;login:

THE MAGAZINE OF USENIX & SAGE
October 2002 volume 27 • number 5

inside:

SYSADMIN

Skvarcek: Remote Monitoring with SNMP

USENIX & SAGE

The Advanced Computing Systems Association &
The System Administrators Guild

remote monitoring with SNMP

by Jozef Skvarcek

Jozef Skvarcek is currently working as a system administrator. He holds a PhD in Physics. Computer technology and science are his long-time hobbies.

jozef@photonfield.net

Introduction

It is possible to monitor and administer a small number of computers individually, for example, by running an interactive session on each one. We would be most likely interested in observing the CPU, memory, and disk space utilizations or verifying that a particular process is running. We could run commands like df or ps at regular intervals in order to assure ourselves that everything is fine. We could also create various smart scripts to perform automatic monitoring functions locally and notify the administrator about the exceptions by, say, email. However, this approach has obvious limitations as the number of servers increases, for it demands a manual modification of the local scripts should the threshold values or the email address change. In general, we would want to have available one centralized point (management station) where we could set up the thresholds and process the notifications about the exceptions. We would prefer to have a unified, simple installation and configuration of the part of the monitoring software which runs on the managed nodes since that would allow for more robust and automated installation procedures for a large number of the systems. Also, we would prefer such a monitoring method to be able to work for the different platforms found in data centers nowadays.

SNMP (Simple Network Management Protocol) fits the task well despite some limitations. It takes care of network communication between the management station and managed nodes. It organizes the management information on the client side so that it can be retrieved and modified by the management station via a small number of SNMP operations. It does not process, filter, or correlate the management information retrieved from the clients. It passes the information to other application programs, or, put differently, the management applications take advantage of SNMP in order to communicate with their clients. One of the compelling reasons for using SNMP is the fact that the SNMP daemon or service is a part of the standard installation of all major modern operating systems. Enterprise-grade database systems, firewalls, and other applications often contain an SNMP module which can communicate the application-specific management information. The components of the network infrastructure such as routers or switches support SNMP; many other devices such as uninterruptible power supplies allow for the installation of a card with an embedded SNMP daemon. Therefore, with relatively little extra burden caused by the planning and configuration, we can have a multi-platform, network-monitoring capability based on open standards.

SNMP standards are defined by the RFC documents created by the Internet Engineering Task Force (IETF). The evolution of SNMP has not been straightforward, varying as different ideas were getting more or less attention. There are three versions of the protocol; the newest (SNMP v3) was standardized on March 27, 2002. From the point of view of the IETF, SNMP is a part of the Internet Standard Management Framework and all three versions share the same basic structure and components:

1. Managed nodes, each with an SNMP entity that provides remote access to management information. These are usually called "agents."
2. One or more SNMP entities with management applications. These are traditionally called "managers."

3. Management protocol to communicate management information between the SNMP entities.
4. Management information.

The architecture of SNMP is modular and it consists of:

1. Data definition language, called Structure of Management Information (SMI). It defines the fundamental data types, object model, and rules for writing the MIB modules.
2. Management Information Base (MIB) modules. They define the objects and event notifications ("traps").
3. Protocol operations. The operations are performed by Protocol Data Units (PDU).
4. Security and administration.

The SMI can be viewed as a collection of management objects residing in a virtual information store, which is the MIB. Collections of related objects are defined in MIB modules written in the SNMP MIB module language. An object in this scheme is called an "Object Identifier" (OID), an ordered sequence of non-negative integers. OID can also be represented by strings; the whole structure resembles DNS. Only a small number of MIBs (the core set) is required to be supported by each conforming agent.

NAME	DEFINITION
MIB-II	RFC 1213
Interfaces MIB	RFC 1573
SNMPv2 Core	RFC 1907

These provide access to a small common set of management information such as the system name, location, contact information and statistics about the network interfaces. There are a large number of non-mandatory MIB modules from the IETF. Another set of MIB modules are written by hardware and software vendors to support the management of their products. These enterprise-specific MIBs specify objects that lie under the "1.3.6.1.4.1" ("iso.org.dod.internet.private.enterprise") branch, sometimes simply called the "enterprise" branch. The administration of the enterprise branch is delegated to the vendors who normally provide their MIB modules as text files on some media, say, CD-ROMs or floppy disks that ship with their products. The administrator can study the files in order to determine which OIDs and traps are available. For example, the administrator would likely be searching for the OIDs holding the current values of input/output voltages or output load if he were reading the MIB which came with a power supply unit.

There are a small number of PDUs; the most frequently used are GET, GET-NEXT, SET, and TRAP. The PDUs are encapsulated into the SNMP messages and then are transmitted over the IP network using the UDP protocol. The SNMP daemon usually listens on port 161 for all PDUs except TRAP, which is sent to port 162. If you have installed a network management application with the GUI such as IBM Tivoli Netview, the PDUs are generated by choosing the appropriate items from the menus. NET-SNMP, the excellent open source SNMP agent, implements the PDUs as command-line executables.

SNMP and Security

The IETF explicitly specifies the following security threats:

1. "Modification of information" – This is the danger of the modification of SNMP information during network transmission by an unauthorized person.
2. "Masquerade" – This is the danger that an unauthorized user may assume the identity of another user with more administrative rights.
3. "Disclosure" – This is the threat that an unauthorized person may observe the SNMP communication between the managed agents and the management station.
4. "Message stream modification" – This is the danger that the SNMP messages can be re-ordered, delayed, or replayed in order to force some unauthorized management operations.

Versions 1 and 2 of the SNMP protocol provided hooks for multiple authentication schemes; however, they did not explicitly specify any but a trivial authentication scheme based on "community" strings. The "community" is included within an SNMP message, the message is transmitted in the clear-text format, and then the community is verified by the receiving entity. Obviously, this security scheme does not really solve the problems posed by the threats and is vulnerable to various sorts of attacks. It is the goal of version 3 to address the threats. It employs the traditional concept of a user who is identified by a username and secret key (passphrase). The security information is associated with the user. The cryptographic algorithms MD5 and SHA are used for the authentication and DES for the encryption. Other protocols are permitted by the standard. The standard also mandates time synchronization between the SNMP entities in order to tackle the message stream modification threat. Note, this time synchronization is performed by the SNMP agent software and is independent of other methods of system-time adjustment such as the NTP. These security methods make SNMPv3 operations safe on the public IP networks. On the other hand, user-based authentication adds another user database not correlated with the system accounts in general. That fact increases the complexity of the administration and implies derived security threats. This problem can be handled by installing an SNMP agent software which supports some kind of centralized and secure user authentication. For example, the latest version of NET-SNMP (version 5) can take advantage of the Kerberos protocol for that purpose.

Which SNMP version to use? SNMPv3 seems to be the trivial answer after what has been said in the previous paragraph. Unfortunately, the standard is too new to be fully supported by all software agents. Your choice will likely by limited by the capabilities of your management station, too. For example, popular enterprise-grade network management applications such as Tivoli Netview or HP OpenView support only SNMPv1 and v2. There are several ways to reduce the risks that stem from running SNMPv1 or v2:

1. Set the community strings. Traditionally, "public" is used as the default for the read-only community and "private" is used for the read-write community. Many agents install with these communities pre-defined. They should be changed before the agent is started for the first time.
2. Disable the write access. In most cases the read-only access is sufficient to satisfy management objectives.
3. Allow PDUs from your management station only. The configuration file of an SNMP agent normally allows you to limit the access only from pre-defined IP

addresses or domain names. If you decide to use domain names, then your DNS server should be safe from unauthorized manipulation of the DNS data.

4. Use a private, closed network for SNMP traffic.

As with other software, security vulnerabilities may be discovered in the code occasionally. Therefore, it is important to keep an eye on the security alerts issued for SNMP software you use and apply the security patches whenever necessary.

We shall take a look at the configuration and utilization of the SNMPv3 protocol in an illustrative case in the following section.

A Practical Example

We used a machine called "Jupiter" as the manager station and a machine called "Europa" as the agent. During preparation of this article, Jupiter was a PC running RedHat Linux 7.2 and Europa was a SPARC box running Solaris 8. NET-SNMP agent software version 4.2.3 was used on both systems. The NET-SNMP software on Jupiter came with the distribution, while the one on Europa was compiled from sources. There were several factors that motivated our use of this particular software. NET-SNMP supports SNMPv3; it is open source, which reduces the cost of the management infrastructure; it is under active development; and it is part of the standard installations of many GNU/Linux distributions. Last but not least, despite the complexity of the SNMP implementation, NET-SNMP has been relatively easy to configure and has been working reliably. It will be assumed in this section that the agent has been properly installed on both systems. (We shall provide more information regarding the installation in the next section.) It is assumed that OpenSSL, required for the encryption, has been installed, too.

Our first objective was to retrieve the system uptime from Europa on Jupiter. How did we know to choose this particular information? Every SNMP agent has to support the core set of MIBs, one of which is MIB-II, defined by RFC 1213. We read the document and found the object sysUpTime with OID 1.3.6.1.2.1.1.3 (iso.org.dod.internet.mgmt.mib-2.system.sysUpTime), described as the time (in hundredths of a second) since the network management portion of the system was last re-initialized. The sysUpTime object is a member of "System Group," which contains other useful objects such as sysName, sysContact, and sysLocation. A MIB may be difficult to read, since the language is meant to be processed by a machine rather than read by a human. (Please consult the books mentioned in the Bibliography section for further information.)

We started by configuring the agent on Europa. We needed to create a user ("john") with a passphrase "secret123" (the passphrase must have at least 8 characters). We put into file /var/ucd-snmp/snmpd.conf the line

```
createUser john MD5 secret123 DES
```

This version of NET-SNMP does not support SHA protocol yet, and therefore we don't really have a choice but to use MD5 for the authentication. It is possible to use a different password for the DES; you would need to put it behind DES on the line. In the next step it is necessary to create the configuration file /usr/local/etc/snmp/snmpd.conf. NET-SNMP uses the View-Based Access Control Model (VACM), defined by RFC 2575. VACM provides the administrator with the capability of allowing a particular user access to only a specified subset of the OID tree at the agent. In our case, we want to give "john" complete read-write access to the management information. Also, we want to enforce SNMPv3 and the highest security level, which means that "john" will have to use the strong authentication and encryption. Below is an example of the content of a simplified file which satisfies these objectives.

```
# /usr/local/etc/snmp/snmpd.conf
#
# Map the security name into a group name:
#
#       groupName   securityModel   securityName
group  johngrp      usm             john

#
# Create a view for us to let the group have rights to:
#
#       name   incl/excl   subtree    mask(optional)
view   all     included    .1

####
# Finally, grant access to the view.
#
#          group     context  sec.model  sec.level  prefix  read  write  notif
access  johngrp   " "      usm        priv       exact   all   all    none

#
# Set value for `system.sysLocation' object
#
syslocation Datacenter

#
# Set value for `system.sysContact' object
#
syscontact Networking

#
END
#
```

For the sake of brevity, we can't discuss the file in detail; please consult the documentation which comes with the NET-SNMP software. Finally, we are ready to launch the SNMP daemon on Europa with the command:

```
# /usr/local/sbin/snmpd
```

At this point the daemon should be listed among the running processes on Europa. An important transformation to the file /var/ucd-snmp/snmpd.conf occurs during the start-up of the daemon. The createUser line is replaced by john's security key. For the calculation of the key, the daemon used the password and the IP address of Europa. This fact is worth noting, since the security key would need to be re-created should the IP change.

The snmpd is running on Europa, but how can we access the information from Jupiter? We can do it with the GET-NEXT PDU, which is implemented by the command snmpwalk. We can run on Jupiter, for example, the following command:

```
$ snmpwalk -v 3 -u john -l authPriv -a MD5 -A secret123 -x DES \
     -X secret123 europa .iso.org.dod.internet.mgmt.mib-2.system
```

which retrieves the values of the OIDs in the "system" branch. We get several lines as the output and recognize the OIDs defined by MIB-II. Among them are the following:

```
system.sysDescr.0 = SunOS europa 5.8 Generic_108528-12 sun4u
system.sysObjectID.0 = OID: enterprises.ucdavis.ucdSnmpAgent.solaris
system.sysUpTime.0 = Timeticks: (383093) 1:03:50.93
system.sysContact.0 = Networking
system.sysName.0 = europa
system.sysLocation.0 = Datacenter
```

The zero trailing the OID names is the "instance." Some OIDs may have multiple instances; for example, the OID describing the mounting point of a disk partition has as many instances as there are partitions.

One may object that the snmpwalk command is too long and poses a security risk. A local user may learn the passwords by running the ps command while we are executing snmpwalk. To deal with this, we can create the file .snmp/snmp.conf in our home directory with the following content:

```
defVersion 3
defSecurityName john
defAuthType MD5
defAuthPassword secret123
defPrivType DES
defPrivPassword secret123
defSecurityLevel authPriv
```

This file should be readable only for the user since it includes the passwords. Having this file in place, we are able to simplify the snmpwalk command to

```
$ snmpwalk europa .iso.org.dod.internet.mgmt.mib-2.system
```

If we recall our original objective, our task was to retrieve the value of the sysUpTime object. We already have the result; the appropriate line is in the output from snmpwalk. We can get the value of that single object by using the GET PDU, which is implemented by the snmpget command. We can run on Jupiter the following:

```
$ snmpget europa .iso.org.dod.internet.mgmt.mib-2.system.sysUpTime.0
```

which returns as output the line

```
system.sysUpTime.0 = Timeticks: (1015919) 2:49:19.19.
```

Our second objective, which will be less trivial, is to monitor the disk space in the root partition on Europa. The necessary information is provided by Host MIB, defined by RFC 1514. Although this MIB is not mandatory, it is supported by many agents, NET-SNMP among them. The MIB specifies the OIDs under the iso.org.dod.internet.mgmt.mib-2.host branch. The command:

```
$ snmpwalk europa .iso.org.dod.internet.mgmt.mib-2.host
```

produces long output containing much interesting information, such as process names currently running on the system, their arguments, partitions, and mounting points, and so on. We are interested in the latest. Among the output were the following lines:

```
host.hrStorage.hrStorageTable.hrStorageEntry.hrStorageIndex.1 = 1
host.hrStorage.hrStorageTable.hrStorageEntry.hrStorageType.1 = OID:
          host.hrStorage.hrStorageTypes.hrStorageFixedDisk
host.hrStorage.hrStorageTable.hrStorageEntry.hrStorageDescr.1 = /
host.hrStorage.hrStorageTable.hrStorageEntry.hrStorageAllocationUnits.1 =
          4096 Bytes
```

```
host.hrStorage.hrStorageTable.hrStorageEntry.hrStorageSize.1 = 756012
host.hrStorage.hrStorageTable.hrStorageEntry.hrStorageUsed.1 = 120936
```

It makes sense, doesn't it? We should verify that the OIDs really are what they seem by reading their descriptions in the MIB file. There were many instances of these OIDs, but we picked up instance 1 since the OID hrStorageDescr.1 returned "/" and that was what we had been looking for. Then we found the rest of the OIDs with that instance number. The size of the file system could now be calculated by multiplying hrStorage-Size by hrStorageAllocationUnits, and the result would be in bytes. Space utilization as a percentage can be calculated using the formula

```
( hrStorageUsed / hrStorageSize ) * 100.
```

The monitoring can be automated by creating, say, a Perl script which could regularly poll the agent. Such a script could parse a configuration file where we could define the threshold. The script would then fire up an alarm if space utilization greater than the threshold were detected. The procedure could be further scaled up to monitor a large number of systems.

NET-SNMP Compilation Notes

The NET-SNMPv4.2.3 sources were downloaded from the project's home site. The agent compiled without any problem on RedHat Linux 7.2 with GCCv2.96 and on Solaris 8 with Sun Work Shop 6.1. After the sources are unpacked we need to run the "configure" script. The script takes several arguments; you can use:

```
# ./configure —help
```

in order to get the list. SNMPv3 is supported as the default, but the Host MIB is not. It is necessary to include the support by the command-line argument:

```
# ./configure —with-mib-modules="host"
```

The agent can support several more MIBs and other functionality; for example, it is possible to include the support for the tcp_wrappers. The configuration uses /usr/local as the default installation prefix, which is why the snmpd.conf file is located in the /usr/local/etc/snmp directory. It is possible to change the location of this and other NET-SNMP files by choosing the appropriate arguments for the configure script. Please read the included documentation for more information. Then we can compile the binaries:

```
# make
```

and install them with

```
# make install
```

The distribution does not provide provisions for those who wish to create the software packages. There are many Makefile files inside the source tree. One has to manually edit the install targets so that the software will be installed in a directory suitable for the build of the package.

Notes on SNMP Agents on Some Operating Systems

RedHat Linux 7.2

Typically, two packages are installed on a production box from RedHat 7.2 CD media:

- ucd-snmp
- ucd-snmp-utils

A third package ucd-snmp-devel which provides the API, can be installed on a development system. Don't be confused by the names; the packages are the NET-SNMP distribution version 4.2.3. (NET-SNMP had been previously known as UCD-SNMP.) The full path to the configuration file is /etc/snmp/snmpd.conf. The packages from the original CDs contained a security vulnerability and the snmpd daemon did not start under certain circumstances. We replaced them with the latest versions of the RPM packages from the home site of the NET-SNMP project and that fixed both problems.

Solaris

The agent is part of the product called "Solstice Enterprise Agents," which is installed automatically during a typical installation. The product consists of five packages:

- SUNWsacom
- SUNWsasnm
- SUNWsadmi
- SUNWmibii
- SUNWsasdk

where the last one provides the development platform and is not installed if only a runtime environment is required. The configuration files are located in the /etc/snmp/conf directory. However, in our opinion this agent is difficult to configure and the documentation is not sufficient. Also, the Host MIB does not seem to be supported and neither does SNMPv3. Therefore we suggest using NET-SNMP instead.

Windows 2000

The agent is installed automatically during a typical installation. Make sure that the "SNMP Agent Service" is enabled for automatic startup. The agent is configured through the SNMP Agent Service Properties window popup in a manner similar to the other system components. The relevant registry keys are located in HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\SNMP.

The documentation can be found in the Windows 2000 Server Resource Kit, TCP/IP Core Networking Guide. The agent works well on small systems, but it seems to have certain problems serving the Host MIB information on larger multi-processor boxes. It does not support SNMPv3.

Conclusions

The scope of this article is limited to describing the configuration of the SNMP agent and demonstrating use of management information for the monitoring of the system parameters. The management information served by the Core and the Host MIBs we discussed is sufficient for monitoring:

1. Disk space utilization
2. CPU load
3. Swap space utilization
4. Running processes
5. TCP/IP statistics

and other parameters. The aim is to give the interested reader enough knowledge for starting practical, secure remote monitoring. The other parts of the SNMP are left to the reader for further research.

Although much can be achieved by scripting SNMP polls, as hinted in the paragraph about the monitoring of disk space usage, in many cases it would be desirable to have a GUI-based network management application with a rich set of features. A few commercial products such as IBM Tivoli Netview or HP OpenView, as well as free ones such as GxSNMP, are available, although they do not support SNMPv3 yet.

We did not write much about the TRAP PDU. When a problem comes up, traps are emitted by certain hardware and software components such as network routers, UPSes, and databases that support the feature. The traps are usually processed by a network management application, which can sort them out, perform correlations, and issue notifications. In many cases the traps are more efficient than a periodical polling, since they do not cyclically consume the network bandwidth and host resources.

Acknowledgments

We would like to thank the members of the UniGroup in New York City for valuable suggestions.

Bibliography and Web Sites

BOOKS

Essential SNMP, by D.R. Mauro and K.J. Schmidt, O'Reilly, 2001

Practical Guide to SNMP v3 and Network Management, by D. Zeltserman, Prentice Hall, 1999

SNMP Network Management, by P. Simoneau, McGraw-Hill, 1999

Understanding SNMP MIBs, by D. Perkins and E. McGinnis, Prentice Hall, 1997

WEB SITES

GxSNMP: *http://www.gxsnmp.org/*

HP OpenView: *http://www.openview.hp.com/*

IBM Tivoli Netview: *http://www.tivoli.com/products/index/netview/*

NET-SNMP (UCD-SNMP): *http://net-snmp.sourceforge.net/*

RFC: *http://www.ietf.org/rfc.html*

SNMPv3: *http://www.ibr.cs.tu-bs.de/projects/snmpv3/*

The details of the standardization of SNMPv3: *http://www.ietf.org/IESG/actions.html*