the other hand, the agent leaves the multicast group specified by **agentGroupIPAddr**, if this address is set to 0.0.0.0 by the manager. As a result of leaving a multicast group, the agents also deletes the corresponding entry in the mcastInfo table (using *McastSet*()). Since multiple managers can configure the same SNMP agent simultaneously, different managers may share the same mcastInfo objects of an agent. For this reason, the agent grants *set* requests only if they are received from the same managers that created the mcastInfo table entries. In other words, a manager can not change (*set*) mcastInfo objects that are created by another manager. If a manager attempts to set **agentGroupIPAddr** that is already in use (i.e., owned by another manager) or if the maximum number of groups indicated by **mcastGroupLimit** is exceeded, then the extended SNMP agent sends a *get-reply* to the manager reporting the problem.

If the manager uses the unicast session to communicate with an agent, then this agent always uses unicast to send the responses to the manager. Therefore, this multicast extension does not affect the standard unicast interaction between an SNMP agent and a manager which is an important design attribute of this framework. On the other hand, if the manager sends a multicast *get* or *get-next* request then, as shown in Figure 3, all agents joining this group fetch the requested values from the MIB and reply to the manager using the **managerGroupIPAddr** and **managerGroupPort** from the mcastInfo table as explained before. This means that manager can request agent to send responses as a unicast or a multicast message. Notice that even if a manager receives multicast replies from a group of agents, the manager can identify the agent IP address for each response. Therefore, the manager at any time can re-organize the agents multicast groups by requesting agents to leave a group (setting **agentGroupIPAddr** to 0.0.0.0) and join another group (setting **agentGroupIPAddr** to a valid multicast IP address).

4. THE FRAMEWORK IMPLEMENTATION AND BENCHMARKING

This framework was implemented and benchmarked in 100 Mbps switched Ethernet LAN environment that contains number of Sun Ultra workstations running Solaris 7 and 300Mz Pentium PCs running Windows

NT. We first extended the MIB II module to include mcastInfo group under *enterprises*. Figure 4 shows the extended MIB incorporated in the MG-Soft MIB browser (http://www.mg-soft.si/download.html). We use the implementation of SNMP agent from University of California at Davis (http://www.ucd-snmp.ucdavis.edu) as case study for developing this framework. The extended (multicast) SNMP manager is implemented using SNMP++ [Mellquist, 1997]. The implementation details of both the agent and the manager, and multicast integration plan is described in [Al-Shaer and Tang, 2000].

Three simple steps are required to integrate IP multicast service into a standard SNMP agent: (1) extending the MIB module to include mcastInfo group, (2) inserting *McastInit*() and *mcastInfoTableLookup*() functions in the proper places as specified in Figure 2, and (3) compile and link the source code of the SNMP agent with the multicast files and the extension library

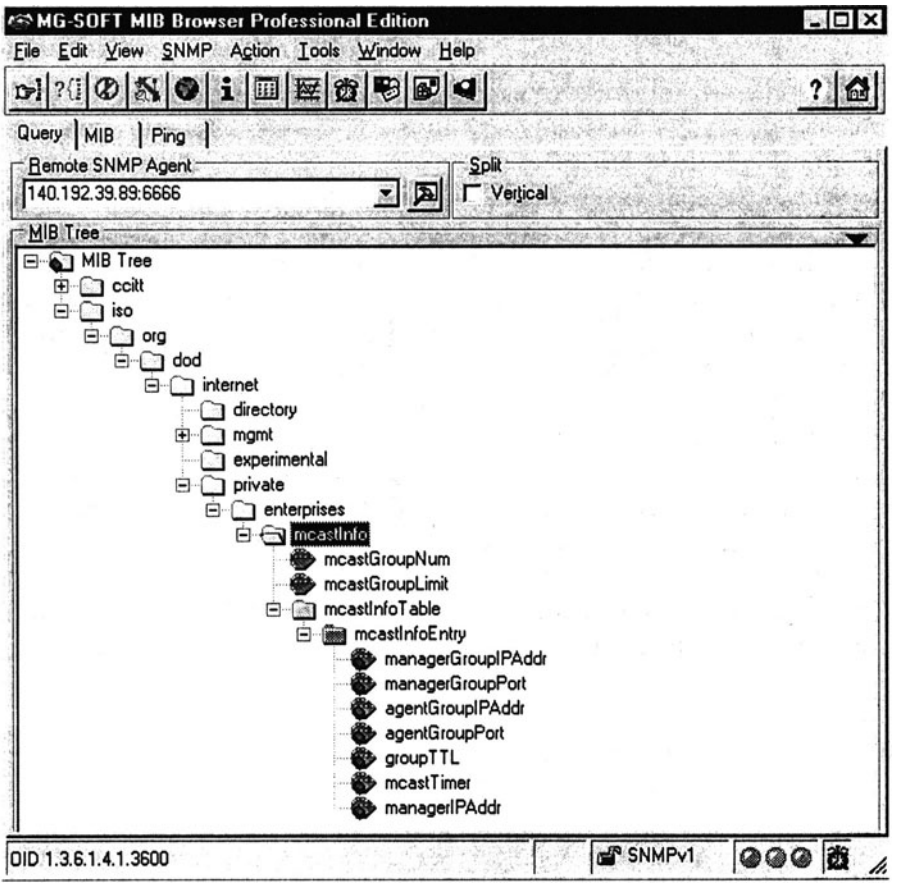

Figure 4. A MIB Browser Incorporating mcastInfo Group

(libmcastsnmp.a). The source code is available from http://www.mnlab.cs.depaul.edu/mcastsnmp/.

We conducted number of experiment to measure the overhead of this framework on SNMP agents. Figure 5 shows the response time of both standard SNMP agent using unicast and extended SNMP using multicasting agent after sending 80 different SNMP *get* requests (different OIDs) for 15 times each. Figure 5 shows that the difference in the response time between the standard and the extended agents is negligible (0 to 4 ms per response).

5. RELATED WORK

Our related-work study focuses on the systems that attempt to develop or use group communication in network management environment. A framework for using IP multicast group communication with SNMP is proposed in [Schonwalder, 1996]. This framework provides a primitive group membership structure and an election algorithm to choose a master agent. The SNMP trap messages over IP multicast are also utilized to exchange control

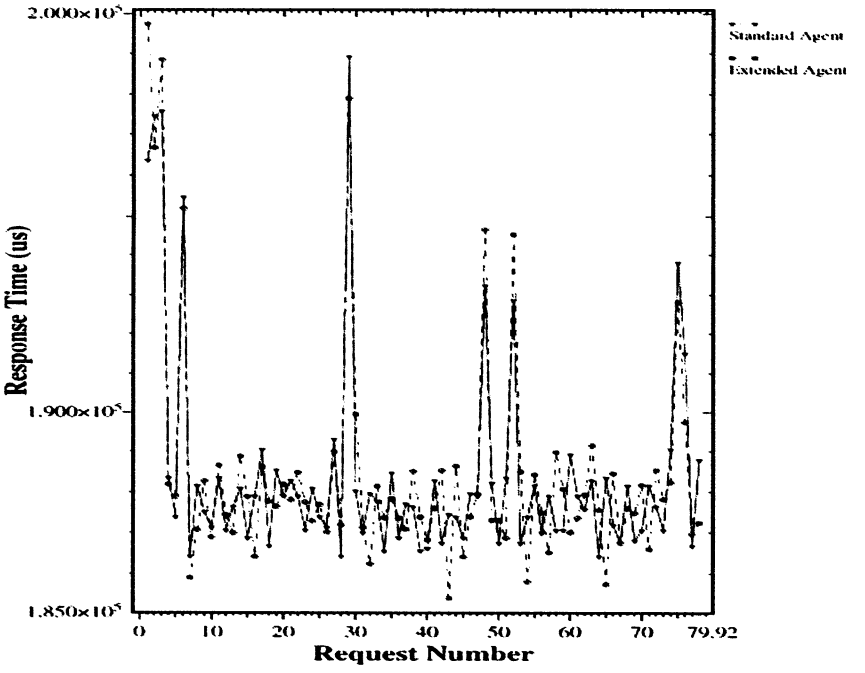

Figure 5. Response Time of the Standard and Extended SNMP Agents

information. Although this framework supports autonomous SNMP agents that use IP multicasting, it does not support an efficient framework for integrating IP multicasting in SNMP because it lacks the flexibility of multicast group and communication model re-configuration, and it requires a significant changes in the SNMP agent implementation.

In [Lee et al., 1995], a reliable group communication protocol for distributed management systems is proposed using a hierarchy of servers. However, it seems that this architecture developed over unicast connections, instead of employing IP multicasting, to emulate group communication. Furthermore, this work does not address integrating multicasting in SNMP agents.

Another group communication infrastructure based on Transis [Amir et al., 1992] group communication system was described in [Amir et al., 1996]. It is used for software installation, simultaneous remote execution and configuration in distributed systems management. A similar framework is proposed in [Parnes and Schefstorm, 1999] that uses IP multicasting for control and management of distributed applications. Although these systems show the advantage of using group communication in distributed management, they do not support an efficient framework for integrating IP multicasting in the standard network management protocols for the same reasons mentioned before in the first related work.

6. CONCLUSION

Integrating IP multicasting in SNMP agents is the most efficient way to provide scalable services in the next-generation network management platforms. Although this integration is feasible, it requires a well-engineered framework that provides high *flexibility*, and *efficiency* while retaining the *simplicity* of SNMP.

This paper presents a highly configurable, lightweight and easy-to-integrate framework for integrating IP multicast in SNMP. The presented framework enables managers to (re)configure agents' group membership (i.e., what/when to join or leave), and the communication model (one-to-one, one-to-many, many-to-one, or many-to-many) for each agent at run-time through sending standard SNMP *set* and *get* requests. This enables managers to organize and control the agents' multicast groups and communication dynamically based on management applications demands, and without introducing a new proprietary protocol. The benchmarking results show that our extended SNMP agent incurs a negligible performance overhead when compared with the standard one. The framework is implemented using UCD

SNMP agent and SNMP++, as an example, and available at http://www.mnlab.cs.depaul.edu/mcastsnmp/.

Our future research agenda includes enhancing the table mcastInfo lookup mechanism, improving the randomized timer technique to consider the group size and the location based on the network addresses.

REFERENCE

[Al-Shaer, 2000] Al-Shaer, E. (2000). Active Management Framework for Distributed Multimedia Systems. *Journal of Network and Systems Management* (JNSM), 8(1).

[Al-Shaer et al., 1999] Al-Shaer, E., Abdel-Wahab, H., and Maly, K. (1999). HiFi: A New Monitoring Architecture for Distributed System Management. In Proceedings of International Conference *on Distributed Computing Systems* (ICDCS'99), pages 171--178, Austin, TX.

[Al-Shaer and Tang, 2000] Al-Shaer, E. and Tang, Y. (2000). Implementation of SNMP Agent Multicast Extension, TR-2000-1. Technical report, School of Computer Science, Telecommunications and Information Systems, DePaul University.

[Amir et al., 1996] Amir, E., Breitgand, D., Chockler, G., and Dolev, D. (1996). Group communication as an infrastructure for distributed system management. In Proceedings of *Third International Workshop on Services in Distributed and Networked Environments*, pages 84--91.

[Amir et al., 1992] Amir, Y., Dolev, D., Kramer, S., and Malki, D. (1992). Transis: a communication sub-system for high availability. In IEEE Workshop on *Fault-Tolerant Parallel and Distributed Systems*, pages 76--84.

[Floyd et al., 1997] Floyd, S., Jacobson, V., Liu, C., McCanne, S., and Zhang, L. (1997). A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing. IEEE/*ACM Transactions on Networking*, 5(6):784--803.

[Handley, 1992] Handley, M. (1992). Multicast address allocation protocol (AAP).

[Lee et al., 1995] Lee, K.-H., Lee, J.-K., and Kim, H.-S. (1995). A multicast protocol for network management system. In Proceedings of *IEEE International Conference on Information Engineering*, pages 364 --368.

[McCloghrie and Rose, 1991] McCloghrie, K. and Rose, M. (1991). *Management Information Base for Network Management of TCP/IP-based internets*: MIB-II, RFC1213.

[Mellquist, 1997] Mellquist, P. E., editor (1997). SNMP++: *An Object-Oriented Approach to Developing Network Management Applications*. Prentice Hall PTR, Roseville, CA, (http://tachyon.rose.hp.com/snmp++/).

[Parnes and Schefstorm, 1999] Parnes, P. and Schefstorm, D. (1999). Real-Time Control and Management of Distributed Application using IP-Multicast. In Proceedings of the IEEE/IFIP *International Symposium on Integrated Network Management (IM),* pages 901--914.

[Rose, 1994] Rose, M. T. (1994). The Simple Book*: An Introduction to Internet Management.* Prentice Hall, Englewood Cliffs, NJ.

[Schonwalder, 1996] Schonwalder, J. (1996). Using multicast-SNMP to coordinate distributed management agents. In *Proceedings of Second IEEE International Workshop on Systems Management*, pages 136 --141.