

# System Security Management via SNMP

F. Stamatelopoulos, G. Koutepas, B. Maglaris

Network Management and Optimal Design (NETMODE) Lab  
Department of Electrical and Computer Engineering  
National Technical University of Athens

Iroon Politechniou 9, Zographou, 157 80 Athens, Greece

tel.: (+301) 772.1448 fax: (+301) 772.1452

e-mail: fotis@netmode.ntua.gr, gkoutep@netmode.ntua.gr, maglaris@ntua.gr

## Abstract

We present a framework for managing system security, based on an SNMP Management Information Base (MIB), namely the System Security MIB (SSEC MIB). We have defined managed objects and completed the ASN.1 description of the MIB that embeds them. The related security management functions are mainly focused on monitoring, external script execution for system security scanning and access control. The main goal of this work is to introduce the semantics and a standard interface that will allow the realisation of specific system security management functions independently of the underlying architecture. Our definitions pertain to multi-user, multi-tasking operating systems that support TCP/IP communications and a prototype of the SSEC MIB is under development for UNIX systems.

The proposed management framework follows the manager/agent paradigm: an agent is installed on every system connected to the network, communicating with one or more central managers through a management protocol. We have tried not to heavily rely on polling for the manager-agent interaction by using as much as possible asynchronous notification mechanisms and allowing some limited delegated functionality for the agent (scheduling and handling of local scripts). The manager scans the agents for security information, sets specific parameters for monitoring and script execution and receives asynchronous notifications on specific events, whereas the agent maintains a MIB that provides the system-independent interface semantics, executes scripts for security scanning, performs monitoring & logging and generates the asynchronous notification PDUs.

**Keywords:** Systems Management, System Security, SNMP, agent, MIB

## I. INTRODUCTION

As data networks grow in size and complexity and the number of networked nodes increases, the need for integrated system and network management architectures becomes more intense. Modern computer systems are potential targets for a wide variety of security threats, especially when the systems are connected to a data communications network (being connected to the Internet seems to be the worst case scenario as far as security is concerned). Efficient system and network administration has always been a difficult task consuming large amounts of specialised man power. If the extra complexity of managing security is added to that, large networks of computer systems may prove unmanageable via the conventional monitoring and remote configuration tools. In addition, the process of remote administration through insecure protocols (like *Telnet* for example) widely used by system administrators, introduces additional security weaknesses.

We propose a management framework for system security, based on the SNMPv2 protocol [3] and an agent with enhanced functionality, that integrates into the conventional network management scheme. SNMP (and SNMPv2) has been designed as a network management protocol, but it has been proven to be able to handle system (Host Resources MIB [ 4,7]) and protocol (FTP MIB [ 1]) management. Further, our work has been inspired by the concept of an intelligent agent [ 7] and aims to add intelligence and functionality to the agent level and even delegate that higher level functionality [ 4] that is usually installed at the manager.

The proposed architecture achieves system security management through the enhancement of the agent with system management functions. These functions are accessible by the manager through an interface that relies more on asynchronous notification than polling. Agent-to-Manager communication is achieved through a special SNMPv2 MIB module that we defined, the system security (SSEC) MIB. The SNMPv2 protocol provides an acceptable level of security at the transport of management information between the agent and the manager.

## II. SYSTEM & NETWORK SECURITY REQUIREMENTS

Networked computer systems face security threats that emanate from both the operating system and the network layer. The most important threats associated to the OS layer are:

- Unauthorised user access.
- System probing for discovery of security weaknesses.
- Viruses and Trojan horses.
- Privileged programs, misuse of files that provide uncontrolled remote access to user accounts (e.g. rhosts).
- Errors in system configuration (especially by personnel without enough expertise for undertaking the administration tasks).
- Operating system bugs.
- Non approved user actions.

Network associated threats include:

- Unauthorised repeating of captured messages.
- Unauthorised modification of information.

- Masquerading of identity or host addresses (forgery).
- Denial of service attacks.

Various tools exist for coping with security problems. Most of them operate on a focused, per-problem basis and do not provide a standard common-user-interface solution. In addition, they provide none or limited extensibility for facing new threats. Also, the majority of these tools support remote configuration and monitoring through insecure protocols like Telnet.

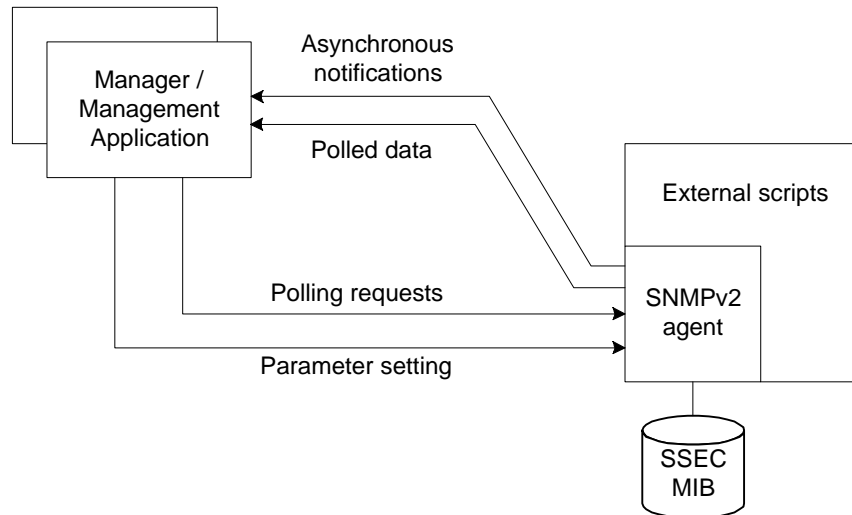
Ideally, an integrated system and network security management tool should:

- (a) be able to successfully and efficiently face all the threats mentioned above, or aid the administrator/ manager to detect and recover from related attacks,
- (b) provide a standard interface, common for all threats,
- (c) be extensible and flexible,
- (d) provide a secure operation both in monitoring and remote configuration.

### **III. MANAGEMENT FRAMEWORK OVERVIEW- GENERAL ARCHITECTURE**

The proposed management framework follows the manager/agent paradigm: an agent is installed on every system connected to the network, communicating with one or more managers through a management protocol. We have tried not to heavily rely on polling for the manager-agent interaction by using as much as possible asynchronous notification and allowing some limited delegated functionality for the agent. The manager scans the agents for security information, sets specific parameters for monitoring and script execution and receives asynchronous notifications on specific events, whereas the agent maintains a MIB that provides the system-independent interface semantics, executes scripts for security scanning, performs monitoring procedures, logging and generates the asynchronous notification PDUs.

Figure 1 presents an overview of the manager(s)-agent interaction. A modified SNMPv2 agent maintains the SSEC MIB and controls the execution of external scripts. Through these external scripts the agent performs system security scanning or provides a first reaction on specific events before the manager is notified. Their execution is driven by the parameters which the manager sets through specific SSEC MIB objects and their output is handled by the agent. The manager(s) sets parameters (access control, external script execution, monitoring, event and notification parameters) and accesses the results of the agent functions through the SSEC MIB: by polling or by receiving asynchronous notifications.



The proposed management framework is presented in three levels:

- *Low level management functions.* These are the functions performed by the agent that maintains the SSEC MIB. The agent driven by the parameters (set by the manager through MIB variables) performs monitoring & control and stores the results back to the SSEC MIB.
- *Data representation.* The MIB structure and the respective ASN.1 description define the communication semantics between the agent and the manager entities.
- *High level management functions.* These are functions performed by the manager using data retrieved from the SSEC MIB. The manager sets the parameters of the low level functions through the SSEC MIB. Feedback is granted to the manager either by asynchronous notification or by polling.

#### IV. LOW LEVEL MANAGEMENT FUNCTIONS (AGENT)

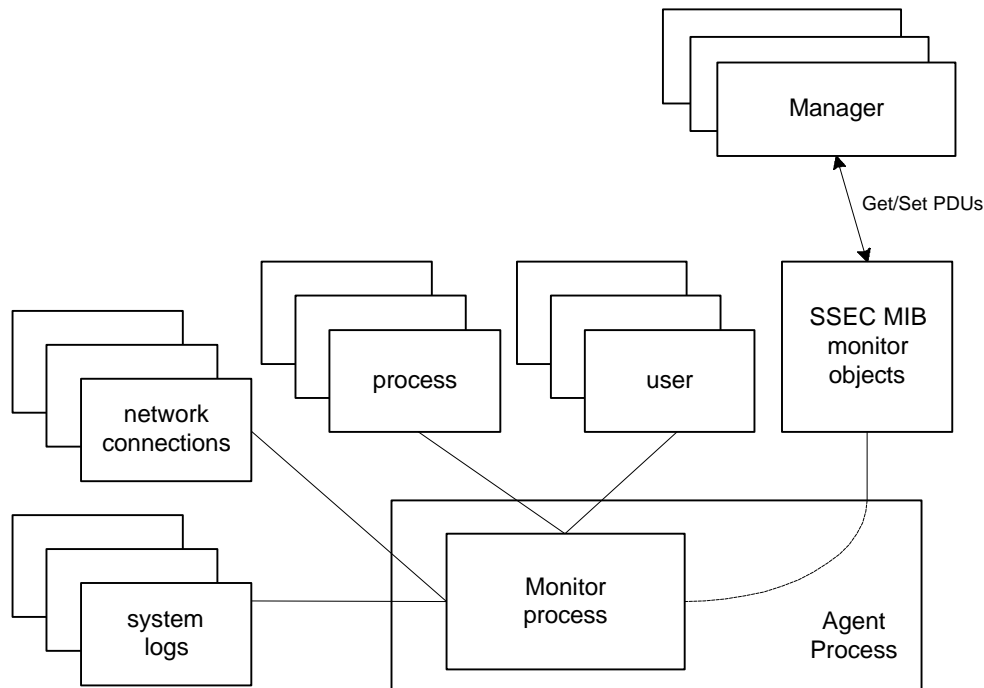
The management functions supported by the agent are focused on: *monitoring, external script execution and access control*

##### A. Monitoring

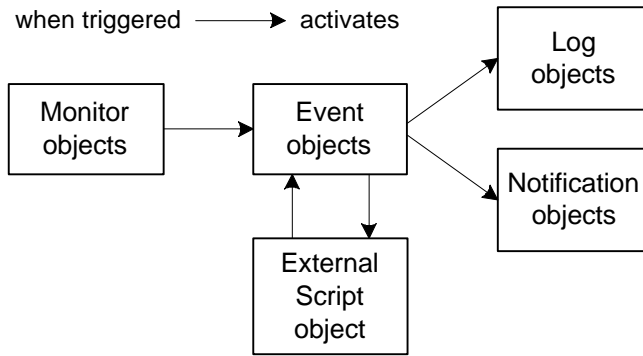
The agent supports monitoring of network connections, user access to the system ( logins), execution of processes (specific programs) and system messages. The agent provides two types of monitoring services:

- *Access and System Monitoring.* The manager may request from the agent to trigger predefined actions on the occurrence of events in these four categories (network connections, user logins, execution of processes, system messages). Actions include logging the event, sending a notification to the manager, executing an external script or performing any combination of the three.
- *System Usage Monitoring.* The manager may request (or poll) the current status of system usage at any time. The agent sends to the manager information on logged users, running jobs and/or network connections at the time of the request. This information is updated on demand and not on a periodical basis, i.e. when the manager attempts to read them.

The agent spawns a child process, the Monitor process ( Figure 2), that is responsible for monitoring network connections, use logs, process execution and system messages.



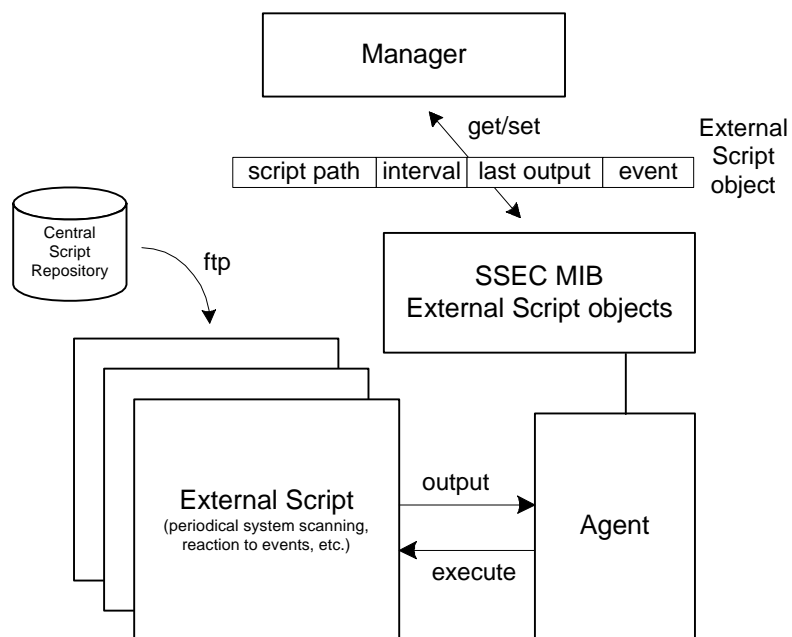
A considerable part of the SSEC MIB objects is associated with monitoring. The Monitor objects (SystemMonitor group) are used to define the parameters of specific procedures executed by the Monitor process. The Event objects (Event group) define the reaction of the agent to the activation of monitored events. An event object may be linked to Log, Notification, or External Script objects for activating when triggered the desired actions (logging, asynchronous notification, external script execution). The Log objects (Log group) provide a MIB interface to the logs maintained by the agent. Log objects are created by triggered events and stored in local files (physically) and MIB tables (conceptually). The asynchronous notification definitions that are linked to the event objects are represented by SSEC MIB Notification objects (Trap group). The External Script objects (ExternalScript group) define the parameters that govern the execution of external scripts and their interaction with the agent and they are described in detail below. When events are linked to such objects, an external script will be executed on a triggered event, aiming to initiating a system scan or to provide an initial reaction before notifying the manager. Figure 3 presents the associations between these objects. Arrows point from the object that is triggered to the object that is activated as a result. A two way association exists between Event and External Script objects, i.e. a triggered event may result in the activation of a script which in turn will trigger another event when executed. Circular linking (link an event to a script object that is linked back to it) is rejected by the agent when the manager configures the Event and External Script parameters.



The Monitor, Event, Log, Notification and External Script objects implement the Access System Monitoring services. The System Usage Monitoring services are supported by the UsersAndJobs and SystemStatus groups, whose objects offer the interface to real-time logged users / running jobs monitoring.

### B. External Script Execution

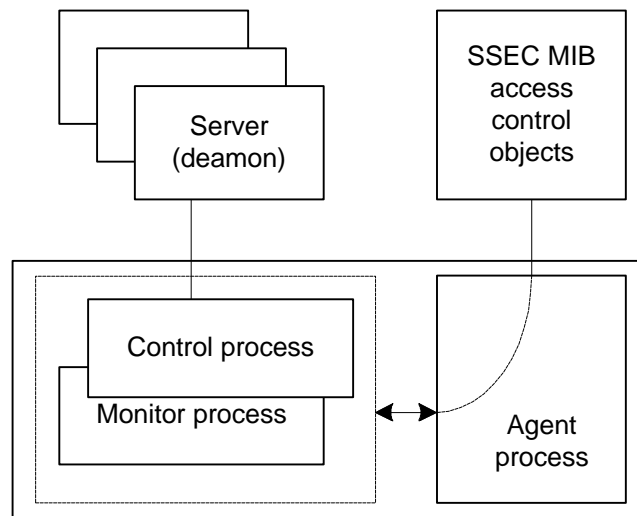
The agent supports a limited delegated functionality by allowing the execution of external scripts. Scripts are transferred manually to the agent site (by ftp, NFS, etc.) but the execution parameters are set by the manager through the External Script objects and the execution is handled by the agent locally ( Figure 4). Every External Script object contains the script local filename, the interval for periodical execution of the script (a 0 value denotes a single execution - when triggered by an event), an associated event (may be a null pointer) and a last output field (octet string). The agent executes the script every “ *interval*” seconds (if *interval* > 0) or when a linked event is triggered. The script output is stored locally and made available to the managers through the “ *last output*” field in the respective External Script object. An associated event is being activated by the agent after the completion of every execution if the script produces output. This allows for a first level of information filtering before reaching the manager but it is the responsibility of the script not the agent’s.



Note that the SSEC agent does not support any kind of scripting language; scripts are handled like external black boxes and the agent does not perform any processing on their execution results other than logging and notifying the manager. The manager is responsible for decision-making driven by the received information.

### C. Access control

In addition to access monitoring, the agent supports the definition and enforcement of system access restrictions. The manager may define restrictions on specific services (defined by port number) for certain users, user groups or network nodes through the SSEC MIB access control objects (§SystemConnRestrict group).



Access control is performed by a Control process in co-operation with the agent Monitor process which is responsible for system monitoring. Driven by the parameters set through the SSEC MIB objects and the status information maintained by the Monitor, the Control process enforces the restrictions (Figure 5). This is, of course, an outline design since details will be heavily dependent on the underlying operating system.

### V. DATA REPRESENTATION(SSEC MIB GROUPS AND OBJECTS)

The SSEC MIB is designed in conformance to the SNMPv2 SMI [ 2] and consists of 11 groups organised in 28 tables containing more than 120 objects in total. The SystemInfo, SystemStatus, SystemAccounts and SystemGroups are mainly informative groups. However, information from the AccountEntry and GroupEntry groups are used by the manager in setting the parameters of system monitoring procedures and access restriction mechanisms. The ExternalScript group maintains the parameters that control the execution of external scripts, the SystemMonitor group maintains the parameters for monitoring, whereas the UsersAndJobs group presents an MIB interface to the monitoring of system usage (logged in users and running jobs). The Event, Log and Trap groups implement the monitoring, logging and notification functionality of the SSEC MIB, respectively. Finally, the SystemConnRestrict group supports the definition of the system access restrictions.

A brief description of the 11 groups follows:

- *SystemInfo group*. This group maintains general information on the system including the host-name, ip addresses of all interfaces (pointers to the MIB II entries), aliases, etc.

It also provides information on the networked file systems (mounted and exported file systems).

- *SystemStatus group*. It maintains information on all current network connections of the system. This group contains descriptive objects for every active connection keeping track of the network service in use (name of service if known: e.g. ftp, telnet, etc.), the initiation time, the remote and local address and the associated ports (udp/tcp port).
- *SystemAccounts group*. This group maintains all the user profiles in a table (*AccountTable*) like the login name, the user ID, last login time, including a subsequent failed login counter. In the initial design, centralised account maintenance protocols, like the UNIX yellow pages (YP) or the latest NIS+ framework are ignored, e.g. YP account information is duplicated in the SSEC MIB of every system and failed logins are handled locally<sup>1</sup>. In the UNIX specific prototype which is currently under development, we have added the *ypDescr* object (*SystemInfo* group) that identifies the host as a YP server or YP client. It will be possible to allow account maintenance through the *SystemAccounts* group, by adding, removing or modifying *AccountTableEntry* objects in the *AccountTable* of the YP server MIB. We have not allowed this in the initial design, mainly for security reasons.
- *SystemGroups group*. It maintains all the group information and associates the groups to the users. All the UNIX groups of the system are included, but also additional groups may be defined without updating system files. The main goal of this group is informative and to provide the means for grouping accounts when defining monitor procedures and access restriction policies.
- *ExternalScript group*. This group maintains a table with information for executing external scripts. Each entry in the table defines the execution of an external script and contains a textual description, the time interval between executions (0 for non periodical scripts), an activation flag, an external script path, a field for last execution output and an associated event (pointer to the *Event* group or NULL).
- *SystemMonitor group*. Monitoring procedure parameters are set by the manager in the form of table entries in this group. The group maintains tables for monitoring procedures of network connections, user logins, repeated failed logins and system messages. Each table entry includes an associated event (pointer to the *Event* group) that is triggered by a monitored connection, user login, repeated failed login or system message.
- *Event group*. All the SSEC MIB events are defined as table entries in this group. All active events (an activation flag is kept in the table) are associated with a log entry, a trap, a scanning procedure or any combination of the three. The associated elements are activated when the event is triggered, i.e. an event may result in a combination of the following:
  - (a) logging the event to a table of the Log group,
  - (b) sending a trap to the manager,
  - (c) activating an external script (which in turn is associated with an event in the table).

---

<sup>1</sup> This is mainly for preserving a UNIX-independent nature. However, since nearly all modern networked, multi-user operating systems support similar protocols we intend to extend the *SystemAccounts* group to support centralized account maintenance.

- *Log group*. Maintains the logs in the form of tables. Log entries are table entries with time stamps and informative attributes. 4 tables (Network Connections, User- logins, External Scripts, System Messages).
- *Trap group*. Contains all the SSEC MIB trap definitions.
- *UsersAndJobs group*. Maintains all the active jobs and connected users at the request time.
- *SystemConnRestrict group*. Maintains the parameters for restricting access to the system through network or serial (modem, terminals) connections. The manager defines restrictions by adding / deleting objects (table entries). For each entry the manager defines a service (ftp, rlogin, etc.), the origin of the requester (address), an account or group and the denial or permission of that service. Wildcards are supported for service, user and request origin.

## VI. HIGH LEVEL MANAGEMENT FUNCTIONS (MANAGER)

The managers access the agents low level functions through the SSEC MIB and implements two types of system security functions: *event-driven* (based on asynchronous notification) and *polling-based*. The proposed framework relies mainly on event driven functions, aiming to minimise management traffic and manager load caused by excessive polling.

### A. *Event-driven functions*

- *Periodical system configuration scans*. The manager configures periodical system scans through scripts that the agents execute locally on every node. Such external scripts should scan the node for errors or intentional modifications in system configuration that may introduce security weaknesses. The agent controls the execution of the scripts locally, but it is the manager's responsibility to process the results, triggered by the asynchronous notifications. The scripts must be created, tested and distributed to the nodes (not through the SSEC MIB) by the operator, and should not be accessible to the users.
- *Execution of specialised scripts*. The manager may configure the periodical execution of scripts for specialised security monitoring function. For example, scripts may be installed for discovering password cracker programs running, programs running with super user privileges, etc. The scripts produce output when the manager should be notified and a trap is generated.
- *Monitoring and access restriction on a per node/service basis*. The manager sets monitoring procedures on system access (user logins, network connections on UDP, TCP ports) and receives notification when connections are initiated. Optionally, the agent may perform logging and execution of scripts on a new connection. Furthermore, restrictions may be defined for users, user groups or hosts on selected or all services (ports). For example sensitive services are restricted to trusted hosts and/or users.
- *Notification and filtering of system messages*. By setting a monitoring procedure on system messages the manager is notified on new messages. Filtering and classification may be performed locally at the node by linking the procedure to an external script that classifies messages, adds a priority indication and notifies the manager or discards the message depending on its class.

### ***B. Polling-based functions***

- *Poll on demand the status of the system using SNMP.* The manager may poll at any time the running processes, logged users and/or active network connections to the node. Gather usage statistics for system performance & planning (redistribution of applications, users, etc.).
- *Poll the agent logs.* The agent can be configured to store non-critical monitoring data in local logs and not generate notifications to the manager on every event. The manager is relieved from handling a flood of short messages for non-critical information. Instead, log contents are bulk-transferred when the manager is not heavily loaded.

## **VII. SECURITY ISSUES IN MANAGEMENT**

It is obvious that a security management framework should not jeopardise the system security that aims to safeguard. It would not be acceptable to use a tool for monitoring and enforcing security and at the same time introduce additional security weaknesses. Two points in the proposed framework may offer opportunities to intruders for breaking-in:

- (a) *Management information transfer between the agent and the manager.* A potential intruder may interfere with this transfer in order to initiate various attacks like masquerade, non-authorised access and modification of security information, etc. [ 4]. In the SSEC MIB design we rely solely on the SNMPv2 protocol security features [ 9] for protecting this information transfer. Access authentication and PDU encryption are used for protecting the system from security threats exploiting the SSEC MIB management services.
- (b) *Execution of external scripts.* The ability of the SSEC MIB agent to execute external scripts is extremely sensitive. An attacker may replace the specified scripts with a break-in program, a Trojan horse or other cracking processes, etc., without interfering with the management protocol. The SNMPv2 security mechanisms protect the scanning parameter setting procedure from external interference, but can not safeguard the actual script. We are investigating the possibility of adding a validation process at the agent before executing an external script.

## **VIII. SYNOPSIS - FUTURE WORK**

We have defined a SNMP-based management framework for system security, focused on monitoring, system scanning and access control. We have designed a MIB (the SSEC MIB<sup>2</sup>) and expanded the conventional agent design in order to encapsulate desired functionality.

Hard-core UNIX system administrators might argue (ours have already done so) that system administration and especially security issues, should be performed through shell scripts, utilities, etc. and not through some “fancy management protocol”. Furthermore, the SNMP people will add that SNMP is not an all-purpose management protocol but it was designed exclusively for network management. Both are probably right, at least to an extent. However, it is our belief that as architectures and operating systems tend to become more complex and get networked, the need for a standard, architecture-independent

---

<sup>2</sup> The ASN.1 description is available on-line at: <http://www.netmode.ntua.gr/ssecMIB/>

management interface will grow proportionally to the number of the nodes of the network. SNMP has been established as a market de facto standard for data network management and SNMPv2 introduced security features that are essential in system management. SNMPv2 could provide the starting point for such an effort.

Currently, a prototype of the SSEC MIB is under development for UNIX systems. We intend to experiment with the prototype implementation for testing the proposed framework for system management and fine-tuning the agent and MIB design. We will, also, try to determine the efficiency of the asynchronous notification mechanism of the SSEC MIB. In the prototype the agent sends SNMPv2 trap PDUs to the manager, i.e. unacknowledged asynchronous notification messages. If the loss of such messages is high in congested LAN conditions we will investigate alternatives to traps, like mixed polling/asynchronous notification techniques or the use of SNMPv2 information-request PDUs that provide an acknowledgement mechanism. Another issue for future work is the external script mechanism. We intend to investigate ways to avoid the manual installation of the scripts to the agent site and to support higher interaction between the agent and the scripts.

## IX. REFERENCES

- [1] Carillo J.A., Madeira E.R.M., "A Scheme for FTP Management", *INET'94 / JENC5*, 1994.
- [2] Case J., McCloghrie K., Rose M., Waldbusser S., "Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2)", SNMP Research, Cisco Systems, Dover Beach Consulting, International Network Services *RFC 1902*, January 1996.
- [3] Case J., McCloghrie K., Rose M., Waldbusser S., "Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2)", SNMP Research, Cisco Systems, Dover Beach Consulting, International Network Services, *RFC 1905*, January 1996.
- [4] International Standard Organization, "Information Processing Systems - Open Systems Interconnection - Basic Reference Model - Part 2: Security Architecture", *International Standard 7498-2* first edition 1989.
- [5] Goldszimdt G., Yemini Y., "Distributed Management by Delegation", *15th International Conference on Distributed Computing Systems*, June 1995.
- [6] Grillo P., Waldbusser S., "Host Resources MIB" *RFC 1514*, September 1993.
- [7] Riecken D., "Intelligent Agents" *Communications of the ACM* 37(7), July 1994.
- [8] Waldbusser S.L., "A Look at the Host Resources MIB", *The Simple Times*, The Bi-Monthly Newsletter of SNMP Technology, Comment, and Events, Volume 2, Number 3, May/June 1993.
- [9] Waters G., "User-based Security Model for SNMPv2 ", Bell-Northern Research Ltd., *RFC 1910*, February 1996.