EVALUATING THE PERFORMANCE OF MOBILE AGENTS IN NETWORK MANAGEMENT

Marcelo Gonçalves Rubinstein and Otto Carlos Muniz Bandeira Duarte
{rubi, otto}@gta.ufrj.br

Grupo de Teleinformática e Automação - GTA
Universidade Federal do Rio de Janeiro
COPPE/EE - Programa de Engenharia Elétrica
P.O. Box 68504 - 21945-970 - Rio de Janeiro - RJ - Brazil

Abstract

Currently, most network management systems are centralized and therefore, they can not scale when the size or the complexity of the network increases. This paper aims to investigate mobile agent scalability in network management, in order to find when mobile agents improve the management efficiency. We compare mobile agent performance with SNMP one in several simulations. Results show that the mobile agent performs better when the number of managed network elements ranges between two bounds and that the mobile agent is less sensitive to changes on a general topology.

1 Introduction

Agents are programs that help users to perform tasks, acting on behalf of these users. If these agents execute their tasks on the network, they are called mobile agents. Mobile agents can move to the place where data are stored and select information the user wants by using intelligence; saving bandwidth, time, and money.

Network management can benefit from the use of mobile agents, due to the fact that, in a general manner, this management is based on a centralized paradigm. This centralized approach does not scale when the size or the complexity of the network increases. Mobile agents decentralize processing and control, and, as a consequence, reduce the traffic around the management station, turn asynchronous agent-management station communication (useful when there are unreliable or lossy links), distribute processing load, and increase the flexibility of the management agents' behavior.

There are several research activities related to decentralization in network management, most of them associated with network management architectures. Management by Delegation (MbD) [1] has been the first paradigm to address decentralization and automation of management tasks by dynamically delegating management functions to agents. Mobile agent applications in the OSI functional areas of network management are being developed by Pagurek et al. [2]. Some investigations related to mobile agent performance in network management are also being performed [3, 4]. Baldi et al. [3] evaluate the tradeoffs of mobile code design paradigms in network management applications by developing a quantitative model that provides the bandwidth used by traditional and mobile code design of management functionalities. Sahai and Morin [4] perform measurements of bandwidth utilization of mobile agent and client-server applications on an Ethernet LAN; in this article we simulate management tasks performed by mobile agents and SNMP ones, considering bandwidth utilization and response time as performance parameters, on a general topology.

This paper aims to investigate mobile agent scalability in network management. It analyzes the applicability of mobile agents in carrying out a management task. The performance of mobile agents is compared with the SNMP one in several simulations. Analyzing the results for a topology similar in shape to the Internet, the mobile agent performs better than the SNMP when the number of managed network elements ranges between two bounds, and the mobile agent is less sensitive to changes on a general topology.

This paper is organized as follows. Section 2 presents network management architectures. Section 3 describes a network management architecture based on mobile agents. Section 4 presents simulation results concerning the applicability of mobile agents in performing a management task. At last, concluding remarks are presented in Section 5.

2 Network Management Architectures

Currently, most network management systems are centralized but some steps towards decentralization have already been taken [5].

The Internet Engineering Task Force (IETF) and the International Organization for Standardization (ISO) have proposed two protocols, SNMP [6] and CMIP (Common Management Information Protocol) [7], which are based on a centralized paradigm. These protocols use the client-server model, on which the management station acts as a client that provides a user interface to the network manager and interacts with agents, which are servers that manage remote access to their local information stored in a Management Information Base (MIB).

IETF and ISO have taken some steps towards decentralization. In event notification, SNMP agents notify the management station upon the occurrence of a few significant events by using traps, i.e., messages sent without an explicit request from the management station. The ISO approach uses more complex agents that have higher processing capacity. In both the approaches, the agent is only responsible for the notification of an event. In SNMPv2, there are multiple top-level management stations, the management servers. Each such server is responsible for managing agents, but it can delegate responsibility to an intermediate manager. The intermediate manager, also called proxy agent, plays the role of a manager, in order to monitor and control the agents under its responsibility; and it also plays the role of an agent to provide information and accept control from a higher-level management server. SNMPv3 makes additional minor functional changes to the SNMPv2 and incorporates a new security approach [6]. IETF has also proposed RMON (Remote MONitoring) that uses network monitoring devices called monitors or probes. By monitoring packet traffic and analyzing the headers, the probes provide information about links, connections among stations, traffic patterns, and status of network nodes. A probe can detect failures, misbehaviors, and identify complex events even when not in contact with the management station.

In order to effectively distribute processing load and control of the management station, new decentralized approaches are being developed. Two basic ones are: hierarchical and flat architectures [5].

The hierarchical architecture uses the concepts of manager per domain and "manager of managers". Each domain manager is only responsible for the management of its domain, and there is no direct communication between domain managers. The manager of managers sits at a higher level and requests information from domain managers. This architecture scales well and a multiple level hierarchy can be achieved by adding another level of manager of managers [5].

The flat approach is a peer-to-peer architecture in which multiple managers, each responsible for a domain, communicate with each other. Whenever information from another domain is required, the corresponding manager is contacted and information is retrieved. By distributing management over several stations throughout the network, the reliability, the robustness, and the performance of the network management increase, while network management costs in communication and computation decrease [5].

3 Proposed Management Architecture

Mobile agents need new network management architectures in order to decentralize processing and control of the management station. A network management architecture based on mobile agents is used in this paper. This architecture uses both multiple managers and manager of managers concepts (Section 2), and provides a two-level hierarchy (Figure 1). Domain Managers (DMs) are responsible for the management of their domain and may be in charge of some functions of a neighbor manager, in order to provide fault tolerance. Domain managers may communicate with each other to perform some management tasks and to analyze the network in a global manner. A Manager of Managers (MoM) is responsible for coordinating the domain managers, requesting information and delegating tasks to them, and MoM's intelligence is greater than domain managers' one.

Domain managers use mobile agents to perform their tasks, in order to automate monitoring, analysis, and

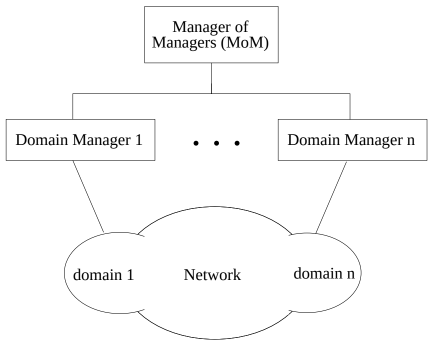

Figure 1: Network management architecture.

control of the managed network elements. A mobile agent is sent from the domain manager to the managed network element to carry out the task locally, dynamically delegating management functions. Mobile agents can perform these functions independently of DM execution, except when they can not solve the problem. In this case, a message is sent to the DM to inform the manager about the problem. If the DM can not solve the problem either, some coordination with the manager of managers is necessary. Domain managers can use different agent types to carry out their tasks. A stationary agent like in RMON, which monitors packet traffic and analyzes the headers, can also be used. This agent provides information about links, connections among stations, and status of network nodes. When detecting failures and misbehaviors, it sends a message to the corresponding DM. A dynamic code change can be performed by sending another mobile agent in order to substitute the "old one". Dynamic MIBs can be used, providing flexibility and efficiency in managed information access. DMs, if given the permission, and MoM control the contents and structure of the databases. Legacy agents like SNMP are also supported by using a translator agent.

The use of mobile agents to perform some domain manager tasks is analyzed by simulation studies.

4 Simulation Results

The applicability of mobile agents in carrying out network management functions is assessed by comparing their performance with the SNMP one in several simulations. The considered performance parameters are the number of bytes transmitted and received by a management station (domain manager for the mobile agent) and the response time in retrieving SNMP variables.

The Network Simulator (NS) is used in this paper [8]. This discrete-event simulator provides several implemented protocols and mechanisms to simulate computer networks with node and link abstractions. Some UDP modules of the NS have had to be changed in order to effectively receive a packet, sending it to the next layer.

All simulations use transit-stub topologies, which are similar in shape to the Internet ones. The GT-ITM [9] generates random topologies, on where each node represents switches or routers and edges represent forwarding paths between switches (Figure 2). Each routing domain in the Internet can be classified as either a stub domain or a transit domain. A domain is a stub domain if the path connecting any two nodes u and v goes through that domain only if either u or v is in that domain. Transit domains do not have this restriction. The purpose of transit domains is to interconnect stub domains efficiently. A transit domain comprises a set of backbone nodes, which are typically fairly well connected to each other. In a transit domain, each backbone node also connects to a number of stub domains, via gateway nodes in the stubs.

We consider that the management station belongs to a node of a stub domain and managed network elements are located in other stub domains (Figure 2). Three different 288-node topologies are used. The management station controls groups of 8 network elements, which is the number of nodes of a stub domain.

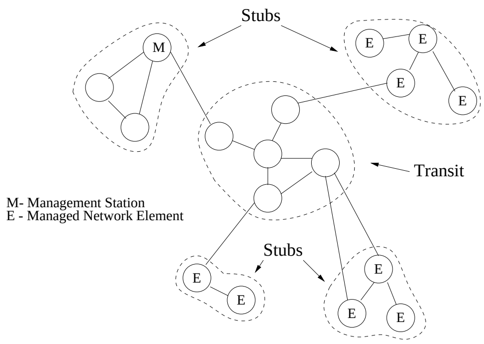

Figure 2: Network management on a transit-stub topology.

The simulation model assumes that links and nodes have no load, links are error-free, and processing time in application layer is not taken into account. UDP is used in all simulations and UDP and IP headers are also taken into account. The Maximum Transmission

Unit (MTU) used in all links is 1500 bytes, therefore, there is no fragmentation of SNMP messages since they are small. Every request/response of a variable is sent on a different message.

SNMP sends requests to all elements to be managed (one after receiving the response from the other); the mobile agent goes to an element to be managed, gathers the variable, and visits all other elements, from a stub domain to another. After finishing, the mobile agent returns to the management station.

First experiment is the retrieval of *ifInErrors* variable, which denotes the number of received packets discarded because of errors, from router's interfaces in three different 288-node networks. All links have a 2 Mbps data rate and latency links are about a few milliseconds. The initial size of a mobile agent depends on the task to be carried out and on the language used to implement it. This task uses 5 kbytes as the initial size of the mobile agent.

For a small number of Managed Network Elements (MNEs), the SNMP uses fewer bytes than the mobile agent to perform the task (Figure 3). As the number of MNEs increases, the overhead associated with several retrievals (PDU GetRequest, UDP, and IP's headers) turns mobile agent utilization more suitable. Response time for the SNMP and for the mobile agent are also analyzed for the three different topologies (Figure 4). Three points per number of MNEs are plotted for the SNMP and for the mobile agent. The mobile agent's behavior does not change significantly with the topology, but for the SNMP, the performance difference between the three topologies increases faster when the number of MNEs is large, due to the fact that SNMP packets traverse oftentimes the different backbone links that depend on the topology. The average response time per number of managed elements is also plotted (Figure 4). For a small number of managed elements, the SNMP performs better than the mobile agent, due to the fact that an SNMP message is smaller than the initial size of the mobile agent. As the number of managed elements increases, response time for the SNMP grows proportionally, since the time to manage a stub domain is approximately the same for all stub domains. For the mobile agent, the response time increases faster when the number of MNEs is large, since the mobile agent has a large size, which turns migration harder. Therefore, the mobile agent performs better that the SNMP when the number of managed network elements ranges between two limits.

The effect of the initial mobile agent size is also analyzed in one of the 288-node networks. Five different initial sizes of mobile agent are used (1, 5, 10, 15, and 20 kbytes).

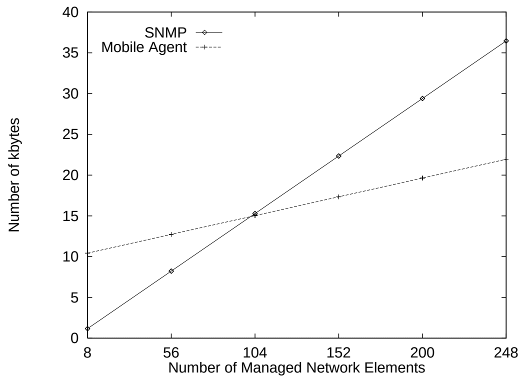

Figure 3: Number of bytes transmitted and received by the management station.

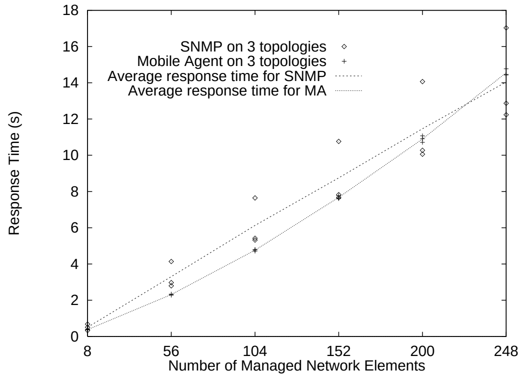

Figure 4: Response time on three different topologies (points) and average response time (lines).

As mobile agent size increases, the number of bytes associated with the management station increases proportionally (Figure 5). For the response time, as the number of MNEs increases, the performance difference between the various mobile agent sizes increases faster (Figure 6). Analyzing the curves MA-10k and SNMP, for a small number of MNEs, the mobile agent performs better than the SNMP, but when the number of MNEs is greater than 150, SNMP performance is better since mobile agent size is significant.

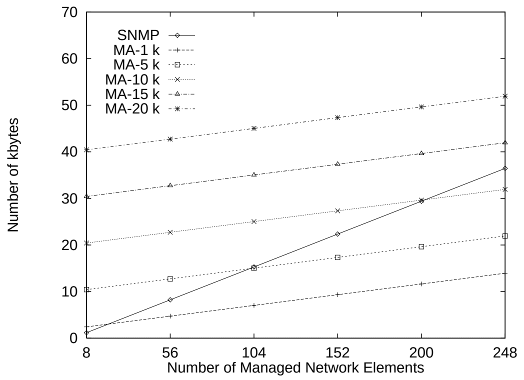

Figure 5: Number of bytes for various mobile agent sizes.

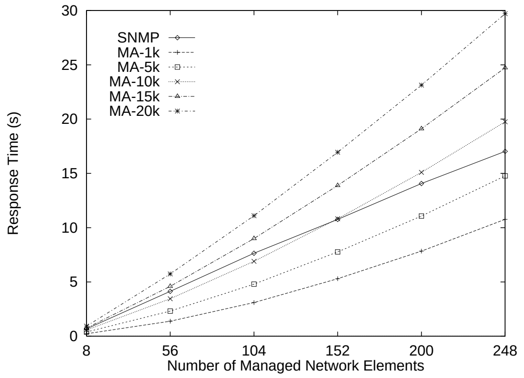

Figure 6: Response time for various mobile agent sizes.

5 Conclusion

This paper has analyzed the applicability of mobile agents in carrying out the retrieval of the *ifInErrors* variable from different router's interfaces on an architecture based on hierarchical and flat ones. Mobile agents have been used by domain managers to decentralize processing and control of the management station. The performance of mobile agents has been compared with the SNMP one in several simulations.

Simulation results for transit-stubs topologies show that a mobile agent performs better than SNMP when the number of managed network elements ranges between two limits: an inferior bound, associated with the number of messages that pass through backbone links, and, a superior bound, related to the incremental size of the mobile agent, which turns migration difficult. A mobile agent is also less sensitive to changes on the general topology due to the fact that for the SNMP, the performance difference between the three topologies increases faster when the number of managed network elements is large.

A solution to the problem of the incremental size of the mobile agent would be to visit a fixed number of managed network elements and after that, return or send all data to the management station (to reduce mobile agent size), continuing the task on the remaining managed network elements.

References

[1] G. Goldszmidt and Y. Yemini, "Delegated agents for network management", *IEEE Communications Magazine*, vol. 36, no. 3, pp. 66–70, Mar. 1998.

[2] A. Bieszczad, B. Pagurek and T. White, "Mobile agents for network management", *IEEE Communications Surveys*, vol. 1, no. 1, Sept. 1998.

[3] M. Baldi and G. P. Picco, "Evaluating the tradeoffs of mobile code design paradigms in network management applications", in *Proceedings of the 20th International Conference on Software Engineering (ICSE'98)*, Kyoto, Japan, pp. 146–155, Apr. 1998.

[4] A. Sahai and C. Morin, "Towards distributed and dynamic network management", in *Proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS'98)*, New Orleans, USA, Feb. 1998.

[5] M. Kahani and H. W. P. Beadle, "Decentralised approaches for network management", *Computer Communications Review*, vol. 27, no. 3, pp. 36–47, July 1997.

[6] W. Stallings, "SNMP and SNMPv2: The infrastructure for network management", *IEEE Communications Magazine*, vol. 36, no. 3, pp. 37–43, Mar. 1998.

[7] Y. Yemini, "The OSI network management model", *IEEE Communications Magazine*, vol. 31, no. 5, pp. 20–29, May 1993.

[8] K. Fall and K. Varadhan, "NS Notes and Documentation", tech. rep., The VINT Project, Jan. 1999.

[9] E. W. Zegura, K. L. Calvert and M. J. Donahoo, "A quantitative comparison of graph-based models for internet topology", *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, pp. 770–783, Dec. 1997.