HP OpenView Network Node Manager SPI for SNMPv3

for the HP-UX, Linux, Solaris, and Windows® operating systems

Software Version: 7.53

User Guide

Document Release Date: July 2008

Software Release Date: July 2008

Legal Notices

Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notices

Documentation Updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for recent updates or to verify that you are using the most recent edition of a document, go to:

http://h20230.www2.hp.com/selfsolve/manuals

This site requires that you register for an HP Passport and sign in. To register for an HP Passport ID, go to:

http://h20229.www2.hp.com/passport-registration.html

Or click the New users - please register link on the HP Passport login page.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HP sales representative for details.

Support

Visit the HP Software Support web site at:

www.hp.com/go/hpsoftwaresupport

This web site provides contact information and details about the products, services, and support that HP Software offers.

HP Software online support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support web site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and sign in. Many also require an active support contract. To register for an HP Passport ID, go to:

http://h20229.www2.hp.com/passport-registration.html

To find more information about support access levels, go to:

http://h20230.www2.hp.com/new_access_levels.jsp

Contents

1 Introduction to this Manual

Introduction

This manual is intended for users of HP OpenView NNM SPI for SNMPv3 (SNMPv3 SPI). It is a guide and reference tool for the speci•c software package it accompanies. This manual does not attempt to be a tutorial for SNMP.

Audience

The content of this manual has been created based on the following assumptions:

- The reader has a basic understanding of SNMP.
- The reader may have a basic understanding of networking concepts.
- The reader has at least a basic familiarity with the contents of a Management Information Base (MIB) document.
- The reader understands •le and directory structures and associated commands.
- The reader knows how to use typical network software tools, such as ping or telnet.

Conventions Used in this Manual

1.3.1 Key Combinations

When key combinations are used, the keys are separated by hyphens. For example, if the Control key and the C key are to be typed together, then the key combination would be represented as follows:

Control-C

1.3.2 Directories and Files

Throughout this document, forward slashes are used to separate directories. On some operating systems, the directory delimiter is a different character. For example, Microsoft Windows systems use a backward slash character (\). Users of these operating systems must interpret the forward slashes printed in the manual as the correct delimiter as required by the platform.

Absolute •le paths (beginning with a forward slash) have an absolute position in the •le system; /etc/srconf/mgr/snmpinfo.dat for example. All other •le paths are implied to be relative to the top-level directory of the distribution unless otherwise speci•ed. For example, bin/brassd refers to the SNMPv3 SPI Server program brassd in the bin subdirectory of the distribution.

1.3.3 Long Lines of Computer Screen Text

If any lines of text that are meant to be shown on a computer screen exceed 80 characters, the line of text will end in a backslash (\)and be indented on the following line. For example, command-lines typed into a computer or lines in a con•guration •le can exceed 80 characters and must be wrapped to the next line:

```
snmpCommunityEntry t0000000 public public localSnmpID - anywhereTag \
        nonVolatile
```

1.4 Supplemental Texts

This manual may make references to chapters contained in supplemental documentation. All necessary supplemental documentation should be shipped with this product. However, note that the supplemental texts have their own index, which is separate from this manual. If this manual references any supplemental documentation that was not shipped to you, contact Hewlett-Packard OpenView.

2 Getting Started

The purpose of this chapter is to provide enough information to “bridge the gap” between receiving a box with a manual and a CD-ROM and using this new software product to work on the project for which it was licensed. Getting started requires a small amount of time and perhaps a bit of help from the local system administrator. By the end of this chapter, the reader should be able to access the programs, files, and examples presented later in this manual.

2.1 Overview

The following definitions describe the processes outlined in this “getting started” chapter:

- Installing
  Deploying the equipment in the actual run-time environment. This includes moving executable programs to the proper directories, setting up default configurations, and setting up online help systems where available.
- Testing
  Running the executable programs with all default settings to make certain that the baseline product is working normally.
- Uninstalling
  Removing all the program’s executable files, configuration files, and directories.

2.2 Prerequisites

This section describes the software that needs to be installed first.

2.2.1 Network Node Manager

The HP NNM SPI for SNMPv3 is only for use with Network Node Manager. Ensure that Network Node Manager version 7.51 or higher is installed before installing HP NNM SPI for SNMPv3.

2.2.2 Java Technology

The Java Runtime Environment, version 1.4.1 or higher required in order to run MIBGuide and the SNMPv3 Configuration Wizard. Sun Microsystems, Inc offers the Java™ 2 Runtime Environment, Standard Edition (J2SE JRE).

On Microsoft Windows systems, the JRE may require an upgrade of the Microsoft Service Pack. Hewlett-Packard OpenView recommends using the latest version of the Service Pack.

To download the Java software, visit Sun Microsystems, Inc Web site: **http://java.sun.com.**

2.3 Installing NNM SPI for SNMPv3

2.3.1 On UNIX Systems

To install NNM SPI for SNMPv3, perform the following steps:

1. Log in as root.
2. Insert the CD-ROM provided in the media pack in the drive.
3. Mount the CD-ROM by typing the following command.
   `$ mount /dev/dsk/device_name /cdrom`
   where device_name is the name of your CD-ROM drive.
4. Change the directory to cdrom directory.
5. Start the installation by executing the following command.
   `$ ./install`
   The installation message appears on the screen.

6 Check the `/tmp/SNMPv3SPI.log` file to review the status of the installation.

7 Unmount the CD-ROM by executing the following commands:

a **`$ cd /`**

b **`$ umount /cdrom`**

If an error occurs during installation, a message that describes the problem appears on the screen.

If a fatal error occurs during installation, a message that describes the problem appears on the screen and the installation program terminates.

To continue installing NNM SPI for SNMPv3, you must fix the error and rerun the installation program.

2.3.2 On Windows Sytems

To install NNM SPI for SNMPv3, perform the following steps:

1 Log in as a user with Administrator privileges.

2 Insert the CD-ROM provided in the media pack in the drive.

3 Open the Windows Command Prompt and do the following:

a Change the current directory to the CD-ROM drive.

b Run install.bat

4 Follow the instructions to complete the installation.

5 Check `%TEMP%\SNMPv3SPI.log` file to review the status of the installation.

2.4 Contents of Distribution

The description given here of the subdirectories may be used as a reference to “what’s where” in this software distribution.

The directories listed in this section contain the software components of the HP NNM SPI for SNMPv3.

On Microsoft Windows installations of Network Node Manager, the contents of distribution are present at the following location %OvInstallDir%\snmpv3. The %OvInstallDir% is the directory into which you installed HP Network Node Manager. On Windows, “brassd” is “brassagt.”

The following table lists the contents of distribution.

Table 1 Contents of Distribution

Directory Path	Contents
/opt/OV/snmpv3/config	Sample configuration files.
/opt/OV/snmpv3/config/agt	Agent configuration files. The directory contains `snmpd.cnf` file.The `snmpd.cnf` file is also present in /etc/srconf/agt/. On Windows systems the `snmpd.cnf` file is present in "%OvInstallDir%"\conf\SNMPAgent directory. See Chapter 8 for more information.
/opt/OV/snmpv3/config/mgr	Manager configuration files. The directory contains the files `mgr.cnf` and `snmpinfo.dat`. The files `mgr.cnf` and `snmpinfo.dat` are also present in /etc/srconf/mgr/ .On Windows systems the files `mgr.cnf` and `snmpinfo.dat` are present in C:\etc\srconf\mgr. See Chapter 7 for more information.
/opt/OV/snmpv3/doc	Documentation files in PDF format.
/opt/OV/snmpv3/java/wizard	Files for the SNMPv3 Configuration Wizard.
/opt/OV/snmpv3/java/wizard/html	Help files for the SNMPv3 Configuration Wizard.
/opt/OV/snmpv3/java/wizard/html/images	Graphics files to show the SNMPv3 Configuration Wizard help pages.
/opt/OV/HPOvLIC/	License-related files (for non Linux operating system).

/opt/OV/HPOvLICA/	License-related files (only for Linux operating system).
/etc/opt/OV/share/conf/OVLicense/SNMPv3spi/	Product definition (PD) file.
/opt/OV/bin/	Contains the brassd executable.
/opt/OV/snmpv3/emanate/	EMANATE master agent. The EMANATE master agent is also present in /opt/OV/bin/.
/opt/OV/snmpv3/mibs/	MIBs used for SNMP query.
/opt/OV/snmpv3/	remove.v3spi removal script used to uninstall HP NNM SPI for SNMPv3.
/opt/OV/snmpv3/utils	This directory conatins the following SNMP utilities: getone, getnext, getmany, setany, getbulk, traprcv, and trapsend.

2.5 Testing the Software

Testing the software indicates whether the software is installed correctly and if the Master Agent is functioning properly.

2.5.1 Testing HP OpenView NNM SPI for SNMPv3 Installation

This section provides testing information for a HP OpenView NNM SPI for SNMPv3 installation. This section provides the steps a user should follow to perform a SNMPv3 request for information about either the local machine or a host running a SNMPv3 Agent. If the SNMPv3 request returns information about the machine, then the test has performed correctly.

2.5.1.1 Starting and Checking the Processes

After installation has completed, start HP OpenView Network Node Manager and make sure that the EMANATE® Master Agent, MIB-II Agent, and the HP NNM SPI for SNMPv3 Server are running.

1 From a command-line prompt, run the ovstart command:

```
# ovstart -c
```

2 Check that the following processes are running:

Issue the ps -ef command and make sure that the following processes are running.

On UNIX Systems

For HP OpenView installations, use ps -ef to check that the snmpdm, mib2agt, and brassd processes are running.

```
#  ps -ef | egrep '(snmpd|agt|brass)'
    root  4500 1  0 07:04:37 ? 0:00 /usr/sbin/hp_unixagt
    root  4520 1  0 07:04:37 ? 0:00 /usr/sbin/trapdestagt
    root  4530 1521 0 07:04:45 ? 0:00 brassd -d -nnm -
subagent
    root  4510 1 0 07:04:37 ?  0:00 /usr/sbin/mib2agt
    root  4534  2967  0 07:05:03 pts/ta    0:00 egrep
(snmpd|agt|brass)
    root  4486 1 0 07:04:34 ? 0:01 /usr/sbin/snmpdm
```

Using ps -ef will show brassd. However, on Network Node Manager the same process is listed as brassagt when ovstatus is used to check the status.•

On Microsoft Windows Systems

The SNMP EMANATE® Master Agent and SNMP EMANATE® Adapter for NT should be installed and running as services.

2.5.1.2 Testing with the MIB Browser

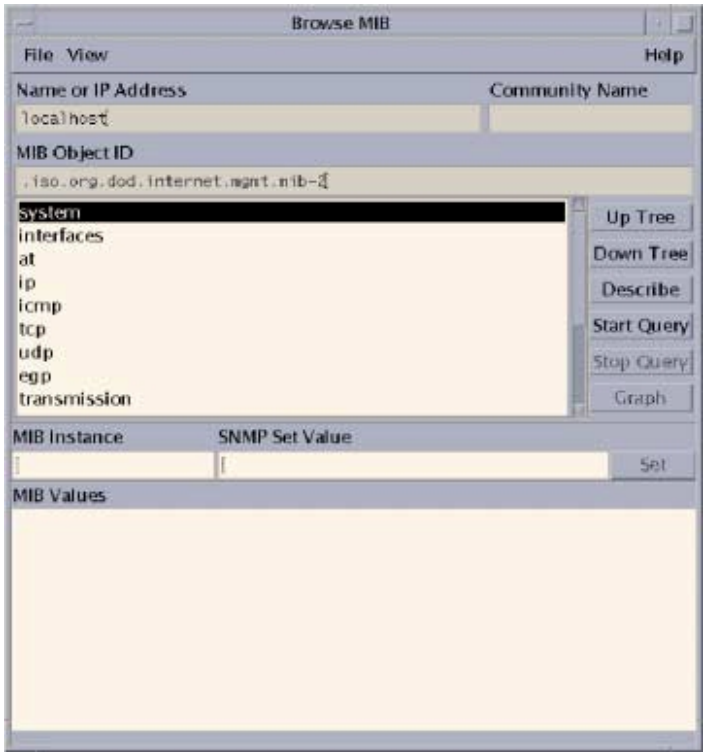
1 Run the Network Node Manager SNMP MIB Browser by doing either of the following:

Execute the following command.

— Xnmbrowser

— From an OVW map, select Tools->SNMP MIB Browser.

Figure 2.2: The Network Node Manager MIB Browser



2 When the Browse MIB window opens, highlight the system MIB by following the path iso.org.dod.internet.mgmt.mib-2.system using one of the following options:

— Double-click an item in the MIB document display window to display the branches.

— Single-click an item to select the branch, and then navigate up or down the branches using the Up Tree and Down Tree buttons.

3 Enter the following information in the corresponding text boxes to perform a SNMPv3 request.

— Type the IP address of a host configured with a SNMPv3 user in the “Network Name or Address” box. (You can perform a test on the localhost using “localhost.”)

— Enter the SNMPv3 user information for a configured user in the “Community Name” box according to the following example (and others listed in Section 2.5.1.3. If the SNMPv3 username is myUser, the authentication passphrase is myAuthPass, and the privacy passphrase is myPrivPass the information is entered as follows:

3P:myAuthPass:myPrivPass/myUser

4 Select the Start Query button.

2.5.1.3 Community String Examples

The following examples provide the format for community strings. The examples are provided as a guide. Obtain specific working usernames and passphrase information from the network administrator. More information is provided in Chapter 3.

The following list shows the community string format for each version of SNMP. SNMPv3 passwords may be “kept” in the local configuration datastore of the SNMPv3 SPI Server.

- SNMPv1
  1/community
- SNMPv2c
  C/community
- SNMPv3 noAuthNoPriv
  3N[/KEEP]/[[contextEngineID] [-contextName]/]username
- SNMPv3 authNoPriv
  3A[;[MD5^|SHA^]authKey[/KEEP]]/[[contextEngineID] [-contextName]/]username
- SNMPv3 authPriv
  3P[;[MD5^|SHA^]authKey[;[DES^|AES^|3DES^]privKey][/KEEP]]/[[contextEngineID] [-contextName]/]username

Example for Network Node Manager
The following community string would be entered in the “Community Name” box.

3P;authpass;privpass/root

2.6 Uninstalling NNM SPI for SNMPv3

2.6.1 On UNIX Systems

To uninstall NNM SPI for SNMPv3, perform the following steps:

1 Log in as root.

2 Execute the following command.
```
$ /opt/OV/snmpv3/remove.v3spi
```

2.6.2 On Windows Systems

To uninstall NNM SPI for SNMPv3, perform the following steps:

1 Log in as a user with Administrator privileges.

2 On the Windows taskbar, click Start, point to Settings, and then select Control Panel. The Control Panel window opens.

3 Double-click Add/Remove Programs. The Add/Remove Programs dialog box opens.

4 Select HP NNM SPI for SNMP v3 support from the list of programs, and then click **Change/Remove**.

2.7 Additional Information

2.7.1 Moving Configuration Files

By default, the configuration files for HP OpenView NNM SPI for SNMPv3 are installed in the /etc/srconf/ directory. After the product is installed, a user may move these configuration files to a different directory. If the files are moved, then the user must set the environment variables that identify the locations of the configuration files. If the environment variable is not defined, the application looks only in the default location.

HP OpenView Emanate agent applications (such as the SNMP Master Agent, snmpdm) look at the value of the SR_AGT_CONF_DIR environment variable to determine which directory contains the configuration files. To set SR_AGT_CONF_DIR, use the set command and specify the new directory location, for example:

On Microsoft Windows Systems

```
set SR_AGT_CONF_DIR=D:\myconf
```

On UNIX Systems
C Shell:

```
# setenv SR_AGT_CONF_DIR /usr/config/myconf
```

Korn or Bourne Shell:

```
# export SR_AGT_CONF_DIR /usr/config/myconf
```

HP OpenView NNM SPI for SNMPv3 utils (such as getmany and setany) look at the value of the SR_MGR_CONF_DIR environment variable to determine which directory contains the configuration files. To set SR_MGR_CONF_DIR, use the set command and specify the new directory location, for example:

On Microsoft Windows Systems

```
set SR_MGR_CONF_DIR=D:\MYCONF
```

On UNIX Systems

C Shell:

```
# setenv SR_MGR_CONF_DIR /usr/config/myconf
```

Korn or Bourne Shell:

```
# export SR_MGR_CONF_DIR /usr/config/myconf
```

Environment variables can be changed each time a terminal window is opened, or the variable can be set in the system-wide configuration. Refer to the operating system documentation for more information.

2.7.2 Using the Software as a Non-root User

There are special cases, determined by network managers, when the Master Agent and/or Subagents may be started by a non-root user. These cases should be carefully considered for the following reasons:

- The Master Agent's function is to help control and monitor a system. This functionality is compromised when any user has access to the Master Agent process.
- The Master Agent must bind to the system's privileged port, 161, to perform SNMP Communication. The operating system demands that only root users have access to the privileged port.
- Many Subagents require privileged access because they manipulate the kernel. It is unsafe to allow any user to have the right to manipulate the kernel.

These issues must be weighed before allowing the Master Agent or Subagents to be run by non-root users.

2.7.2.1 On Microsoft Windows Systems

If the user is not logged in as the Administrator, then the Master Agent must be started on a non-privileged port (that is, a port other than 161 and higher than 1024). Set the system environment variable SR_SNMP_TEST_PORT before attempting to start the Master Agent.

2.7.2.2 On UNIX Systems

Running Subagents as a Non-root User

Remotely Coupled Subagents, by default, can be started by non-root users on the same machine as the Master Agent. In order to allow Remotely Coupled Subagents to connect to the Master Agent, start the Master Agent (snmpdm) process with the -tcplocal argument.

There are three ways to run Loosely Coupled Subagents as non-root users:

— Change the permissions of the /tmp/.AgentSockets directory and the /tmp/.AgentSockets/A file to full permissions for any user (for example, chmod 777 /tmp/.AgentSockets). The file and directory are created by the Master Agent for master-to-agent communications. This is only a temporary solution because the permissions on the directory and file are reset when the host is rebooted.

— Change the permissions of the Subagent using chmod 755 <Subagent>. This sets the userid to be root. The Subagent then runs with root permissions though it can be started by non-root users. This is effective until the Subagent is rebuilt and the current Subagent binary version is overwritten. In the case that the Subagent is rebuilt, perform this method again.

— Run the Master Agent as a non-root user and run the Subagents using the same non-root user.

Running the Master Agent as a Non-root User

There are two ways to run the Master Agent as a non-root user:

— Change the permissions of the Master Agent using chmod 4755 snmpdm . This sets the userid to be root. The Master Agent then runs with root permissions though it can be started by non-root users.

— Set the Master Agent to bind to a non-privileged port (see section 2.7.3), that is, a port other than 161 and greater than 1024. Then set the Subagents to use the same non-privileged port. Give the Master Agent permission to create files in the /tmp/.AgentSockets directory

and to write to the file /tmp/.AgentSockets/A in order to set up the UDP socket for Subagent connections.

If the Master Agent is running as a non-root user, make sure that it has permission to create files and write to the /tmp/.AgentSockets and to the /tmp/.AgentSockets/A file. This allows the Master Agent to set up the UDP socket for Subagent connections. If the Master Agent is bound to a non-privileged port, the socket file name will have the current SNMP port number appended to the name of the file.

2.7.3 Editing the Services File

On Microsoft Windows Systems

If the user wishes to use the Microsoft® Native Agent Adapter for Microsoft Windows XP, then the snmp port should be set to something other than 161, for example, use 8161. The
EMANATE Master Agent uses port 161, by default. To change the SNMP port, perform the following steps:

- Halt the native SNMP agent:
  C:**\>net stop snmp**
- Set the port one of two ways:
  - Set the port using the environment variable SR_SNMP_TEST_PORT:
    set SR_SNMP_TEST_PORT=8161
  - Edit the C:\WINDOWS\system32\drivers\etc\services file and make sure the SNMP port (used by the native SNMP agent)is 8161:
    snmp 8161/udp
    snmp-trap 8162/udp

On Microsoft Windows XP and Windows Server 2003 systems, the services file is present in C:\WINDOWS\system32\drivers\etc directory.

On Microsoft Windows 2000 systems, the services file is present in C:\WINNT\system32\drivers\etc directory.

- Restart the native SNMP agent (optional):
  C**:\>net start snmp**

On UNIX Systems

Make sure the SNMP ports are defined in the /etc/services file (This is automatically done during installation):

```
# grep snmp /etc/services
snmp            161/udp
snmp-trap       162/udp
```

To use an alternative port (for example 8161), set the variable SR_SNMP_TEST_PORT:
C Shell:

```
# setenv SR_SNMP_TEST_PORT 8161
```

Korn or Bourne Shell:

```
# export SR_SNMP_TEST_PORT 8161
```

2.7.4 Setting the Path for Binary Components

Executable binary components, such as the Hewlett-Packard OpenView Utilities, are located in a default directory.

- On UNIX systems the directories are: /opt/OV/bin and /opt/OV/snmpv3/utils.
- On windows directories are: NNMInstallDir\bin and NNMInstallDir\snmpv3\utils

2.7.4.1 On UNIX Systems

Put the bin directory in the shell's execution path. For example, to set the binary path to the program utilities, use the following command if the distribution was installed in the /opt/OV/bin directory on a UNIX operating system:

```
# setenv PATH ${PATH}:/opt/OV/bin:/opt/OV/snmpv3/utils
```

If the path is not set, then any commands issued must be issued within the bin directory and contain the characters "./" before the command.

2.7.4.2 On Microsoft Windows Systems

On Microsoft Systems, put the bin directory containing the executable programs explicitly in the execution path. For example, if HP NNM SPI for SNMPv3 is installed on a 32-bit Intel machine in the default directory, the command would be as follows:

C:\>set path= "C:\Program Files\HP OpenView\bin";"C:\Program Files\HP OpenView\snmpv3\utils";%PATH%

This command can be issued at the start of each DOS session, or the path may be permanently set in the system-wide configuration. Refer to the operating system documentation for more information.

If the command to add the execution path is typed incorrectly, the path variable may be incorrectly set overwritten. Use the set command to verify that the path has been correctly appended to the list of entries for the path variable.

If the path is not set, then any commands issued must be issued within the bin directory and contain the characters ".\" before the command.

2.8 Troubleshooting

2.8.1 SNMPv3 Configuration Wizard Does Not Open

The SNMPv3 Configuration Wizard Requires Java Runtime Environment version 1.4.1 or higher. If Java is installed on your operating system, check the version as follows:

- Change directories into the directory in which the Java application is located.
- Execute the 'java –version' command.

The Java application should return a product banner that contains a version number. If the version is lower than 1.4.1, install an upgrade of the software.

2.8.2 Host Unknown

There are two reasons the "Host Unknown" error would be given:

1. The Hewlett-Packard OpenView Product cannot find the configuration files. Check the value of the environment variable SR_MGR_CONF_DIR and SR_AGT_CONF_DIR, then check that the configuration files are in the directory indicated by the environment variables, or change the value of the environment variables.

2 The named host has not been configured for SNMP Operations. Use the SNMPv3 Configuration Wizard utility to create an entry in the configuration file, or manually configure the host in the configuration file.

2.8.3 Product Manual cannot be opened

Make sure Adobe Acrobat Reader is installed on your host.

3 HP OpenView NNM SPI for SNMPv3

3.1 Introduction

HP OpenView NNM SPI for SNMPv3 is a combination of software which allows Network Node Manager (which speaks SNMPv1, or SNMPv1 and SNMPv2c) to send SNMPv3 requests and receive SNMPv3 notifications, including both Trap and Inform messages. SNMPv3 provides authentication and privacy mechanisms to network management.

This chapter provides an overview of HP OpenView NNM SPI for SNMPv3, describes the components of HP OpenView NNM SPI for SNMPv3, and tells how to send SNMPv3 requests.

3.2 SNMPv3 Security Protect

SNMPv3 offers five main types of threat protection:

- Masquerade: SNMPv3 can verify the identity of the SNMPv3 message's origin so that an unauthorized user can not carry out management operations by assuming the identity of an authorized user.
- Message payload modification: SNMPv3 thwarts accidental or intentional alterations of in-transit messages by checking the integrity of the data and including a time stamp.
- Message stream modification: SNMPv3 can verify that a received message is timely by checking message stream integrity and including a time stamp, so that (by malice or error) management messages are not reordered, replayed, or delayed.
- Disclosure: SNMPv3 uses encryption to protect message contents from unauthorized users eavesdropping on the management messages.
- Unauthorized Access: SNMPv3 verifies operator authorization and protects critical data from intentional and/or accidental corruption by using an Access Control Table. (SNMPv3 supports policy-based management.)

3.3 SNMPv3 SPI Components

The following sections describe each of the components of HP NNM SPI for SNMPv3. Figure 3.2 depicts how the components of the SNMPv3 SPI Server interact with one another and with SNMP managers, such as the Network Node Manager.

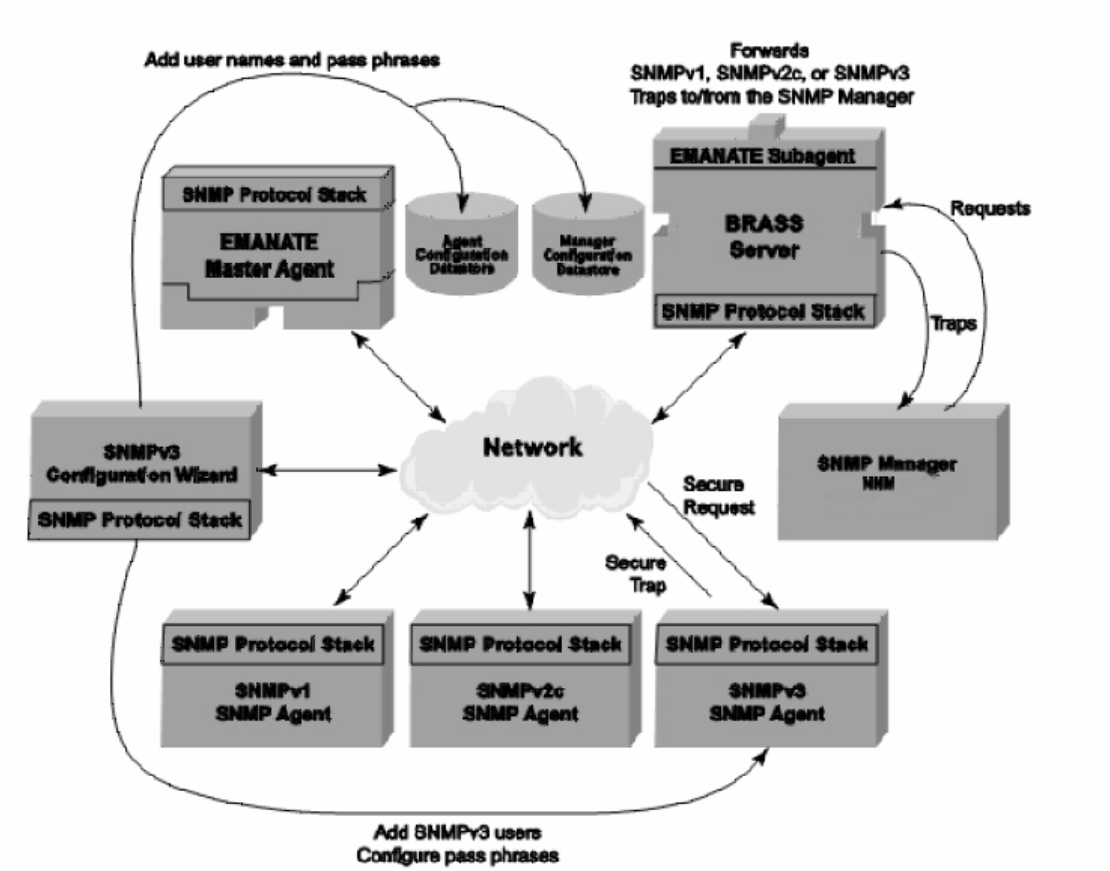


Figure 3.2: The SNMP Security Pack Architecture

3.3.1 The SNMPv3 SPI Server

The SNMPv3 SPI Server (the brassd program) acts on behalf of HP OpenView NNM to send SNMP requests and to receive SNMP responses and notifications (including Trap messages). The SNMPv3 SPI Server is the component of HP OpenView NNM SPI for SNMPv3 which allows SNMPv1- and SNMPv2c-capable management applications to communicate with SNMPv3 agents.

The SNMPv3 SPI Server, when used with the NNM Secure Polling Agent, allows HP OpenView NNM to manage networks separated from the NNM management server by one or more firewalls.

For more information about the SNMPv3 SPI Server refer to Chapter 4.

3.3.2 The EMANATE Master Agent

The SNMPv3 SPI Server can connect to the EMANATE® Master Agent so it can be managed through SNMP. SNMP manager applications can modify existing SNMPv3 users or create new SNMPv3 users in the SNMPv3 SPI Server's local configuration datastore by sending SNMP requests to the Master Agent.

The EMANATE Master Agent can replace any existing SNMP agent that may already be installed.[1] For more information about the EMANATE Master Agent, refer to Chapter 5.

3.3.3 Local Configuration Datastore

The local configuration datastore for HP NNM SPI for SNMPv3 consists of two files containing information about SNMPv3 users. One configuration file, called mgr.cnf, is used by the SNMPv3 SPI Server. The other configuration file, called snmpd.cnf, is used by the EMANATE Master Agent. More information about the contents of these files can be found in Chapter 9 and in Chapter 10.

---

[1] On systems with Network Node Manager, the SNMPv3-enabled EMANATE Master Agent replaces the older EMANATE Master Agent distributed by Hewlett-Packard.

3.3.4 The SNMPv3 Configuration Wizard

The SNMPv3 Configuration Wizard is a Java application that allows the operator to install SNMPv3 user name and pass phrase information into remote SNMPv3 agents. After the remote SNMP agents have been configured, then they can be accessed through HP NNM SPI for SNMPv3. The HP NNM SPI for SNMPv3 installation configures a default user "root" with authentication pass phrase as "authpass" and privacy pass phrase "privpass".

In Network Node Manager, the SNMPv3 Configuration Wizard can be launched from the toolbar. Do this by selecting SNMPv3 Configuration Wizard in the Tools menu.

The SNMPv3 Configuration Wizard can be launched from the command line with the run_v3wizard command. On UNIX systems, this command is a shell script located in the directory /opt/OV/bin. On Microsoft Windows systems, this command is a batch file located in the directory C:\Program Files\Hp OpenView\bin\

The SNMPv3 Configuration Wizard is described in Chapter 6.

3.4 Using SNMPv3 SPI

When NNM wants to send a SNMPv3 Get or Set request to a device, it must send the request in either a SNMPv1 or SNMPv2c message to the SNMPv3 SPI Server. In this message, the community string contains all of the information necessary for the SNMPv3 SPI Server to forward the request to the correct device in a SNMPv3 message.

The following sections describe the format for including this additional information in a community string.

3.4.1 Sending SNMPv3 Requests

When HP OpenView NNM SPI for SNMPv3 is installed and running, there should be no apparent change in the way the Network Node Manager operates.

However, HP OpenView NNM SPI for SNMPv3 causes Network Node Manager to recognize special prefix fields in the community string. This special community string format is called an overloaded community string.

Each prefix field is delimited within the community string by a forward slash ('/').

SNMPv3 overloaded community strings can have multiple fields which are as follows:

- location
- version
- authentication
- privacy
- context
- SNMPv3 username

The overloaded community string consists of at least a version prefix, a forward slash, and the community or user name.

3.4.1.1 The Version Field

The version identifies the version of SNMP to use for sending the SNMP request. Valid version prefixes are as follows:

- 1 SNMPv1
- 2C SNMPv2c
- 3N SNMPv3 with no authentication and no privacy (noAuthNoPriv)
- 3A SNMPv3 with authentication but no privacy (authNoPriv)
- 3P SNMPv3 with authentication and privacy (authPriv)

If the version prefix is not recognized, then the request is forwarded using the default SNMP version selected by Network Node Manager (SNMPv1 or SNMPv2c), using the entire community string, as is.

For example, "C/public" directs Network Node Manager to send a SNMPv2c request containing the community string "public." Likewise, "3N/public" directs Network Node Manager to send a SNMPv3 noAuthNoPriv request using the user "public."

3.4.1.2 The Community Name or User Name

When SNMPv1 or SNMPv2c is used, the community name is the SNMPv1 or SNMPv2c community that is used for sending the SNMP Request.

3.4.1.3 The Authentication Field

SNMPv3 messages require the user's authentication password or key if the user is configured to use authentication. If the user is not configured to use authentication, this field is not required. The MD5 authentication protocol is used by default. However, the user can change the default authentication protocol when starting the SNMPv3 SPI Server, or change the protocol used for a particular SNMP Request. The prefix "MD5^" or "SHA^" indicates the authentication algorithm to be used instead of the default for the request.

3.4.1.4 The Privacy Field

SNMPv3 messages require the user's authentication password or key if the user is configured to use privacy. If the user is not configured to use privacy, this field is not required. DES is the default privacy protocol. However, the user can change the default privacy protocol when starting the SNMPv3 SPI Server, or change the protocol used for a particular SNMP Request. The prefix "DES^", "3DES^", or "AES^" indicates the encryption algorithm to be used instead of the default.

3.4.1.5 The Keep Field

When a password (or key) is included in the community string for a SNMPv3 message, the user may indicate for a password to be kept in the local configuration so that it is not necessary to enter it again on subsequent requests to the same IP address for the same SNMPv3 username. If the KEEP qualifier is specified in the community string, and if the password(s)/key(s) are successfully used to communicate with an agent,[2] then the localized password(s)/key(s) [3] are stored in the SNMPv3 SPI Server's local configuration datastore.

---

[2] This feature might be used to automatically populate the local configuration datastore. The Network Node Manager operator can attempt one SNMPv3 request for each of the SNMPv3 passwords that are known to be in use, allowing BRASS™ to keep track of the passwords that work.

[3] If passwords or keys are stored in a localized form, it means that the original plain text passwords are not feasibly Recoverable. This is a security feature of SNMPv3.

3.4.1.7 The Context Field

The contextEngineID uniquely identifies a SNMP Agent. The user can specify which SNMP Agent should respond to a request in the case that the receiving SNMP Agent should forward the message to a third SNMP Agent at a different address.

The Network Node Manager does not permit the use of colons in the community string. So, the contextEngineID for a SNMPv3 message is specified using hexadecimal values separated by semicolons (';'). For example:

00;00;00;63;00;00;00;a1;0a;00;00;01

The contextName identifies the original context, in the case that the request should be sent in a context other than the default context (use GetNext requests to walk the vacmContextTable to see a list of the valid contexts for the device).

3.4.1.8 Community String Format for Network Node Manager.

The following list shows the community string format for each version of SNMP. SNMPv3 passwords may be "kept" in the local configuration datastore of the SNMPv3 SPI Server.

- SNMPv1
  1/community
- SNMPv2c
  C/community
- SNMPv3 noAuthNoPriv
  3N[/KEEP]/[[contextEngineID] [-contextName]/]username
- SNMPv3 authNoPriv
  3A[;[MD5^|SHA^]authKey[/KEEP]]/[[contextEngineID] [-contextName]/]username
- SNMPv3 authPriv
  3P[;[MD5^|SHA^]authKey[;[DES^|AES^|3DES^]privKey][/KEEP]]/[[contextEngineID] [-contextName]/]username

3.4.1.9 Examples for Network Node Manager

The following examples illustrate how to include SNMPv3 user information in the community string for Network Node Manager. These community strings must not include the location field because it is automatically provided by Network Node Manager. The community string used by Network

Node Manager does require the use of the semicolons. If the community string does not match the correct format, the SNMPv3 SPI Server will send the request as a SNMPv1 message using the entire community string as the community name.

Example for Network Node Manager

The following community string (default user) would be entered in the "Community Name" box.

3P;authpass;privpass/root

3.4.2 Receiving SNMPv3 Notifications

All received notifications are forwarded to the NNM (with a few exceptions, see Section 3.4.2.1.

When the SNMPv3 SPI Server receives a Trap message, it forwards the Trap message to the
NNM. When the SNMPv3 SPI Server receives an Inform message,[4] it sends an acknowledgement message back to the notification originator (that is, the SNMP agent that sent the Inform), it converts the Inform message into a Trap message, and it forwards the Trap message to the NNM.

By default, the SNMPv3 SPI Server forwards all notifications to the NNM as SNMPv2c Trap messages. To change the default behavior so that the SNMPv3 SPI Server will convert all incoming notifications to SNMPv1 Trap messages, start the SNMPv3 SPI Server with the -v1traps port command line argument.

For more information about command line arguments that affect HP OpenView NNM SPI for SNMPv3, refer to Chapter 4.

When a notification is forwarded to the NNM, the community string in the SNMPv2c (or SNMPv1) message contains the source IP address and authentication information of the original notification message.

The following list shows the possible formats of the community string that may be received by the NNM.

---

[4] An Inform is like a Trap, except that it must be acknowledged. A SNMPv2c or SNMPv3 agent that sends an Inform message will typically resend the message again and again until it receives the acknowledgement message from the SNMP manager. This is SNMPv3's way to ensure the reliable delivery of fault warning or other unsolicited message.

The following list shows the community string format for each version of SNMP. SNMPv3 passwords must already be "kept" in the local configuration datastore of the SNMPv3 SPI Server.

- SNMPv1
  ipaddr/1/community
- SNMPv2c
  ipaddr/C/community
- SNMPv3 noAuthNoPriv
  ipaddr/3N/[[contextEngineID] [-contextName]/]username
- SNMPv3 authNoPriv
  ipaddr/3A/[[contextEngineID] [-contextName]/]username
- SNMPv3 authPriv
  ipaddr/3P/[[contextEngineID] [-contextName]/]username

3.4.2.1 Cases When Notifications Are Not Forwarded

There are two cases when the SNMPv3 SPI Server does not forward received notifications (Traps or Informs) to the NNM.

The first case is when the incoming notification is in an authenticated (and possibly encrypted) SNMPv3 message. To receive this type of message, the SNMPv3 SPI Server must first be configured with the appropriate SNMPv3 password(s). Otherwise, the message can not be deemed as authentic, and therefore it will be ignored by the SNMPv3 SPI Server. There are a few different ways to configure the SNMPv3 SPI Server so that this does not happen:

- Store localized keys in the SNMPv3 SPI Server's local configuration datastore using the /KEEP qualifier.
- Enter plain text passwords directly into the SNMPv3 SPI Server's configuration file, `mgr.cnf`. Refer to Section 7.1.3.

The second case is when the -v1traps port command line argument is specified to the SNMPv3 SPI Server, and the received notification contains a Counter64 type MIB object. The Counter64 type MIB object is not valid in SNMPv1 messages.[5] If the command line argument has been specified and

[5] The NNM's pmd process does not accept counter64 variables in a v1 Trap. As a step in making a SNMPv1-capable management application into a SNMPv3-capable management application, some vendors have chosen to make their applications accept a SNMPv1 message containing a Counter64 type MIB object. This is a modification that is outside the SNMP

the SNMPv3 SPI Server receives a Trap containing a Counter64 type MIB object, the SNMPv3 SPI Server ignores the Trap message and does not forward it to the NNM. If the command line argument has been specified and the SNMPv3 SPI Server receives an Inform containing a Counter64 type MIB object, the SNMPv3 SPI Server sends an acknowledgement message back to the notification originator and does not forward the Inform message to the NNM.

3.4.3 Viewing Error Notifications in Network Node Manager

This section describes configuring Network Node Manager to view "HP OpenView NNM SPI for SNMPv3" SNMPv3 report and error notifications.

New functionality has been added so that in the case of an error, a SNMPv2c Trap is sent back to NNM to provide error or SNMPv3 report information.

In order to view the Trap in the NNM Event Viewer the user will need to verify that the HP OpenView NNM SPI for SNMPv3 MIBs (snmp-res.mib and secpack.mib) are loaded into NNM[6] and then modify the Event Configuration window so that the message displayed is easier to understand.

3.4.3.1 Load MIB documents

First, verify whether or not the HP OpenView NNM SPI for SNMPv3 MIBs are loaded.

1. Select Options, then Load/Unload MIBs. Scroll down to and check to see if either secpack.my or snmp-res.my is on the list of loaded mibs. If they are, then skip to section 3.4.3.2.
2. To load a mib, select Load.
3. In the Load/Unload MIBs:SNMP /Load MIB from File window, locate the mibs to be loaded in the /opt/OV/snmpv3/mibs/ directory (for example, secpack.my or snmp-res.my). Load snmp-res.my first. See Figure 3.4 for an example. Then load secpack.my. When the MIBs are being loaded,

---

standards. Nevertheless, to support such hybrid applications, the SNMPV3 SPI Server has another command line argument, -c64, that allows notifications containing Counter64 type MIB objects to be forwarded in SNMPv1 messages.

[6] On a Solaris installation, the Hewlett-Packard OpenView MIBs are loaded automatically. On Microsoft Windows XP and HP-UX installations, the user must manually add them.

NNM should detect that there are Notifications. The user will be prompted to select to add the information to trapd.conf.

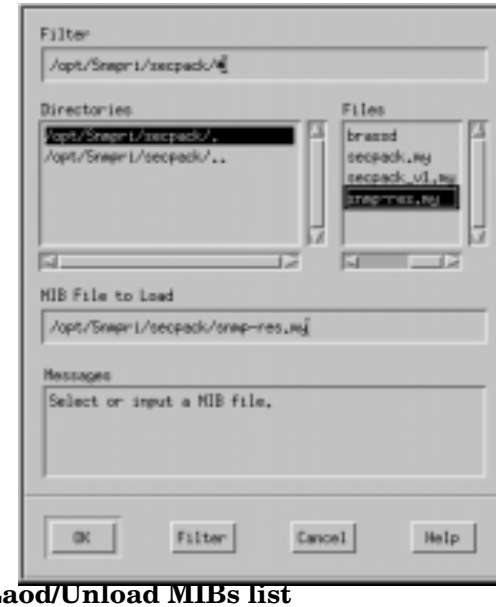


Figure 3.3: Check the Laod/Unload MIBs list

Figure 3.4: Load MIBs

3.4.3.2 Configure the Events

The following steps allow the user to view the Traps in the NNM Event Viewer from the Error Events window when they are sent by the SNMPv3 SPI Server.

1 From the OpenView GUI Main Window, Select Options, then EventConfiguration.

2 Select snmpSecurityPackMIB. If the notifications were not automatically added in step 1, then perform the following steps:

3 Set Enterprise Identification. Select Edit, Add, Enterprise Identification (See Figure 3.5.). Then, type the following information in the text boxes in the Add New Enterprise window (See Figure 3.6):

Enterprise Name: snmpSecurityPackMIB Enterprise ID: .1.3.6.1.4.1.99.12.45

Figure 3.5: Event configuration window

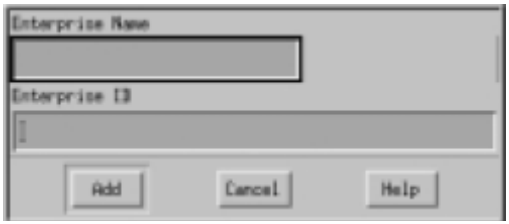


Figure 3.6: Add new enterprise window

4 Select ADD.

5 Set the Event Notification type. From the Event Configuration window, select,Edit, Add, and then Event and type the following text in the Add New Event window.

Event Name: snmpSecurityPackError
EventObjectIdentifier: .1.3.6.4.1.99.12.45.3.1

6 Select ADD.

7 In the Event Configuration window, select in the Category scroll box Then, type
"Security Pack Error: $1" in the Event Log Message text box.

8 Select OK.

9 Set the Event Notification type. From the Event Configuration window, select Edit, Add,
then Event and type the following text in the Event Configuraton / Add Event ...
window (see Figure 3.7):

Event Name: snmpSecurityPackError
EventObjectIdentifier: .1.3.6.4.1.99.12.45.3.2

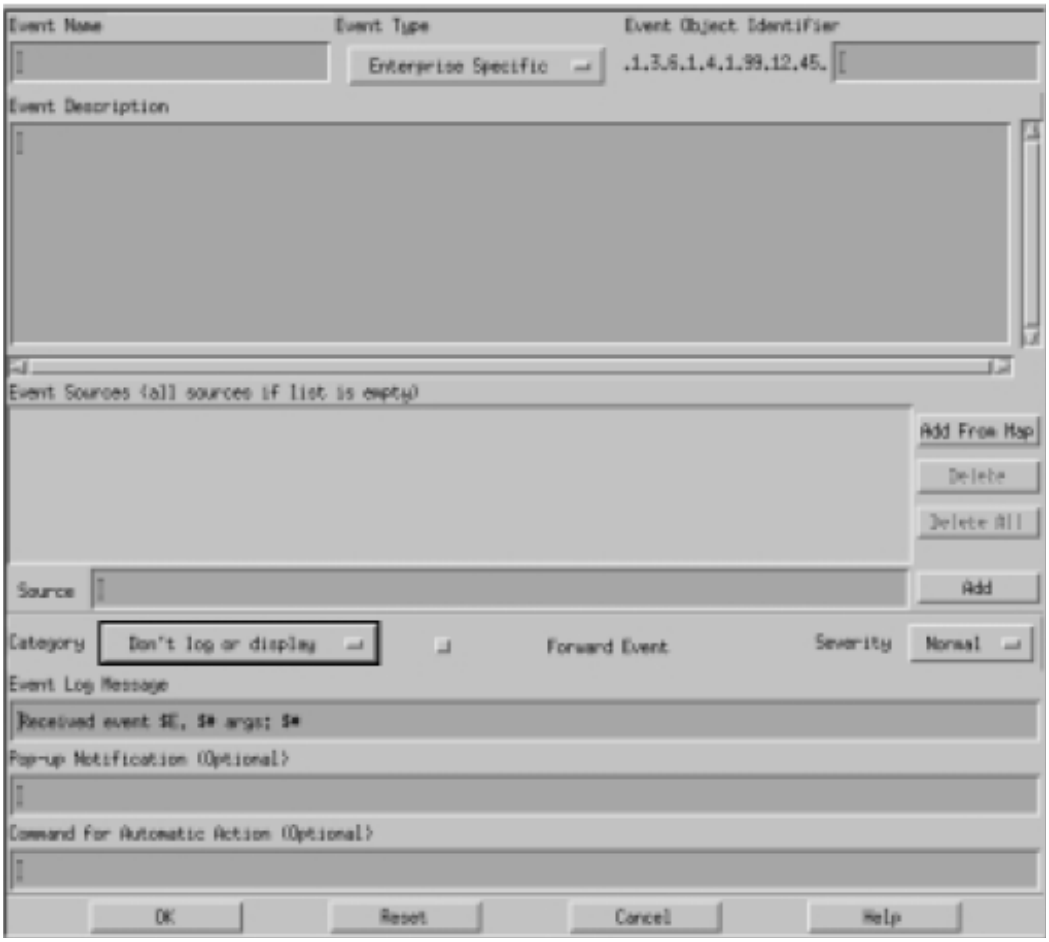


Figure 3.7: Event Configurator / Add Event window

10 Select Error Alarms in the Category scroll box. Then, type “Security Pack SNMPv3
Report: $1” in the Event Log Message text box.

11 Select OK.

4 The SNMPv3 SPI Server

This chapter provides instructions about how to run the SNMPv3 SPI Server-Subagent (hereafter referred to simply as the “SNMPv3 SPI Server”) on various platforms.

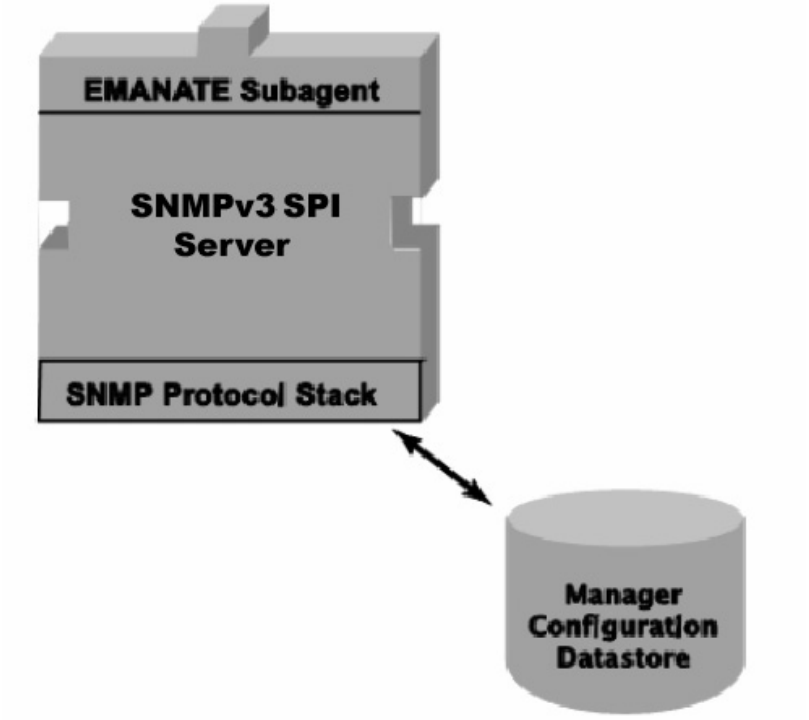


Figure 4.1: The SNMPv3 SPI Server

4.1 Before Starting the SNMPv3 SPI Server

4.1.1 Under HP OpenView

The HP OpenView NNM SPI for SNMPv3 version 1.00 is supported only with NNM 7.51 or later versions.

The name “brassagt” may be substituted for “brassd” for Network

Node Manager installations.

1. Make sure that the brassd executable program has been copied into HP OpenView's bin directory.

   On UNIX systems, the full path name of the file is /opt/OV/bin/brassd.

   On Microsoft Windows systems, the full path name of the file is C:\Program Files\HP OpenView\bin\brassagt.exe.

2. Make sure that the SNMPv3 SPI Server's configuration files have been copied into HP's standard configuration directory.

   The files are mgr.cnf and snmpinfo.dat.

   On UNIX systems, the full path name of the configuration directory is /etc/srconf/mgr. On Microsoft Windows systems, the full path name of the file is C:\etc\srconf\mgr. If the files are not found in the correct directory, copy them to that location.

3. Examine the services file to look for the snmpv2s and snmpv2s-trap services. On UNIX systems, the full path name of the file is /etc/services. On Microsoft Windows XP, the full path name of the file is C:\WINDOWS\system32\drivers\etc\services.

   ```
   # egrep "snmp|unm" /etc/services
   snmpv2s                         4747/udp
   snmpv2s-trap                    4748/udp
   #
   ```

   On Microsoft Windows XP and Windows Server 2003 systems, the services file is present in C:\WINDOWS\system32\drivers\etc directory.

   On Microsoft Windows 2000 systems, the services file is present in C:\WINNT\system32\drivers\etc directory.

   If these services are not found in the services file, then add to the file the two lines exactly as they appear above. These lines define the ports that Network Node Manager and HP OpenView NNM SPI for SNMPv3 use to communicate with one another. The port number assigned to the snmpv2s service is used to forward SNMP requests and reply messages. If no port number is assigned to this service, then no SNMP request messages will be exchanged between HP OpenView and HP OpenView NNM SPI for SNMPv3, and Network Node Manager will not be able to communicate using SNMPv3. The port number assigned to the snmpv2s-trap service is used to forward SNMP notifications; i.e., traps. If no port

number is assigned to the snmpv2s-trap service, but a port number is assigned to the snmpv2s service, then the port number that will be used for the snmpv2s-trap service will be one greater then the port number assigned to the snmpv2s service.

If no port number is assigned to either the snmpv2s service or the snmpv2s-trap service, it is still possible for HP OpenView NNM SPI for SNMPv3 to forward SNMPv3 traps to Network Node Manager if a port number for the snmpv2s-trap service is specified using the -nnmtrap port command-line argument to the SNMPv3 SPI Server. Substitute “port” with the actual port number to be used.

4 Verify that the file brassagt.lrf is installed correctly.

On UNIX systems, this file is located in the directory /etc/opt/OV/share/lrf.

```
# cd etc/opt/OV/share/lrf

# ls -l

-r--r--r- 1bin   sys    388Aug 16    1996brassagt.lrf

#
```

On Microsoft Windows systems, this file is located in the directory C:\Program Files\HP OpenView\lrf.

```
C:\>cd "Program Files\HP OpenView\lrf"

C:\Program Files\HP OpenView\lrf>dir brassagt.lrf

Volume in drive C has no label.

Directory of C:\Program Files\HP OpenView\lrf

01/02/02        02:45p             2,667brassagt.lrf

C:\Program Files\HP OpenView\lrf>
```

If the file is not found in the correct directory, copy it to that location. Then run ovaddobj to register brassagt.lrf.

On UNIX Systems

```
# /opt/OV/bin/ovaddobj
/etc/opt/OV/share/lrf/brassagt.lrf
```

On Microsoft Windows Systems

```
C:\Program Files\HP OpenView\bin\ovaddobj

C:\Program Files\HP OpenView\lrf\brassagt.lrf
```

5 View the brassd.lrf file and ensure that the correct command-line arguments will be used when the SNMPv3 SPI Server is invoked. On Microsoft Windows systems, the "-nnm" argument is required.

brassagt:brassd:

OVs_YES_START::-nnm :OVs_NON_WELL_BEHAVED:15:

On UNIX systems, the "-d" and "-nnm" arguments are both required.

brassd:/opt/SecurityPack/bin/brassagt:brassd:
OVs_YES_START::-d -nnm :OVs_NON_WELL_BEHAVED:15:

If any other command-line arguments are desired for the SNMPv3 SPI Server, edit the brassd.lrf file and add the additional arguments between the second and third colons of the second line.

4.1.1.1 Special Instructions for Traps on Microsoft Windows Systems

Network Node Manager receives Traps through through Microsoft's SNMP Trap Service, so these additional steps are required.

1 Edit the C:\WINDOWS\system32\drivers\etc\services file and change port number for traps. Find this line in the file:
snmp-trap 162/udp

Change it like this:
snmp-trap 4748/udp

2 Restart the SNMP Trap Service:

a Open the Control Panel window and double-click on the Services icon.

b Find "SNMP Trap Service" in the list and select it by clicking on it once.

c If the Status is "Started", then stop the service by clicking on the Stop button.

d Start the service again by clicking on the S tart button.

3 Restart ovtrapd:

a Open a "DOS" window by double-clicking on the Command Prompt icon.

b Type the following commands.
C:\>**ovstop ovtrapd**
C:\>**ovstart ovtrapd**

4.2 Running the SNMPv3 SPI Server

To start the SNMPv3 SPI Server on all systems with HP OpenView, use the ovstart program with the brassagt argument.

ovstart brassagt

To stop the SNMPv3 SPI Server, use the ovstop program with the brassagt argument. Failure to specify the brassagt argument will halt all of the components of Network Node Manager, not just the SNMPv3 SPI Server.

To enable network management through •rewalls, refer to the NNM Secure Polling Agent User Guide.

4.3 Setting Access Parameters for BRASS Clients

It may be desirable to restrict access made by BRASS™Clients (brassagt) to the SNMPv3 SPI Server. This section describes the various ways that access to the SNMPv3 SPI Server can be restricted.

4.3.1 On All Supported Operating Systems

By default, the SNMPv3 SPI Server listens for connection requests on TCP ports 6842 and 6843, and it accepts connection requests that originate from BRASS™ Clients running on the same host as the SNMPv3 SPI Server. These defaults can be changed before the SNMPv3 SPI Server is started using the following environment variables.

BRASS_ASYNC_TCP_ADDR

If defined, the value of this variable indicates a single network address from which asynchronous communication requests from BRASS Clients will be accepted (requests sent from other addresses will be ignored). If not defined, the SNMPv3 SPI Server will accept asynchronous communication requests from localhost (127.0.0.1) only. Define this variable with 0.0.0.0 to allow asynchronous connections from any host.

BRASS_ASYNC_TCP_PORT

If defined, the value of this variable indicates the TCP port which the SNMPv3 SPI Server uses to listen for asynchronous communication requests from BRASS Clients (requests sent to other ports will not be received). If not defined, the SNMPv3 SPI Server will accept asynchronous communication requests on port 6842.

BRASS_SYNC_TCP_ADDR

If defined, the value of this variable indicates a single network address from which synchronous communication requests from BRASS Clients will be accepted (requests sent from other addresses will be rejected). If not defined, the SNMPv3 SPI Server will accept synchronous communication requests from localhost (127.0.0.1)only. Define this variable with 0.0.0.0 to allow synchronous connections from any host.

BRASS_SYNC_TCP_PORT

If defined, the value of this variable indicates the TCP port which the SNMPv3 SPI Server uses to listen for synchronous communication requests from BRASS Clients (requests sent to other ports will not be received). If not defined, the SNMPv3 SPI Server will accept synchronous communication requests on port 6843.

Changing the port numbers can prevent an unauthorized BRASS Client from connecting to the SNMPv3 SPI Server, but only as long as the user of the BRASS™Management Application does not know the new port number. Changing the network address defaults can prevent a BRASS Client on an unauthorized system from connecting to the SNMPv3 SPI Server, but only one authorized system may be specified; i.e., the local host (127.0.0.1) or one other network address.

4.3.2 On UNIX Systems

On UNIX Systems, in addition to TCP, BRASS Clients can communicate with the SNMPv3 SPI Server using UNIX Domain Sockets. The names of the associated special files are .BRASS SYNC and .BRASS, and these files are located in /etc/srconf/mgr (or the directory specified by the SR_MGR_CONF_DIR environment variable). The permission modes of these special files allow the SNMPv3 SPI Server to restrict the access of BRASS Clients on the local system.

The following command-line arguments to brassd set the permission bits for the UNIX Domain Socket special files used by BRASS.

- -access u
  Allows only BRASS Clients to connect which have been started by the same user who started the SNMPv3 SPI Server.
- -access g
  Allows only BRASS Clients to connect which have been started by a different user in the same group as the user who started the SNMPv3 SPI Server.
- -access o
  Allows only BRASS Clients to connect which have been started by users who are not in the same group as the user who started the SNMPv3 SPI Server.

Note that access categories may be used in combinations. For example, -access ug allows BRASS Clients to connect which have been started by the same user who started the SNMPv3 SPI Server or by a user in the same group as the user who started the SNMPv3 SPI Server.

4.4 Running the SNMPv3 SPI Server With the EMANATE Master Agent

The SNMPv3 SPI Server can connect to an EMANATE® Master Agent as a Subagent to enable remote configuration. The SNMPv3 SPI Server contains agent method routines that allow the local configuration datastore containing SNMPv3 security information to be managed remotely. That is, Set requests can be used to modify the security information in RAM and the configuration on disk.

To enable the Subagent feature of the SNMPv3 SPI Server, execute brassd with the
-subagent option. This will cause the SNMPv3 SPI Server to connect to an EMANATE® Master Agent. On systems where no Master Agent is installed, this may slow the performance of the SNMPv3 SPI Server dramatically, because the SNMPv3 SPI Server will periodically attempt to establish a connection. To cause the SNMPv3 SPI Server to connect to a Master Agent on a different machine, specify that machine's address using the IP address with the tcpaddr <IP address> option.

4.5 Running the SNMPv3 SPI Server in Debug Mode

At times, it may be necessary to run the SNMPv3 SPI Server in debug mode to diagnose a problem running the program. To execute the SNMPv3 SPI Server in debug mode, run the program with one or more of the command line arguments described in this section. The command line arguments tell the SNMPv3 SPI Server which kinds of messages to generate. There are two general kinds of messages which can be generated.

1 SNMP Library Messages

These messages originate in the Hewlett-Packard OpenView SNMP libraries and report on such activities as program initialization, reading configuration files, parsing SNMP packets, and so on. There are three types of library messages: error, warning, and tracing (informational). User messages are also included in these library messages: They are user-friendly versions of the Warning and Error messages. By default, all library messages are printed to standard error.

- Warning messages indicate an unexpected, but non-fatal, condition.
- Error messages indicate an unexpected condition that can seriously affect the operation of the program.
- User messages are application specific (user-level) log messages.

2 BRASS™ Tracing Messages

These messages report on the specific activities of the SNMPv3 SPI Server and provide such information as packet dumps. By default, all BRASS™ tracing messages are printed to standard output. Tracing messages can be directed into a file by specifying the -tracefile filename command line argument[6]. Note that when tracing messages are turned on in the SNMPv3 SPI Server, the default behavior of all connected BRASS™ Clients (and all of their sessions) will be to generate tracing messages. The following categories of trace messages can be enabled and disabled independently in most Hewlett-Packard OpenView applications. Uncategorized trace messages may also be enabled and disabled independently of categorized trace messages. The categories of trace messages are as follows:

- Configuration messages are log messages printed while reading lines from a configuration file.

[6] BRASS™ tracing messages are appended to the file filename.

- Packet messages are log messages printed while parsing or constructing a SNMP packet.
- Trap messages are log messages printed while constructing a SNMP trap or inform notification.
- Access messages are log messages printed while a SNMP request is being processed by the access control service.
- EMANATE® messages are log messages printed while executing sections of program code specific to EMANATE.
- Verbose messages are additional log messages printed throughout the program.

The AP debugging facility generates diagnostics messages for general application programming interface issues related to use of the Hewlett-Packard OpenView libraries. These options are position dependent. Thus, if you wanted only master agent message diagnostics, the command-line option -apnone followed by -apemanate would disable warning and error messages, then only turn on packet messages. The following switches alter the SNMP library message logging.

Warning and error messages (-apwarn, -aperror) are already turned on.

- -apaccess
  Log agent processing messages.
- -apall
  Log all library messages (error, warning, and trace messages).
- -apconfig
  Log configuration file i/o messages.
- -apemanate
  Log master/subagent messages.
- -aperror
  Generate error messages if the Subagent determines that any errors occur.
- -appacket
  Log SNMP packet build/parse messages.
- -aptrap
  Log trap and inform messages.

- -aptrace
  Log library tracing messages (activities such as writing a configuration file).
- -apuser
  Log messages generated by user-modified code.
- -apverbose
  Log verbose debug messages.
- -apwarn
  Generate warning messages if anything unusual happens in the Subagent.

The ER diagnostics are used to display various kinds of information related to SNMP packets being handled by BRASS on behalf of a management application. Unless otherwise specified, no message-level diagnostics are issued.

- -ertrace
  Issue trace messages.
- -ererror
  Issue error messages.
- -ervarbinds
  Display the contents of all variable binding lists.
- -erpackdump
  Display a hex packet dump of all SNMP messages.
- -erpdus
  Display the protocol data unit header contents.
- -erauths
  Display request and response security (authentication and privacy) wrappers.
- -erreports
  Display report PDUs.
- -erall
  Display all SNMP traffic.

4.6 Using Nonstandard Ports

RFC1157 specifies that SNMP entities should receive Trap messages on UDP port 162. The default port used by the SNMPv3 SPI Server to listen for Traps can be configured, however. It may be desirable to run a SNMP entity on a nonstandard port for a few reasons:

- To test a BRASS™ application before putting it into actual service
- To use a BRASS™ application in a customized network environment

The following sections describe how to change the default setting.

4.6.1 Environment Variable

The first place the SNMPv3 SPI Server will look to find out which port to use for Traps is an environment variable. The environment variable that controls the default port is called SR_TRAP_TEST_PORT. If defined, the value of this variable indicates the default UDP port number to be used to listen for all SNMP Trap messages.

4.6.2 The Services File

The /etc/services file on a system defines the network services provided by the local network host. This file should contain an entry defining which port is to be used for SNMP Traps, usually:

snmp-trap 162/udp

If the appropriate environment variable is not defined, the SNMPv3 SPI Server will look at this file to determine which port to use as the default for SNMP Traps.

To select a nonstandard port for SNMP Traps, change 162 in /etc/services to another number. The number selected will be the default port number used by the SNMPv3 SPI Server. Any number below 65536 may be used, but it is important that the port number selected be unique for all UDP entries in /etc/services. For reasons beyond the scope of this manual, it is “safer” to choose a number between 5000 and 9999.

On Microsoft Windows XP and Windows Server 2003 systems, the name of the “/etc/services” file is actually C:\WINDOWS\system32\drivers\etc\services.

On Microsoft Windows 2000 systems, the name of the “/etc/services” file is actually C:\WINNT\system32\drivers\etc\services.

4.6.3 Running the SNMPv3 SPI Server with Notification Throttling

The SNMPv3 SPI Server automatically filters incoming notifications from an agent if the agent generates more than the predefined number of notifications per second (10 is the default). The number of notifications per second is the stormrate. The SNMPv3 SPI Server throttles the agent by dropping incoming notifications from the agent for 300 seconds, the default, after incoming notification exceed the stormrate. The number of seconds to drop notifications is the stormtime.

The following arguments control the SNMPv3 SPI Server notification throttling:

- -notrap throttle
  Disable notification throttling. Enabled by default.
- -stormrate num
  Set the stormrate, the rate at which the SNMPv3 SPI Server filters incoming notifications if one agent generates notifications exceeding the predefined number of notifications per second. The default is 10.
- -stormtime num sec
  Set the stormtime, the length of time for the SNMPv3 SPI Server to drop notifications from the agent exceeding the stormrate. The default is 300.

4.7 Other SNMPv3 SPI Server Command Line Arguments

This section describes other command-line options for the SNMPv3 SPI Server (the brassd program).

- -help
  Print a brief listing of all supported switches.

- -c64
  Return Counter64 types to SNMPv1-only managers. If the manager is not capable of handling Counter64 objects, packets containing them are dropped, unless this option is specified. This is used when the management application's SNMPv1 implementation has been modified to accept Counter64 objects in response packets.
- -d
  Do not daemonize. On hosts that support the fork()operation, prevent BRASS from
  forking and releasing control of the terminal session. This switch is ignored on non-Unix-like platforms.
- -listen <port>
  Bind to port port, listening there for SNMP requests from the management application. This overrides the snmpv2s or sr-unm service port specification from the services file or the NIS services map. This switch may only be specified once in a command line.
- -nnm
  Act as a SNMPv3 proxy for Network Node Manager or any other SNMPv1 or SNMPv2c-only manager.
- -nnmtrap <port>
  Forward incoming notifications to port port as SNMPv2c Trap messages. In forwarded Traps, the community string will be the same as the incoming community string, and a special Network Node Manager variable binding identifying the source of the trap will be added to the variable bindings list. This argument overrides the port associated with the snmpv2s service in the services file or the NIS services map.
- -pkt size <size>
  Set the maximum packet size that can be received from a management application via the SNMP proxy (-listen)port. The default value of 2048 octets is adequate for most situations.
- -rcvsocksize <size>
  Change the size of the receive socket buffer for messages sent by BRASS Clients. If the socket's recv buffer is being filled too quickly, traps are dropped. Set the recv socket buffer size larger than the default using the "-rcvsocksize <size>" command-line option. To check the default socket buffer size on UNIX systems, use getsockopt().
- -secpack
  Enable HP OpenView NNM SPI for SNMPv3 functionality.

- -sndsocksize <size>
  Change the size of the send socket buffer for messages being sent to the BRASS Client.
- -tcpaddr <IP address>
  To cause a SNMPv3 SPI Server to connect to a Master Agent on a different machine, specify the machine's IP address. The default is 127.0.0.1.
- -sha
  Make the default authentication protocol SHA-1, not MD5. Ordinarily the authentication passwords are assumed to use MD5. By default, if "MD5^" or "SHA^" is not given in the authKey specifier of the community string[7], MD5 is assumed.

This option may not be available in HP OpenView NNM SPI for SNMPv3 releases outside of the United States.

- -subagent
  Attempt to connect to an EMANATE® Master Agent (only available with EMANATE®).
- -tcpaddr <IP address>
  Cause the SNMPv3 SPI Server to connect to a Master Agent on the machine located at the IP address specified. The default is 127.0.0.1, the "localhost."
- -trapport <port>
  Resend traps to "port." The agent address is prepended to the community string. SNMPv3 traps are rewritten as SNMPv2c traps. This option uses the context string to identify the trap. In forwarded Traps, the community string will be prefixed by the IP address of the notification originator. This argument overrides the port associated with the sr-unm service in the services file or the NIS services map.
- -trapsto <port>
  Resend traps to "port." This uses the snmpSecurityPackNotificationAddress object
  defined in the HP OpenView NNM SPI for SNMPv3 MIB to convey originating agent
  information, and is the preferred approach. All SNMPv3 traps are rewritten as SNMPv2c
  traps. If your management application can accept SNMPv2c packets, use

[7] Refer to Section 3.4.

this switch to get traps. This option uses snmpSecurityPackNotificationAddress to identify the trap source.

- -unm
  Act as a SNMPv3 proxy for any SNMPv1 or SNMPv2c-only manager. This switch currently has the same functionality as -nnm.
- -v1traps <port>
  Resend traps as v1 traps to “port.” The agent address is prepended to the community
  string. SNMPv3 and SNMPv2c traps are rewritten as SNMPv1 traps. In forwarded Traps, the community string will be prefixed by the IP address of the agent address (the notification originator address). This argument overrides the port associated with the sr-unm service in the services file or the NIS services map. This option uses the context string to identify the trap.
- -v1trapsto <port>
  Resend traps to “port.” This uses the snmpSecurityPackNotificationAddress object defined in the HP OpenView NNM SPI for SNMPv3 MIB to convey originating agent information, and is the preferred approach. All SNMPv3 traps and SNMPv2c traps are rewritten to SNMPv1 traps. If your management application can only accept SNMPv1 packets, use this switch to get traps. This option uses snmpSecurityPackNotificationAddress to identify the trap source.
- -v2errors
  Return SNMPv2c and SNMPv3 errors and exceptions (unaltered)to SNMPv1-only managers. Ordinarily, these errors and exceptions are mapped to SNMPv1 packets using the rules of the coexistence document (RFC3584).
- -wbufnum <number>
  Change the maximum number of buffers that can be used for messages going to a BRASS Client. This prevents BRASS messages from being dropped when a BRASS Client cannot accept messages as fast as a SNMPv3 SPI Server is sending them. The default number of buffers is 512. Usually, a value of 2048 is more than adequate.

4.8 Options for Integrating HP OpenView NNM SPI for SNMPv3 with Secure Polling Agent

- -du -dropunconnected
  SNMP packets destined for a Secure Proxy Agent will be dropped if the destination forwarder is not connected. Otherwise, BRASS™ will emit requests ordinarily destined for the Secure Proxy Agent locally.
- -remoteconnect <IP address>:<port>
  Attempt to establish a connection to a Secure Proxy Agent located at the specified <IP address>. By default, the Secure Proxy Agent listens at port 6844. The <port> only needs to be specified when overriding the default port.
- -remoteaccept <IP address>:<port>
  Listen for and accept a connection from a Secure Proxy Agent located at the specified <IP address>. The <port> specifies the port at which the SNMPv3 SPI Server will listen. If the port is not specified, the default port (6844) is assumed.
- -remoterange <IP address>/<masklength>
  Specify the range of addresses for which management requests will be forwarded. This switch applies to the previous -remoteconnect or -remoteaccept specification. If there was no previous -remoteconnect or -remoteaccept specification, then a diagnostic is printed. More than one -remoterange may be specified. For example, to forward all traffic for 10.1.1.0 and 10.2.5.5to 192.168.2.5, you would add the following options to the brassd command line:
  - -remoteconnect 192.168.2.5
  - -remoterange 10.1.1.0/24
  - -remoterange 10.2.5.5

  When specifying the IP address, all four octets must be speci•ed, due to a limitation in the IP address parsing code native to many systems.
- -secure
  This switch specifies to open a SSL connection to the host specified using the previous
  -remoteconnect or -remoteaccept option. If there have been no -remoteconnect or
  -remoteaccept specifications, then all HP OpenView Secure Polling Agent

connections will be opened using SSL over TCP rather than just TCP. If security is requested, the data stream is encrypted, and mutual cryptographic authentication is enforced. This option is ignored if either the SNMPv3 SPI Server or the Secure Proxy Agent does not request a secure connection.

- -certdir <directory>
  The directory where the certificates required for mutual authentication are stored. By default, this is /etc/srconf/mgr on UNIX Systems, and C:\etc\srconf\mgr on Microsoft Windows Systems. The SNMPv3 SPI Server (brassd or brassagt.exe) must have a certificate authority certificate (snmpricacert.pem) and the server certificate (SNMPv3 SPImastercert.pem) in the certificate directory. This option is ignored if a secure connection is not requested by either the SNMPv3 SPI Server or SNMPv3 SPI Secure Proxy Agent.

- -privpassword <pw>
  This option is only used when a non-Hewlett-Packard OpenView-supplied certificate is used. The private keys within the application certificates are encrypted to avoid accidental disclosure. The SNMPv3 SPI Server knows the passwords for the privacy keys within Hewlett-Packard OpenView-supplied certificates; if you replace SNMPv3 SPImastercert.pem, use this option to specify the encryption key used when you created the certificate.

5 The EMANATE Master Agent

This chapter provides instructions about how to run the EMANATE® Master Agent (hereafter referred to simply as the “agent”) that is customized for the Hewlett-Packard Corporation. This chapter also discusses how to change certain default behaviors when running the agent.

5.1 Configuring the Agent

This section describes how to configure the agent. Configuration is a required prerequisite to running the agent in a new environment (on a new machine, in a new subnet, etc.). To simplify the task of initial product testing, a sample configuration file is provided. This configuration file should be replaced with a custom configuration file before the agent is made fully operational on a new system.

A SNMP agent is an entity which has a SNMP engine, a command responder application, and a notification originator application. This EMANATE agent supports objects from MIB-II. Configuration of this entity, therefore, consists of the following steps:

1. Providing default values for MIB-II system group variables (Chapter 11).
2. Specifying that authentication-failure Traps should or should not be generated (Chapter 11).
3. Defining security access rights for SNMP manager entities. This procedure includes specifying which SNMP manager entities should receive Trap messages and other notifications (see below in Section 5.1.1).
4. Identifying Static Subagents (if any) to be loaded by the EMANATE® Master Agent (Chapter 8).
5. Setting performance parameters for the EMANATE® Master Agent (Chapter 8).

5.1.1 Normal Mode of Configuration

The normal way for the agent to obtain its configuration information is to read the `snmpd.cnf` configuration file at startup. Any time that the configuration information in the agent changes (for example, as the result of a SNMP Set request to a SNMPv3 Administration MIB object) the agent will save its configuration information by re-writing the snmpd.cnf configuration file.

Configuration for the SNMP protocol in the agent is described in Chapter 8.

5.1.2 Backwards-Compatible Configuration

The snmpd.conf configuration file[1] was designed to configure the pre-EMANATE® SNMP agent supplied by Hewlett-Packard. For the sake of backwards-compatibility, the EMANATE® Master Agent reads[2] the snmpd.conf configuration file in addition to its normal configuration file, snmpd.cnf.

For details about backwards-compatible configuration, refer to Chapter 11 (Section 11.4) and to Chapter 10.

5.2 Starting the Agent

The procedure for starting the agent is different for the different operating systems. To start the agent, follow the steps described in the following sections.

5.2.1 On UNIX Systems

5.2.1.1 Preparing to Run the Agent

Before running the agent, perform the following steps.

---

[1] The snmpd.conf file is located in the directory /etc/SnmpAgent.d/. The snmpd.cnf file is located by default in the /etc/srconf/agt directory.

[2] The EMANATE Master Agent never writes to the `snmpd.conf` configuration file.

1. Become root. The agent must be started by a superuser.
2. Put the bin directory containing the executable programs in the shell's execution path (section 2.7.4).
3. Ensure that the SNMP ports are defined in the /etc/`services file` (section 2.7.3).
4. Kill[3] any existing SNMP agent.

5.2.1.2 Running the Agent with the Hewlett-Packard Script

Hewlett-Packard provides a Korn Shell (ksh) script that emulates the pre-EMANATE ® SNMP agent. This script, called snmpd, starts both the EMANATE® Master Agent and all Subagents that have been configured in /sbin/SnmpAgtStart.d,

To run the agent with the snmpd script, type the name of the script, optionally followed by command-line arguments. Note that many of the command-line arguments described later in this chapter will cause the snmpd script to exit without performing any action. The snmpd script will accept only the command-line arguments known to the pre-EMANATE® SNMP agent from Hewlett-Packard.

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

5.2.1.3 Running the Agent as a Daemon

To run the agent as a system daemon, perform the following steps.

1. Type the name of the agent followed by the name of any desired command-line arguments (described later in this chapter). Note that some arguments may cause the agent to run in the foreground and not as a daemon.

   ```
   # snmpdm -tcplocal
   ```

   After a few seconds, the program banner will be printed. Also, some informational messages may be printed. If the agent runs successfully, the command prompt will not return.
2. Verify that the agent is running. It is possible that the agent failed silently.

---

[3] On some versions of UNIX, the argument to ps may be -guax or aux instead of -ef. Also, the argument to kill may be -KILL instead of -9.

```
# ps -ef | grep "snmp"
root 24825      0.0    0.32296 ?    S      00:07  0:00 snmpdm
```

If the agent is not running, check for the presence of a file called `snmpd.log` (in the /tmp directory on most UNIX systems).

5.2.1.4 Running the Agent as a Foreground Process

To run the agent as a foreground process, type the name of the agent followed by "-d" and any other desired command-line arguments (described later in this chapter).

```
# snmpdm -d -tcplocal
```

After a few seconds, the program banner will be printed. Also, some informational messages may be printed. If the agent runs successfully, the command prompt will not return. While the agent is running, it will receive signals which are issued at the keyboard, for example, to stop the running agent, type Control-C.

5.2.1.5 Running the Agent as a Background Process

To run the agent as a background process, type the name of the agent followed by "-d", any other desired command-line arguments (described later in this chapter), and an ampersand (&) character[4].

```
# snmpdm -d &
```

After a few seconds, the process ID will appear, the program banner will be printed, and the command prompt will return.

At any time after the agent is started successfully, informational messages may be printed to the terminal screen (interrupting anything else that's going on). If the agent does not run successfully, a message may be printed indicating that the agent has exited.

[4] This works in both the Bourne Shell (sh) and in the C Shell (csh). For other shells, refer to the appropriate man page.

5.2.2 On Microsoft Windows XP or Windows Server 2003 Systems

This section describes how to start the agent on Microsoft Windows XP or Windows Server 2003 systems.

5.2.2.1 Preparing to Run the Agent

Before running the agent, perform the following steps.

1. Log in as Administrator. The agent must be started by the administrator.
2. Open a DOS window.
3. Put the w32.bin directory containing the executable programs in the execution path (section 2.7.4).
4. Ensure that the SNMP ports are defined in the Services file (section 2.7.3).

5.2.2.2 Running the Agent as a Service

To run the agent as a system service, perform the following steps.

1. If the Hewlett-Packard OpenView agent has never been installed as a service, type the name of the agent followed by -install:

   C:\>**snmpdm -install**

   To uninstall the agent as a service, type the name of the agent followed by -remove.

2. To start the agent, type the name of the agent followed by -start:
   C:\>**snmpdm -start**

   To stop the running agent, type the name of the agent followed by -stop.

5.2.2.3 Running the Agent as a DOS Program

To run the agent in a DOS window, type the name of the agent followed by -d and any other desired command-line arguments (described later in this chapter).

C:\>**snmpdm -d**

After a few seconds, the program banner will be printed. Also, some informational messages may be printed. If the agent runs successfully, the

command prompt will not return. While the agent is running, it will receive signals which are issued at the keyboard, for example, to stop the running agent, type Control-C.

5.2.3 On Microsoft Windows 2000 Systems

This section describes how to start the agent on Microsoft Windows 2000 systems.

5.2.3.1 Preparing to Run the Agent

Before running the agent, perform the following steps.

1. Log in as Administrator. The agent must be started by the administrator.
2. Open a DOS window
3. Put the w32.bin directory containing the executable programs in the execution path (section 2.7.4).
4. Ensure that the SNMP ports are defined in the Services file (section 2.7.3).
5. Stop any existing SNMP Agent process.

5.2.3.2 Running the Agent as a "DOS" Program

To run the agent in a "DOS" window, type the name of the agent followed by "-d" and any other desired command-line arguments (described later in this chapter).

```
C:\>snmpdm -d
```

After a few seconds, the program banner will be printed. Also, some informational messages may be printed. If the agent runs successfully, the command prompt will not return. While the agent is running, it will receive signals which are issued at the keyboard, for example, to stop the running agent, type Control-C.

5.3 Using the Message Logging Facility

5.3.1 Introduction to Log Messages

There are several types of messages that can be generated by the running program. The “Log Level” indicates which types of messages will be generated. The types of messages that can be printed are:

Warning messages: log messages printed when an unexpected but non-fatal condition is encountered.

Error messages: log messages printed when an unexpected condition is experienced that can seriously affect the operation of the program.

Trace messages: informational log messages printed during various “interesting” points of activity. These messages can be very useful for troubleshooting, but they are undesirable for normal operations since a significant amount of system resources are consumed to generate them.

Configuration messages: log messages printed while reading lines from a configuration
file.

Packet messages: log messages printed while parsing or constructing a SNMP packet.

Trap messages: log messages printed while constructing a SNMP Trap or Inform notification.

Access messages: log messages printed while a SNMP request is being processed by the access control service.

EMANATE® messages: log messages printed while executing sections of program code specific to EMANATE®.

Verbose messages: extraneous log messages printed throughout the program. User messages: application specific (user-level) log messages.

5.3.2 Changing the Log Level

By default, the agent will generate warning messages and error messages. The default Log Level can be changed, however, by specifying one or more of the following command-line arguments:

- -apnone
  Turns off all log messages.
- -aperror
  Turns on error messages. This is the default.

  Note that this argument has no effect on the default Log Level. However, if -apnone has already been used on the command line to turn off the default messages, then -aperror can be used to turn the error messages back on.
- -apwarn
  Turns on warning messages. This is the default.

  Note that this argument has no effect on the default Log Level. However, if -apnone has already been used on the command line to turn off the default messages, then -apwarn can be used to turn the warning messages back on.
- -aptrace
  Turns on uncategorized trace messages.

  Prior to Release 15.1, this argument produced the side effect of causing the agent to run the foreground. Beginning with Release 15.1, this side effect is disabled to allow the Master Agent to generate trace messages internally while running as a daemon or system service.
- -apconfig
  Turns on trace messages for configuration processing.
- -appacket
  Turns on trace messages for packet processing.
- -aptrap
  Turns on trace messages for Trap (and Inform)processing.
- -aptimer
  Turns on trace messages for timer messages.
- -apthread
  Turns on trace messages for thread messages.
- -apaccess
  Turns on message logging for agent processing messages.
- -apaudit
  Turns on trace messages for audit SET processing.
- -apemanate
  Turns on trace messages for EMANATE®-specific processing.

- -apverbose
  Turns on extraneous trace messages.
- -apuser
  Turns on user messages.
- -apall
  Turns on all types of trace messages plus error, warning, and user messages. Prior to Release 15.1, this argument produced the side effect of causing the agent to run the foreground. Beginning with Release 15.1, this side effect is disabled to allow the Master Agent to generate trace messages internally while running as a daemon or system service.

5.3.3 Changing the Destination of Log Messages

By default, the agent runs as a daemon (or system service) and will write the generated warning messages and error messages into a file called snmpd.log. Also by default, any messages that are generated by the Master Agent are available to Subagents at their request. Beginning with Release 15.1, the destination(s) of log messages can be explicitly controlled with command line arguments.

- -log closefile
  Sends the trace messages to the log file, opening and closing the file for each trace message. This allows the user to back up the file during runtime.
- -log stderr
  Allow log messages to go to standard error.
- -log nostderr
  Do not allow log messages to go to standard error.
- -log stdout
  Allow log messages to go to standard output.
- -log nostdout
  Do not allow log messages to go to standard output.
- -log file
  Allow log messages to go to the snmpd.log file. This is the default.

To prevent disk space from being rapidly consumed by trace messages, this argument only allows warning messages and error messages to go to the snmpd.log •le. To force the agent to put trace messages into the snmpd.log •le, use the -log trace•le argument

together with -log •le.

- -log nofile
  Do not allow log messages to go to the snmpd.log file.
- -log tracefile
  Allow all enabled types of trace messages to go to the snmpd.log file.
- -log notracefile
  Do not allow trace messages to go to the snmpd.log file. This is the default.
- -log append
  Do not overwrite the contents of the snmpd.log file. Append log messages to the existing snmpd.log file.
- -log noappend
  Overwrite the existing contents of the snmpd.log file. This is the default.
- -log mtos
  Allow log messages to go to Subagents.
- -log nomtos
  Do not allow log messages to go to Subagents.

5.3.3.1 Location of the Log File

The default location of the `snmpd.log` file on most UNIX operating systems is the /var/opts/OV/share/log directory. On Microsoft Windows XP and Microsoft Windows 2000, the default is %OV_LOG%.

To change the location of the `snmpd.log` file, assign the full path name of the desired directory to the environment variable SR_LOG_DIR before running the agent. On Microsoft Windows XP and Microsoft Windows 2000, the path name may begin with a drive specification, and the subdirectory delimiters must be forward slashes instead of backslashes; for example, C:/WINDOWS/MYDIR.

5.3.4 Changing the Format of Log Messages

There are two output formats which the agent can use to print log messages. The original format, which has been used by Hewlett-Packard OpenView agents for many years, looks like this:

```
ProgramName: something just happened at line 999 in file myfile.c
```

A new format available since Release 15.1 is shown below. The timestamp value shows amount of time which has passed since the Master Agent was started (in 100ths of a second)[6].

```
LogLevel:     APTRACE
timestamp:    1720439434
progname:     The program name can be any string
filename:     foo.c
linenum:      28
message:      Hello, world.
UserData:     0x0
```

By default, the agent will print log messages using the older format. The following command line arguments can be used to explicitly select the log message format.

- -log format 0
  Print log messages using the original (older) format.
- -log format 1
  Print log messages using the newer format.

5.3.5 Submission of Log Messages by Subagents

An EMANATE® Subagent can send its log messages to the Master Agent for handling. An EMANATE® Master Agent can receive log messages from Subagents and will process the Subagent messages the same way it processes its own messages.

The command line arguments to the Master Agent can be used to explicitly enable and disable the processing of Subagent log messages.

- -log stom
  Process any log messages received from Subagents in the same way as log messages generated locally.
- -log nostom
  Discard any log messages received from Subagents.

[6] The timestamp value of 1720439434 indicates that the agent has been running continuously for just over 199 days.

5.4 Using Nonstandard Ports

RFC1157 specifies that SNMP entities should receive all messages except traps on UDP port 161, and traps should be received on UDP port 162. Hewlett-Packard OpenView entities can be configured to communicate on ports other than these, however. It may be desirable to run an SNMP entity on a nonstandard port for several reasons:

- To test an agent before putting it into actual service
- To use an agent in a customized network environment
- To configure an agent for a proxy relationship

Hewlett-Packard OpenView entities will use ports 161 and 162 by default. The following sections describe how to change the default settings.

5.4.1 Environment Variables

The first place a Hewlett-Packard OpenView entity will look to find out which ports to use is the environment variables. The following environment variables can be defined to set or change which ports will be used:

SR_SNMP_TEST_PORT

If defined, the value of this variable indicates the UDP port number to be used for all SNMP communication other than traps[7].

This environment variable also changes the way the EMANATE® Master Agent
communicates with EMANATE® Subagents. For example, on most UNIX Systems, this variable modifies the name of the UNIX Domain Socket created by the Master Agent to receive connection requests from Loosely Coupled Subagents. As another example, on systems where the Master Agent and Subagents can communicate with TCP, this environment variable changes the TCP port number on which the Master Agent listens for connection requests from Remotely Coupled (or Loosely Coupled)Subagents.

SR_TRAP_TEST_PORT

[7] If SR_TRAP_TEST_PORT is not defined, the port number used for traps will be one greater than the value of this variable.

If defined, the value of this variable indicates the UDP port number to be used for all SNMP traps.

5.4.2 Services File

On UNIX systems, the /etc/services file defines the network services provided by the local network host. This file should contain entries defining which ports are to be used for most SNMP network traffic, usually:

```
snmp        161/udp
snmp-trap   162/udp
```

On Microsoft Windows XP and Windows Server 2003 systems, the services file is present in C:\WINDOWS\system32\drivers\etc directory.

On Microsoft Windows 2000 systems, the services file is present in C:\WINNT\system32\drivers\etc directory.

If the appropriate environment variables are not defined, Hewlett-Packard OpenView entities will look at this file to determine which ports to use for SNMP communication.

To select a nonstandard port for SNMP communication, change 161 in /etc/services to another number. The number selected will be the port number used by the agent. Any number below 65536may be used, but it is important that the port number selected be unique for all UDP entries in /etc/services. For reasons beyond the scope of this manual, it is “safer” to choose a number between 5000 and 9999.

5.5 Troubleshooting

This section describes some of the more common problems that can be encountered when running the agent. For each problem, a description of the symptoms as well as suggestive steps for correction is provided.

- “When I try to run the program, it says ‘Permission denied.’” On UNIX systems, this problem can happen to a privileged user if the mode of the executable image is set incorrectly. First, check the mode with ls:

  ```
  # ls -lg snmpdm
  -rw------ 1root  daemon  1966080Apr 30 11:02 snmpdm
  ```

If the permission bits do not contain execute privilege (an “x” as shown below), then change the permissions with the chmod command:

```
# chmod 700 snmpdm

# ls -lg snmpdm

-rwx-----  1root  daemon  1966080Apr 30 11:02 snmpdm
```

Unauthorized users should contact the local system administrator to apply for access.

- “After I start the program, nothing about it seems to work.”
  On UNIX systems, the agent prints a banner message at startup and then daemonizes to become a background process.

  ```
  %snmpdm
  Hewlett-Packard OpenView SNMP Agent Resident Module Version
  1.10.0.0 Copyright 1989-2004 Hewlett-Packard OpenView, Inc.
  %
  ```

  Afterwards, there is no direct interaction with the user at the console or terminal. When the agent does not appear to be working, usually it is because the program encountered some problem and quit. First, check to see if the program is running with the ps command (in the following example, no SNMP agent program is running):

  ```
  %ps -gaux | grep snmp
  joeuser   10696  0.1   0.7   800   584pts/5
     S 12:15:06 0:00 grep snmp%
  ```

  To determine the problem, check the snmpd.log file. If the following message appears[8], it means that the SNMP port is already in use and the agent could not continue (two SNMP agents cannot bind to the same port).

  AgentSocketCreate: bind

  at line 479 in file tcp.c

  Address already in use

  To work around this problem, either kill the SNMP agent which is currently running, or use a different UDP port.

  If the following message appears, it means that the agent does not have the privilege to bind to the SNMP port.

[8] The line number and file name are unimportant if different from what the agent program actually generates

%AgentSocketCreate: bind: Permission denied

at line 398 in file uds.c

This problem can happen to a non-privileged user if the mode of the executable image is not “setuid” correctly. First, check the mode with ls:

```
%ls -lg snmpdm  -r-s-----   1root  daemon
   1966080Apr 30 11:02 snmpdm
```

If the permission bits do not contain an ‘s’ as shown above, then only a privileged user may execute the program. To allow a non-privileged user to start the agent, the system administrator should change the permissions with the chmod command:

```
# chmod 4500 snmpdm
```

- “The program runs okay, but the configuration isn’t right.”
  If the person who created the configuration files has a good understanding of the entries in snmpd.cnf and knows how the configuration setup should make the agent behave, there are a few things to check in the configuration log messages which may cause unexpected agent behavior. So, run the agent using the -apconfig (or -apall) option.

  The following message tells the location where the agent looks for configuration files. If the directory where the agent is looking for configuration files is not the location where it should be looking, an environment variable may be set incorrectly.

  ```
  init_fnames: searching for configuration files in
  /tmp/srconf/agt from getenv("SR_AGT_CONF_DIR")
  ```

  The following message indicates that the agent does not understand a string on line 187 in the snmpd.cnf configuration file while parsing a vacmViewTreeFamilyEntry tag. The agent does a lot of name-to-OID translation information-it only has a small list of “well-known” strings. If the English form of the OBJECT IDENTIFIER is not in the list, then the agent skips the configuration entry. To work around this problem, use the numeric representation of the OBJECT IDENTIFIER in the configuration file.

  Reading config recordtype vacmViewTreeFamilyEntry at line 187
  at line 1092 in file scanfile.c

  MakeOIDFragFromDot, hash table lookup failed: rmon
  at line 132 in file oidtran.c

  MakeOIDFromDot: MakeOIDFragFromDot(rmon) failed
  at line 228 in file oidtran.c

Can't make 'rmon' into an OID
at line 385 in file scanfile.c

ProcessConfigRecord: Error, cannot parse token rmon
at line 835 in file scanfile.c

The following message indicates that the agent has found a duplicate entry in the snmpd.cnf configuration file while parsing a usmUserEntry tag. A duplicate entry for any tag is an indication of human error, typically a cut-and-paste error.

CreateTableEntry: attempt to create row that already exists in table: usmUserEntry.
at line 1229 in file scanfile.c

The following message indicates that the agent has found an incomplete entry on line 11 in the snmpd.cnf configuration file. An incomplete entry is often an indication of human error. For example, when a person changes the file to configure a new user, one of the fields might be forgotten.

ProcessConfigRecord: Error, incomplete entry at line 11
at line 776 in file scanfile.c

The agent re-writes the con•guration •le with valid con•guration entries and predetermined comments only. User-added comments are discarded. Invalid ("junk") con•guration entries are appended to a •le called snmpd.jnk, which is located in the same directory as `snmpd.cnf`. The `snmpd.jnk` •le is created by the agent if it does not already exist.

6 SNMPv3 Configuration Wizard

6.1 Introduction

The SNMPv3 Configuration Wizard is an easy-to-use graphical interface that configures the SNMPv3 administrative user, community string, security group, and notification destination configurations.

SNMPv3 Con•guration Wizard can:

- Create, modify, or delete SNMPv3 USM users;
- Create, modify, or delete SNMPv1 and SNMPv2c community strings;
- Define security groups and access views; and
- Define notification destinations.

The SNMPv3 Configuration Wizard requires that an Administrative user be configured on the system in order to perform configurations. Hewlett-Packard OpenView recommends using default configuration file that is installed with the product.

6.1.1 Launching the Wizard

The SNMPv3 Configuration Wizard can be launched as a stand-alone process or from the NNM Toolbar.

To start the SNMPv3 Configuration Wizard as a stand-alone process, type the command run_v3wizard in the HP OpenView NNM bin directory.

Generally, use the following list to identify the necessary input information, before using the wizard:

- The hostname or IP address of the device to be configured.
- The SNMPv3 username to use when accessing the node to be configured. This user must already be configured on the node and must support the use of SNMPv3 authentication (with or without privacy).
- The authentication passphrase for the SNMPv3 user accessing the node.
- The privacy passphrase for the SNMPv3 user if, required.

6.2 Operating the Wizard

6.2.1 Accessing Remote SNMP nodes

The wizard's first five steps require the user to enter the information necessary to perform configurations: Which node to configure, an SNMPv3 user already configured on that node, and a level of security to use when accessing the SNMP node.

To retrieve and modify the remote SNMP node configurations:

1 Enter the hostname or IP address of the node to be configured.

2 Enter the SNMPv3 USM username and security passphrases.

The wizard requires a SNMPv3 user to have authentication in order to access nodes. To securely configure an SNMP node, you must use a SNMPv3 user that already exists on that node and has been configured to use either the authentication or the authentication with privacy security level. You must provide passphrase(s) for authentication and privacy, depending on the security level you use.

6.2.2 The Configuration Menu

The main configuration menu appears after the wizard collects all the initial configuration information. From this configuration menu, you may choose from the following two options:

1 Configure users or community strings by selecting Configure Get and/or Set Access to the Node.

   — To configure users, select Configure SNMPv3 USM User. Then follow instructions in section 6.2.4.

   — To configure community strings, select Configure Community String. Then, follow instructions in section 6.2.3.

2 Create new or modify existing notification entries by selecting Configure Notifications (Trap/Inform). Then, follow instructions in section 6.2.5.

6.2.3 Configuring Community Strings

To configure community strings on the remote node, perform the steps outlined in the following sections for one of the following options:

1. Create a new community string by clicking Create New. Then, follow the instructions in section 6.2.3.1.
2. Modify an existing community string by selecting a configured community string and clicking Modify Existing. Then, follow the instructions in section 6.2.3.2.
3. Delete an existing community string by selecting a configured community string and clicking the **Delete Existing**. Then, follow the instructions in section 6.2.3.3.

6.2.3.1 Creating New Community Strings

To create a new community string, perform the following steps from the configuration menu:

1. Enter a name for the community string.
2. Select access restrictions for the community. You can limit the community string's access to this node by specifying the IP addresses from the community string to which the node would allow access.
3. Configure the security for this community string by choosing one of the predefined security configurations (minimum, moderate, or maximum access)or by defining a custom security configuration.
   - To customize the security configuration, select an existing security group on the node or define a new security group.
   - To define a new security group, select an existing access view or define a new view.
   - To define a new view, enter a name for the new access view. Then, select or enter OBJECT IDENTIFIERs to be included or excluded in this view.
4. Click Commit.

6.2.3.2 Modifying an Existing Community String

To modify a community string, perform the following steps from the configuration menu:

1. Select access restrictions for the community. You can limit the community string's access to this node by specifying the IP addresses from the community string to which the node would allow access.
2. Configure the security for this community string by choosing one of the predefined security configurations (minimum, moderate, or maximum access) or by defining a custom security configuration.
   - To customize the security configuration, select an existing security group on the node or define a new security group.
   - To define a new security group, select an existing access view or define a new view.
   - To define a new view, enter a name for the new access view. Then, select or enter OBJECT IDENTIFIERs to be included or excluded in the view.
3. Click Commit.

6.2.3.3 Deleting an Existing Community String

To delete a community string, confirm the delete operation by clicking Commit.

6.2.4 Configuring SNMPv3 USM Users

The SNMPv3 Configuration Wizard provides the ability to create new SNMPv3 USM user configurations and modify or delete existing SNMPv3 USM user configurations on the remote SNMP node.

To configure SNMPv3 USM users on the remote node, perform the steps outlined in the following sections for one of the following options:

1. Create a new SNMPv3 USM user by clicking Create New. Then, follow the instructions in section 6.2.4.1.
2. Modify an existing SNMPv3 USM user by selecting a configured SNMPv3 USM user and clicking Modify Existing. Then, follow the instructions in section 6.2.4.2.
3. Delete an existing SNMPv3 USM user by selecting a configured SNMPv3 USM user and clicking Delete Existing. Then, follow the instructions in section
4. Deleting an Existing SNMPv3 USM User.

6.2.4.1 Creating a new SNMPv3 USM User

To create new SNMP USM users remotely, the wizard requires that the new user be cloned from an existing user. As a result, you must provide information about an existing USM user already configured on the remote SNMP node.

To create a new SNMPv3 USM user, perform the following steps from the configuration menu:

1. Enter a new SNMPv3 USM username to be created.
2. Select the maximum security level for the new user.
3. Select an existing SNMPv3 USM to use for cloning your new user. You must provide the secret passphrases for the existing user, as well as create passphrases for the new user.
4. Configure the security for this new user by selecting one of the predefined security configurations (minimum, moderate, or maximum access) or by defining a custom security configuration.
   - To customize the security configuration, select an existing security group on the node or define a new security group.
   - To define a new security group, select an existing access view or define a new view.
   - To define a new view, enter a name for the new access view. Then, select or enter OBJECT IDENTIFIERs to be included or excluded in the view.
5. Click Commit.

6.2.4.2 Modifying an Existing SNMPv3 USM User

To modify an existing SNMPv3 USM user, perform the following steps from the configuration menu:

1. Modify the SNMPv3 USM user's passphrase(s) by entering the old and new passphrases.
2. Modify the user's security by selecting one of the predefined security configurations (minimum, moderate, or maximum access) or by defining a custom security configuration.
   - To customize the security configuration, select an existing security group on the node or define a new security group.

— To define a new security group, select an existing access view or define a new view.

— To define a new view, enter a name for the new security group. Then, select or enter object identifiers to be included or excluded in the view.

3 Click Commit.

6.2.4.3 Deleting an Existing SNMPv3 USM User

To delete a SNMPv3 USM user, confirm the delete operation by clicking Commit.

If the node is configured with more than one SNMPv3 USM user with the same selected name, then a new window will display a more detailed summary of each of the user's configurations.

6.2.5 Configuring Notification Targets and Security Parameters

The SNMPv3 Configuration Wizard can create, modify, or delete notification destinations and security parameters.

To configure notification targets on the remote node, perform the steps outlined in the following sections for one of the following options:

1 Create a new notification target by clicking Create New. Then, follow the instructions in section 6.2.5.1.

2 Modify an existing notification target by selecting a configured notification target and clicking Modify Existing. Then, follow the instructions in section 6.2.5.2.

3 Delete an existing notification target by selecting a configured notification target and clicking Delete Existing. Then, follow the instructions in section 6.2.5.3.

6.2.5.1 Creating New Notification Targets

To create a new notification target, perform the following steps from the configuration menu:

1 Select Configure Notifications (Trap/Inform).

2 Select **Create New**.

3 Enter the new destination's hostname and notification port number.

4 Select the SNMP protocol (SNMPv1, SNMPv2c, or SNMPv3) that should be used to send the notifications to the target.

5 Select the type of notifications to send to the target (trap or inform).

— If you selected the SNMPv1 protocol, either define a new community string or choose one from the list.

a If you selected the SNMPv2c protocol, perform the following steps:

(a) Select the notification type (trap or inform).

(b) Define a new community string or choose one from the list.

b If you selected the SNMPv3 protocol, perform the following steps:

(a) Select the notification type (trap or inform).

(b) Enter the SNMP Engine ID of the notification receiver if it is not detected (if the notification receiver is not running).

(c) Select the security level to be used to send notifications to this target.

(d) Select or define a SNMPv3 USM user already configured on the node.

(e) Define a new SNMPv3 USM user or select a user already configured on the node.

6 Click **Commit**.

6.2.5.2 Modifying an Existing Notification Target

To modify a notification target, perform the following steps from the configuration menu:

1 Select **Con• gure Noti• cations (Trap/Inform)**.

2 Select an existing notification target from the list.

3 Click **Modify Existing**.

4 Select the type of notifications to send to the target (trap or inform).

— For the SNMPv1 protocol, either define a new community string or choose one from the list.

— For the SNMPv2c protocol, perform the following steps:
(a) Select the notification type (trap or inform).
(b) Define a new community string or choose one from the list.

— For the SNMPv3 protocol, perform the following steps:
(a) Select the notification type (trap or inform).
(b) Enter the SNMP Engine ID of the notification receiver if it is not detected (if the notification receiver is not running).
(c) Select the security level to be used to send notifications to this target.
(d) Select or define a SNMPv3 USM user already configured on the node.
(e) Define a new SNMPv3 USM user or select a user already configured on the node.

5 Click Commit.

6.2.5.3 Deleting an Existing Notification Target

To delete a notification target, perform the following steps from the configuration menu:

1 Select **Con• gure Noti• cations (Trap/Inform).**

2 Select an existing notification target from the list.

3 Click **Delete Existing**.

4 Click **Commit**.

6.3 Errors

The wizard can produce two types of errors: one for configuration information and one for configuration processes. The first type of error can occur while the operator is entering configuration information; these are due to an invalid value or incorrect formatting. The second type of error can occur after the configuration information has been submitted; these errors are produced because an operation fails. If an error occurs, the wizard presents a pop-up window containing an error name and brief description. Refer to the help screens for more information about each error message.

7 Configuring the SNMPv3 SPI Server

The SNMPv3 SPI Server reads information from two configuration files, mgr.cnf and snmpinfo.dat.

The `mgr.cnf` configuration file contains SNMPv3 user information that is critical for receiving secure Trap and Inform messages, and it can be used as a source of convenience for sending Get and Set requests. The SNMPv3 SPI Server may write to this file if its local configuration datastore in RAM memory is updated at run time.

The `snmpinfo.dat` configuration file contains information for translating OBJECT IDENTIFIERs from name to number form and back. This feature is important for management applications that interact directly with humans. The information in the main snmpinfo.dat file can be supplemented at run time with additional translation information from other <base>info.dat files.

The following sections describe each of these configuration files in detail.

7.1 The mgr.cnf Configuration File

7.1.1 The Purpose of this Configuration File

BRASS™Client applications that communicate with devices that support SNMPv3 need to utilize the credentials of SNMPv3 users.

Interactive applications can obtain a user's information by prompting the user for his or her user name and passwords. However, as a convenience to users, it may be desirable to pre-configure the user's password so repeated prompting is unnecessary.

There are many types of management applications that run as a background process and so do not have an human interface. For example, polling applications that send SNMPv3 Get requests may not be able to prompt for a user name and password. It would be ludicrous for an application that receives SNMPv3 Trap or Inform messages to prompt for user credentials, because the application has no way to know in advance what user information will be required. Such an application would have to summon a

user to the console any time a possibly authentic notification message was received.

For the reasons described above, the SNMPv3 SPI Server supports the SNMPv3 Administration Framework so user names and passwords can be configured in the local configuration datastore. This following sections describe how to configure the SNMPv3 SPI Server to contain this SNMPv3 information.

7.1.2 Configuration File Format

Each line of the configuration file has the format TAG VALUE where TAG is a keyword and VALUE is a valid configuration value. Entries may be continued across multiple lines by using a backslash (\). White space (tabs, spaces, line-feeds/carriage-returns) and blank lines in the file are ignored. Values which are strings containing white space must be delimited with quotation marks (").

7.1.3 Configuring SNMPv3 Users

Configuration for at least one SNMPv3 user must be provided for an SNMP engine to send or receive SNMPv3 messages on behalf of certain SNMP applications. Table 7.1 lists the SNMP applications which require this configuration. For each application, the file is named where user configuration information is expected.

Table 7.1 SNMP applications and corresponding configuration files

SNMP Application	Configuration File
Command Generator	`mgr.cnf`
Command Responder	`snmpd.cnf`
Notification Originator	`snmpd.cnf`
Notification Receiver	`mgr.cnf`
Proxy Forwarder	`snmpd.cnf`

To configure an SNMPv3 user, add a line to the appropriate configuration file with the usmUserEntry TAG. The format of the VALUE clause is:

usmUserEngineID usmUserName usmUserAuthProtocol usmUserPrivProtocol \ usmUserStorageType usmTargetTag AuthKey PrivKey

where:

- usmUserEngineID is an OctetString which is the authoritative SNMP engine's
  administratively unique identifier. For a detailed explanation of snmpEngineID, refer to Section 7.1.4.

  For Get, GetNext, GetBulk, and Set requests, the SNMP entity containing the command responder application is authoritative. Therefore, the value of the
  usmUserEngineID field of the usmUserEntry in the agent's configuration file will be localSnmpID. The value of the usmUserEngineID field of the usmUserEntry in the manager's configuration file will be value of the agent's snmpEngineID.0 object, specified as hexadecimal numbers (00-FF) and delimited with colons.

  For Trap messages, the SNMP entity containing the notification generator application is authoritative. Therefore, the value of the usmUserEngineID field of the usmUserEntry in the agent's configuration file will be localSnmpID. The value of the usmUserEngineID field of the usmUserEntry in the manager's configuration file will be value of the agent's snmpEngineID.0 object, specified as hexadecimal numbers (00-FF)and delimited with colons.

  For Inform messages, the SNMP entity containing the notification receiver application is authoritative. Therefore, the value of the usmUserEngineID field of the usmUserEntry in the agent's configuration file will be the manager's snmpEngineID, specified as hexadecimal numbers (00-FF) and delimited with colons. The value of the usmUserEngineID field of the usmUserEntry in the manager's configuration file will be localSnmpID.

- usmUserName is a human readable string representing the name of the user. This is the user-based security model dependent security ID.

- usmUserAuthProtocol is an OBJECT IDENTIFIER that indicates whether messages sent on behalf of this user to or from the SNMP engine identified by usmUserEngineID can be authenticated, and if so, the type of authentication protocol which is used. The value of

  usmUserAuthProtocol can be usmNoAuthProtocol,

  usmHMACMD5AuthProtocol, or usmHMACSHAAuthProtocol.

- usmUserPrivProtocol is an OBJECT IDENTIFIER that indicates whether messages sent on behalf of this user to or from the SNMP engine identified by usmUserEngineID can be protected from disclosure, and if so, the type of privacy protocol which is used. The value of

usmUserPrivProtocol can be usmNoPrivProtocol , usmDESPrivProtocol , usm3DESPrivProtocol, usmAES128CfbPrivProtocol,

usmAES192CfbPrivProtocol, usmAES256CfbPrivProtocol, or some other value (contact HP Support for other usmUserPrivProtocol values).

- usmUserStorageType is “nonVolatile”, “permanent”, or “readOnly”.
- usmTargetTag is a human readable string that is used to select a set of entries in the snmpTargetAddrTable for source address checking. If the SNMP entity does not have a command responder application, or if the SNMP entity should not perform source address checking, then this field should contain a dash (-).
- AuthKey is an OctetString represented as a sequence of hexadecimal numbers separated by colons. Each octet is within the range 0x00 through 0xff. If usmUserAuthProtocol is usmNoAuthProtocol, then this user does not have an AuthKey, and this field should contain a dash (-).

  This field can also be set to a human readable string representing the user’s authentication password; the password will be converted to a key at run time.
- PrivKey is an OctetString represented as a sequence of hexadecimal numbers separated by colons. Each octet is within the range 0x00 through 0xff. If usmUserPrivProtocol is usmNoPrivProtocol, then this user does not have a PrivKey, and this field should contain a dash (-).

  This field can also be set to a human readable string representing the user’s privacy password; the password will be converted to a key at run time.

7.1.4 Breakdown of a snmpEngineID

An snmpEngineID is a globally unique identifier for an SNMP entity. All SNMPv3 entities must possess a snmpEngineID. In general, a snmpEngineID may vary in length from 5 to 32octets, though Hewlett-Packard OpenView entities automatically generate a 12-octet snmpEngineID.

The snmpEngineID of an SNMP agent can be retrieved by sending a Get request to the agent for the MIB object snmpEngineID.0:

%getone -v1 localhost public snmpEngineID.0 snmpEngineID.0 =

00 00 00 63 00 00 00 a1 c0 93 8e 81

SNMP entities created by Hewlett-Packard OpenView can automatically compute their own unique value for snmpEngineID. The snmpEngineID can also be configured manually. The choice is determined by the value of SNMP ENGINE ID SRC in the SNMP entity's configuration file.

The possible values of SNMP ENGINE ID SRC are listed in Table 7.2. These are based on the two algorithms described on pages 41-42 of RFC3411. The value SIMPLE ALG (1) causes a simple, fixed-length algorithm to be used. The simple algorithm is summarized in Table 7.3, and an example is shown in Table 7.4. The value COMPLEX ALG (2) is not yet implemented.

The snmpEngineID can be configured manually in two steps by editing the configuration file. First, set the value of SNMP ENGINE ID SRC to MANUAL (3). Next, set the value in the snmpEngineID record to the value that should be given to the snmpEngineID object.

The following example demonstrates four cases in which a snmpEngineID is created:

1. If no entry exists for either snmpEngineID or SNMP ENGINE ID SRC, then the snmpEngineID is automatically created and written into the configuration file.
2. If snmpEngineID is stored in the configuration file, and the value of SNMP ENGINE ID SRC is set to 1, then the snmpEngineID is automatically created and written into the configuration file, overwriting the previous entry in snmpEngineID.
3. If snmpEngineID is stored in the configuration file, and the value of SNMP ENGINE ID SRC is set to 2, then the snmpEngineID is automatically created and used as a local snmpEngineID. The entry will not be changed in the configuration file.
4. If snmpEngineID is stored in the configuration file, and the value of SNMP ENGINE ID SRC is set to 3, then the snmpEngineID that is stored in the configuration file will be used.

Table 7.2 Possible values of SNMP ENGINE ID SRC

Macro Name	Value	Meaning
SIMPLE ALG	1	Use algorithm 1
COMPLEX ALG	2	Use algorithm 2
MANUAL	3	Manually Configured

Table 7.3 Meanings of the individual bytes of a simple snmpEngineID

Byte Number	Meaning
1-4	the vendor's enterprises assignment
5	identifier of an enterprise-specific algorithm
6-12	data for the enterprise- specific algorithm

Table 7.4 Algorithm 'zero' used by HP OpenView to compute snmpEngineID

Macro Name	Value	Meaning
1-4	00:00:00:63	Hewlett-Packard OpenView's enterprise number, enterprises.99 (the value '63' in hexadecimal is '99' in decimal).
5	00	A Hewlett-Packard OpenView snmpEngineID algorithm identifier. Hewlett-Packard OpenView uses the value zero to identify the algorithm which uses a boolean value (agent/manager), the IP address, and the SNMP request number to form a unique value.
6	00	Is the SNMP entity a manager or an agent? In algorithm number 'zero', this byte is zero if the SNMP entity is an agent, and this byte is one if the SNMP entity is a manager.
7-8	00:al	A Hewlett-Packard OpenView snmpEngineID algorithm identifier. Hewlett-Packard OpenView

		uses the value zero to identify the algorithm which uses a boolean value (agent/manager), the IP address, and the SNMP request number to form a unique value.
9-12	c0:93:8e:81	The IP address of the SNMP entity (the example value shown in hexadecimal is 192.147.142.129in dotted-decimal format).

7.1.5 Examples

7.1.5.1 Configuring a Manager to Send Requests and Receive Traps

When an SNMP manager sends a secure SNMPv3 request to an SNMP agent, it must use security keys to prepare the packet for sending. The management application can either prompt for the user's passwords within an interactive session, or the keys can be preconfigured in the mgr.cnf configuration file.

When an SNMP manager desires to receive Trap messages from an SNMP agent, the recipient user must be known to the manager's SNMP engine. If the Trap is sent in a secure packet, the manager must use the user's security keys to authenticate (and possibly decrypt)the message. For this operation, the keys must be preconfigured in the mgr.cnf configuration file.

For each of the following examples, the IP address of the agent is 192.147.142.129. Note that the snmpEngineID for the agent is used, because the receiving SNMP engine is authoritative for the security of Get, GetNext, GetBulk, and Set requests, and the sending SNMP engine is authoritative for SNMP Trap messages.

Configuration for Authentication and Privacy

The following usmUserEntry configures an SNMP manager engine with information about a SNMPv3 user whose name is "myV3AuthPrivUser." This entry contains the user's authentication password and the user's privacy password (for encryption). An SNMP request may be issued to this user from

the SNMP manager using no security, authentication (no encryption), or authentication with encryption, and the application will not need to prompt for the user's passwords[1]. The SNMP manager can receive Trap messages for this user sent using no security, MD5 authentication (no encryption), or MD5 authentication with DES encryption.

```
usmUserEntry 00:00:00:63:00:00:00:a1:c0:93:8e:81 myV3AuthPrivUser \
usmHMACMD5AuthProtocol usmDESPrivProtocol nonVolatile - \
myV3UserAuthPassword \
 myV3UserPrivPassword
```

Configuration for Authentication and No Privacy

The following usmUserEntry configures an SNMP manager engine with information about a SNMPv3 user whose name is "myV3AuthNoPrivUser". This entry contains the user's authentication password. This user does not have a privacy password, so the last field contains a dash (-). An SNMP request may be issued to this user from the SNMP manager using no security or using MD5 authentication (no encryption), and the application will not need to prompt for the user's password[2]. The SNMP manager can receive Trap messages for this user sent using no security or using MD5 authentication.

```
usmUserEntry 00:00:00:63:00:00:00:a1:c0:93:8e:81 myV3AuthNoPrivUser \
usmHMACMD5AuthProtocol usmNoPrivProtocol nonVolatile - \
myV3UserAuthPassword \
```

Configuration for No Authentication

The following usmUserEntry configures an SNMP manager engine with information about a SNMPv3 user whose name is "myV3NoAuthNoPrivUser". This user does not have an authentication password or a privacy password, so the last two fields contain a dash (-). An SNMP request may be issued to this user from the SNMP manager using no security. The SNMP manager can receive Trap messages for this user sent using no security.

```
usmUserEntry 00:00:00:63:00:00:00:a1:c0:93:8e:81 myV3NoAuthNoPrivUser
\ usmNoAuthProtocol usmNoPrivProtocol nonVolatile - -
```

---

[1] The Hewlett-Packard OpenView Utilities will automatically use MD5 authentication with DES encryption, but a more sophisticated application could choose to send the request using a less secure mechanism.

[2] The Hewlett-Packard OpenView Utilities will automatically use MD5 authentication, but a more sophisticated application could choose to send the request using a less secure mechanism.

7.1.5.2 Configuring a Manager to Receive Informs

When an SNMP agent sends an SNMPv3 Inform message to an SNMP manager, the user to whom the message is being communicated must be known to the manager's SNMP engine. If the Inform is sent in a secure packet, the manager must use the user's security keys to authenticate (and possibly decrypt) the message. For this operation, the keys must be preconfigured in the mgr.cnf configuration file.

For each the following examples, the snmpEngineID for the manager is used (localSnmpID), because the SNMP engine receiving the message is authoritative for the security of SNMP Inform messages.

Configuration for Authentication and Privacy

The following usmUserEntry configures an SNMP manager engine with information about a SNMPv3 user whose name is "myV3AuthPrivUser". This entry contains the user's authentication password and the user's privacy password (for encryption). An SNMP Inform message from another SNMP entity for this user can be received if the message was sent using no security, MD5 authentication (no encryption), or MD5 authentication with DES encryption.

```
usmUserEntry localSnmpID myV3AuthPrivUser \
        usmHMACMD5AuthProtocol usmDESPrivProtocol nonVolatile - \
        myV3UserAuthPassword \
        myV3UserPrivPassword
```

Configuration for Authentication and No Privacy

The following usmUserEntry configures an SNMP manager engine with information about a SNMPv3 user whose name is "myV3AuthNoPrivUser". This entry contains the user's authentication password. This user does not have a privacy password, so the last field contains a dash (-). An SNMP Inform message from another SNMP entity for this user can be received if the message was sent using no security or using MD5 authentication (no encryption).

```
usmUserEntry localSnmpID myV3AuthNoPrivUser \
        usmHMACMD5AuthProtocol usmNoPrivProtocol nonVolatile - \
        myV3UserAuthPassword \
```

Configuration for No Authentication

The following usmUserEntry configures an SNMP manager engine with information about an SNMPv3 user whose name is "myV3NoAuthNoPrivUser". This user does not have an

authentication password or a privacy password, so the last two fields contain a dash (-). An SNMP Inform message from another SNMP entity for this user can be received if the message was sent using no security.

```
usmUserEntry localSnmpID myV3NoAuthNoPrivUser usmNoAuthProtocol \
        usmNoPrivProtocol nonVolatile - -
```

7.1.5.3 Configuring an Agent to Receive Requests and Send Traps

This section describes how to con•gure SNMPv3 user information only. Additional con•guration is required for an SNMP agent to actually receive SNMP requests and send SNMP Traps3. Refer to Section 10.4.

When a SNMP agent receives a SNMPv3 request from a SNMP manager, the user sending the message must be known to the agent's SNMP engine. If the request is sent in a secure packet, the agent must use the user's security keys to authenticate (and possibly decrypt) the message. For this operation, the keys must be preconfigured in the snmpd.cnf configuration file.

When an SNMP agent sends a SNMPv3 Trap to a SNMP manager, the recipient user must be known to the agent's SNMP engine. If the Trap is sent in a secure packet, the agent must use the user's security keys to compute an authentication digest for (and possibly encrypt) the message. For this operation, the keys must be preconfigured in the snmpd.cnf configuration file.

For each of the following examples, the snmpEngineID for the agent is used (localSnmpID), because the receiving SNMP engine is authoritative for the security of SNMP request messages, and the sending SNMP engine is authoritative for the security of SNMP Trap messages.

Configuration for Authentication and Privacy

The following usmUserEntry configures an SNMP agent engine with information about a SNMPv3 user whose name is "myV3AuthPrivUser". This entry contains the user's authentication password and the user's privacy password (for encryption). An SNMP request message from this user (originating from another SNMP entity) can be received if the message was sent using no security, MD5 authentication (no encryption), or MD5 authentication with DES encryption. The SNMP agent can send Trap messages to this user using no security, MD5 authentication (no encryption), or MD5 authentication with DES encryption.

```
usmUserEntry localSnmpID myV3AuthPrivUser \
        usmHMACMD5AuthProtocol usmDESPrivProtocol nonVolatile \
```

whereValidRequestsOriginate \
myV3UserAuthPassword \
myV3UserPrivPassword

Configuration for Authentication and No Privacy

The following usmUserEntry configures an SNMP agent engine with information about a SNMPv3 user whose name is "myV3AuthNoPrivUser". This entry contains the user's authentication password. This user does not have a privacy password, so the last field contains a dash (-). An SNMP request message from this user (originating from another SNMP entity) can be received if the message was sent using no security or using MD5 authentication (no encryption). The SNMP agent can send Trap messages to this user using no security or using MD5 authentication (no encryption).

```
usmUserEntry localSnmpID myV3AuthNoPrivUser \
        usmHMACMD5AuthProtocol usmNoPrivProtocol nonVolatile \
        whereValidRequestsOriginate \
    myV3UserAuthPassword \
```

Configuration for No Authentication

The following usmUserEntry configures an SNMP agent engine with information about a SNMPv3 user whose name is "myV3NoAuthNoPrivUser". This user does not have an authentication password or a privacy password, so the last two fields contain a dash (-). An SNMP request message from this user (originating from another SNMP entity) can be received if the message was sent using no security. The SNMP agent can send Trap messages to this user using no security.

```
usmUserEntry localSnmpID myV3NoAuthNoPrivUser \
        usmNoAuthProtocol usmNoPrivProtocol nonVolatile \
        whereValidRequestsOriginate -
```

7.1.5.4 Configuring an Agent to Send Informs

This section describes how to Con•gure SNMPv3 user information only. Additional Con•guration is required for an SNMP agent to actually send SNMP Informs. Refer to Section 8.4.

When an SNMP agent sends a SNMPv3 Inform message to an SNMP manager, the user to whom the message is being communicated the message must be known to the agent's SNMP engine. If the Inform is sent in a secure packet, the agent must use the user's security keys to compute an

authentication digest for (and possibly encrypt) the message. For this operation, the keys must be preconfigured in the snmpd.cnf configuration file.

For each the following examples, the IP address of the manager is 192.147.142.35. Note that the snmpEngineID for the manager is used, because the receiving SNMP engine is authoritative for the security of SNMP Inform messages.

Configuration for Authentication and Privacy

The following usmUserEntry configures an SNMP agent engine with information about an SNMPv3 user whose name is “myV3AuthPrivUser”. This entry contains the user’s authentication password and the user’s privacy password (for encryption). The SNMP agent can send Inform messages to this user using no security, MD5 authentication (no encryption), or MD5 authentication with DES encryption.

```
usmUserEntry 00:00:00:63:00:01:00:a2:c0:93:8e:23 myV3AuthPrivUser \
    usmHMACMD5AuthProtocol usmDESPrivProtocol nonVolatile - \
    myV3UserAuthPassword \

myV3UserPrivPassword
```

Configuration for Authentication and No Privacy

The following usmUserEntry configures an SNMP agent engine with information about an SNMPv3 user whose name is “myV3AuthNoPrivUser”. This entry contains the user’s authentication password. This user does not have a privacy password, so the last field contains a dash (-). The SNMP agent can send Inform messages to this user using no security or using MD5 authentication (no encryption).

```
usmUserEntry 00:00:00:63:00:01:00:a2:c0:93:8e:23
myV3AuthNoPrivUser \
    usmHMACMD5AuthProtocol usmNoPrivProtocol nonVolatile - \
    myV3UserAuthPassword \
```

Configuration for No Authentication

The following usmUserEntry configures an SNMP agent engine with information about an SNMPv3 user whose name is “myV3NoAuthNoPrivUser”. This user does not have an authentication password or a privacy password, so the last two fields contain a dash (-). The SNMP agent can send Inform messages to this user using only no security.

```
usmUserEntry      00:00:00:63:00:01:00:a2:c0:93:8e:23
myV3NoAuthNoPrivUser \ usmNoAuthProtocol usmNoPrivProtocol
nonVolatile - -
```

7.1.6 Remote Configuration

If the SNMPv3 SPI Server is started as an EMANATE®Subagent[4], the mgr.cnf configuration file can be configured remotely through SNMPv3 Set requests. Hewlett-Packard OpenView offers several different products that automates the configuration of SNMPv3 information. The SNMPv3 Configuration Wizard provides an easy way to configure a single machine for SNMPv3. The Simple PolicyPro® application (part of the EnterPol product) can be used to easily configure SNMPv3 information in dozens, hundreds, or even thousands of machines.

To enable applications like these to configure the SNMPv3 SPI Server, at least one SNMPv3 user must be already configured. This user is called a template user, because it will be used as a pattern from which to clone, or copy other users.

Users can be configured remotely only with the same properties as the template user. If the template user is a noAuthNoPriv user, for example (meaning that it is a user with no passwords), then the only type of new user that can be automatically configured is a noAuthNoPriv user. To configure new users with varying levels of security and using different combinations of authentication and privacy protocols, multiple template users should be pre-configured.

7.2 The snmpinfo.dat Configuration File

7.2.1 Purpose of Configuration File

All MIB objects have an associated OBJECT IDENTIFIER (OID) which uniquely identifies the object in a global tree of objects. The OID can be represented by a string of "dotted decimals," but because these strings are usually very long, all objects also have a short English name form called an object descriptor. For example, the MIB object 1.3.6.1.2.1.1.1 is also known as sysDescr.

The human operator of an SNMP management application typically identifies a MIB object using the object descriptor. An SNMP Agent, however, only

---

[4] For information about starting the SNMPV3 SPI Server as an EMANATE Subagent, refer to Section 4.5.

recognizes a MIB object when it is identified by its numeric OID. Therefore, SNMP programs that interact directly with a human operator must be capable of performing name-to-OID translation.

Hewlett-Packard OpenView software which performs name-to-OID translation reads a configuration file called snmpinfo.dat. Figure 9.1 summarizes the operation of name-to-OID translation.

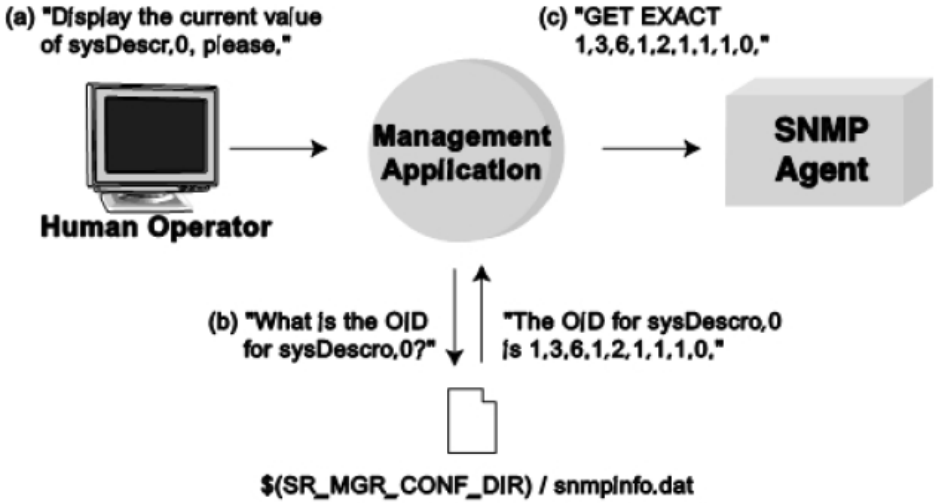


Figure 9.1: Object identifier name resolution:

(a)a human operator specifies an English name for a MIB object, (b)the management application consults snmpinfo.dat for translation information, (c) the management application sends to the SNMP agent an SNMP request containing the numeric form of the OID

7.2.2 Configuration File Format

The snmpinfo.dat file is a flat ASCII file which contains mappings of OIDs to a human-readable form. This file can be read as a data file by SNMP applications which perform OID manipulation. This file also includes the type and access for each MIB object. An example of a `snmpinfo.dat` file is shown here.

/*

* Arguments used to create this file are automatically generated by

* the mib compiler - do not edit

*/

Iso 1		nonLeaf
org	1.3	nonLeaf
dod	1.3.6	nonLeaf
internet	1.3.6.1	nonLeaf
directory	1.3.6.1.1	nonLeaf
mgmt	1.3.6.1.2	nonLeaf
mib_2	1.3.6.1.2.1	nonLeaf
experimental	1.3.6.1.4.1	nonLeaf
private	1.3.6.1.4	nonLeaf
enterprises	1.3.6.1.4.1	nonLeaf
snmpResearch	1.3.6.1.4.1.99	nonLeaf
snmpResearchMIBs	1.3.6.1.4.1.99.12	nonLeaf
srExamples	1.3.6.1.4.1.99.12.19	nonLeaf
srBasics	1.3.6.1.4.1.99.12.19.1	nonLeaf
srScalars	1.3.6.1.4.1.99.12.19.1.1	nonLeaf
srScalarGet	1.3.6.1.4.1.99.12.19.1.1.1	nonLeaf
srInteger	1.3.6.1.4.1.99.12.19.1.1.1.1	nonLeaf
thermometer	1.3.6.1.4.1.99.12.19.1.1.1.1.1	nonLeaf
thrmTemperature	1.3.6.1.4.1.99.12.19.1.1.1.1.1.1	INTEGER (-460..2147483187) read-only
thrmUnits	1.3.6.1.4.1.99.12.19.1.1.1.1.1.2	INTEGER read-only

(

1 celsius

2 fahrenheit

)

7.2.3 Creating the Configuration File

On a particular host, Hewlett-Packard OpenView management applications looks for just one snmpinfo.dat file. This “main” file is normally located in a well-known directory, but the location can be changed on some systems with an environment variable (refer to Section 7.3).

The “main” snmpinfo.dat file is built from a collection of smaller <base>info.dat files. There is generally one “smaller” file for each MIB document. The procedure to build the “main” snmpinfo.dat file is:

1 Use the MIB tools called premosy, mosy, and mgrtool to automatically generate each of the “smaller” <base>info.dat files (Figure 9.2).

Figure 9.2: Generating a <base>info.dat file from MIB documents

2 Use the MIB tool called mgrtool to merge the <base>info.dat files into the “main” snmpinfo.dat file (Figure 9.3).

Figure 9.3: Merging "smaller" <base>info.dat files with the "main" snmpinfo.dat file

7.2.4 Loading Additional MIB Information into a Running SNMPv3 SPI Server

While the SNMPv3 SPI Server is running, it may be necessary to merge new MIB information into the SNMPv3 SPI Server OBJECT IDENTIFIER translation table in RAM memory. For example, an SNMPagent containing new MIB objects might be started in the network, and a BRASS™ Client may need to begin polling it.

A program called mergemib is provided to load additional MIBs into the SNMPv3 SPI Server's memory. This program is actually a small BRASS™ Client that connects to SNMPv3 SPI Server. After it connects, the program calls a library function to merge MIB information from a specified file.

To run mergemib, type the name of the program followed by the name of a <base>info.dat
file:

%**mergemib vcrinfo.dat**

Another program called newmib is also provided. This program is very similar to mergemib, except that instead of merging the new MIBs into the SNMPv3 SPI Server's memory, newmib replaces the information in memory with the information contained in the <base>info.dat file. Under most circumstances, the user should probably use mergemib instead of newmib.

Neither mergemib nor newmib affect the contents of the "main" snmpinfo.dat. These programs only update the MIB information in the memory of a SNMPv3 SPI Server while it is running. If the SNMPv3 SPI Server is

restarted, the changes to the information in memory is lost, and the information from the snmpinfo.dat is loaded again.

7.3 Location of the Configuration Files

7.3.1 On UNIX Systems

By default, the configuration files are found in the directory /etc/srconf/mgr.

The location of the directory containing the configuration files can be changed by setting the environment variable SR_MGR_CONF_DIR. For example,

%**setenv SR_MGR_CONF_DIR /tmp/myconf**

will cause the Master Agent to look in the directory /tmp/myconf for its configuration files instead of the default location.

7.3.2 On Microsoft Windows Systems

By default, the configuration files are located in the C:\ETC\SRCONF\MGR directory.

The location of the directory containing the configuration files can be changed by setting the environment variable SR_MGR_CONF_DIR. After the files are moved. For example,

C:\> **SET SR_MGR_CONF_DIR=C:\MYCONF**

will cause the Master Agent to look in the directory C:\MYCONF on the C drive for the configuration files instead of the default location.

8 EMANATE Configuration

This chapter describes configuration information that is used only by the EMANATE® Master Agent. The information described in this section is required by the EMANATE® Master Agent in addition to other configuration information needed by all SNMP agent entities.

8.1 The snmpd.cnf Configuration File

The configuration information described in this chapter should be entered into the `snmpd.cnf` configuration file. Each line in this file has the format TAG VALUE where TAG is a keyword and VALUE is a valid configuration value. Entries may be continued across multiple lines by using a backslash (\). White space (tabs, spaces, line-feeds/carriage-returns)and blank lines in the file are ignored. Values which are strings containing white space must be delimited with quotation marks (").

8.2 Configuring EMANATE Performance Parameters

EMANATE® system parameters can be “fine tuned” for peak performance in a target environment. Entries in the snmpd.cnf file can control the way that the EMANATE® Master Agent behaves under certain conditions. The following sections describe each configuration file entry and the performance parameter it controls.

8.2.1 MAX_THREADS

The value of MAX_THREADS determines the maximum number of threads permitted to execute in the Master Agent. Since a thread is created for each PDU, this is equivalent to the maximum number of SNMP requests which can be processed asynchronously in the Master Agent. So, a larger value for MAX_THREADS means that more requests can potentially be serviced in a shorter period of time. A larger value also means that more memory will be consumed to process additional requests in parallel.

When choosing a value for MAX_THREADS, remember that EMANATE® Subagents are NOT asynchronous, and each Subagent can process only one PDU at a time. So, two requests can only be processed in parallel if the objects to be retrieved are in different Subagents. For example, if two Get requests, A and B, are processed by the Master Agent in parallel, and the objects in both PDU A and PDU B are supported in a single Subagent, Xagt, then resources will be used in the Master Agent for two requests, but Xagt will have to completely process request A before processing request B. As a second example, if request A can be serviced by Xagt and request B can be serviced by a second Subagent, Yagt, then Xagt can process request A at the same time that Yagt processes request B.

Conclusions which can be drawn from the two examples given above are as follows[1]:

1. If there is a small number of Subagents each containing a large number of MIB variables, a smaller value for MAX_THREADS will probably optimize performance.
2. If there is a large number of Subagents each containing only a few MIB variables, a larger value for MAX_THREADS will probably optimize performance.

   The default value for MAX_THREADS is 10.

8.2.2 MAX_PDU_TIME

The value of MAX_PDU_TIME determines the maximum length of time the Master Agent will wait for a PDU to be processed. This is equivalent to the amount of time that the Master Agent will wait for a response from a Subagent before the Subagent "times out."

The default value for MAX_PDU_TIME is 2500 centiseconds.

8.2.3 MAX_OUTPUT_WAITING

The value of MAX_OUTPUT_WAITING determines the maximum amount of serialized EMANATE®communication data which will be allowed to buffer

[1] Note that the examples and conclusions presented here should be correct for any combination of Subagent types.

before the communication stream is considered to “overflow.” An overflow condition can occur if:

- The sender is writing data to the EMANATE® stream faster than the receiver can read the data from stream; or
- The receiving process is paused or halted (crashed); or
- A single EMANATE® message to be sent is larger than the buffer.

The last scenario can happen if a Subagent attempts to register a large number of MIB objects in a single registration event. This scenario can be averted even with a smaller value for MAX_OUTPUT_WAITING if large MIBs are registered piecemeal in several events.

The default value for MAX_OUTPUT_WAITING is 65536 bytes.

8.2.4 MAX_SUBAGENTS

The value of MAX_SUBAGENTS determines the maximum number of Subagents which can be connected to the Master Agent at a given time. This maximum includes all types of Subagents, including Shared Library Subagents.

The default value for MAX_SUBAGENTS is 10.

8.3 Loading Static Subagents

On architectures that support shared libraries, the EMANATE® Master Agent can load a Shared Library Subagent directly into its process space at startup time. This allows the Master Agent to service requests for MIB variables supported by the Subagent. When a Shared Library Subagent is loaded by the Master Agent at startup time using the instructions provided in this section, it becomes a functional part of the Master Agent and can not be unloaded at a later point of execution.

8.3.1 On UNIX Systems

To load a Static (Shared Library) Subagent, add a line to the snmpd.cnf configuration file that instructs the Master Agent to load the Subagent. The line to add consists of the TAG subagent followed by the full pathname of the Shared Library Subagent to load. Shared Library Subagents end with the

extension .so, so to load the example Subagent, the line may look like the following:

subagent /usr/local/bin/exmp_shlib.so

NOTE: The Subagent image file must be owned by root, and the the file's permission bits must be turned off for 'group' and 'others'. For example, if the Subagent is named exmp shlib.so:

```
# ls -l exmp_shlib.so
-rw------   1root      425984Nov 18 17:49 exmp_shlib.so
```

8.3.2 On Microsoft Windows XP Systems

To load a Static (Shared Library) Subagent, add a line to the snmpd.cnf configuration file that instructs the Master Agent to load the Subagent. The line to add consists of the TAG subagent followed by the full pathname of the Shared Library Subagent to load. On Microsoft Windows XP system, Shared Library Subagents are Dynamically Linked Libraries and end with the extension .DLL, so to load the example Subagent, the line may look like the following:

subagent C:/SNMP/BIN/EXMP.DLL

The directory separators in the Subagent's path name must be forward slashes (/), not back slashes (\).

8.4 Additional Configuration for SNMPv3 Agent Entities

Certain SNMP applications (which are normally associated with an SNMP entity acting in the "agent" role) require more information in addition to the information about SNMPv3 users. Table 10.1 lists the SNMP applications which require additional configuration information. For each application, the file is named where the additional configuration information described by this section is expected.

8.4.1 Configuring View-based Access Control

Configuration of view-based access control must be provided for the SNMP engine to correctly process SNMPv1, SNMPv2c, or SNMPv3 messages. Configuring view-based access control is a process that requires three steps:

Table 8.1: SNMP applications and corresponding configuration files

SNMP Application	Configuration File
Command Responder	`snmpd.cnf`
Notification Originator	`snmpd.cnf`

1. Define a family of view subtrees.
2. Define a group and its associated access rights.
3. Assign an SNMPv3 user (or SNMPv1 community string, etc.) to the group defined in step 2.

The following sections describe each step of this process in more detail.

8.4.1.1 Defining Families of View Subtrees

To configure a view tree family, add a line to the snmpd.cnf file with the vacmViewTreeFamilyEntry TAG. The format of the VALUE clause is:

vacmViewTreeFamilyViewName vacmViewTreeFamilySubtree \
vacmViewTreeFamilyMask vacmViewTreeFamilyType \
vacmViewTreeFamilyStorageType

where:

- vacmViewTreeFamilyViewName is a human readable string representing the name of this family of view subtrees.
- vacmViewTreeFamilySubtree is an OBJECT IDENTIFIER that identifies a subtree of the MIB; e.g., enterprises.99. This value and vacmViewTreeFamilyMask are used to determine if an OBJECT IDENTIFIER is in this family of view subtrees.
- vacmViewTreeFamilyMask is an OctetString represented as a sequence of hexadecimal numbers separated by colons. Each octet is within the range 0x00 through 0xff. A zero-length OctetString is represented with a dash (-).

- vacmViewTreeFamilyType is “included” or “excluded” and indicates if the vacmViewTreeFamilySubtree is explicitly accessible or not accessible in this family of view subtrees.
- vacmViewTreeFamilyStorageType is nonVolatile, permanent, or readOnly. For example, vacmViewTreeFamilyEntry All iso - included nonVolatile defines a subtree for the view named “All” that includes the entire set of MIB objects (iso is the root node of the MIB tree).

The vacmViewTreeFamilyMask field allows restriction of the MIB view at a finer granularity than that of the vacmViewTreeFamilySubtree and vacmViewTreeFamilyType pair. For instance, a view can be restricted to one row of a table (see the example below).

The value - causes the corresponding vacmViewTreeFamilyMask to be a NULL string, which in turn allows all entries ‘below’ the vacmViewTreeFamilySubtree entry to be visible, unless cancelled by another vacmViewTreeFamilyEntry.

The vacmViewTreeFamilyMask is built using octets that correspond to the OID being restricted. For example, one may wish to restrict a user’s view of the ifTable to only the second row, all columns. The OID for ifEntry.0.2 is:

1.3.6.1.2.1.2.2.1.0.2

The vacmViewTreeFamilyMask is a series of ones and zeros used for masking out parts of the tree. A zero indicates a ‘wild card’ (i.e., matches anything), and a one indicates an exact match must be made. So:

OID	1	3	6	1	2	1	2	2	1	0	2
vacmViewTreeFamilyMask	1	1	1	1	1	1	1	1	1	0	1

would require an exact match on all fields except the table column (i.e., the 0 in ifEntry.0.2).

Using the above example, the bits of the vacmViewTreeFamilyMask would be grouped into bytes, and then the right end padded with ones if necessary to fill out the last byte:

byte1		byte 2		
1111	1111	101		original mask
1111	1111	1011	1111	padded with 1’s

ff	bf	hex value

So the vacmViewTreeFamilyMask entry would be

ff:bf

With this value for vacmViewTreeFamilyMask and all other appropriate entries in the configuration file, performing a **`getmany`** on the ifTable would return:

```
ifIndex.2 = 2

ifDescr.2 = lo0

ifType.2 = softwareLoopback(24)
ifMtu.2 = 1536

ifSpeed.2 = 0
ifPhysAddress.2 =

ifAdminStatus.2 = up(1)
ifOperStatus.2 = up(1)
ifLastChange.2 = 0
ifInUcastPkts.2 = 182945
ifInErrors.2 = 0

ifOutUcastPkts.2 = 182949
ifOutErrors.2 = 0

ifOutQLen.2 = 0

ifSpecific.2 = ccitt.0
```

Of course, the second row would have to exist before retrieving it would be successful.

8.4.1.2 Defining Groups and Access Rights

To define a group and its associated access rights, add one or more lines to the snmpd.cnf file with the vacmAccessEntry TAG. The format of the VALUE clause is:

vacmAccessEntry vacmGroupName vacmAccessContextPrefix vacmAccessSecurityModel \

vacmAccessSecurityLevel vacmAccessContextMatch \

vacmAccessReadViewName vacmAccessWriteViewName \

vacmAccessNotifyViewName vacmAccessStorageType

where:

- vacmGroupName is a human readable string which is the group name.
- vacmAccessContextPrefix is a human readable text string which is an entire or partial context name used to match the context name in (or derived from)a management request. A dash (-)represents the default context.
- vacmAccessSecurityModel is “snmpv1” for SNMPv1, “snmpv2c” for SNMPv2c, or “usm” for SNMPv3.
- vacmAccessSecurityLevel is “noAuthNoPriv” for no authentication and no privacy, “authNoPriv” is for MD5 or SHA-1 authentication with no privacy, and “authPriv” is for MD5 or SHA-1 authentication with DES, 3DES, or AES encryption.
- vacmAccessContextMatch is either “exact” or “prefix” to indicate how the context of a request must match vacmAccessContextPrefix.

  For example, if an authenticated management request is sent in context “UPS1,” and if the value of vacmAccessContextPrefix and vacmAccessContextMatch are “UPS” and “prefix,” then the context name in (or derived from)the request is determined to be a correct match to the values in this vacmAccessEntry.
- vacmAccessReadViewName is a vacmViewTreeFamilyViewName (defined by at least one vacmViewTreeFamilyEntry)identifying the view subtrees accessible for Get, GetNext, and GetBulk requests.
- vacmAccessWriteViewName is a vacmViewTreeFamilyViewName (defined by at
  least one vacmViewTreeFamilyEntry)identifying the view subtrees accessible for Set
  requests.
- vacmAccessNotifyViewName is a vacmViewTreeFamilyViewName (defined by at least one vacmViewTreeFamilyEntry)identifying the view subtrees from which objects can be included in notifications.
- vacmAccessStorageType is nonVolatile, permanent, or readOnly. If the row is configured as permanent, it may be changed via SNMP, but not deleted. If the row is configured as readOnly, it may be neither changed nor deleted via SNMP.

8.4.1.3 Assigning Principals to Groups

A “principal” is a generic term to refer to an SNMPv3 user or an SNMPv2c or SNMPv1 community string (see RFC3411). To assign a principal to a group,

add one or more lines to the snmpd.cnf file with the vacmSecurityToGroupEntry TAG. The format of the VALUE clause is:

vacmSecurityModel vacmSecurityName vacmGroupName \

vacmSecurityToGroupStorageType

where:

- vacmSecurityModel is “snmpv1” for SNMPv1, “snmpv2c” for SNMPv2c, or “usm” for SNMPv3.
- vacmSecurityName is a human readable string. This is the principal.
- vacmGroupName is a human readable string which is the group name. The group name must be defined by at least one vacmAccessEntry.
- vacmSecurityToGroupStorageType is nonVolatile, permanent, or readOnly. If the row is configured as permanent, it may be changed via SNMP, but not deleted. If the row is configured as readOnly, it may be neither changed nor deleted via SNMP.

8.4.2 Configuring Notifications

Configuration of notifications must be provided for the SNMP engine to correctly send SNMPv1 Traps, SNMPv2c Traps, SNMPv2c Informs, SNMPv3 Traps, or SNMPv3 Informs. Configuring notification is a process that requires three steps:

1. Define a notification.
2. Define a set of network addresses to which a notification should be sent. .
3. Define parameters to use when sending notifications to each of the target addressed identified in step 2.

The following sections describe each step of this process in more detail.

8.4.2.1 Defining Notifications

To configure a notification, add a line to the snmpd.cnf file with the snmpNotifyEntry TAG. The format of the VALUE clause is:

snmpNotifyName snmpNotifyTag snmpNotifyType snmpNotifyStorageType where:

- snmpNotifyName is a human readable string representing the name of this notification.

- snmpNotifyTag is a human readable string that is used to select a set of entries in the snmpTargetAddrTable.
- snmpNotifyType is “trap(1)” or “inform(2)”.
- snmpNotifyStorageType is “nonVolatile”, “permanent”, or “readOnly”.

8.4.2.2 Defining Target Addresses

To configure a target address (to which a notification should be sent), add a line to the snmpd.cnf file with the snmpTargetAddrEntry TAG. The format of the VALUE clause is:

snmpTargetAddrName snmpTargetAddrTDomain snmpTargetAddrTAddress \ snmpTargetAddrTimeout snmpTargetAddrRetryCount snmpTargetAddrTagList \ snmpTargetAddrParams snmpTargetAddrStorageType snmpTargetAddrTMask snmpTargetAddrMMS

where:

- snmpTargetAddrName is a human readable string representing the name of this target.
- snmpTargetAddrTDomain is an OID which indicates the network type (UDP/IP, IPX, etc.). RFC1906 defines several possible values[2] for this field. The dotted-decimal values and their English equivalents are listed in the table in Table 10.2. The value of this OID may be specified in dotted-decimal format or by the English name.
- snmpTargetAddrTAddress is a valid address in the snmpTargetAddrTDomain. For example, if the snmpTargetAddrTDomain is “snmpUDPDomain,” a valid address would be 192.147.142.35:0, where the value trailing the colon is the UDP port number.

  This address is used as the destination address for outgoing notifications.

If the port number is speci•ed as zero, the actual destination port used for the outgoing noti•cation message will be determined as follows. If the environment variable SR_TRAP_TEST_PORT is set, then its value becomes the destination port number. Otherwise, if the environment variable SR_SNMP_TEST_PORT is set, then one greater than its value becomes the destination port number.

---

[2] As of the printing of this document, the valid values for snmpTargetAddrTDomain are snmpUDPDomain, and transportDomainUdpIpv4.

Otherwise, the destination port number defaults to 162.

- snmpTargetAddrTimeout is an integer which identifies the expected maximum round trip time (in hundredths of seconds)for communicating with the snmpTargetAddrTAddress. When an Inform is sent to this address, and a response is not received within this time period, the SNMP entity will assume that the response will not be delivered. The default value of 1500 (15 seconds) is suggested by RFC3413. If the outgoing message type is not Inform, then this field is ignored.
- snmpTargetAddrRetryCount is an integer which identifies the number of times the SNMP entity will attempt to retransmit an Inform when a response is not received. The default value of 3 is suggested by RFC3413. If the outgoing message type is not Inform, then this field is ignored.
- snmpTargetAddrTagList is a quoted string containing one or more (space-separated)
  tags. These tags correspond to the value of snmpNotifyTag in the snmpNotifyTable. A notification defined in the snmpNotifyTable will be sent to the address specified in
  snmpTargetAddrTDomain if the notification's snmpNotifyTag appears in this list of
  tags.
- snmpTargetAddrParams is a human readable string that is used to select a set of entries in the snmpTargetParamsTable.
- snmpTargetAddrStorageType is "nonVolatile", "permanent", or "readOnly".
- snmpTargetAddrTMask is a bit field mask for the snmpTargetAddrTAddress and appears in the snmpd.cnf file in the same format as the snmpTargetAddrTAddress. For notifications, the value must be a dash ("-") to indicate that the Trap or Inform message will be sent to a specific address.

  SNMP does not allow for the broadcasting of noti•cations. However, a noti•cation may be sent to more than one speci•c address by con•guring more than one snmpTargetAddrEntry with the same tag(s) in the snmpTargetAddrTagList •eld.

- snmpTargetAddrMMS is an integer which is the maximum message size (in bytes)that can be transmitted between the local host and the host with address
  snmpTargetAddrTAddress without risk of fragmentation. The default value is 2048.

Table 8.2: English and numeric names of snmpDomains from RFC-1906.

English Names	Numeric OIDs
snmpUDPDomain	1.3.6.1.6.1.1
snmpCLNSDomain	1.3.6.1.6.1.2
snmpCONSDomain	1.3.6.1.6.1.3
snmpDDPDomain	1.3.6.1.6.1.4
snmpIPXDomain	1.3.6.1.6.1.5

8.4.2.3 Defining Target Parameters

To configure parameters to be used when sending notifications, add one or more lines to the `snmpd.cnf` file with the snmpTargetParamsEntry TAG. The format of the VALUE clause is:

snmpTargetParamsName snmpTargetParamsMPModel \
snmpTargetParamsSecurityModel snmpTargetParamsSecurityName \
snmpTargetParamsSecurityLevel snmpTargetParamsStorageType

where:

- snmpTargetParamsName is a human readable string representing the name of this parameter.
- snmpTargetParamsMPModel is 0 for SNMPv1, 1 for SNMPv2c, or 3 for SNMPv3. The value of this field together with the value of snmpTargetParamsSecurityModel indicates which type of notification should be sent.
- snmpTargetParamsSecurityModel is “snmpv1” for SNMPv1, “snmpv2c” for SNMPv2c, or “usm” for SNMPv3. The value of this field together with the value of snmpTargetParamsMPModel indicates which type of notification should be sent.
- snmpTargetParamsSecurityName is a human readable string which is the principal (an SNMPv3 or an SNMPv2c or SNMPv1 community string)to be used in the notification.
- snmpTargetParamsSecurityLevel identifies the security level of the notification to send. When an SNMPv1 or SNMPv2c notification is configured, the only valid value is “noAuthNoPriv”. When an SNMPv3 notification is configured, the value of this field is “noAuthNoPriv” for no

authentication and no privacy, “authNoPriv” for authentication without privacy, or “authPriv” for authentication with privacy.

- snmpTargetParamsStorageType is “nonVolatile”, “permanent”, or “readOnly”.

8.4.3 Configuring Notification Filters

After the SNMP entity has been properly configured to send notifications, the SNMP engine will dutifully send SNMPv1, SNMPv2c, and SNMPv3 notification messages on behalf of the notification generator application. Depending upon the nature of the specific notification generator application, this may result in the sending of few or many notifications. A well-designed notification generator application will send enough notifications to be useful to a notification receiver application, but not too many notifications that it produces “noise.”.

The SNMPv3 administration framework allows an SNMP entity which contains both a notification receiver application and a command generator application to “turn down the noise” by filtering notifications at the source. In the SNMP entity containing the notification originator, there are two MIB tables which control notification filtering: the

snmpNotifyFilterProfileTable and the snmpNotifyFilterTable. By sending SNMP Set

requests to create new rows in these tables, the SNMP entity with the notification receiver application can specify what kinds of notifications should not be sent to it.

This section describes the snmpNotifyFilterProfileTable and the snmpNotifyFilterTable in terms of the corresponding entries in the snmpd.cnf file. Using this information, some notification filters can be preconfigured before the “agent” entity is launched.

Configuring a notification filter is a process that requires two steps:

1. Create a notification filter.
2. Associate the notification filter with one or more notification parameters.

8.4.3.1 Creating a Notification Filter

To create a notification filter, add a line to the snmpd.cnf file with the snmpNotifyFilterEntry TAG. The format of the VALUE clause is:

snmpNotifyFilterProfileName snmpNotifyFilterSubtree snmpNotifyFilterMask \ snmpNotifyFilterType snmpNotifyFilterStorageType

where:

• snmpNotifyFilterProfileName is a human readable string representing the name of this notification filter.

- snmpNotifyFilterSubtree is an OID which specifies the MIB subtree containing notifications objects to be filtered. The value of this OID may be specified in dotted-decimal format or by the English name.
- snmpNotifyFilterMask modifies the set of notifications and objects identified by snmpNotifyFilterSubtree (a detailed explanation follows). This object is an OctetString represented as a sequence of hexadecimal numbers separated by colons. Each octet is within the range 0x00 through 0xff. A zero-length OctetString is represented with a dash (-).
- snmpNotifyFilterType is “included” or “excluded”. This object indicates whether the family of filter subtrees defined by this entry are included in or excluded from a filter.
- snmpNotifyFilterStorageType is “nonVolatile”, “permanent”, or “readOnly”.

The snmpNotifyFilterMask field allows filtering of MIB view at a finer granularity than that of the snmpNotifyFilterSubtree and snmpNotifyFilterType pair alone. For instance, a filter can be made to apply to one row of a table only (see the example below).

The value - causes the corresponding snmpNotifyFilterMask to be a NULL string, which in turn allows all object ‘below’ the snmpNotifyFilterSubtree entry to be filtered.

The snmpNotifyFilterMask is built using octets that correspond to the OID being filtered. For example, one may wish to restrict a filter of the ifTable to only the second row, all columns. The OID for ifEntry.0.2 is:

1.3.6.1.2.1.2.2.1.0.2

The snmpNotifyFilterMask is a series of ones and zeros used for masking out parts of the filter. A zero indicates a ‘wild card’ (i.e., matches anything), and a one indicates an exact match must be made. So:

OID	1	3	6	1	2	1	2	2	1	0	2
snmpNotifyFilterMask	1	1	1	1	1	1	1	1	1	0	1

would require an exact match on all fields except the table column (i.e., the 0 in ifEntry.0.2).

Using the above example, the bits of the snmpNotifyFilterMask would be grouped into bytes,

and then the right end padded with ones if necessary to fill out the last byte:

byte1		byte 2		
1111	1111	101		original mask
1111	1111	1011	1111	padded with 1's
ff		bf		hex value

So the snmpNotifyFilterMask entry would be

ff:bf

With this value for snmpNotifyFilterMask and all other appropriate entries in the configuration file, a notification containing values from any of the following ifTable objects would match the filter and would not be sent:

ifIndex.2
ifDescr.2
ifType.2
ifMtu.2
ifSpeed.2

ifPhysAddress.2
ifAdminStatus.2
ifOperStatus.2
ifLastChange.2
ifInUcastPkts.2
ifInErrors.2

ifOutUcastPkts.2
ifOutErrors.2
ifOutQLen.2
ifSpecific.2

8.4.3.2 Associating a Filter with a Notification Parameter

To associate a notification filter with a notification parameter, add a line to the `snmpd.cnf` file with the snmpNotifyFilterProfileEntry TAG. The format of the VALUE clause is:

snmpTargetParamsName snmpNotifyFilterProfileName \

snmpNotifyFilterProfileStorageType

where:

- snmpTargetParamsName is a snmpTargetParamsName defined in the snmpTargetParamsTable.
- snmpNotifyFilterProfileName is a snmpNotifyFilterProfileName defined in the snmpNotifyFilterTable.
- snmpNotifyFilterProfileStorageType is “nonVolatile”, “permanent”, or “readOnly”.

8.4.4 Configuring Source Address Checking

A feature of Hewlett-Packard OpenView software allows the SNMP engine to perform additional authentication of an incoming SNMPv1, SNMPv2c, or SNMPv3 message by checking the source address of the message.

To configure a source address (from which a message will be received), add a line to the snmpd.cnf file with the snmpTargetAddrEntry TAG. The format of the VALUE clause is:

snmpTargetAddrName snmpTargetAddrTDomain snmpTargetAddrTAddress \
snmpTargetAddrTimeout snmpTargetAddrRetryCount
snmpTargetAddrTagList \ snmpTargetAddrParams
snmpTargetAddrStorageType snmpTargetAddrTMask \
snmpTargetAddrMMS

where:

- snmpTargetAddrName is a human readable string representing the name of this target.

- snmpTargetAddrTDomain is an OID which indicates the network type (UDP/IP, IPX, etc.). RFC1906 defines several possible values[3] for this field. The dotted-decimal values and their English equivalents are listed in the table in Table 10.2. The value of this OID may be specified in dotted-decimal format or by the English name.
- snmpTargetAddrTAddress is a valid address in the snmpTargetAddrTDomain. For example, if the snmpTargetAddrTDomain is "snmpUDPDomain," a valid address would be 192.147.142.35:0. The value trailing the colon is the source UDP port number.[4] This address is compared to the source address of an incoming message to determine if the message should be received or rejected. The scope of this comparison is controlled by the value of snmpTargetAddrTMask.[5]
- snmpTargetAddrTimeout is an integer which must be present but is ignored by the SNMP engine. This field should be set to zero.
- snmpTargetAddrRetryCount is an integer which must be present but is ignored by the SNMP engine. This field should be set to zero.
- snmpTargetAddrTagList is a quoted string containing one or more (space-separated) tags. These tags correspond to the value of usmTargetTag in the usmUserTable and to the value of snmpCommunityTransportTag in the snmpCommunityTable. An incoming SNMPv1 or SNMPv2c message will not be rejected if:

1. the community string in the incoming message matches a configured snmpCommunityName, and
2. the snmpCommunityEntry has a snmpCommunityTransportTag with one or more corresponding tag(s)in the snmpTargetAddrTable, and

---

[3] As of the printing of this document, the valid values for snmpTargetAddrTDomain are snmpUDPDomain and transportDomainUdpIpv4.

[4] Many SNMP manager applications randomly select a port (called an ephemeral port) from which to originate an SNMP request. Therefore, the value trailing the last colon in the snmpTargetAddrTMask field should contain a zero to mask out this port number, so that it will be ignored in the comparison between snmpTargetAddrTAddress and the source address of an incoming message. Otherwise, SNMP requests sent to an agent can randomly fail.

[5] If the snmpTargetAddrTMask field is set to a dash ('-'), or if the snmpTargetAddrTMask MIB object is changed to a zero-length OCTET STRING by an SNMP Set request, the mask's port number will be treated as a zero and ignored in comparisons.

3 the source address of the incoming message is validated by snmpTargetAddrTAddress (masked by snmpTargetAddrTMask)of a corresponding snmpTargetAddrEntry.

An incoming SNMPv3 message will not be rejected if:

1 the user identified by the incoming message matches a configured usmUserName, and

2 the usmUserEntry has a usmTargetTag with one or more corresponding tag(s)in the snmpTargetAddrTable, and

3 the source address of the incoming message is validated by snmpTargetAddrTAddress (masked by snmpTargetAddrTMask)of a corresponding snmpTargetAddrEntry.

- snmpTargetAddrParams is a human readable string which must be present but is ignored by the SNMP engine. This field should be set to a dash (-).
- snmpTargetAddrStorageType is “nonVolatile”, “permanent”, or “readOnly”.
- snmpTargetAddrTMask is a bit field mask for the snmpTargetAddrTAddress and appears in the snmpd.cnf file in the same format as the snmpTargetAddrTAddress. For example, if snmpTargetAddrTDomain is “snmpUDPDomain,” a valid mask would be 255.255.255.0:0. This mask is used in conjunction with the snmpTargetAddrTAddress to determine if an incoming request has arrived from an authorized address.

The value trailing the colon should ALWAYS be zero (see footnote 5).

The value of snmpTargetAddrTMask identifies which bits of the source address should be compared to the value of snmpTargetAddrTAddress. A bit value of ‘1’ in the mask means that the corresponding bit in the source address should be compared to the corresponding bit in the value of snmpTargetAddrTAddress. A bit value of ‘0’ in the mask means that corresponding bit in the source address is a “don’t care” case in the comparison.

- snmpTargetAddrMMS is an integer which is the maximum message size (in bytes)that can be transmitted between the local host and the host with address snmpTargetAddrTAddress without risk of fragmentation. The default value is 2048.

8.4.4.1 Matching Exactly One Source Address

The following example indicates that the source address must exactly match the value of

snmpTargetAddrTAddress, or the incoming SNMP request will be rejected. If

snmpTargetAddrTMask is 255.255.255.255:0, then all bits have a value of '1':

byte 1		byte 2		byte 3		byte 4		
255		255		255		255		decimal
1111	1111	1111	1111	1111	1111	1111	1111	binary

8.4.4.2 Matching Any Source Address

The following example indicates that none of the bits of the source address will be compared to the value of snmpTargetAddrTAddress, and consequently, an incoming SNMP request will not be rejected based on its source address. If snmpTargetAddrTMask is 0.0.0.0:0, then all bits have a value of '0':

byte 1		byte 2		byte 3		byte 4		
0		0		0		0		decimal
0000	0000	0000	0000	0000	0000	0000	0000	binary

8.4.4.3 Matching a Source Address in a Subnet

If the high-order bits of snmpTargetAddrTMask are set to '1' and the low-order bits are set to '0', the mask can be used to reject an SNMP request which does not come from a particular subnet. If snmpTargetAddrTMask is 255.255.255.128:0, then only the most significant 25 bits of the source address must match the most significant 25 bits of the value of snmpTargetAddrTAddress.

byte 1	byte 2	byte 3	byte 4	
255	255	255	255	decimal

1111	1111	1111	1111	1111	1111	1000	0000	binary

Consider the case where the value of snmpTargetAddrTAddress is 192.147.142.35:

byte 1		byte 2		byte 3		byte 4		
192		147		142		35		decimal
1100	0000	1001	0011	1000	1110	0010	0011	binary

To not be rejected, the source address of an incoming SNMP request must begin with 192.147.142.In the fourth byte, only the first bit will be compared to the same bit of the value of snmpTargetAddrTAddress. The remaining bits are “don’t care” cases (shown as ‘?’ below).

byte 4		
1000	0000	snmpTargetAddrTMask (binary)
0010	0011	snmpTargetAddrTAddress (binary)
0???	????	source address of SNMP request

Therefore, to not be rejected, the source address of an incoming SNMP request must be 192.147.142.xxx where ‘xxx’ is a value between 0 (expressed as ‘00000000’ in binary) and 127 (expressed as ‘01111111’ in binary).

8.4.5 Examples

This section contains examples of SNMP configuration for SNMP agent entities. The sample SNMP request messages in this section are sent using a command-line manager utility available from Hewlett-Packard OpenView called getone. The SNMP Trap and Inform messages in this section are received using a command-line manager utility available from Hewlett-Packard OpenView called traprcv.

The examples which demonstrate an SNMP request message query for the MIB object sysDescr.0. The value of this object, like the SNMP user information, is stored in the snmpd.cnf configuration file.

8.4.5.1 For noAuthNoPriv SNMPv3 Users

1 To authorize the receipt of SNMPv3 noAuthNoPriv Get and Set[6] requests from the user “myV3NoAuthNoPrivUser” from exactly one manager station (one IP address), add the following lines to the snmpd.cnf configuration file together with the usmUserEntry for the user “myV3NoAuthNoPrivUser” as shown in Section 7.1.5.3.

vacmAccessEntry myV3NoAuthNoPrivGroup - usm noAuthNoPriv exact \ All All - nonVolatile

vacmSecurityToGroupEntry usm myV3NoAuthNoPrivUser myV3NoAuthNoPrivGroup \ nonVolatile

vacmViewTreeFamilyEntry All iso - included nonVolatile

snmpTargetAddrEntry myV3Manager_allRequests snmpUDPDomain 192.147.142.35:00 \
0
whereValidRequestsOriginate - nonVolatile 255.255.255.255:0 2048

The following is a sample query from 192.147.142.35 with a valid usmUserEntry in the mgr.cnf file for the user “myV3NoAuthNoPrivUser”. The application utilizes the
information in the configuration file to obtain the requested information without additional interaction with the user.

%getone -v3 192.147.142.129 myV3NoAuthNoPrivUser sysDescr.0
sysDescr.0 = Target SNMP Agent at 192.147.142.129

The following is a sample query from 192.147.142.35 with no usmUserEntry in the mgr.cnf file for the user “myV3NoAuthNoPrivUser”. Note that the application prompts for the user’s password, because there is no information in the configuration file to inform the application that it should send the request using the noAuthNoPriv protocol.

%getone -v3 192.147.142.129 myV3NoAuthNoPrivUser sysDescr.0
Enter Authentication password:

sysDescr.0 = Target SNMP Agent at 192.147.142.129

#snmpget -c “3N/myV3NoAuthNoPrivUser” 192.147.142.129 sysDescr.0

sysDescr.0 = Target SNMP Agent at 192.147.142.129

[6] To authorize Get requests without authorizing Set requests, the fields “All All -” in the vacmAccessEntry should be changed to “All - -”.

If 3N/myV3NoAuthNoPrivUser is configured in Get Community for Target 192.147.142.129 then snmpget command without using –c parameter should also work.

#snmpget 192.147.142.129 sysDescr.0

sysDescr.0 = Target SNMP Agent at 192.147.142.129

The following is a sample query from a host machine other than 192.147.142.35.

%getone -v3 192.147.142.129 myV3NoAuthNoPrivUser sysDescr.0
REPORT received, cannot recover:

at line 723 in file util_v3.c

usmStatsUnknownUserNames.0 = 1

To relax the agent configuration so that this user can access the MIB objects from additional hosts, change the snmpTargetAddrTMask to perform wildcard matching of the source address of the incoming request message. For more information, refer to Section 10.4.4. To relax the agent configuration so that this user can access the MIB objects from any host, change “whereValidRequestsOriginate” in the usmUserEntry to “0.0.0.0:0.”[7].

usmUserEntry localSnmpID myV3NoAuthNoPrivUser \

usmNoAuthProtocol usmNoPrivProtocol nonVolatile - -

2 To authorize the sending of SNMPv3 noAuthNoPriv Trap messages to a user at exactly one SNMP manager station (one IP address), add the following lines to the snmpd.cnf
configuration file together with the usmUserEntry for the user “myV3NoAuthNoPrivUser” as shown in Section 7.1.5.3.

vacmAccessEntry myV3NoAuthNoPrivGroup - usm noAuthNoPriv exact \ --All nonVolatile

vacmSecurityToGroupEntry usm myV3NoAuthNoPrivUser myV3NoAuthNoPrivGroup \
nonVolatile

vacmViewTreeFamilyEntry All iso - included nonVolatile
snmpNotifyEntry myTrap whereMyNotificationsGo trap nonVolatile
snmpTargetAddrEntry myV3Manager_noAuthNoPrivNotifications

[7] With this change, the snmpTargetAddrEntry is no longer referenced for this user, and it may be deleted if no other usmUserEntry entries reference it.

snmpUDPDomain \
    192.147.142.35:0 100 3 whereMyNotificationsGo
myV3NoAuthNoPrivParams \
    nonVolatile 1.2.3.4:0 2048

snmpTargetParamsEntry myV3NoAuthNoPrivParams 3 usm
myV3NoAuthNoPrivUser \ noAuthNoPriv nonVolatile

The following is a sample execution of traprcv on 192.147.142.35. There is a valid usmUserEntry in the mgr.cnf file for the user "myV3NoAuthNoPrivUser". The Trap message received indicates that the SNMP agent entity at 192.147.142.129 just booted.

```
# traprcv

Waiting for traps.

Received SNMPv3 noAuthNoPriv Trap:
From: 192.147.142.129

sysUpTime.0 = 42

snmpTrapOID.0 = coldStart
```

To configure additional Trap destinations (additional IP addresses where the user is authorized to operate a management station), add additional snmpTargetAddrEntry entries to the snmpd.cnf configuration file. For example, to authorize 192.147.142.111 as an additional Trap destination, add the following line to the snmpd.cnf configuration file.

snmpTargetAddrEntry anotherV3Manager_noAuthNoPrivNotifications
snmpUDPDomain \ 192.147.142.111:0 100 3 whereMyNotificationsGo
myV3NoAuthNoPrivParams \ nonVolatile 1.2.3.4:0 2048

3 To authorize the sending of SNMPv3 noAuthNoPriv Inform messages to a user at exactly
    one SNMP manager station (one IP address), add the following lines to the snmpd.cnf
    configuration file together with the usmUserEntry for the user "myV3NoAuthNoPrivUser" as shown in Section 9.1.5.3.

vacmAccessEntry myV3NoAuthNoPrivGroup - usm noAuthNoPriv exact
\ --All nonVolatile

vacmSecurityToGroupEntry usm myV3NoAuthNoPrivUser
myV3NoAuthNoPrivGroup \ nonVolatile

vacmViewTreeFamilyEntry All iso - included nonVolatile
snmpNotifyEntry myInform whereMyNotificationsGo inform nonVolatile
snmpTargetAddrEntry myV3Manager_noAuthNoPrivNotifications

snmpUDPDomain \ 192.147.142.35:0 100 3 whereMyNotificationsGo myV3NoAuthNoPrivParams \ nonVolatile 1.2.3.4:0 2048

snmpTargetParamsEntry myV3NoAuthNoPrivParams 3 usm myV3NoAuthNoPrivUser \ noAuthNoPriv nonVolatile

The following is a sample execution of traprcv on 192.147.142.35. There is a valid usmUserEntry in the mgr.cnf file for the user "myV3NoAuthNoPrivUser". The Inform message received indicates that the SNMP agent entity at 192.147.142.129 just booted.

```
# traprcv
Waiting for traps.
Received SNMPv3 noAuthNoPriv Inform:
From: 192.147.142.129
sysUpTime.0 = 42
snmpTrapOID.0 = coldStart
```

To configure additional Inform destinations (additional IP addresses where the user is authorized to operate a management station), add additional snmpTargetAddrEntry entries to the snmpd.cnf configuration file. For example, to authorize 192.147.142.111 as an additional Inform destination, add the following line to the snmpd.cnf configuration file.

snmpTargetAddrEntry anotherV3Manager_noAuthNoPrivNotifications snmpUDPDomain \ 192.147.142.111:0 100 3 whereMyNotificationsGo myV3NoAuthNoPrivParams \ nonVolatile 1.2.3.4:0 2048

8.4.5.2 For authNoPriv SNMPv3 Users

1 To authorize the receipt of SNMPv3 authNoPriv Get and Set[8] requests from the user
"myV3AuthNoPrivUser" from exactly one manager station (one IP address), add the
following lines to the snmpd.cnf configuration file together with the usmUserEntry for
the user "myV3AuthNoPrivUser" as shown in Section 7.1.5.3.

vacmAccessEntry myV3AuthNoPrivGroup - usm authNoPriv exact \
All All - nonVolatile

[8] To authorize Get requests without authorizing Set requests, the fields "All All -" in the vacmAccessEntry should be changed to "All - -".

vacmSecurityToGroupEntry usm myV3AuthNoPrivUser myV3AuthNoPrivGroup \
    nonVolatile

vacmViewTreeFamilyEntry All iso - included nonVolatile

snmpTargetAddrEntry myV3Manager_allRequests snmpUDPDomain 192.147.142.35:0 0 \ 0whereValidRequestsOriginate - nonVolatile 255.255.255.255:0 2048

The following is a sample query from 192.147.142.35 with a valid usmUserEntry in the mgr.cnf file for the user “myV3AuthNoPrivUser”. The application utilizes the information in the configuration file to obtain the requested information without additional interaction with the user.

%getone -v3 192.147.142.129 myV3AuthNoPrivUser sysDescr.0
sysDescr.0 = Target SNMP Agent at 192.147.142.129

The following is a sample query from 192.147.142.35 with no usmUserEntry in the mgr.cnf file for the user “myV3NoAuthNoPrivUser”. Note that the application prompts for the user’s password, because there is no information in the configuration file to inform the application that it should send the request using the noAuthNoPriv protocol.

%getone -v3 192.147.142.129 myV3AuthNoPrivUser sysDescr.0 Enter Authentication password : myV3UserAuthPassword

Enter Privacy password :

sysDescr.0 = Target SNMP Agent at 192.147.142.129

#snmpget –c “3A;MD5^MyV3UserAuthPassword/myV3AuthNoPrivUser “ 192.147.142.129 sysName.0

sysDescr.0 = Target SNMP Agent at 192.147.142.129

snmpget command will not prompt for password because the password is included as a part of –c parameter.

If 3A;MD5^MyV3UserAuthPassword/myV3AuthNoPrivUser is configured in Get Community for Target 192.147.142.129 then snmpget command without using –c parameter should also work.

The following is a sample query from a host machine other than 192.147.142.35.

%getone -v3 192.147.142.129 myV3AuthNoPrivUser sysDescr.0 REPORT received, cannot recover:

at line 723 in file util_v3.c

usmStatsUnknownUserNames.0 = 1

To relax the agent configuration so that this user can access the MIB objects from additional hosts, change the snmpTargetAddrTMask to perform wildcard matching of the source address of the incoming request message. For more information, refer to Section 8.4.4. To relax the agent configuration so that this user can access the MIB objects from any host, change “whereValidRequestsOriginate” in the usmUserEntry to “0.0.0.0:0.” [9]

2 To authorize the sending of SNMPv3 authNoPriv Trap messages to a user at exactly one
SNMP manager station (one IP address), add the following lines to the snmpd.cnf
configuration file together with the usmUserEntry for the user “myV3AuthNoPrivUser”
as shown in Section 7.1.5.3.

vacmAccessEntry myV3AuthNoPrivGroup - usm authNoPriv exact \
--All nonVolatile

vacmSecurityToGroupEntry usm myV3AuthNoPrivUser myV3AuthNoPrivGroup \
nonVolatile

vacmViewTreeFamilyEntry All iso - included nonVolatile
snmpNotifyEntry myTrap whereMyNotificationsGo trap nonVolatile
snmpTargetAddrEntry myV3Manager_authNoPrivNotifications snmpUDPDomain \
192.147.142.35:0 100 3 whereMyNotificationsGo myV3AuthNoPrivParams \
nonVolatile 1.2.3.4:0 2048

snmpTargetParamsEntry myV3AuthNoPrivParams 3 usm myV3AuthNoPrivUser \
authNoPriv nonVolatile

The following is a sample execution of traprcv on 192.147.142.35. There is a valid usmUserEntry in the mgr.cnf file for the user “myV3AuthNoPrivUser”. The Trap message received indicates that the SNMP agent entity at 192.147.142.129 just booted.

```
# traprcv
Waiting for traps.
```

[9] With this change, the snmpTargetAddrEntry is no longer referenced for this user, and it may be deleted if no other usmUserEntry entries reference it.

```
Received SNMPv3 authNoPriv Trap:
From: 192.147.142.129

sysUpTime.0 = 30

snmpTrapOID.0 = coldStart
```

To configure additional Trap destinations (additional IP addresses where the user is authorized to operate a management station), add additional snmpTargetAddrEntry entries to the snmpd.cnf configuration file. For example, to authorize 192.147.142.111 as an additional Trap destination, add the following line to the snmpd.cnf configuration file.

snmpTargetAddrEntry anotherV3Manager_authNoPrivNotifications snmpUDPDomain \
    192.147.142.111:0 100 3 whereMyNotificationsGo myV3AuthNoPrivParams \
    nonVolatile 1.2.3.4:0 2048

3 To authorize the sending of SNMPv3 authNoPriv Inform messages to a user at exactly one SNMP manager station (one IP address), add the following lines to the snmpd.cnf
    configuration file together with the usmUserEntry for the user “myV3AuthNoPrivUser”
    as shown in Section 9.1.5.3.

vacmAccessEntry myV3AuthNoPrivGroup - usm authNoPriv exact \
    --All nonVolatile

vacmSecurityToGroupEntry usm myV3AuthNoPrivUser myV3AuthNoPrivGroup \
    nonVolatile

vacmViewTreeFamilyEntry All iso - included nonVolatile
snmpNotifyEntry myInform whereMyNotificationsGo inform nonVolatile
snmpTargetAddrEntry myV3Manager_authNoPrivNotifications snmpUDPDomain \
    192.147.142.35:0 100 3 whereMyNotificationsGo myV3AuthNoPrivParams \
    nonVolatile 1.2.3.4:0 2048

snmpTargetParamsEntry myV3AuthNoPrivParams 3 usm myV3AuthNoPrivUser \
    authNoPriv nonVolatile

The following is a sample execution of traprcv on 192.147.142.35. There is a valid usmUserEntry in the mgr.cnf file for the user “myV3AuthNoPrivUser”. The Inform message received indicates that the SNMP agent entity at 192.147.142.129 just booted.

```
# traprcv
Waiting for traps.
Received SNMPv3 authNoPriv Inform:
From: 192.147.142.129
sysUpTime.0 = 30
snmpTrapOID.0 = coldStart
```

To configure additional Inform destinations (additional IP addresses where the user is authorized to operate a management station), add additional snmpTargetAddrEntry entries to the snmpd.cnf configuration file. For example, to authorize 192.147.142.111 as an additional Inform destination, add the following line to the snmpd.cnf configuration file.

```
snmpTargetAddrEntry anotherV3Manager_authNoPrivNotifications
snmpUDPDomain \
    192.147.142.111:0 100 3 whereMyNotificationsGo
myV3AuthNoPrivParams \
    nonVolatile 1.2.3.4:0 2048
```

8.4.5.3 For authPriv SNMPv3 Users

1. To authorize the receipt of SNMPv3 authPriv Get and Set [10]requests from the user "myV3AuthPrivUser" from exactly one manager station (one IP address), add the following lines to the snmpd.cnf configuration file together with the usmUserEntry for the user "myV3AuthPrivUser" as shown in Section 9.1.5.3.

   ```
   vacmAccessEntry myV3AuthPrivGroup - usm authPriv exact \
       All All - nonVolatile

   vacmSecurityToGroupEntry usm myV3AuthPrivUser
   myV3AuthPrivGroup \
       nonVolatile

   vacmViewTreeFamilyEntry All iso - included nonVolatile

   snmpTargetAddrEntry myV3Manager_allRequests snmpUDPDomain
   192.147.142.35:0 0   \
       0
   whereValidRequestsOriginate - nonVolatile 255.255.255.255:0 2048
   ```

[10] To authorize Get requests without authorizing Set requests, the fields "All All -" in the vacmAccessEntry should be changed to "All - -".

The following is a sample query from 192.147.142.35 with a valid usmUserEntry in the mgr.cnf file for the user “myV3AuthPrivUser”. The application utilizes the information in the configuration file to obtain the requested information without additional interaction with the user.

%getone -v3 192.147.142.129 myV3AuthPrivUser sysDescr.0 sysDescr.0 = Target SNMP Agent at 192.147.142.129

The following is a sample query from 192.147.142.35 with no usmUserEntry in the mgr.cnf file for the user “myV3NoAuthPrivUser”. Note that the application prompts for the user’s password, because there is no information in the configuration file to inform the application that it should send the request using the noAuthNoPriv protocol.

%getone -v3 192.147.142.129 myV3AuthPrivUser sysDescr.0 Enter Authentication password : myV3UserAuthPassword

Enter Privacy password : myV3UserPrivPassword

sysDescr.0 = Target SNMP Agent at 192.147.142.129

snmpget –c “3P;MD5^MyV3UserAuthPassword;DES^MyV3UserPrivPassword/myV3AuthPrivUser” 192.147.142.129 sysDescr.0

sysDescr.0 = Target SNMP Agent at 192.147.142.129

The above snmpget command will not prompt for authentication and privacy password because it is included as a part of –c parameter.

If 3P;MD5^MyV3UserAuthPassword;DES^MyV3UserPrivPassword/myV3AuthPrivUser is configured in Get Community for Target 192.147.142.129 then snmpget command without using –c parameter should also work.

The following is a sample query from a host machine other than 192.147.142.35.

%getone -v3 192.147.142.129 myV3AuthPrivUser sysDescr.0 REPORT received, cannot recover:

at line 723 in file util_v3.c
usmStatsUnknownUserNames.0 = 1

To relax the agent configuration so that this user can access the MIB objects from additional hosts, change the snmpTargetAddrTMask to perform wildcard matching of the source address of the incoming request

message. For more information, refer to Section 8.4.4. To relax the agent configuration so that this user can access the MIB objects from any host, change “whereValidRequestsOriginate” in the usmUserEntry to “0.0.0.0:0.”[11].

2 To authorize the sending of SNMPv3 authPriv Trap messages to a user at exactly one
SNMP manager station (one IP address), add the following lines to the snmpd.cnf
configuration file together with the usmUserEntry for the user “myV3AuthPrivUser” as
shown in Section 9.1.5.3.

vacmAccessEntry myV3AuthPrivGroup - usm authPriv exact \
--All nonVolatile

vacmSecurityToGroupEntry usm myV3AuthPrivUser myV3AuthPrivGroup \
nonVolatile

vacmViewTreeFamilyEntry All iso - included nonVolatile
snmpNotifyEntry myTrap whereMyNotificationsGo trap nonVolatile
snmpTargetAddrEntry myV3Manager_authPrivNotifications snmpUDPDomain \
192.147.142.35:0 100 3 whereMyNotificationsGo myV3AuthPrivParams \
nonVolatile 1.2.3.4:0 2048

snmpTargetParamsEntry myV3AuthPrivParams 3 usm myV3AuthPrivUser \
authPriv nonVolatile

The following is a sample execution of traprcv on 192.147.142.35. There is a valid usmUserEntry in the mgr.cnf file for the user “myV3AuthPrivUser”. The Trap message received indicates that the SNMP agent entity at 192.147.142.129 just booted.

```
# traprcv

Waiting for traps.

Received SNMPv3 authPriv Trap:
From: 192.147.142.129

sysUpTime.0 = 42
```

[11] With this change, the snmpTargetAddrEntry is no longer referenced for this user, and it may be deleted if no other usmUserEntry entries reference it.

```
snmpTrapOID.0 = coldStart
```

To configure additional Trap destinations (additional IP addresses where the user is authorized to operate a management station), add additional snmpTargetAddrEntry entries to the snmpd.cnf configuration file. For example, to authorize 192.147.142.111 as an additional Trap destination, add the following line to the snmpd.cnf configuration file.

snmpTargetAddrEntry anotherV3Manager_authPrivNotifications snmpUDPDomain \
    192.147.142.111:0 100 3 whereMyNotificationsGo myV3AuthPrivParams \
    nonVolatile 1.2.3.4:0 2048

3 To authorize the sending of SNMPv3 authPriv Inform messages to a user at exactly one
    SNMP manager station (one IP address), add the following lines to the snmpd.cnf
    configuration file together with the usmUserEntry for the user “myV3AuthPrivUser” as
    shown in Section 9.1.5.3.

vacmAccessEntry myV3AuthPrivGroup - usm authPriv exact \
    --All nonVolatile

vacmSecurityToGroupEntry usm myV3AuthPrivUser myV3AuthPrivGroup \
    nonVolatile

vacmViewTreeFamilyEntry All iso - included nonVolatile
snmpNotifyEntry myInform whereMyNotificationsGo inform nonVolatile
snmpTargetAddrEntry myV3Manager_authPrivNotifications snmpUDPDomain \
    192.147.142.35:0 100 3 whereMyNotificationsGo myV3AuthPrivParams \
    nonVolatile 1.2.3.4:0 2048

snmpTargetParamsEntry myV3AuthPrivParams 3 usm myV3AuthPrivUser \
    authPriv nonVolatile

The following is a sample execution of traprcv on 192.147.142.35. There is a valid usmUserEntry in the mgr.cnf file for the user “myV3AuthPrivUser”. The Inform message received indicates that the SNMP agent entity at 192.147.142.129 just booted.

```
# traprcv
Waiting for traps.
```

```
Received SNMPv3 authPriv Inform:
From: 192.147.142.129
sysUpTime.0 = 42
snmpTrapOID.0 = coldStart
```

To configure additional Inform destinations (additional IP addresses where the user is authorized to operate a management station), add additional snmpTargetAddrEntry entries to the snmpd.cnf configuration file. For example, to authorize 192.147.142.111 as an additional Inform destination, add the following line to the snmpd.cnf configuration file.

```
snmpTargetAddrEntry anotherV3Manager_authPrivNotifications
snmpUDPDomain \
    192.147.142.111:0 100 3 whereMyNotificationsGo
myV3AuthPrivParams \
    nonVolatile 1.2.3.4:0 2048
```

8.4.5.4 Source Address Checking

To add source address checking, add an entry to the `mgr.cnf` file that is similar to the following snmpTargetAddrEntry

```
snmpTargetAddrEntry myV3Manager snmpUDPDomain \
        192.147.142.35:0 0 0 whereValidRequestsOriginate \
        -nonVolatile 255.255.255.255:0 2048
```

8.4.5.5 Destination Address Checking

To add destination address checking, add an entry to the mgr.cnffile that is similar to the following snmpTargetAddrEntry.

```
snmpTargetAddrEntry myV3Manager snmpUDPDomain \
        192.147.142.35:0 100 3 whereMyNotificationsGo \
        myV3NoAuthNoPrivParams nonVolatile - 2048
```

8.5Additional Configuration for Bilingual and Trilingual Agent Entities

This section describes SNMP configuration which is required for SNMP entities which support SNMPv1 and/or SNMPv2c in addition to SNMPv3.

8.5.1 Configuring Communities

Configuration of at least one community string must be provided for an SNMP engine to send or receive SNMPv1 or SNMPv2c messages. To configure an SNMPv1 or SNMPv2c community, add a line to the snmpd.cnf file with the snmpCommunityEntry TAG. The format of the VALUE clause is:

snmpCommunityIndex snmpCommunityName snmpCommunitySecurityName \ snmpCommunityContextEngineID snmpCommunityContextName \ snmpCommunityTransportTag snmpCommunityStorageType

where:

- snmpCommunityIndex is a human readable string which is an arbitrary index. The value of this field is unimportant, other than it must unique from other values in this field in other snmpCommunityEntry entries.
- snmpCommunityName is the community string, which may be a human readable string or a hexadecimal representation containing unprintable characters.

  For example, if the community string was the word “public” with an unprintable ‘<bell>’ character (ASCII code 7)at the end, then the value of this field would be 70:75:62:6c:69:63:07 (the ASCII codes for ‘p,’ ‘u,’ ‘b,’ ‘l,’ ‘i,’ ‘c,’ and ‘<bell>’).
- snmpCommunitySecurityName is a human readable string which identifies the security
  name for this community string. This string should appear in at least one vacmSecurityToGroupEntry to assign the community string (principal) to an access control group.

  For example, if the community string was the word “public” with an unprintable ‘<bell>’ character (ASCII code 7)at the end, then the value of this field could be “public”, or it could be any other arbitrary string containing only printable characters.
- snmpCommunityContextEngineID is an OctetString, usually “localSnmpID”.
- snmpCommunityContextName is the SNMPv3 context implied by the community string. A dash (-)in this field represents the default context.
- snmpCommunityTransportTag is a human readable string that is used to select a set of entries in the snmpTargetAddrTable for source address checking. Entries in the snmpTargetAddrTable are selected if the value of snmpCommunityTransportTag appears in the list of (space-

separated)tags in snmpTargetAddrTagList. If the SNMP entity should not perform source address checking, then this field should contain a dash (-).

- snmpCommunityStorageType is “nonVolatile”, “permanent”, or “readOnly”.

8.5.2 Examples

1 To receive SNMPv1 requests from exactly one SNMP manager station:

```
snmpCommunityEntry      61targetV1Community targetV1Community
localSnmpID - \
whereValidRequestsOriginate nonVolatile
vacmAccessEntry myV1Group - snmpv1 noAuthNoPriv exact All All All \
    nonVolatile
vacmSecurityToGroupEntry snmpv1 targetV1Community myV1Group \
    nonVolatile
vacmViewTreeFamilyEntry All iso - included nonVolatile
snmpTargetAddrEntry myV1Manager_allRequests snmpUDPDomain
192.147.142.35:0    0          \
0 whereValidRequestsOriginate - nonVolatile 255.255.255.255:0 2048
```

2 To send SNMPv1 Trap messages to exactly one SNMP manager station:

```
snmpCommunityEntry      61targetV1Community targetV1Community
localSnmpID         \
whereValidRequestsOriginate nonVolatile
vacmAccessEntry myV1Group - snmpv1 noAuthNoPriv exact All All All \
    nonVolatile
vacmSecurityToGroupEntry snmpv1 targetV1Community myV1Group \
    nonVolatile
vacmViewTreeFamilyEntry All iso - included nonVolatile
snmpNotifyEntry myTrap whereMyNotificationsGo trap nonVolatile
snmpTargetAddrEntry myV1Manager_allNotifications snmpUDPDomain
\
    192.147.142.35:0 100 3 whereMyNotificationsGo
myV1ExampleParams \
    nonVolatile 1.2.3.4:0 2048
```

```
snmpTargetParamsEntry myV1ExampleParams 0 snmpv1
targetV1Community \
    noAuthNoPriv nonVolatile
```

3 To receive SNMPv2c requests from exactly one SNMP manager station:

```
snmpCommunityEntry      62targetV2cCommunity targetV2cCommunity
localSnmpID             \

whereValidRequestsOriginate nonVolatile

vacmAccessEntry myV2cGroup - snmpv2c noAuthNoPriv exact All All All
\
    nonVolatile

vacmSecurityToGroupEntry snmpv2c targetV2cCommunity \
    myV2cGroup nonVolatile

vacmViewTreeFamilyEntry All iso - included nonVolatile

snmpTargetAddrEntry myV2cManager_allRequests snmpUDPDomain
192.147.142.35:0    0       \

0 whereValidRequestsOriginate - nonVolatile 255.255.255.255:0 2048
```

4 To send SNMPv2c Trap messages to exactly one SNMP manager station:

```
snmpCommunityEntry      62targetV2cCommunity targetV2cCommunity
localSnmpID             \

whereValidRequestsOriginate nonVolatile

vacmAccessEntry myV2cGroup - snmpv2c noAuthNoPriv exact All All All
\
    nonVolatile

vacmSecurityToGroupEntry snmpv2c targetV2cCommunity \
    myV2cGroup nonVolatile

vacmViewTreeFamilyEntry All iso - included nonVolatile
snmpNotifyEntry myTrap whereMyNotificationsGo trap nonVolatile
snmpTargetAddrEntry myV2cManager_allNotifications
snmpUDPDomain \
    192.147.142.35:0 100 3 whereMyNotificationsGo
myV2cExampleParams \
    nonVolatile 1.2.3.4:0 2048

snmpTargetParamsEntry myV2cExampleParams 1 snmpv2c \
    targetV2cCommunity noAuthNoPriv nonVolatile0 \

0whereValidRequestsOriginate - nonVolatile 255.255.255.255:0 2048
```

5 To send SNMPv2c Inform messages to exactly one SNMP manager station:

```
snmpCommunityEntry      62targetV2cCommunity targetV2cCommunity
localSnmpID             \

whereValidRequestsOriginate nonVolatile

vacmAccessEntry myV2cGroup - snmpv2c noAuthNoPriv exact All All All
\
    nonVolatile

vacmSecurityToGroupEntry snmpv2c targetV2cCommunity \
    myV2cGroup nonVolatile

vacmViewTreeFamilyEntry All iso - included nonVolatile
snmpNotifyEntry myInform whereMyNotificationsGo inform nonVolatile
snmpTargetAddrEntry myV2cManager_allNotifications
snmpUDPDomain \
    192.147.142.35:0 100 3 whereMyNotificationsGo
myV2cExampleParams \
    nonVolatile 1.2.3.4:0 2048

snmpTargetParamsEntry myV2cExampleParams 1 snmpv2c \
    targetV2cCommunity noAuthNoPriv nonVolatile
```

9 SNMP Configuration: The Basics

This chapter describes the sample SNMP configuration provided by Hewlett-Packard OpenView. The sample SNMP configuration files demonstrate how a site security policy can be realized through SNMPv3 authentication and access control mechanisms. The configuration file described in this chapter is included, but not installed by default. The default configuration file does not include all of the configuration entries described in this chapter. This example configuration file is included in the directory v1c3-full.cnf.

9.1 Introduction

SNMPv3 security provides strong mechanisms to ensure that:

- People who should have SNMP access to a system can gain access. That is, important information contained in SNMP-manageable devices can be retrieved and modified, and notifications about alert conditions can be delivered in SNMP Trap and Inform messages.
- People who should not have SNMP access to a system can not gain access.

Before one can configure an SNMPv3 system correctly, one must understand the site security policy. A site security policy will answer all of the following questions about the system to be configured.

1. Who are the individuals that may access the information?
2. What type of access is allowed for each individual? Read access? Write access? Should the individual see alert conditions?
3. Where can each individual perform authorized access operations? That is, on which parts of MIB can the user read and write MIB objects?
4. Are the individuals gaining access across a nonsecure network? Is data source authentication, modification protection or replay protection needed?
5. Is the information that an individual is accessing sensitive? Could it be seen by an unauthorized third party on your network?

The sample SNMP configuration provided by Hewlett-Packard OpenView implements a site security policy which is described in the remainder of this chapter.

9.2 Configuration Policies

This company has defined a specific site security policy which all computers should be configured to enforce. The site security policy consists of Security Groups, Access Views, and Users and/or Communities.

9.3 User Configuration

When users are created, a entries are either added to the configuration file manually, or it is generated by one of Hewlett-Packard OpenView configuration tools such as the SNMPv3 Configuration Wizard or the snmpdcfg utility.

User

The term user refers to the SNMPv3 USM users configured on an SNMP agent. The user is used when making a SNMPv3 request to an agent that supports SNMPv3. A user is set up to support one or more of the following security levels:

- Authentication with privacy
- Authentication without privacy
- No authentication and no privacy

The username can be the name of a person, group or management application. For example:

Administrator

public

EnterPol

Security Level

SNMPv3 users can be configured to use one or more of the following security levels:

- Authentication with privacy
- Authentication without privacy
- No authentication and no privacy

For security levels that use authentication, an authentication protocol must be specified in
the configuration entry. For security levels that use privacy, a privacy protocol must also be specified in the configuration entry. Support privacy if this user might be used to
communicate sensitive information that should not be transmitted across the network as
plain text.

Authentication Passphrase

The authentication passphrase is used to calculate a unique authentication key for each SNMP agent on which the new SNMPv3 user is configured. The authentication key verifies the authenticity of the SNMPv3 messages sent by a manager using the new SNMPv3 username. A passphrase is similar to a password except that spaces are acceptable. Example: auth password for DayShiftSupervisor

The best authentication passphrase is one that can not be guessed easily. Passphrases should be at lease eight characters long.

Privacy Passphrase

The privacy passphrase is used to calculate a privacy key which is used to encrypt SNMP messages sent by this user. Encryption of the SNMP message ensures privacy as the message is transmitted across the network. A passphrase is similar to a password except that spaces are acceptable. Example:

priv password for HelpDesk

The best privacy passphrase is one that can not be guessed easily. Passphrases should be at lease eight characters long.

Community

The term community refers to the SNMPv1 or SNMPv2c configured request name. A community is used when making SNMPv1 or SNMPv2c requests to a SNMP agent.

Administrative User

In order for EnterPol to configure SNMP agents through EnterPol modules like Simple Policy Pro or CIAgent Policy Pro, the EnterPol database must contain an administrative user that is already configured on the SNMP agent. This is because configuration is applied using SNMP set requests. This administrative user must have permission to set all MIB objects that may be set during the distribution of a policy. The Administration panels allow you to discover the administrative users in the EnterPol database.

This section describes the configuration entries in the snmpd.cnf file that are specific to each user. The user is created using an entry with the tag, usmUserEntry. The user is assigned general authentication and security privileges in this entry. The following sections describe the user list their respective configuration file entries that are included in the sample configuration file.

9.3.1 NetworkAdministrator

NetworkAdministrator has complete read-write access to all of the network elements. Consequently, authentication is very important for NetworkAdministrator's configuration. If another user could convince a machine that he is NetworkAdministrator, then the other user would also have complete access to that machine. All configurations of NetworkAdministrator should include his authentication password ("auth password for NetworkAdministrator").

NetworkAdministrator's duties may sometimes require him to examine data considered sensitive and confidential to the company. Remote access to sensitive data must be encrypted-especially if the data traverses an untrusted internet-because the company could be harmed if the sensitive information were to be seen by an unauthorized person. Therefore, all configurations of NetworkAdministrator should include his privacy password ("priv password for NetworkAdministrator").

To configure the user NetworkAdministrator with the appropriate passwords, the `snmpd.cnf` file contains the following entry:

```
usmUserEntry localSnmpID NetworkAdministrator
usmHMACMD5AuthProtocol \
        usmDESPrivProtocol nonVolatile anywhereTag \
  "auth password for NetworkAdministrator" \
  "priv password for NetworkAdministrator"
```

9.3.2 DayShiftSupervisor, EveningShiftSupervisor, and NightShiftSupervisor

DayShiftSupervisor, EveningShiftSupervisor, and NightShiftSupervisor are all different users, but have the same privileges, so they are described together.

These users have read-write access to nearly all of the network elements So, like NetworkAdministrator, all configurations of DayShiftSupervisor, EveningShiftSupervisor, and NightShiftSupervisor should include their authentication password. Also, like NetworkAdministrator, these users may need to remotely access systems and applications which contain sensitive and confidential information. So, all configurations of DayShiftSupervisor should also include his privacy password.

To configure the user DayShiftSupervisor, EveningShiftSupervisor, and NightShiftSupervisor with the appropriate passwords, the snmpd.cnf file contains the following entry.

```
usmUserEntry localSnmpID DayShiftSupervisor usmHMACMD5AuthProtocol
usmDESPrivProtocol \
        nonVolatile operationsCenterTag "auth password for
DayShiftSupervisor" \
        "priv password for DayShiftSupervisor"
usmUserEntry localSnmpID NightShiftSupervisor usmHMACMD5AuthProtocol
\ usmDESPrivProtocol nonVolatile operationsCenterTag \
  "auth password for NightShiftSupervisor" \
  "priv password for NightShiftSupervisor"
```

9.3.3 HelpDesk

HelpDesk has complete read access over the information categorized under MIB-II. This includes MIB objects found in such documents as the Host Resources MIB, which allows the user to view installed and/or running software applications. Consequently, authentication is important for HelpDesk's configuration. If another user could convince a machine that he is HelpDesk, then the other user would also have the same level of access to that machine. This could be a potential security hole, since having information about installed software, its version, and at what times the software is executing could be useful to a malicious attacker with knowledge about the weaknesses of the software. All configurations of HelpDesk should include the authentication password ("password for HelpDesk").

To configure the user HelpDesk with the appropriate password, the snmpd.cnf file contains the following entry.

usmUserEntry localSnmpID HelpDesk usmHMACMD5AuthProtocol usmDESPrivProtocol \ nonVolatile HelpDeskTag "auth password for HelpDesk" "priv password for HelpDesk"

9.3.4 DayOperator, Evening Operator, and Night Operator

DayOperator, EveningOperator, and Night Operator are all different users, but belong to the same group and have the same privileges, so they are described together.

To configure the user DayOperator, Evening Operator, and Night Operator, with the appropriate password, the snmpd.cnf file contains the following entries.

usmUserEntry localSnmpID DayOperator usmHMACMD5AuthProtocol \
usmDESPrivProtocol nonVolatile operationsCenterTag\
"auth password for DayOperator" "priv password for DayOperator"
usmUserEntry localSnmpID EveningOperator usmHMACMD5AuthProtocol \ usmDESPrivProtocol nonVolatile operationsConsoleTag \
"auth password for EveningOperator" \ "priv password for EveningOperator"

usmUserEntry localSnmpID NightOperator usmHMACMD5AuthProtocol \
usmDESPrivProtocol nonVolatile operationsConsoleTag \
"auth password for NightOperator" \

"priv password for NightOperator"

9.3.5 public

A public user is useful for debugging and initial installs. This SNMPv3 user has a very limited privileges. The entry for public is as follows:

usmUserEntry localSnmpID public usmNoAuthProtocol \
usmNoPrivProtocol nonVolatile anywhereTag -

9.4 Community String Configuration

The sample configuration uses the community string “public” for public access. To configure this community string for use by SNMPv1 and SNMPv2c protocols, the snmpd.cnf file contains the following entries.

The snmpCommunityEntry creates the community string in the same manner that the user is created using the usmUserEntry.

snmpCommunityEntry t0000000 public public localSnmpID - anywhereTag \
nonVolatile

vacmSecurityToGroupEntry snmpv1 public public nonVolatile
vacmSecurityToGroupEntry snmpv2c public public nonVolatile

The following explanations for some of the fields of this snmpCommunityEntry are provided to answer common questions.

“t0000000” is an arbitrary index. However, it must be different from other values in this field in other snmpCommunityEntry entries.

“public” appears twice in the entry. The reason is that a community string in SNMPv1 can
contain unprintable ASCII characters, but the SNMPv3 “security name” representation for
the community string can not contain unprintable characters. So, the first ‘public’ is the
actual community string, and the second ‘public’ is the security name representation.
If the community string contained the unprintable ‘<bell>’ character (ASCII code 7)at the
end, then the first field in the pair would be 70:75:62:6c:69:63:07 (the ASCII codes for
‘p,’ ‘u,’ ‘b,’ ‘l,’ ‘i,’ ‘c,’ and ‘<bell>’), and the second field in the pair could still be simply
‘public.’

localSnmpID indicates that the context for SNMPv1 and SNMPv2c requests received with the ‘public’ community string is the default context. “–” The dash indicates that SNMPv1 and SNMPv2c requests received with this community string will be accepted from a sender at any location. In place of the dash, one of the authorized workstations defined in Section 11.7 could be substituted (e.g., “HelpDesk” or “operationsCenter”)to perform source address checking on any received requests with the ‘public’ community string.

The community string is a member of a group called “public,” just as NetworkAdministrator is a member of the group called “Administrator.”

Members of the ‘public group can read all parts of the MIB, but can write to no MIB objects.

```
vacmAccessEntry public          -snmpv1 noAuthNoPriv exact
ApplicationsView - \
   ApplicationsView nonvolatile
vacmAccessEntry public          -snmpv2c noAuthNoPriv exact
ApplicationsView - \
   ApplicationsView nonVolatile
```

SNMPv1 and SNMPv2c are completely unsecure protocols. The appropriate function of SNMPv1 and SNMPv2c configuration for a managed device should only be to enable public read-only access. The sample SNMP configuration supplied by Hewlett-Packard OpenView defines an narrow MIB view that exemplifies what is probably the appropriate level of public access to a system.

```
vacmViewTreeFamilyEntry ApplicationsView iso 0 included nonVolatile
vacmViewTreeFamilyEntry ApplicationsView snmpCommunityTable -
excluded \
        nonvolatile
vacmViewTreeFamilyEntry ApplicationsView vacmAccessTable - excluded \
        nonvolatile
vacmViewTreeFamilyEntry ApplicationsView vacmSecurityToGroupTable -
excluded \
        nonvolatile
vacmViewTreeFamilyEntry ApplicationsView usmUserTable - excluded \
        nonvolatile
vacmViewTreeFamilyEntry ApplicationsView vacmViewTreeFamilyTable -
excluded         \
        nonVolatile
```

9.5 Security Groups

The user is given specific security access privileges by being assigned to a Security Group. A security group assigns users to an access view. The user is assigned to the security group with an entry called vacmSecuritytoGroupEntry.

Security Group

The security group defines:

- The security model that will be supported (SNMPv1, SNMPv2c, or SNMPv3).
- The security level that will be supported (if SNMPv3 is the security model).
- The access level (or views) (read, write and notify permissions).

Users assigned to a security group take on the access limitations defined for the group. It is often useful to name a security group to indicate the access level. It is also helpful to create a security group for a specific level of access.

Examples:

Administrator
ShiftSupervisor
public

Multiple users can be assigned to the same security group; however, each user can only be assigned to one security group.

Context

Contexts are generally used when a SNMP agent has multiple subagents that support the same MIB. By making a request with a context the agent can correctly forward the request to the subagent that has registered for the context. Typically most MIB objects will be supported under the default SNMP context. A non-default context should only be specified if you know the SNMP agent being configured has MIB objects supported under a different context.

The users are assigned to the following groups.

- Administrator
  - NetworkAdministrator
  - EnterPol
- ShiftSupervisor
  - DayShiftSupervisor
  - EveningShiftSupervisor
  - NightShiftSupervisor
- HelpDesk
  - HelpDesk
- Operator
  - DayOperator

- EveningOperator
- NightOperator

- public
  - public
- Application
  - OpenView
  - Spectrum

vacmSecurityToGroupEntry usm NetworkAdministrator Administrator nonVolatile vacmSecurityToGroupEntry usm DayShiftSupervisor ShiftSupervisor nonVolatile vacmSecurityToGroupEntry usm NightShiftSupervisor ShiftSupervisor nonVolatile vacmSecurityToGroupEntry usm HelpDesk HelpDesk nonVolatile vacmSecurityToGroupEntry usm DayOperator Operator nonVolatile vacmSecurityToGroupEntry usm OpenView Applications nonVolatile vacmSecurityToGroupEntry usm Spectrum Applications nonVolatile

Security groups must be set to use an access view. The following entries name the security group and which access view the group uses.

The vacmAccessEntry table is used to associate views with each security group. Remember, that views are associated with the group a user is in, not with the user itself.

9.5.1 Administrator

A user in this group has read privilege to all branches of the MIB at any security level. A user in this group has write privilege to all branches of the MIB, but an authentication password is required to perform a write operation. Also, a user in this group can receive notifications of alert conditions at any security level.

To implement this site security policy, the snmpd.cnf file contains the following entries.

- For the default context:

  vacmAccessEntry Administrator - usm noAuthNoPriv exact publicView - publicView \
  nonVolatile
  vacmAccessEntry Administrator - usm authNoPriv exact All All All nonVolatile vacmAccessEntry Administrator - usm authPriv exact All All All nonVolatile

- For any “UPS” context:

  vacmAccessEntry Administrator UPS usm noAuthNoPriv prefix All - All nonVolatile vacmAccessEntry Administrator UPS usm authNoPriv prefix All All All nonVolatile vacmAccessEntry Administrator UPS usm authPriv prefix All All All nonVolatile

- For the MIB View:

  vacmViewTreeFamilyEntry All iso - included nonVolatile

9.5.2 ShiftSupervisor

A user in this group has read privilege to all branches of the MIB at any security level. A user in this group has write privilege to all branches of the MIB, but an authentication password is required to perform a write operation. Finally, a user in this group can receive notifications of alert conditions at any security level.

To implement this site security policy, the snmpd.cnf file contains the following entries.

- For the default context:

  vacmAccessEntry ShiftSupervisor - usm noAuthNoPriv exact publicView - \
  publicView nonVolatile ShiftSupervisorView

  vacmAccessEntry ShiftSupervisor - usm authPriv exact ShiftSupervisorView \
  ShiftSupervisorView ShiftSupervisorView nonVolatile

- For any “UPS” context:

  vacmAccessEntry ShiftSupervisor UPS usm noAuthNoPriv prefix publicView - \
  publicView nonVolatile

  vacmAccessEntry ShiftSupervisor UPS usm authNoPriv prefix ShiftSupervisorView \
  ShiftSupervisorView ShiftSupervisorView nonVolatile

  vacmAccessEntry ShiftSupervisor UPS usm authPriv prefix ShiftSupervisorView \
  ShiftSupervisorView ShiftSupervisorView nonVolatile

- For the MIB: View[1]

vacmViewTreeFamilyEntry ShiftSupervisorView iso - included nonVolatile

9.5.3 HelpDesk

A user in this group must always provide an authentication password. Also, a user in this group can only request information in the default context.

To implement this site security policy, the snmpd.cnf file contains the following entries.

vacmAccessEntry HelpDesk - usm noAuthNoPriv exact publicView - publicView \
nonVolatile

vacmAccessEntry HelpDesk - usm authNoPriv exact HelpDeskView HelpDeskView \
HelpDeskView nonVolatile

vacmAccessEntry HelpDesk - usm authPriv exact HelpDeskView HelpDeskView \
HelpDeskView nonVolatile

9.5.4 Operator

A user in this group has both read and write privileges to only those branches of the MIB which contain confidential information related to business transactions and company records. Also, a user in this group can receive notifications of alert conditions related to the business transactions and company records; e.g., alerts generated by the corporate database software. Therefore, the specific MIB branches available to a user in this group are specified by the view “publicView.” To implement this site security policy, the snmpd.cnf file contains the following entries.

vacmAccessEntry Operator - usm noAuthNoPriv exact publicView - publicView \
nonVolatile

[1] This definition of the “ShiftSupervisorView” MIB view is shared with the System Administrator and appears only once in the snmpd.cnf file.

vacmAccessEntry Operator - usm authNoPriv exact OperatorView OperatorView \
OperatorView nonVolatile

vacmAccessEntry Operator - usm authPriv exact OperatorView OperatorView \
OperatorView nonVolatile

9.5.5 Public

A user in this group has both read and write privileges to only the minimum set of MIB branches necessary for demonstrations. Also, a user in this group can receive notifications of alert conditions that originate from product demonstrations. Therefore, the specific MIB branches available to a user in this group for reading are specified by the view “ApplicationsView.”

To implement this site security policy, the snmpd.cnf file contains the following entries.

vacmAccessEntry public - usm noAuthNoPriv exact ApplicationsView - \
ApplicationsView nonVolatile

9.5.6 Applications

vacmAccessEntry Applications - usm noAuthNoPriv exact publicView - \
publicView nonVolatile

vacmAccessEntry Applications - usm authNoPriv exact ApplicationsView \
ApplicationsView ApplicationsView nonVolatile

vacmAccessEntry Applications - usm authPriv exact ApplicationsView \
ApplicationsView ApplicationsView nonVolatile

9.6 Access Views

Access View

The access level determines the MIB objects that members (users/communities)of a security group can retrieve with SNMP Get requests or set with SNMP Set requests. There are two types of access views, read and write:

- The read access view determines objects that can be retrieved using SNMP Get requests.
- The write access view determines objects that can be set using SNMP Set requests.

A view describes, in detail, the objects in the MIB tree that members of a Security Group have access to. A access view is created and defined using an entry with the tag vacmAccessEntry. Items can be included or excluded from a view. If an object is not explicitly excluded, then the user has read, but not write access to that MIB object. Items excluded from the view are not viewable to the members of the group. The following views describe the objects viewable by the members of the groups listed in section 11.5. The groups are listed by view as follows:

- All
  - Administrator
- ShiftSupervisorView
  - ShiftSupervisor
- HelpDeskView
  - HelpDesk
- OperatorView
  - Operator
- publicView publicView is the default view for any of the following users for which the user's privileges are noAuthNoPriv.
  - ShiftSupervisor
  - Operator
  - Administrator
  - HelpDesk
  - Applications
- ApplicationsView
  - public
  - Applications

9.6.1 All

The All view supports nonVolatile access to the entire MIB tree, iso. The vacmViewTreeFamilyMask in the All entry (and in some others)is deliberately 0,so that the vacmViewTreeFamilySubtree value iso (1.3) will match anything. This has become necessary as some MIB documents have received registrations outside of the ISO tree.

vacmViewTreeFamilyEntry All iso 0 included nonVolatile

9.6.2 ShiftSupervisorView

The ShiftSupervisorView view supports nonVolatile access to the entire MIB tree,

iso. vacmViewTreeFamilyEntry ShiftSupervisorView iso 0 included nonVolatile

9.6.3 HelpDeskView

The HelpDeskView includes all of the MIB documents accessible by CIAgent and other Hewlett-Packard OpenView MIB documents.

vacmViewTreeFamilyEntry HelpDeskView system - included nonVolatile
vacmViewTreeFamilyEntry HelpDeskView mib_2 - included nonVolatile
vacmViewTreeFamilyEntry HelpDeskView critApp - included nonVolatile
vacmViewTreeFamilyEntry HelpDeskView htmlpage - included nonVolatile
vacmViewTreeFamilyEntry HelpDeskView siServiceMIB - included nonVolatile
vacmViewTreeFamilyEntry HelpDeskView siFsMonitor - included nonVolatile
vacmViewTreeFamilyEntry HelpDeskView siLog - included nonVolatile
vacmViewTreeFamilyEntry HelpDeskView smExtensionMIB - included nonVolatile vacmViewTreeFamilyEntry HelpDeskView httpSecurityMIB - included nonVolatile vacmViewTreeFamilyEntry HelpDeskView srSmExt - included nonVolatile
vacmViewTreeFamilyEntry HelpDeskView eventMIB - included nonVolatile

9.6.4 OperatorView

The OperatorView view supports nonVolatile access to the entire MIB tree, iso.

vacmViewTreeFamilyEntry OperatorView iso 0 included nonVolatile

9.6.5 publicView

The publicView view supports nonVolatile access to the system, srExamples, snmpTrap, snmpTraps, and srBasics MIB objects. The publicView view also contains an instance-based view limitation, “ifEntry” which limits what a user can do with the mib objects in their view.

```
vacmViewTreeFamilyEntry publicView system - included nonVolatile
vacmViewTreeFamilyEntry publicView srExamples - included nonVolatile
vacmViewTreeFamilyEntry publicView snmpTrap - included nonVolatile
vacmViewTreeFamilyEntry publicView snmpTraps - included nonVolatile
vacmViewTreeFamilyEntry publicView srBasics - included nonVolatile
vacmViewTreeFamilyEntry publicView ifEntry.0.2 ff:bf included nonVolatile
```

9.6.6 ApplicationsView

The ApplicationsView supports nonVolatile access to the entire MIB tree, iso. This view specifically excludes read and write access to the following MIB items: snmpCommunityTable, vacmAccessTable, vacmSecurityToGroupTable, usmUserTable, and vacmViewTreeFamilyTable.

```
vacmViewTreeFamilyEntry ApplicationsView iso 0 included nonVolatile
vacmViewTreeFamilyEntry ApplicationsView snmpCommunityTable -
excluded \
        nonVolatile
vacmViewTreeFamilyEntry ApplicationsView vacmAccessTable - excluded \
        nonVolatile
vacmViewTreeFamilyEntry ApplicationsView vacmSecurityToGroupTable -
excluded \
                        nonVolatile
vacmViewTreeFamilyEntry ApplicationsView usmUserTable - excluded \
   nonVolatile
vacmViewTreeFamilyEntry ApplicationsView vacmViewTreeFamilyTable -
excluded        \
   nonvolatile
```

9.7 Access Restriction and Notifications

The snmpTargetAddrEntry is used to specify target addresses for notifications, and valid source address ranges for snmpCommunityTable entries and usmUserTable entries.

9.7.1 Access Restriction

Access Restriction

Access to the SNMP agents configured with a community string can be restricted based
upon the source address of requests. If address restriction is selected, only managers
included in the list of IP addresses will be able to query the agent using the new community string.

Each user is expected to work only from a predetermined set of workstations. If a user attempts to perform work from an unauthorized workstation, this will raise suspicion that the user is attempting a act which is harmful to the company or possibly illegal.

The example workstations are organized into the following groups.

9.7.1.1 OperationsCenter

The operations center consists of all the computers that are essential for the operation of the company. This network consists of a block of 255 TCP/IP network addresses at 10.1.2.xxx. To configure these network addresses; the `snmpd.cnf` file contains the following entry.

```
snmpTargetAddrEntry opsCenter snmpUDPDomain 10.1.2.0:0 0 0
operationsCenterTag none \
        nonVolatile 255.255.255.0:0
```

9.7.1.2 HelpDesk

The HelpDesk is on a separate network than the OpsCenter network. It is important to restrict access to these machines, so the IP addresses in the HelpDesk network must be configured for an Address Restriction. SNMP configuration allows the use of a subnet mask in order to configure a range of network addresses using only one configuration entry. To configure these network addresses using a subnet mask, the snmpd.cnf file contains the following entry.

```
snmpTargetAddrEntry HelpDesk snmpUDPDomain 10.1.3.128:0 0 0 \
        HelpDeskTag none nonVolatile 255.255.255.240:0
```

This configuration entry specifies a network address and a subnet mask. The entry “255.255.255.240” is a mask (equivalent to “10.1.3/28”) that indicates that 16 machines ranging from “10.1.3.128” through “10.1.3.144” belong to

the HelpDesk network. Thus the access view, HelpDesk, can only be used at this range of addresses.

9.7.1.3 Operations Console

This refers to a single machine in the Operations Console Office. To configure this machine's network address, the snmpd.cnf file contains the following entry.

```
snmpTargetAddrEntry opsConsole snmpUDPDomain 10.1.2.110:0 0 0
operationsConsoleTag \
   none nonVolatile 255.255.255.255:0
```

9.7.2 Notifications

When a machine experiences a situation which needs to be reported to a human operator, the site security policy should state that SNMP notifications (Traps or Informs) should be sent to the specific destinations, called notification targets.

Notification Targets

Notification targets are managers or agents that are selected to receive information from SNMP events. Notifications can be sent as either Traps or Informs. A Trap is a one way communication from an agent to a manager. An Inform contains the same information as a Trap; however, with an Inform the manager sends a verification response back to the agent. The user sending the notification must be assigned to a security group that has access to the notification OID and OID of other objects within the notification.

The sample configuration file contains the following notification entries:

9.7.2.1 Notification Entries

The snmpNotifytable selects snmpTargetAddrtable entries to receive Traps or Informs.

```
snmpNotifyEntry Traps TrapTag trap nonVolatile

snmpNotifyEntry Informs InformTag inform nonVolatile
```

9.7.2.2 Notification Targets

The snmpTargetAddrEntry is used to specify target addresses for notifications, and valid source address ranges for snmpCommunityTable entries and usmUserTable entries.

The sample configuration file contains the following specifications for sending notifications:

```
snmpTargetAddrEntry localHostV1 snmpUDPDomain 127.0.0.1:0 100 3
TrapTag \
        v1ExampleParams nonVolatile 255.255.255.255:0

snmpTargetAddrEntry localHostV2c snmpUDPDomain 127.0.0.1:0 100 3
InformTag \
        v2cExampleParams nonVolatile 255.255.255.255:0
```

SNMPv1 Traps sent by the localhost are sent to the machine identified by localHostV1. SNMPv2c Informs sent by the localhost are sent to the machine identified by localHostV2c.

9.7.2.3 Notification Restrictions

The following entries are used by the snmpCommunityEntry and usmUserEntries to restrict from which hosts SNMP requests are honored. The sample configuration honors the tgtAddressMask for both USM and non-USM requests.

```
snmpTargetAddrEntry opsCenter snmpUDPDomain 10.1.2.0:0 0 0 \
        operationsCenterTag none nonVolatile 255.255.255.0:0
snmpTargetAddrEntry opsConsole snmpUDPDomain 10.1.2.100:0 0 0 \
        operationsConsoleTag none nonVolatile 255.255.255.255:0
snmpTargetAddrEntry anywhere snmpUDPDomain 0.0.0.0:0 0 0 \
        anywhereTag none nonVolatile 0.0.0.0:0

snmpTargetAddrEntry HelpDesk snmpUDPDomain 10.1.3.128:0 0 0 \
        HelpDeskTag none nonVolatile 255.255.255.240:0
```

9.7.2.4 Notification Parameters

The snmpTargetParamsTable speci•es what SNMP version, user name, and security level should be used with noti•cations.

```
snmpTargetParamsEntry v1ExampleParams 0 snmpv1 public noAuthNoPriv
\
        nonVolatile
```

snmpTargetParamsEntry v2cExampleParams 1 snmpv2c public noAuthNoPriv \
nonVolatile

snmpTargetParamsEntry v3ExampleParams 3 usm root authNoPriv \
nonVolatile

9.7.2.5 Notification Filters

The snmpNotifyFilterProfileEntry is used to specify which snmpNotifyFilterEntry is to be used with notifications generated with a particular snmpTargetParamsName. This table is used to do source squelching of notifications. This particular example limits SNMPv2c notifications to the well known traps, for example, coldStart, warmStart, and authenticationFailure.

snmpNotifyFilterProfileEntry v2cExampleParams wellKnownTraps nonVolatile

The snmpNotifyFilterEntry specifies the restrictions on which notifications should be sent to targets implied by the snmpNotifyFilterProfileEntry that selected it.

snmpNotifyFilterEntry wellKnownTraps snmpTraps - included nonVolatile

10 Backwards-Compatible SNMP Configuration: snmpd.conf

The `snmpd.conf` configuration file[1] was designed to configure the pre-EMANATE® SNMP agent previously supplied by Hewlett-Packard with very old versions of HP/UX. For the sake of backwards-compatibility, the EMANATE® Master Agent reads[2] the snmpd.conf configuration file in addition to its normal configuration file, snmpd.cnf.

This chapter describes how the EMANATE® Master Agent interprets the information it finds in snmpd.conf to preserve (as much as possible) the SNMPv1 behavior of the pre-EMANATE® SNMP agent when provided with the same configuration file.

10.1 Recognized Configuration Entries

When the EMANATE® Master Agent reads the snmpd.conf configuration file, it recognizes and uses the following entries.

get-community-name: Configures a community string that can be used by an SNMPv1 manager to access the Master Agent with Get and GetNext requests.

The same community string can be used by an SNMPv2c manager to access the Master Agent with Get, GetNext, and GetBulk requests.

set-community-name: Configures a community string that can be used by an SNMPv1 manager to access the Master Agent with Get, GetNext, and Set requests. The same community string can be used by a SNMPv2c manager to access the Master Agent with Get, GetNext, GetBulk, and Set requests.

---

[1] The snmpd.conf file is located in the directory /etc/SnmpAgent.d/. The snmpd.cnf file is located by default in the /etc/srconf/agt directory.

[2] The EMANATE Master Agent never writes to the snmpd.conf configuration file.

IP: Places a source address restriction on a community string. The community string
will not work unless the manager's IP address is identified in a space-separate list following IP: The IP: tag and address list must appear on the same line as the

get-community-name: or set-community-name: tag.

VIEW: Places a MIB object restriction on a community string. The community string
will work only in MIB subtrees identified in a space-separate list following VIEW:.
The VIEW: tag and address list must appear on the same line as the

get-community-name: or set-community-name: tag.

IMPORTANT NOTE: The following comment appears in the file `snmpd.conf`:

```
#       The '\' character may be used at the end of a line to denote
#       continuation of the configuration entry. For example,
#
#       get-community-name operator IP: 15.2.112.90 15.2.114.101 \
#        VIEW: mib-2 hp -system
```

The Master Agent, however, does not correctly recognize this type of line continuation in the snmpd.conf file, so it should not be used. The configuration
information within a single entry should appear all on the same line. It is okay if
the line is so long that it can not be seen within the display width of a typical text editor.

Also in the above example, "mib-2" should be "mib 2". Note that the Master Agent can recognize several hundred object names, but the list is finite. If the Master Agent does not recognize an object name in the snmpd.conf file (such as "mib-2"), it will generate an error message saying MakeOIDFragFromDot(mib-2) failed. This message may be printed to the screen or into the snmpd.log log file.

trap-dest: Configures a destination for sending SNMPv1 Trap messages. Traps
sent to destinations specified with entries of this type will contain the community
string "sendtrap". The list of VarBinds in these Trap messages is not restricted
by a MIB view.

The max-trap-dest: tag is not used by the EMANATE ®Master Agent. The Master Agent will never write to the snmpd.conf configuration file except for a set request of "Contact" or "location".

10.2 Interpretation of Configuration Information

This section describes by example how entries in the snmpd.conf configuration file are interpreted within the SNMPv3 Administration Framework of EMANATE®. For a detailed description of the configuration for the SNMPv3 Administration Framework, refer to Section 10.4.

10.2.1 Example 1 (get-community-name:)

get-community-name: secret

When the above get-community-name: entry in snmpd.conf is processed by the Master Agent, the resulting configuration for the SNMPv3 Administration Framework is equivalent to the following configuration entries in `snmpd.cnf`.

```
snmpCommunityEntry hpconf0002 secret secret localSnmpID - - readOnly
vacmAccessEntry secret - snmpv1 noAuthNoPriv exact default - default -
readOnly
vacmAccessEntry secret - snmpv2c noAuthNoPriv exact default - default \
        readOnly

vacmSecurityToGroupEntry snmpv1 secret secret readOnly
vacmSecurityToGroupEntry snmpv2c secret secret readOnly
vacmViewTreeFamilyEntry default iso - included readOnly
```

This configuration information is stored in RAM and is used by the Master Agent, but it is not saved to the snmpd.cnf configuration file. The configuration information can be seen by sending Get requests to the Master Agent for MIB objects in the SNMPv3 Administration Framework.

```
%getmany -v1 hpux11 secret snmpCommunityTable | egrep 'secret'
snmpCommunityName.104.112.99.111.110.102.48.48.48.50 = secret
snmpCommunitySecurityName.104.112.99.111.110.102.48.48.48.50 = secret
%
```

The above execution of the getmany utility walks the snmpCommunityTable with GetNext requests. Piping the output into the UNIX program egrep allows one to see just the desired lines of output. In this case, the output is

limited to lines that show the “secret” community string. Note that the snmpCommunityEntry contains the word “secret” twice, and there are two lines in the output of getmany that contain “secret”.

```
%gettab -v1 hpux11 secret vacmAccessTable | egrep
’115.101.99.114.101.116|^$’ | uniq

vacmAccessContextMatch.6.115.101.99.114.101.116.0.1.1 = exact(1)
vacmAccessReadViewName.6.115.101.99.114.101.116.0.1.1 = default
vacmAccessWriteViewName.6.115.101.99.114.101.116.0.1.1 =
vacmAccessNotifyViewName.6.115.101.99.114.101.116.0.1.1 = default
vacmAccessStorageType.6.115.101.99.114.101.116.0.1.1 = readOnly(5)
vacmAccessStatus.6.115.101.99.114.101.116.0.1.1 = active(1)

vacmAccessContextMatch.6.115.101.99.114.101.116.0.2.1 = exact(1)
vacmAccessReadViewName.6.115.101.99.114.101.116.0.2.1 = default
vacmAccessWriteViewName.6.115.101.99.114.101.116.0.2.1 =
vacmAccessNotifyViewName.6.115.101.99.114.101.116.0.2.1 = default
vacmAccessStorageType.6.115.101.99.114.101.116.0.2.1 = readOnly(5)
vacmAccessStatus.6.115.101.99.114.101.116.0.2.1 = active(1)

%
```

The above execution of the gettab utility walks the vacmAccessTable with GetNext requests and presents each row of the table together. The egrep program searches for rows[3] that are indexed by the ASCII codes for the letters ‘s,’ ‘e,’ ‘c,’ ‘r,’ ‘e,’ and ‘t.’

Note that each instance of vacmAccessWriteViewName has an empty value. This indicates that the Master Agent will not allow the “secret” community string to perform Set requests. The “secret” community string can obviously perform Get requests, because the gettab execution above is using the “secret” community string to retrieve the MIB objects.

Similar executions of the gettab and getmany utilities could be performed to demonstrate that the get-community-name: entry in snmpd.conf is being utilized by the Master Agent. However, this exercise is left to the reader to perform.

[3] By including the “+|+/\ $” character sequence in the egrep search pattern, and by piping the output to the UNIX program uniq, the spacing between selected table rows is preserved.

10.2.2 Example 2 (get-community-name:)

get-community-name: operator IP: 15.2.112.90 15.2.114.101 VIEW: mib_2 hp -system

When the above get-community-name: entry in snmpd.conf is processed by the Master Agent, the resulting configuration for the SNMPv3 Administration Framework is equivalent to the following configuration entries in snmpd.cnf.

```
snmpCommunityEntry hpconf0002 operator operator localSnmpID -
operatorTran \
        readOnly

vacmAccessEntry operator - snmpv1 noAuthNoPriv exact operatorView - \
        operatorView readOnly

vacmAccessEntry operator - snmpv2c noAuthNoPriv exact operatorView - \
        operatorView readOnly

vacmSecurityToGroupEntry snmpv1 operator operator readOnly
vacmSecurityToGroupEntry snmpv2c operator operator readOnly
vacmViewTreeFamilyEntry operatorView mib_2 - included readOnly
vacmViewTreeFamilyEntry operatorView system - excluded readOnly
vacmViewTreeFamilyEntry operatorView hp - included readOnly
snmpTargetAddrEntry hpt3 snmpUDPDomain 15.2.112.90:0 0 0
operatorTran - \
        readOnly 255.255.255.255:0

snmpTargetAddrEntry hpt4 snmpUDPDomain 15.2.114.101:0 0 0
operatorTran - \
        readOnly 255.255.255.255:0
```

Compared to the configuration for Example 1 (Section 12.2.1), one should first notice that there an additional type of entries for this example.

- The snmpTargetAddrEntry entries correspond to the network addresses following the IP: tag. Next, one should notice that the entries which appear in both examples are slightly different for this example.
- The sixth field in the value portion of the snmpCommunityEntry entry is “operatorTran” instead of a dash (-). This causes the community string “operator” to be associated with the snmpTargetAddrEntry entries, which also contain the value “operatorTran”.
- The vacmViewTreeFamilyEntry entries correspond to the MIB object names following the VIEW: tag. The mib 2 and hp MIB branches are

"included" in the MIB view, and the system branch is "excluded" from the MIB view.

- The sixth and eighth fields in the value portion of each vacmAccessEntry entry is
"operatorView" instead of "default". The default MIB view for community strings defined in the snmpd.conf configuration file is the entire MIB tree. However, the "operator" community string has a smaller MIB view, so these vacmAccessEntry entries must be associated with the vacmViewTreeFamilyEntry entries which define the smaller MIB view.

10.2.3 Example 3 (set-community-name:)

set-community-name: control

When the above set-community-name: entry in snmpd.conf is processed by the Master Agent, the resulting configuration for the SNMPv3 Administration Framework is equivalent to the following configuration entries in snmpd.cnf.

```
snmpCommunityEntry hpconf0002 control control localSnmpID - - readOnly
vacmAccessEntry control - snmpv1 noAuthNoPriv exact default default
default \
        readOnly

vacmAccessEntry control - snmpv2c noAuthNoPriv exact default default
default \
        readOnly

vacmSecurityToGroupEntry snmpv1 control control readOnly
vacmSecurityToGroupEntry snmpv2c control control readOnly
vacmViewTreeFamilyEntry default iso - included readOnly
```

Compared to the configuration for Example 1 (Section 12.2.1), the important difference is that the definition of MIB views for read operations, write operations, and notifications, respectively, has changed from "default – default" and has become "default default default".

```
%gettab -v1 hpux11 control vacmAccessTable | egrep
'99.111.110.11.1014.111.108|^$' | uniq

vacmAccessContextMatch.7.99.111.110.11.1014.111.108.0.1.1 = exact(1)
vacmAccessReadViewName.7.99.111.110.11.1014.111.108.0.1.1 = default
vacmAccessWriteViewName.7.99.111.110.11.1014.111.108.0.1.1 = default
vacmAccessNotifyViewName.7.99.111.110.11.1014.111.108.0.1.1 = default
vacmAccessStorageType.7.99.111.110.11.1014.111.108.0.1.1 = readOnly(5)
vacmAccessStatus.7.99.111.110.11.1014.111.108.0.1.1 = active(1)
```

vacmAccessContextMatch.7.99.111.110.11.1014.111.108.0.2.1 = exact(1)
vacmAccessReadViewName.7.99.111.110.11.1014.111.108.0.2.1 = default
vacmAccessWriteViewName.7.99.111.110.11.1014.111.108.0.2.1 = default
vacmAccessNotifyViewName.7.99.111.110.11.1014.111.108.0.2.1 = default
vacmAccessStorageType.7.99.111.110.11.1014.111.108.0.2.1 = readOnly(5)
vacmAccessStatus.7.99.111.110.11.1014.111.108.0.2.1 = active(1)

%

The above execution of the gettab demonstrates that the vacmAccessWriteViewName for the “control” community string has a value of “default,” meaning that it has the associated default MIB view defined by the corresponding vacmViewTreeFamilyEntry.

%getone -v1 hpux11 control sysName.0
sysName.0 = hpux11

%

%setany -v1 hpux11 control sysName.0 -D "myHpuxMachine" sysName.0 = myHpuxMachine

%

%getone -v1 hpux11 control sysName.0
sysName.0 = myHpuxMachine

%

The above executions of getone and setany demonstrate the “control” community string being used to send a successful SNMP Set request.

10.2.4 Example 4 (trap-dest:)

trap-dest: 15.2.113.223

When the above trap-dest: entry in snmpd.conf is processed by the Master Agent, the resulting con•guration for the SNMPv3 Administration Framework is equivalent to the following configuration entries in snmpd.cnf.

vacmAccessEntry sendtrap - snmpv1 noAuthNoPriv exact default - default \
readOnly

vacmSecurityToGroupEntry snmpv1 sendtrap sendtrap readOnly
vacmViewTreeFamilyEntry default iso - included readOnly snmpNotifyEntry hptrap hptrap trap readOnly

snmpTargetAddrEntry hptrap00000000 snmpUDPDomain 15.2.113.223:162 0 0 \

hptrap hptrap readOnly 0.0.0.0:0 2048

snmpTargetParamsEntry hptrap 0 snmpv1 sendtrap noAuthNoPriv readOnly

The following execution of the traprcv (on host 15.2.113.223)demonstrates that the EMANATE® Master Agent sends a SNMPv1 Trap message as expected.

%traprcv

Waiting for traps.

Received SNMPv1 Trap:
Community: sendtrap
Enterprise: 0.0
Agent-addr: 192.147.142.67
Cold start trap.
Time Ticks: 292

11 MIB-II Configuration

A SNMP entity which has a command responder application and which supports MIB-II must maintain certain configuration information in non-volatile storage. This includes the values of five MIB objects in the system group and one MIB object in the snmp group. This section describes the MIB objects to be configured and this information is configured in Hewlett-Packard OpenView software products.

11.1 The snmpd.cnf Configuration File

The configuration information described in this chapter should be entered into the snmpd.cnf configuration file. Each line in this file has the format

TAG VALUE

where TAG is a keyword and VALUE is a valid configuration value. Entries may be continued across multiple lines by using a backslash (\). White space (tabs, spaces,
line-feeds/carriage-returns) and blank lines in the file are ignored. Values which are strings containing white space must be delimited with quotation marks (").

11.2 Configuring System GroupVariables

The system group of MIB-II identifies the SNMP entity. The value for each system group variable is defined by adding an entry to the snmpd.cnf configuration file.

For example:

sysDescr "Fred Router from Flintstones, Inc."
sysObjectID 1.3.6.1.4.1.4242.1.1
sysLocation "Telephone closet, 3rd floor"
sysContact "Hewlett-Packard OpenView, Inc. +1 423 573 1434"
sysName fred.snmp.com

Each of these system group variables is described in the following sections. To configure default values for MIB-II, select the appropriate value for each variable and add lines similar to the example above (replacing the example values with the correct values)to the snmpd.cnf file.

Note that it is possible to assign values to sysDescr, sysLocation, and sysContact in the snmpd.conf configuration file and from command-line arguments. Values assigned in either of these ways will override (and overwrite)any values specified in the snmpd.cnf configuration file.

11.2.1 sysDescr

The variable sysDescr is a description of the managed entity. This variable should indicate the full name and version identification of the system's hardware type, operating system, and networking software.

11.2.2 sysObjectID

The variable sysObjectID is the vendor's authoritative identification of the network
management subsystem contained in the entity. This value is allocated within the SMI enterprises subtree 1.3.6.1.4.1 and provides an easy and unambiguous means for determining what kind of system is being managed.

As an example, if the vendor "Flintstones, Inc." was assigned the subtree 1.3.6.1.4.1.4242, it could assign the identifier 1.3.6.1.4.1.4242.1.1 to its "Fred Router."

11.2.3 sysLocation

The variable sysLocation tells the physical location of the managed entity.

11.2.4 sysContact

The variable sysContact identifies the contact person for the managed entity and provides information on how to contact this person.

11.2.5 sysName

The variable sysName identifies the administratively assigned name for the managed node. By convention, the value of this variable should be the node's fully qualified domain name.

11.3 Configuring SNMP GroupVariables

When a SNMP engine receives a SNMP packet, the message processing subsystem consults with the security subsystem to make sure that the packet is 'okay' (authentic and/or secure, according to the SNMP configuration). If the packet passes the security check, the SNMP engine will dispatch the PDU to the appropriate SNMP application. If the packet fails to pass the security check, the engine will:

1. Discard the original packet,
2. Send a Report PDU to the sender of the original packet (only if the original packet is not
   a SNMPv1 packet), and
3. If the value of snmpEnableAuthenTraps is enabled (1), send an
   authentication-failure Trap to all configured SNMP entities which have a notification
   receiver application.

Figure 11.1 depicts the role of authentication failure traps. Like all other traps,
authentication-failure traps must be configured for destination and security, but the sending of these traps (when an authentication-failure event occurs) must be enabled for the traps to actually be generated.

The snmpEnableAuthenTraps MIB object is a member of the snmp of MIB-II. This variable enables or disables authentication-failure traps. The variable should be set to enabled(1) to enable traps or disabled(2) to disable traps. Authentication-failure traps are disabled by default.

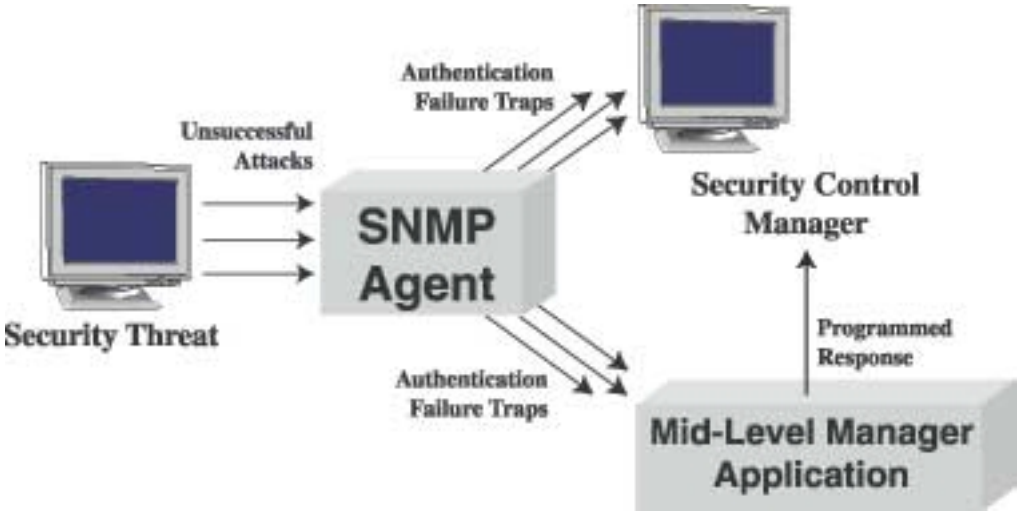
To enable authentication-failure traps, add the following line to the snmpd.cnf file:

snmpEnableAuthenTraps 1

If the value of the object is at its maximum, the value wraps to zero.

To explicitly disable authentication-failure traps, use the -authfail command-line argument when launching the EMANATE® Master Agent. This will override (and overwrite) any value specified in the snmpd.cnf configuration file.

Figure 11.1: The role of authentication-failure traps



11.4 Backwards-Compatible MIB-II Configuration: snmpd.conf

The snmpd.conf configuration file[1] was designed to configure the pre-EMANATE® SNMP agent previously supplied by Hewlett-Packard with very old versions of HP/UX. For the sake of backwards-compatibility, the EMANATE® Master Agent reads[2] the snmpd.conf configuration file in addition to its normal configuration file, snmpd.cnf.

This section describes how the EMANATE® Master Agent uses the information it finds in snmpd.conf to preserve the behavior of the pre-EMANATE® SNMP agent when provided with the same configuration file. This section also addresses a potential problem with this behavior and how to make an appropriate correction.

11.4.1 Recognized Configuration Entries

When the EMANATE® Master Agent reads the snmpd.conf configuration file, it recognizes and uses the following entries.

contact

Configures the value of the MIB object sysContact.0. If this configuration entry appears in snmpd.conf, it will override (and overwrite) any sysContact value specified in the snmpd.cnf configuration file. If the Master Agent is launched with the command-line argument -Contact name, the specified name will override any other sysContact value.

location

Configures the value of the MIB object sysLocation.0. If this configuration entry appears in snmpd.conf, it will override (and overwrite) any sysLocation value specified in the snmpd.cnf

[1] The snmpd.conf file is located in the directory /etc/SnmpAgent.d/. The snmpd.cnf file is located by default in the /etc/srconf/agt directory.

[2] The EMANATE Master Agent never writes to the snmpd.conf configuration file.

configuration file. If the Master Agent is launched with the command-line argument -Location place, the specified place will override any other sysLocation value.

sys-descr

Configures the value of the MIB object sysDescr.0. If this configuration entry appears in snmpd.conf, it will override (and overwrite)any sysDescr value specified in the snmpd.cnf configuration file. If the Master Agent is launched with the command-line argument -sysDescr string, the specified string will override any other sysDescr value.

11.4.2 Caveats

Each time the EMANATE® Master Agent is started, the values of contact:, location:, and sys-descr: from the snmpd.conf file are saved in the snmpd.cnf file, overwriting any value of sysContact, sysLocation, or sysDescr, respectively, that exists in that file. When a SNMP Set request is processed for sysContact.0, sysLocation.0, or sysDescr.0, the new value for the MIB object is also saved in snmpd.cnf. Because the EMANATE® Master Agent does not write to the snmpd.conf configuration file except for some set requests to "Contact" or "location". Therefore, any change to the value of sysContact.0, sysLocation.0, or sysDescr.0 made via SNMP Set requests are lost when the EMANATE® Master Agent restarts.

To correct this undesirable behavior, the lines of snmpd.conf that begin with contact:, location:, or sys-descr: should be made into a comment after the EMANATE® Master Agent has been executed for the first time on a system. To make a line into a comment, insert a hash mark, or pound sign ('#') character at the beginning on the line.

12 Command-line Utilities

This chapter gives a brief introduction to the Hewlett-Packard OpenView command-line Utilities. The Utilities are management tools designed to perform basic queries of a SNMP agent. They are provided to the user as a set of convenient commands to use to access MIB variables in the agent. The utilities described in this chapter are:

- getone
  get the MIB variable(s)specified
- getnext
  get the MIB variable(s)which is(are)lexicographically “next” of the variable(s)specified
- getmany
  get the MIB variables in the specified MIB subtree
- setany
  set one or more MIB variables to the values specified
- getbulk
  get as many MIB variables as possible in one request beginning with the variable specified
- traprcv
  display the contents of Traps received
- trapsend
  send a Trap to the specified manager

12.1 Basic Operations

The HP OpenView NNM SPI for SNMPv3 Utilities are small manager applications which, when connected to the BRASS™ server, perform the same kinds of basic functions found in robust SNMP manager applications[1]

[1] One utility, trapsend, performs a basic function of an SNMP agent, NOT a manager. The focus of this section is to describe the utilities acting in the manager role, so some information may not apply to trapsend.

such as HP OpenView. The primary manager function is to send a SNMP request and to receive and process the SNMP reply. The action of the robust SNMP manager as well as the Utilities is directed by the user. Both kinds of manager applications contain SNMP configuration information and require the same types of input from the user to send requests.

There are two primary differences between the HP OpenView NNM SPI for SNMPv3 utilities and robust manager applications:

- A robust application may attempt to hide more of the operational detail from the user, so it may make more assumptions about the kind of information being requested and may ask the user for less information;
- A robust application may provide a “preferences” configuration which is configured once at startup and never again, whereas the Hewlett-Packard OpenView Utilities examine the current environment variables for operational preferences;

This section will explain the operational details common to all of the HP OpenView NNM SPI for SNMPv3 utilities and how the preferred operations may be set in the runtime environment.

12.1.1 Command-Line Arguments

The HP OpenView NNM SPI for SNMPv3 utilities accept input parameters as command-line arguments. For each utility[2] the user can specify the following information on the command line.

These arguments may be specified in any order, but they must appear before the Agent Address (agent addr) argument (described below).

- SNMP Version
  The SNMP version may be either SNMPv1 (-v1), SNMPv2c (-v2c), or SNMPv3 (-v3).
- Context
  By default, the utility sends all requests in the “default” context. If the user wishes to send a request in a different context, the context is specified using the -ctx string argument where “string” is the contextName.

[2] Except for traprcv, which takes only the -d argument to turn on message logging (debug output).

For example, the following invocations of the getmany Utility send Get requests for MIB objects in the vcr group[3] to a SNMP agent running on the machine called "myhost." The agent is programmed to provide a different response based on the context of the request. If the Get requests are sent in the default context, the SNMP agent responds to the manager with an indication that it contains no information about the VCR-MIB:

```
%getmany -v3 myhost Administrator vcr
Enter Authentication Password:
%
```

If the Get requests are sent in the "UPS1" context, the SNMP agent responds to the manager with the values of MIB objects from the VCR-MIB:

```
%getmany -v3 -ctx UPS1 myhost Administrator vcr
Enter Authentication password :
vcrChannel.0 = 6
vcrPowerState.0 = on(1)
vcrProgramChannel.4 = 3
vcrProgramStartTime.4 = 199501240600
vcrProgramStopTime.4 = 199501240700
vcrProgramSpeed.4 = 1
vcrProgramStatus.4 = active(1)
%
```

- Context Engine ID

  By default, the utility assumes that the response to a request should be provided by the agent residing at agent addr. If the user wishes to send a request in a proxy context, the snmpEngineID of the agent which should provide the response is specified using the -ctxid string argument where "string" is the snmpEngineID.

  For example, the following invocation of the getone Utility sends a Get request for the MIB object sysDescr.0 to a proxy agent located at the IP address 192.147.142.35. The proxy agent forwards[4] the request to another SNMP agent whose snmpEngineID equals the value of the "string" (refer to Figure 12.1).

---

[3] The vcr group of MIB objects is described in the VCR-MIB, an example MIB document from Hewlett-Packard OpenView.

[4] The proxy agent can only forward an SNMP request if the context engine ID is known to the proxy forwarder application.

%getone -v3 -ctxid 00:00:00:63:00:00:00:a1:c0:93:8e:81 192.147.142.35 \
v3v1ProxyUser sysDescr.0

Enter Authentication password:

sysDescr.0 = Target SNMP Agent at 192.147.142.129
%

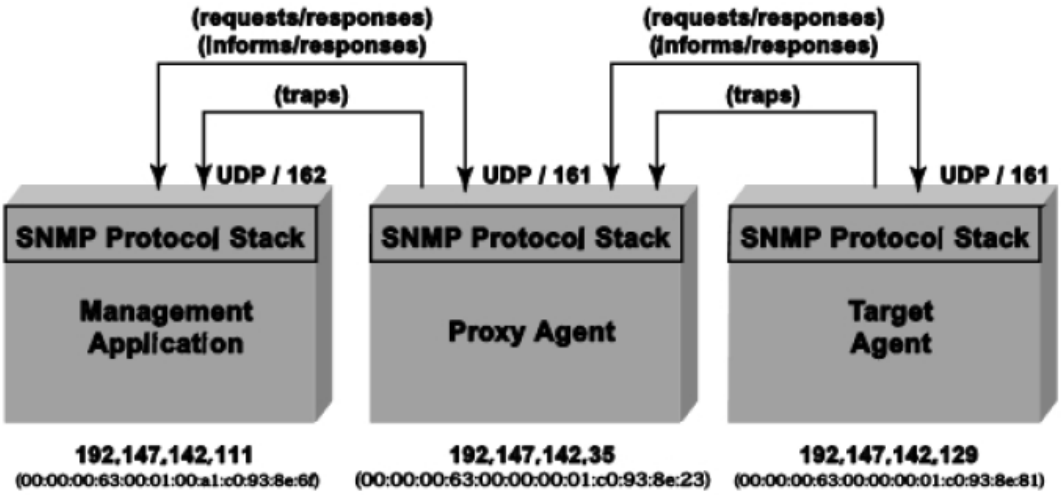


Figure 12.1: The -ctxid flag can be used to forward requests through a proxy agent

- Debug Output

  The -d argument turns on message logging in most[5] of the Utilities. When this feature is selected, the utility prints messages containing important information at various points of execution. This is useful for problem diagnosis. For example, if the utility exits with the error message Failure in snmpinfo.dat, the user may be able to determine that the environment variable SR_MGR_CONF_DIR points to a directory which does not contain the configuration file. The debug message in this circumstance would be similar to the following:

  OpenConfigFile: can't open /home/myconfig/snmpinfo.dat with mode 1
  at line 221 in file k_fileio.c

  Refer to Section 7.2 for information about the snmpinfo.dat configuration file.

[5] The trapsendagt utility uses -aperror, -aperror, etc. Refer to Section 12.2.7.2.

- Timeout

  By default, the utility will wait a total of ten (10) seconds to receive a response from a SNMP agent. If no response is received within that timeout period, the utility may try again to send the request. The -timeout seconds (or -t seconds) argument can be used to change the length of the timeout period where “seconds” is the time in seconds.

- Retries

  By default, if the utility receives no response from the SNMP agent, it will try again five(5) more times to send the request. The -retries number (or -r number)argument can be used to change the number of times the utility will send the request where “number” is the number of retries.

- Maximum Message Size

  By default, the utility will not transmit a SNMP request or receive a SNMP notification if the Basic Encoding Rules-encoded length of the packet exceeds 2048 bytes. The -pkt size n argument can be used to set the maximum supported packet size for all SNMPv1, SNMPv2c, and SNMPv3 messages to n bytes. The value of n must be in the range from 484 to 2147483647.

After the previous arguments have been specified, the following arguments should be provided. Unlike the previous arguments, it is important to use the following arguments in order. Refer to the sections in this chapter that detail the precise usage for each of these utilities.

- Destination Address

  This argument is the destination address for the SNMP message (agent addr or dst addr) specified as a host name or numeric IPv4 address in the form “xxx.xxx.xxx.xxx.” Most of the utilities, like getone, act in the manager role and send SNMP requests to an agent entity. Some of the utilities, like trapsend, act in the agent role and send SNMP messages a manager entity.

- Community String or User Name

  If the SNMP version was set to SNMPv1 by the -v1 argument or to SNMPv2c by the -v2c argument, the community/userName argument is the community string to in for the SNMP message. If the SNMP version was set to SNMPv3 by the -v3 argument, the community/userName argument is the user name to use in the SNMP message.

  If the SNMPv3 user has been configured to use authentication, the user will be prompted

for an authentication password: if no authentication password has been set via the
SR_UTIL_AUTH_PASSWORD environment variable, and no authentication password has been placed in the configuration file for this user. The authentication protocol is specified by prefixing the authentication passphrase with one of the following authentication indicator tags:

-MD5: for the HMAC-MD5-96 authentication protocol

MD5: is the default, and may be omitted if the user has been configured in the agent for MD5-based authentication.

-SHA: for the HMAC-SHA-96 authentication protocol.

For example, if a SNMPv3 user is configured to use SHA-1, and the authentication pass phrase is authpass, the user would enter SHA:authpass at the pass phrase prompt.

If the SNMPv3 user has been configured to use privacy, the user will be prompted for a privacy password: if no privacy password has been set via the SR_UTIL_PRIV_PASSWORD environment variable, and no privacy password has been placed in the configuration file for this user. The privacy protocol is specified by prefixing the privacy passphrase with one of the following privacy indicator tags:

-DES: CBC-DES Symmetric Encryption Protocol DES: is the default, and may be
omitted if the user has been configured in the agent for DES-based privacy.

The following privacy protocols may be specified if the product has been configured to support Extended Security Options (ESO):

-3DES: 3DES-EDE Privacy Protocol[6]

-AES: CFB128-AES-128 Symmetric Encryption Protocol[7] -AES128: CFB128-AES-128 Symmetric Encryption Protocol[7]

---

[6] 3DES-EDE is defined in the Extended Security Options Consortium document, entitled "Reeder Triple DES draft with changes" (located at http://www.snmp.com/eso/)

[7] Hewlett-Packard OpenView agents and command-line utilities support the AES-128 protocol, which is not standardized.It is documented in http://www.snmp.com/eso/draft-blumenthal-aes-usm-06.txt and is likely to become a standard soon.

-AES192: CFB128-AES-192 Symmetric Encryption Protocol[8] -AES256: CFB128-AES-256 Symmetric Encryption Protocol[8]

For example, if a SNMPv3 user is configured to use 3DES, and the privacy pass phrase is privpass, the user would enter 3DES:privpass at the pass phrase prompt.

After these arguments have been specified, the utility may require additional arguments that are specific to the application. These arguments will be described here. Refer to the section for each specific utility for precise usage information.

- MIB Variable Name

  A MIB variable is known to a SNMP agent only by its fully qualified numeric object identifier. For example, the only way that a SNMP agent can identify the MIB variable sysDescr is by the OID value, which is 1.3.6.1.2.1.1.1.0. An agent does not know that the string "sysDescr" also refers to the same MIB object.[9] Humans, however, tend to use "sysDescr" instead of "1.3.6.1.2.1.1.1.0" because it is shorter to type and easier to remember.

  The Hewlett-Packard OpenView Utilities read a configuration file called snmpinfo.dat which contains "name-to-OID" OID translation information.[10] The Utilities use this information to attempt to recognize a MIB variable by its various names containing English strings. The variable name [. . .] argument(s) may be specified using any valid combination of English and number OID subidentifiers. To demonstrate the flexibility of this argument, Figure 12.2 shows all of the valid ways to specify the MIB variable sysDescr to the Utilities.

[8] CFB128-AES-192 and CFB128-AES-192 are documented in http://www.snmp.com/eso/draft-blumenthal aes-usm-04.txt and not included in the later revisions of the AES USM document. At this time, these protocols appear unlikely to be standardized.

[9] Just as some readers may not be aware that mgmt.1.1.1.0 also refers to the MIB variable sysDescr.

[10] Refer to Section 7.2 for information about the snmpinfo.dat configuration file.

Names for sysDescr
1.3.6.1.2.1.1.1.0
iso.3.6.1.2.1.1.1.0
org.6.1.2.1.1.1.0
dod.1.2.1.1.1.0
internet.2.1.1.1.0
mgmt.1.1.1.0
mib 2.1_1.0
system.1.0
sysDescr.0

Figure 12.2: Equivalent names for the MIB variable sysDescr

- MIB Table Name

  Specifying a MIB table name works exactly the same way as specifying a MIB variable name. The only difference is that the English name or numeric OID refers to a table (a group of objects) rather than to a single MIB variable. Figure 12.3 shows examples of some common table names.

English Names	Numeric OIDs
ifTable	1.3.6.1.2.1.2.2
atTable	1.3.6.1.2.1.3.1
ipAddrTable	1.3.6.1.2.1.4.20
ipRouteTable	1.3.6.1.2.1.4.21
ipNetToMediaTable	1.3.6.1.2.1.4.22
tcpConnTable	1.3.6.1.2.1.6.13
udpTable	1.3.6.1.2.1.7.5
egpNeighTable	1.3.6.1.2.1.8.5
snmpORTable	1.3.6.1.6.3.1.1.3.2
srCommunityTable	1.3.6.1.4.1.99.12.33.2
userNameTable	1.3.6.1.6.3.1135.3.3
usmUserTable	1.3.6.1.6.3.15.2.2.2
vacmSecurityToGroupTable	1.3.6.1.6.3.1.10.2
vacmAccessTable	1.3.6.1.6.3.1.10.4

Figure 12.3: English and numeric names of some MIB tables

- MIB Variable Type and Value

  Every MIB variable is defined to be of a specific type, and there is a different way to denote

  a value of each type as a command-line argument. To specify a MIB variable and value to

  the utility, use the -type value argument[11] where "-type" is one of the following flags

-i	INTEGER
-o	OCTET STRING
-d	OBJECT IDENTIFIER
-a	IP Address
-c	Counter
-j	Counter64
-g	Gauge
-t	TimeTicks
-T	DateAndTime
-b	BITS
-D	Display String
-N	NULL

  and "value" is the desired value. The value for INTEGER, Counter, Gauge, and TimeTicks types should be specified as a simple number. The value for an enumerated INTEGER may be also is specified with the

[11] Note that "-type" is an optional flag for all values except -b (BITS) and -D (Display String). The type may still be put on the command line, but it is only necessary in the case of BITS and Display Strings.

enumeration (in ASCII text). The specification for OCTET STRING values is a quoted string containing a hexadecimal value (one or more hex digits, upper or lower case)followed by a space, with more hex values and spaces to finish out the string. The specification for Display String values is a quoted string containing NVT ASCII characters. The specification for OBJECT IDENTIFIER values is a quoted string containing an OID in name-number form (such as sysDescr.0) or number form (such as

1.3.6.1.2.1.1.1.0).The specification for IP Address values is a quoted string containing the internet address in network byte order (bytes ordered from left to right)in the form "xxx.xxx.xxx.xxx." The specification for BITS values is a comma-separated list of individual bit positions (in decimal) or enumerations (in ASCII text) to be turned on; i.e., set to '1.' Unspecified bits are turned off; i.e., cleared to '0'. The flags -setall and -clearall are recognized in the value field to mean "all bits turned on" and "all bits turned off," respectively. The value field for NULL is ignored but must be present (use '0').

- Subnet Address

  A subnet address is a partial IP address (a substring of an IP address). This argument is interpreted as first octet(s) of the IP address. In other words, '128' will match addresses 128.xxx.xxx.xxx, but not addresses with xxx.128.xxx.xxx or xxx.xxx.128.xxx.

12.1.2 Environment Variables

The Hewlett-Packard OpenView Utilities allow the user to specify operational preferences through the use of environment variables. This section describes the available environment variables and how each one changes the behavior of the Utilities.

- SR_MGR_CONF_DIR

  This environment variable changes the location where the Utilities look for the configuration files mgr.cnf and snmpinfo.dat.[12] The value of this variable is the full path of the directory containing the configuration files, such as /etc/srconf/mgr on UNIX systems or C:\ETC\SRCONF\MGR on Microsoft Windows systems.

- SR_SNMP_TEST_PORT

[12] See Chapter 9 for information about the mgr.cnf and snmpinfo.dat configuration files.

This environment variable changes the UDP port number which the Utilities use to send and receive SNMP packets (normally 161). This environment variable also changes the UDP port number which the Utilities use to receive SNMP traps (normally 162). The new port number for SNMP requests is the value of the environment variable, and the new port number for SNMP traps is one(1)plus the value of the environment variable.

- SR_TRAP_TEST_PORT

  This environment variable changes the UDP port number which the Utilities use to receive SNMP traps (normally 162). The new port number for SNMP traps is the value of the environment variable. This environment variable takes precedence over SR_SNMP_TEST_PORT, so SR_TRAP_TEST_PORT can be used to override an unwanted change to the trap port caused by SR_SNMP_TEST_PORT.

- SR_UTIL_SNMP_VERSION

  This environment variable sets the default SNMP version to use for all subsequent requests. The value of this environment variable may be "-v1", "-v2c", or "-v3." If this variable is set, the user does not need to specify -v1, -v2c, or -v3 on the command line.

- SR_UTIL_COMMUNITY

  This environment variable sets the default community string for all subsequent SNMPv1 and SNMPv2c requests. If this variable is set, the user does not need to specify a community string on the command line for SNMPv1 or SNMPv2c requests.

- SR_UTIL_USERNAME

  This environment variable sets the default user name for all subsequent SNMPv3 requests. If this variable is set, the user does not need to specify his or her user name on the command line for SNMPv3 requests.

- SR_UTIL_AUTH_PASSWORD

  This environment variable sets the authentication password for all subsequent requests. If this variable is set to a null string, the user will not be prompted for a password and a noAuthNoPriv request will be sent; otherwise, the user will be prompted only for a privacy password.

- SR_UTIL_PRIV_PASSWORD

This environment variable sets the privacy password for all subsequent requests. If SR_UTIL_AUTH_PASSWORD is not set, this variable is ignored.

12.2 Specific Operations

This section describes each of the Hewlett-Packard OpenView Utilities in detail. The command line for each utility is shown, and command-line arguments which have not been previously described are explained in detail. An example of how to use each utility is also provided.

12.2.1 The getone Utility

12.2.1.1 getone Description

The getone utility is a SNMP application used to retrieve a set of individual variables from a SNMP entity using a Get request.

12.2.1.2 getone Command Line

```
usage: getone [-v1] [-v2c] [-v3] \
        [-ctxid contextID] [-ctx contextName] \
        [-d] [-timeout seconds] [-retries number] \
        agent_addr community/userName \
variable_name [variable_name . . .]
```

12.2.1.3 getone Examples

Variables may be in name form, number form, or as they appear in the MIB document. Since the operation is Get, as opposed to a GetNext, the variable MUST be fully qualified (include instance information) for the request to be successful.

For example, each of the following fully qualified entries would return the variables sysDescr.0 and snmpInPkts.0:

%getone -v1 myagent public sysDescr.0 snmpInPkts.0 %getone -v2c myagent public sysDescr.0 snmpInPkts.0 %getone -v3 myagent Administrator sysDescr.0 snmpInPkts.0

However, the following calls would return an error from the entity since it is not a fully qualified SNMP variable:

%getone -v1 myagent public system
%getone -v2c myagent public system
%getone -v3 myagent Administrator system

12.2.1.4 getone Limitations

If any one of the variables in the list is a malformed OID or unrecognized object descriptor, nothing will be sent out and an error will be displayed.

12.2.2 The getnext Utility

12.2.2.1 getnext Description

The getnext utility is a SNMP application used to retrieve the lexicographically 'next' value after the specified variable from a SNMP entity using a GetNext request.

12.2.2.2 getnext Command Line

```
usage: getnext [-v1] [-v2c] [-v3]              \
          [-ctxid contextID] [-ctx contextName] \
          [-d] [-timeout seconds] [-retries number] \
          agent_addr community/userName \
variable_name [variable_name . . .]
```

12.2.2.3 getnext Examples

For example, each of the following requests would return the variables sysDescr.0 and snmpInPkts.0:

%getnext -v1 myagent public sysDescr snmp %getnext -v2c myagent public sysDescr snmp %getnext -v3 myagent Administrator sysDescr snmp

However, each of the following calls would return snmpOutPkts.0, the 'next' variable after snmpInPkts.0:

%getnext -v1 myagent public snmpInPkts.0 %getnext -v2c myagent public snmpInPkts.0 %getnext -v3 myagent Administrator snmpInPkts.0

12.2.2.4 getnext Limitations

If any one of the variables in the list is a malformed OID or an unrecognized object descriptor, nothing will be sent out and an error will be displayed.

12.2.3 The getmany Utility

12.2.3.1 getmany Description

The getmany utility retrieves the entire variable class (variable class)by sending a GetNext request to a SNMP entity to retrieve the first variable in the class. It then sends a GetNext to the entity using the variable name returned in the previous call to retrieve the next variable in the class. Traversal stops when all of the classes being polled return a variable of a class different than what was requested. In other words, it progressively 'walks' through the MIB until the end of the variable class.

12.2.3.2 getmany Command Line

```
usage: getmany [-v1] [-v2c] [-v3] \
        [-ctxid contextID] [-ctx contextName] \
        [-d] [-timeout seconds] [-retries number] \
        agent_addr community/userName \
variable_name [variable_name . . .]
```

12.2.3.3 getmany Examples

The following will traverse the SNMP entity's system variable class:

%getmany -v1 myagent public system

%getmany -v2c myagent public system
%getmany -v3 myagent Administrator system

The SNMP entity's entire MIB tree can be traversed using one of the following examples: %getmany -v1 myagent public iso

%getmany -v2c myagent public iso
%getmany -v3 myagent administrator iso

12.2.3.4 getmany Limitations

If more than one variable class is requested, odd effects may be seen. Specifically, since 'The traversal stops when all of the classes being polled return a variable of a class different than what was requested,' once the first variable class completes, everything quits whether all the variables were returned or not. Another side-effect of this behavior is if the first variable class is 'large,' the other variable classes will continue to be retrieved whether they're finished or not. This can result in "End of MIB" or other messages being displayed within the values from other requests.

If more than one variable class is requested and any one of the classes does not exist, nothing will be returned for any of the classes.

12.2.4 The setany Utility

12.2.4.1 setany Description

The setany utility performs a Set request on variables passed to it.

12.2.4.2 setany Command Line

```
usage: setany [-v1] [-v2c] [-v3] \
        [-ctxid contextID] [-ctx contextName] \
        [-d] [-timeout seconds] [-retries number] \
        agent_addr community/userName \

variable_name type value [variable_name type value . . .]
```

12.2.4.3 setany Examples

For setany to succeed, the variable's MAX-ACCESS as defined by the MIB must be read-write (or read-create), and the userName or community string must be authorized to perform a Set request on the indicated variable.

In the following example, the SNMPv1 community string "public" is authorized to get the value of sysContact.0 but is not authorized to set the value:

```
%getone -v1 myagent public sysContact.0
sysContact.0 = Hewlett-Packard OpenView, Inc. (865)573-1434
%setany -v1 myagent public sysContact.0 -D "support@snmp.com"
Error code set in packet - No such variable name. Index: 1.
```

In the following example, the SNMPv2c community string “public” is authorized to get the value of sysContact.0 but is not authorized to set the value:

```
%getone -v2c myagent public sysContact.0
sysContact.0 = Hewlett-Packard OpenView, Inc. (865)573-1434
%setany -v2c myagent public sysContact.0 -D "HP Support" Error code set in
packet - NO_ACCESS_ERROR: 1.
```

In the following example, the SNMPv3 user “public” is not authorized to send Set requests to this SNMP entity but can Get the value of sysContact.0:

```
%getone -v3 myagent Administrator sysContact.0
Enter Authentication password:
sysContact.0 = Hewlett-Packard OpenView, Inc.          (865)573-1434
%setany -v3 myagent Administrator sysContact.0 -D "HP Support" Enter
Authentication password:
Error code set in packet - NO_ACCESS_ERROR: 1.
```

In the following example, the variable is sysContact.0 and the SNMPv3 user is “HelpDesk”. This user is authorized to send Set requests to this SNMP entity, but the variable is not in the user’s MIB view for Get or Set requests:

```
%getone -v3 localhost HelpDesk sysContact.0
Enter Authentication password:
Error code set in packet - AUTHORIZATION_ERROR: 1.
%setany -v3 localhost HelpDesk sysContact.0 -D "HP Support"
Enter Authentication password:
Error code set in packet - AUTHORIZATION_ERROR: 1.
```

In the following example, the SNMPv3 user “Administrator” is authorized to send Set requests to this SNMP entity, and the user’s MIB view for Get and for Set requests includes both sysContact.0 and sysDescr.0. The value of sysDescr.0 can not be set because the variable’s MAX-ACCESS as defined by the MIB is read-only:

```
%getone -v3 myagent Administrator sysContact.0 sysDescr.0
Enter Authentication password:
sysContact.0 = Hewlett-Packard OpenView, Inc.          (865)573-1434
sysDescr.0 = SNMPv3 agent from Hewlett-Packard OpenView, Inc.
%setany -v3 myagent Administrator sysContact.0 -D "HP Support"
Enter Authentication password:
sysContact.0 = support@hp.com
%setany -v3 myagent Administrator sysDescr.0 -D "SNMPv3 agent"
Enter Authentication password:
Error code set in packet - NOT_WRITABLE_ERROR: 1.
```

The setany utility can read DateAndTime strings in the same format that getmany prints them. So, for example, the vcr example subagent setany command could use the string 2004-Jan-10,12:30:00.0.

12.2.5 The trapsend Utility

12.2.5.1 trapsend Description

The trapsend utility demonstrates how to send Trap messages to SNMP entities containing a notification receiver application. This utility can send SNMPv1, SNMPv2c, and SNMPv3 Traps. The arguments to trapsend depend on which kinds of Traps are being sent.

12.2.5.2 trapsend Command Line

usage: trapsend [-v1] dst_addr community generic_trap \
[specific_trap enterprise time_ticks] \

variable_name type value [variable_name type value . . .]

usage: trapsend [-v2c] [-v3 [-alt_engine_id]] \
[-ctxid contextID] [-ctx contextName] \
dst_addr community/userName snmpTrapOID_value(OID) \
variable_name type value [variable_name type value . . .]

In addition to the command line arguments described in Section 14.1.1, trapsend may use some additional arguments which are as follows:

Arguments for SNMPv1 traps

generic trap

The generic trap number of SNMPv1 pre-defined traps.

generic trap

The specific trap number of SNMPv1 enterprise specific traps. This value is defined in the SMIv1 MIB document, and when it is used, generic trap must be equal to ‘6’.

enterprise

The enterprise argument is the value defined by the ENTERPRISE clause in the SMIv1 MIB document for this SNMPv1 trap.

time ticks

The time ticks argument is the actual value of the variable sysUpTime.0.

Arguments for SNMPv3 traps

snmpTrapOID value(OID)

The snmpTrapOID value (OID) argument is the OBJECT IDENTIFIER of the trap as defined by the SMIv2 MIB document. This argument will become the value of snmpTrapOID.0 in the Trap packet. Name forms, such as coldStart or myWonderfulEntTrap are permitted.

-alt engine id

Use an alternate snmpEngineID in SNMPv3 traps.

Use this qualifier when running all three of the following on the same host:

- A SNMPv3 management application (such as an Hewlett-Packard OpenView SNMPv3 SPI Server or the traprcv utility),
- A SNMPv3 agent that sends traps to the same host as it is running on, using the default Hewlett-Packard OpenView snmpEngineID algorithm, and
- trapsend to send authenticated (SNMPv3 authNoPriv or authPriv)traps to the management application on the same host.

  In this set of circumstances, the management application will be unable to differentiate between traps from the SNMP agent and trapsend. Because each authenticated trap bears a timeliness indicator, and SNMPv3 management applications must maintain a table of timeliness indicators indexed by snmpEngineID, if the timeliness indicators vary between the agent and the trapsend utility (and they most likely will), the management application is likely to decide that traps from one source or the other are stale, and disregard them. When the -alt engine id argument is specified, the seventh and eighth byte positions in the original snmpEngineID are altered to”00:15” as follows:

  Original snmpEngineID: 00:00:00:63:00:01:00:a1:c0:93:8e:c0

  Altered snmpEngineID: 00:00:00:63:00:01:00:15:c0:93:8e:c0

When an alternate snmpEngineID is used, the user’s password must be con•gured with that snmpEngineID in the SNMPv3 management application so that it can receive a SNMPv3 Trap properly. For security reasons, SNMPv3 Traps have build in timeliness parameters which are often cached in management applications (such as the SNMPv3 SPI Server) using the snmpEngineID of the sender as the key. Changing the snmpEngineID in packets sent by trapsend avoid collisions in this cache.

12.2.5.3 trapsend Examples

Sending generic traps

The following generic trap with SNMPv1 would send a SNMPv1 coldStart trap to the machine mymgr:

```
%trapsend -v1 mymgr public 0
```

The following generic trap with SNMPv2c would send a SNMPv2c warmStart trap to the machine mymgr:

```
%trapsend -v2c mymgr public snmpTraps.2
```

The following generic trap with SNMPv3 would send a SNMPv3 linkUp trap to the machine mymgr:

```
%trapsend -v3 mymgr Administrator linkUp
```

Sending enterprise specific traps

The following examples are based on an Hewlett-Packard OpenView example MIB called the SURGE-PROTECTOR-MIB.

Sending enterprise specific traps with SNMPv1

The following trap would send a SNMPv1 enterprise specific trap with an ENTERPRISE of surgeProtector, specific trap of 5, sysUpTime.0 of 4974123, and the appropriate variable in the VarBind list:

%trapsend -v1 myagent public 6 5 surgeProtector \
4974123surgeBreakerStatus -i closed

Sending enterprise specific traps with SNMPv2c

The following trap would send a SNMPv2c enterprise specific trap surgeBreakerAlarm, with sysUpTime.0 of 4974123 and the appropriate variable in the VarBind list.

%trapsend -v2c myagent public surgeBreakerAlarm \
4974123surgeBreakerStatus -i open

Sending enterprise specific traps with SNMPv3

The following trap would send a SNMPv3 enterprise specific trap, surgeBreakerAlarm, with sysUpTime.0 of 4974123 and the appropriate variable in the VarBind list.

%trapsend -v3 myagent Administrator surgeBreakerAlarm \
4974123surgeBreakerStatus -i unknown

Sending enterprise specific traps with SNMPv3 using an altered snmpEngineID

The following trap would send a SNMPv3 enterprise specific trap using an altered snmpEngineID.

%trapsend -v3 -alt_engine_id myagent Administrator surgeBreakerAlarm \ 4974123surgeBreakerStatus -i unknown

The following message is displayed after trap is sent. The user's password in the SNMPv3 management application has to be configured with an altered snmpEngineID shown in the message to receive a SNMPv3 Trap properly.

Sent SNMPv3 Trap with altered snmpEngineID:
0:0:0:63:0:1:0:15:c0:93:8e:ad

12.2.6 The traprcv Utility

12.2.6.1 traprcv Description

The traprcv utility is a program that receives SNMP Trap messages and responds to SNMP Inform requests from remote SNMP entities. It binds to the SNMP trap port (udp/162)to listen for the notifications, and thus must be run as root. It prints to standard output messages about the notifications it has received. Note that traprcv can receive SNMPv1 Traps, SNMPv2c Traps, SNMPv2c Informs, SNMPv3 Traps, and SNMPv3 Informs.

12.2.6.2 traprcv Command Line

traprcv [-d]

12.2.6.3 traprcv Examples

To receive a trap, type the following command:
%traprcv

12.2.7 The trapsendagt Utility

12.2.7.1 trapsendagt Description

The trapsendagt utility is an EMANATE® Subagent that initiates the sending of Trap and Inform messages by the EMANATE® Master Agent.

This utility is provided only with products that include an EMANATE® Master Agent.

The primary purpose of the trapsendagt utility is to allow one to test the Master Agent's configuration to ensure that notifications initiated by Subagents will be delivered to all of the desired destinations.

12.2.7.2 trapsendagt Command Line

The trapsendagt command line:

```
usage: trapsend -smiv1 [ [Subagent options] [-help] ] \
        [-ctx contextName] generic_trap [specific_trap enterprise] \
        [time_ticks [variable_name type value ...]]

usage: trapsend -smiv2 [ [Subagent options] [-help] ] \
        [-ctx contextName] snmpTrapOID_value(OID) \
        [sysUpTime_value [variable_name type value ...]]
```

A typical Subagent is concerned only about the application that it manages. If the application triggers an alarm condition, the Subagent's responsibility is to initiate the sending of SNMP notification messages to any and all destinations that are configured in the Master Agent's local configuration datastore. It is not normal for a Subagent to have knowledge of the destination addresses of SNMP Managers that should receive the notifications. Consequently, the trapsendagt utility does not accept parameters that indicate the IP address of a SNMP Manager, the type of protocol message to send (i.e., Trap or Inform), or the protocol version to use for sending the notification (e.g., SNMPv1 or SNMPv3). Instead, all of this information comes from the Master Agent's configuration.

On the next page is a list of command line arguments for trapsendagt.

- -smiv1

  This option causes trapsendagt to interpret the arguments that follow as information from a SMIv1 MIB document, specifically, generic trap number, specific trap number, and enterprise.

  generic trap

  The generic trap number of SNMPv1 pre-defined traps.

  generic trap

  The specific trap number of SNMPv1 enterprise specific traps. This value is defined in the SMIv1 MIB document, and when it is used, generic trap must be equal to '6'.

  enterprise

The enterprise argument is the value defined by the ENTERPRISE clause in the SMIv1 MIB document for this SNMPv1 trap.

- -smiv2
  This option causes trapsendagt to interpret the argument that follows as information from an SMIv2 MIB document, specifically, the OBJECT IDENTIFIER value that identifies the notification to send.

  snmpTrapOID value(OID)

  The snmpTrapOID value(OID) argument is the OBJECT IDENTIFIER of the trap as defined by the SMIv2 MIB document. This argument will become the value of snmpTrapOID.0 in the Trap packet. Name forms, such as coldStart or myWonderfulEntTrap are permitted.

- time ticks
  The time ticks argument is the actual value of the variable sysUpTime.0.

- -ctx string
  By default, trapsendagt sends all notifications in the “default” context. If the user wishes to send a notification in a different context, the context is specified using the -ctx string argument where “string” is the contextName.

- -ipaddr ipv4 address
  This option is used to specify the IPv4 address of the machine where the EMANATE® Master Agent is running. If not specified, trapsendagt will attempt to connect the Master Agent running on the local host. The ipv4 address value should be specified as a string in the form “xxx.xxx.xxx.xxx.”

- -retry n
  If trapsendagt can not immediately communicate with the EMANATE® Master Agent, it will try again. This option changes the number of seconds between consecutive attempts to contact the Master Agent.

- -apnone
  This option causes trapsendagt to generate no log messages.

- -apall
  This option causes trapsendagt to generate all log messages.

- -aperror
  This option causes trapsendagt to generate error messages if any errors occur. This is the default.

- -apwarn
  This option causes trapsendagt to generate warning messages if anything unusual happens. This is the default.
- -aptrace
  This option causes trapsendagt to generate trace messages during miscellaneous “interesting” points of activity.
- -apemanate
  This option causes trapsendagt to generate log messages about any communication that occurs with the EMANATE® Master Agent.
- -help
  This option causes trapsendagt to print a brief description of the command-line arguments it understands, then exit.

12.2.7.3 trapsendagt Examples

Sending generic traps

The following command sends a warmStart notification (generic Trap number one)to all configured destinations. The output indicates that two notifications were actually sent. Both notifications were sent in Trap messages, and both notifications were sent to the local host (loopback address), because the Master Agent’s configuration indicates that this is what should be done to deliver the notification.

```
%trapsendagt -smiv1 1
trap sent to 127.0.0.1:0

trap sent to 127.0.0.1:0

snmpOutTraps.0 = 6
```

The following command is similar to the above command, except the SMIv2 OBJECT IDENTIFIER warmStart (1.3.6.1.6.3.1.1.5.2)is used to indicate what type of notification to send.

```
%trapsendagt -smiv2 warmStart

trap sent to 127.0.0.1:0
trap sent to 127.0.0.1:0
snmpOutTraps.0 = 8
```

The output of trapsendagt ends with the current value of the MIB object snmpOutTraps.0. This is a counter inside the Master Agent that increases by 1 every time a Trap or Inform message is sent by the Master

Agent. The count includes not only the notifications that are initiated by trapsendagt but also any other notifications initiated by other Subagents or by the Master Agent itself.

Sending enterprise specific traps

The following example is based on an Hewlett-Packard OpenView example MIB called the SURGE-PROTECTOR-MIB (an SMIv2 MIB document).

The following command sends a surgeBreakerAlarm notification to all configured destinations. The outgoing Trap and Inform messages include the MIB object sysUpTime.0 with a value of 4974123and also the MIB object surgeBreakerStatus with a value of open(2).

%trapsendagt -smiv2 surgeBreakerAlarm 4974123 surgeBreakerStatus.0 -i open trap sent to 127.0.0.1:0

trap sent to 127.0.0.1:0
snmpOutTraps.0 = 10

The OBJECT IDENTIFIERs for Trap OID, the MIB object names, and enumerated values may also be specified in numeric form:

%trapsendagt -smiv2 1.3.6.1.4.1.99.12.19.1.3.1.5 4974123 \
1.3.6.1.4.1.99.12.19.1.3.1.3.0-i 2
trap sent to 127.0.0.1:0
trap sent to 127.0.0.1:0
snmpOutTraps.0 = 12

12.3 Error Messages

This section briefly describes some of the error messages which can be generated by the HP OpenView NNM SPI for SNMPv3 utilities.

- Could not initialize ARL application.

  This message is only produced by some versions of the utilities.

BRASS™

In the version of the utilities designed to work with BRASS™, this message indicates that the utility could not establish communication with the SNMPv3 SPI Server. Most likely, the SNMPv3 SPI Server is simply not running. If the SNMPv3 SPI Server is running, the utility may not have permission to talk to the SNMPv3 SPI Server (on UNIX systems,

the utility may need to be run as root). It is also possible that the utility is trying to use the wrong socket (check environment variables).

Asynchronous Request Library

In the version of the utilities designed to work with the ARL, this message indicates that a failure occurred during startup. Most likely, a configuration file is missing or unreadable (check mgr.cnf and snmpinfo.dat).

- Failure in snmpinfo.dat

  This message indicates that there is a problem with the named configuration file. Check to see that there is a snmpinfo.dat file located in the default directory or in the directory indicated by the environment variable SR_MGR_CONF_DIR. If the file exists, ensure that the file is readable and contains the correct information.

- Cannot translate MIB variable

  This message indicates that the snmpinfo.dat configuration file does not contain OID translation information for the indicated MIB variable's English name.

- Error code set in packet - No such variable name

This message is generated to indicate which VarBind in the PDU caused an request to fail at the agent.

A Trademarks

Accelerated Technology, Microtec, Nucleus, XRAY, and VRTX are registered trademarks of Mentor Graphics Corporation.

Adobe, Adobe Acrobat, Adobe Acrobat Reader, PostScript are trademarks or registered trademarks of Adobe Systems, Incorporated.

AlphaStation, Compaq, DEC, DIGITAL, HP OpenView Network Node Manager, HP/UX, OpenVMS, VMS, and VT-100 are trademarks or registered trademarks of Hewlett-Packard Corporation.

A/UX, Apple, and AppleTalk are registered trademarks of Apple Computer, Incorporated.

CodeView, Microsoft, FrontPage, MS-DOS, Windows, Windows NT, and Windows Server are trademarks or registered trademarks of Microsoft Corporation.

CORBA is a registered trademark of Object Management Group, Inc. Cygwin and Red Hat are trademarks of Red Hat, Incorporated. DG/UX is a trademark of Data General Corporation.

eHealth is a trademark and SPECTRUM is a registered trademark of Concord Communications, Inc.

EmWeb is a trademark of Agranat Systems, Incorporated.

FreeBSD is a registered trademark of The FreeBSD Foundation.

Green Hills and INTEGRITY are registered trademarks and velOSity is a trademark of Green Hills Software, Inc.

Intel is a registered trademark of the Intel Corporation.

IPNET and IPLITE are trademarks or registered trademarks of Interpeak AB. Java, Solaris, and SunOS are trademarks of Sun Microsystems, Incorporated. LINUX is a registered trademark of Linus Torvalds.

Lucent and Lucent Technologies are registered trademarks of Lucent Technologies.

Motif, Nucleus, OSF/1, OSF/Motif, and UNIX are registered trademarks and The Open Group is a trademark and X Window System are trademarks of The Open Group.

MOTOROLA is a registered trademark of Motorola, Inc.

Glossary

An understanding of the following general terms will help in comprehending this manual:

Access Restriction

Access to the SNMP agents configured with a community string can be restricted based
upon the source address of requests. If address restriction is selected, only managers
included in the list of IP addresses will be able to query the agent using the new community string.

Access View

The access level determines the MIB objects that members (users/communities)of a security group can retrieve with SNMP Get requests or set with SNMP Set requests. There are two types of access views, read and write:

- The read access view determines objects that can be retrieved using SNMP Get requests.
- The write access view determines objects that can be set using SNMP Set requests. Administrative User

In order for EnterPol to configure SNMP agents through EnterPol modules like Simple Policy Pro or CIAgent Policy Pro, the EnterPol database must contain an administrative user that is already configured on the SNMP agent. This is because configuration is applied using SNMP set requests. This administrative user must have permission to set all MIB objects that may be set during the distribution of a policy. The Administration panels allow you to discover the administrative users in the EnterPol database.

AES

Advanced Encryption Standard. a symmetric 128-bit block data encryption technique developed by Belgian cryptographers Joan Daemen and Vincent Rijmen. The U.S government adopted the algorithm as its encryption technique in October 2000, replacing the DES encryption. The National Institute of Standards and Technology (NIST) of the U.S.

Department of Commerce selected the algorithm called Rijndael. AES is not available outside the United States due to export restrictions.

API

Application Programming Interface.

ASN.1

Abstract Syntax Notation One. A description language used to describe SNMP data types in a machine architecture-independent format.

authentication

Authentication within the context of SNMP means that the SNMP entity can assert with certainty that the purported sender of a message is in fact the sender of that message.

Authentication Passphrase

The authentication passphrase is used to calculate a unique authentication key for each
SNMP agent on which the new SNMPv3 user is configured. The authentication key verifies the authenticity of the SNMPv3 messages sent by a manager using the new SNMPv3 username. A passphrase is similar to a password except that spaces are acceptable.
Example:

auth password for DayShiftSupervisor

The best authentication passphrase is one that can not be guessed easily. Passphrases should be at lease eight characters long.

BER

Basic Encoding Rules. The Basic Encoding Rules describe how SNMP data should be encoded “on the wire” in such a way that machines with potentially very different architectures can understand it.

CMIP

The ISO-OSI network management protocol. The Common Management Information Protocol.

COEX

This is an informal reference to RFC3584, “Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework.”

Community

The term community refers to the SNMPv1 or SNMPv2c configured request name. A community is used when making SNMPv1 or SNMPv2c requests to a SNMP agent.

Configuration Policy

A configuration policy consists of one or more security groups and the users or communities assigned to those groups. When a security group is included in a policy, all the users or communities assigned to that group will be configured on the SNMP agent. Note: SNMP agents that will be configured using the new policy should support the security models and security levels defined for the security groups selected. For example, a policy containing a SNMPv3 user configuration should not be configured on an agent that supports only SNMPv1 and/or SNMPv2c.

connectionless protocols

Connectionless protocols allow packets between network correspondents to be routed
individually rather than through a pre-established “connection.” IP is such a connectionless protocol.

connection-oriented protocols

Connection-oriented protocols transmit packets between network correspondents along predetermined routes which are established at connection setup.

Context

Contexts are generally used when a SNMP agent has multiple subagents that support the same MIB. By making a request with a context the agent can correctly forward the request to the subagent that has registered for the context. Typically most MIB objects will be supported under the default SNMP context. A non-default context should only be specified if you know the SNMP agent being configured has MIB objects supported under a different context.

CVS

Concurrent Versions System. A front end to the revision control system (see RCS). CVS keeps a single copy of shared files in a source “repository”; it contains all the information to permit extracting previous software releases at any time based on either a symbolic revision tag, or a date in the past.

DES

Data Encryption Standard. An encryption algorithm often used as a privacy mechanism.

DNS

The Domain Name System. A networked database primarily used to identify mail handlers and to resolve IP addresses from symbolic names.

HMAC

Hash-based Message Authentication Codes. A mechanism (defined in RFC2085)for message authentication using cryptographic hash functions. HMAC can be used with any iterative cryptographic hash function, e.g., MD5, SHA-1, in combination with a secret shared key. The cryptographic strength of HMAC depends on the properties of the underlying hash function.

IESG

The Internet Engineering Steering Group. A standards body responsible for approving technology as Internet Standards.

IETF

The Internet Engineering Task Force. A standards body that forms Working Groups to develop technology for the Internet community. When a protocol is deemed ready to move forward in the standards process, the IETF sends its recommendations to the IESG.

instrumentation

Instrumentation refers to the system-dependent program code written by an agent developer to gather the information that can be accessed using SNMP. For example, the number of packets in and out of an interface must be counted in order that this information can be retrieved. The instrumentation does the counting.

IP

Internet Protocol. IP is a connectionless network-layer protocol.

ISO

International Standards Organization. A standards body responsible for many different kinds of standards. The ‘networking branch’ of standards is usually referred to as the OSI.

ISODE

ISODE is a freely available development environment created as a research tool and represents an effort to promote the use of the International Organization for Standardization (ISO)interpretation of open systems interconnection (OSI), particularly in the Internet and RARE research communities. For more information, see How to Manage Your Network Using SNMP: The Network Management Practicum by Marshall Rose and Keith McCloghrie. The full reference information is provided on page C-6.

Master Agent

The EMANATE® Master Agent. The EMANATE® architecture includes the Master Agent and zero to many Subagents. The Master Agent includes such things as authentication, privacy, packet receipt and sending, BER processing, Subagent management, and so forth.

As much as is possible, the difficult processing has been centralized in the Master Agent to leave the Subagents as simple as possible.

MD5

Message Digest Algorithm 5. A “fingerprinting” algorithm (defined in RFC1321)that is often used as an authentication mechanism. Using a shared secret, the recipient of a [SNMP] message can verify that the message was not altered “in flight.”

MIB

Management Information Base. Each SNMP agent implements a set of “managed objects.” These objects are described in MIB documents written in the ASN.1 data description language.

MIB family

For the purpose of writing method routines, SNMP variables are separated into families. A family consists of all of the leaf MIB variables with the same immediate parent node, or root (the Object Identifier without the instance information). For example, in MIB-II the following variables form a single family since they are all children of ifEntry (1.3.6.1.2.1.2.2.1):

ifIndex 1.3.6.1.2.1.2.2.1.1

ifDescr 1.3.6.1.2.1.2.2.1.2

...skipping entries between...

ifOutQLen 1.3.6.1.2.1.2.2.1.21

ifSpecific 1.3.6.1.2.1.2.2.1.22

Note that ifNumber (1.3.6.1.2.1.2.1) is also a member of the interfaces group, but it is not a member of the same family since it is not a child of ifEntry.

MIB view

A MIB view is a subset of MIB objects at a SNMP entity which can be managed. monolithic agent

A compile-time extensible SNMP agent. In contrast to a run-time extensible agent, a monolithic agent requires that new MIB objects be incorporated into the agent through recompilation and relinking. EMANATE®/Lite is an example of a monolithic agent.

Notification Targets

Notification targets are managers or agents that are selected to receive information from SNMP events. Notifications can be sent as either Traps or Informs. A Trap is a one way communication from an agent to a manager. An Inform contains the same information as a Trap, however, with an Inform the manager sends a verification response back to the agent. The user sending the notification must be assigned to a security group that has access to the notification OID and OID of other objects within the notification.

NVT ASCII

Network Virtual Terminal ASCII. A subset of the ASCII code defined by RFC854 for use with the telnet protocol. NVT ASCII consists of printable ASCII characters and selected control characters such as carriage-return.

OID

Object Identifier. Each object in a SNMP MIB has an associated Object Identifier which uniquely identifies the object in a global tree of objects.

OSI

Open Systems Interconnect. A set of networking standards endorsed by the ISO. privacy

Privacy within the context of SNMP means that the contents of a SNMP packet can be interpreted correctly only by the sender and intended recipient of the SNMP packet.

Privacy Passphrase

The privacy passphrase is used to calculate a privacy key which is used to encrypt SNMP messages sent by this user. Encryption of the SNMP message ensures privacy as the message is transmitted across the network. A passphrase is similar to a password except that spaces are acceptable. Example:
priv password for HelpDesk

The best privacy passphrase is one that can not be guessed easily. Passphrases should be at lease eight characters long.

RCS

Revision Control System. A system of managing multiple revisions of files. RCS is useful for text that is revised frequently, for example C programs, documentation, graphics, papers, and form letters.

RFC

Request for Comment. Documents maintained by the IETF standards body containing standards in various stages of completion. RFC documents are available via the Internet for no fee and in printed form for a nominal printing charge.

Security Group

The security group defines:

— The security model that will be supported (SNMPv1, SNMPv2c, or SNMPv3).

— The security level that will be supported (if SNMPv3 is the security model).

— The access level (or views)(read, write and notify permissions).

Users assigned to a security group take on the access limitations defined for the group. It is often useful to name a security group to indicate the access level. It is also helpful to create a security group for a specific level of access.

Examples:

Administrator
ShiftSupervisor
public

Multiple users can be assigned to the same security group, however, each user can only be assigned to one security group.

Security Level

SNMPv3 users can be configured to use one or more of the following security levels:

— Authentication with privacy

— Authentication without privacy

— No authentication and no privacy

For security levels that use authentication, an authentication protocol must be specified in the configuration entry. For security levels that use privacy, a privacy protocol must also be specified in the configuration entry. Support privacy if this user might be used to communicate sensitive information that should not be transmitted across the network as
plain text.

SHA-1

Secure Hash Algorithm. A “fingerprinting” algorithm (similar to MD5)that is often used as an authentication mechanism. Using a shared secret, the recipient of a [SNMP] message can verify that the message was not altered “in flight.”

SNMP

Simple Network Management Protocol. The specification for this Historic protocol is published in RFC1157.

SNMPv2c

Community-based SNMPv2. An Historic protocol published in RFC1901 which combines SNMPv2 operations (such as GetBulk)with SNMPv1 trivial authentication.

SNMPv2*

Simple Network Management Protocol version 2 “star”. An Historic proposed protocol (published as Internet Drafts)which predates SNMPv3 and should no longer be used.

SNMPv3

Simple Network Management Protocol version 3. The specification for this Full Standard protocol is published in RFC3410-3418. SNMPv3 provides a Full Standard administrative framework (authorization, access control, etc.)and remote configuration/remote administration MIB.

Subagent

An EMANATE® Subagent. See “Master Agent” for a description of the EMANATE ® architecture. A Subagent traditionally implements a single MIB document, such as the FDDI-MIB or the Host Resources MIB.

TCP

The Transmission Control Protocol is a connection-oriented transport-layer protocol. It attempts to achieve reliability through retransmission.

Triple-DES or 3DES

The 3DES-EDE privacy protocol(3DES)is an extension to the User-based Security Model(USM). The designated portion of a SNMP message is encrypted and included as part of the message sent to the recipient.

UDP

The User Datagram Protocol is a connectionless end-to-end transport-layer protocol.

User

The term user refers to the SNMPv3 USM users configured on a SNMP agent. The user is used when making a SNMPv3 request to an agent that supports SNMPv3. A user is set up to support one or more of the following security levels:

— Authentication with privacy

— Authentication without privacy

— No authentication and no privacy

The username can be the name of a person, group or management application. For example:

Administrator
public
EnterPol
VarBind

A SNMP variable binding. A VarBind includes an OBJECT IDENTIFIER and a value (which may be NULL).

Index

B

C

D

G

O

P

R

S

T

U

V

W