IBM Tivoli Netcool/System Service Monitors
Version 4.0.1
for AIX, HP-UX, Linux, Solaris, and Windows

Administration Guide

SC23-8761-02

Note

Before using this information and the product it supports, read the information in"Notices and trademarks" on page 171.

This edition applies to version 4.0.1 of Netcool/SSM (product numbers 5724-P39, 5724-P40, 5724-P41, 5724-P43) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Figures

Tables

About this guide

This guide describes how to install, administer, and use Netcool/SSM and the Netcool/ASM suite of products.

Who should read this guide

This guide is intended for network administrators and engineers who install and use Netcool/SSM to monitor networks and hosts. It provides detailed, cross-platform information about the tools, functions, and capabilities of Netcool/SSM. Use this guide to assist you in designing and configuring your network management and monitoring environment.

To use Netcool/SSM effectively, and to understand the information in this guide, you should already be familiar with network technologies, network management practices, and the Simple Network Management Protocol (SNMP).

Publications

This section lists publications in the Netcool/SSM library. It also describes how to access Tivoli publications online and how to order them.

Documentation library

The following documents are available in the Netcool/SSM library:

- *Netcool/SSM Administration Guide*

  Provides information about installing and using Netcool/SSM.
- *Netcool/SSM Reference Guide*

  Provides detailed reference material covering the subagents and MIB modules included in Netcool/SSM.
- *Netcool/SSM Patch Installation Guide*

  Provides instructions on installing patches to Netcool/SSM.
- *Netcool/SSM Release Notes*

  Provides the latest information about Netcool/SSM.

Standards

Netcool/SSM and Netcool/ASM MIB modules are based on the Internet standard Network Management Framework which consists of the following components:

- RFC 2578 which defines the SMI [SMIv2], the mechanisms used for describing and naming objects for the purpose of management.
- RFC 1213 which defines MIB-2, the core set of managed objects for the Internet suite of protocols.
- RFC 1905 which defines the SNMP [SNMPv2], the protocol used for network access to managed objects.

Another important component is RFC 2021 (RMON2) the network monitoring Internet standard on which Netcool/SSM is based. RMON2 is a MIB that defines objects for managing remote network monitoring devices over TCP/IP networks.

Accessing terminology online

The *Tivoli Software Glossary* includes definitions for many of the technical terms related to Tivoli software. The *Tivoli Software Glossary* is available at the following Tivoli software library Web site:

http://publib.boulder.ibm.com/tividd/glossary/tivoliglossarymst.htm

The IBM Terminology Web site consolidates the terminology from IBM product libraries in one convenient location. You can access the Terminology Web site at the following Web address:

http://www.ibm.com/ibm/terminology

Accessing publications online

IBM posts publications for this and all other Tivoli products, as they become available and whenever they are updated, to the Tivoli software information center Web site. Access the Tivoli software information center by first going to the Tivoli software library at the following Web address:

http://www.ibm.com/software/tivoli/library

Scroll down and click the **Product manuals** link. In the Tivoli Product Documents Alphabetical listing window, click **M** to access all IBM Tivoli Monitoring product manuals.

Note: If you print PDF documents on other than letter-sized paper, set the option in the File -> Print window that allows Adobe Reader to print letter-sized pages on your paper.

Ordering publications

You can order many Tivoli publications online at the following Web site:

IBM Publications Center(`http://www.elink.ibmlink.ibm.com/public/applications/publications/cgibin/pbi.cgi`)

You can also order by telephone by calling one of these numbers:
- In the United States: 800-879-2755
- In Canada: 800-426-4968

In other countries, contact your software account representative to order Tivoli publications. To locate the telephone number of your local representative, perform the following steps:
1. Go to Directory of worldwide contacts(`http://www.ibm.com/planetwide/`).
2. Select the letter that your country starts with and click the name of your country. A list of numbers for your local representatives is displayed.

Tivoli technical training

For information about Tivoli technical training, refer to the following IBM Tivoli Education Web site:

http://www.ibm.com/software/tivoli/education

Support information

If you have a problem with your IBM software, you want to resolve it quickly. IBM provides a number of ways for you to obtain the support you need.

- Searching knowledge bases: You can search across a large collection of known problems and workarounds, Technotes, and other information.
- Obtaining fixes: You can locate the latest fixes that are already available for your product.
- Contacting IBM Software Support: If you still cannot solve your problem, and you need to work with someone from IBM, you can use a variety of ways to contact IBM Software Support.

Conventions used in this guide

This guide uses several conventions for operating system-dependent commands and paths, special terms, actions , and user interface controls.

Typeface conventions

This guide uses the following typeface conventions:

Bold

- Lowercase commands and mixed case commands that are otherwise difficult to distinguish from surrounding text
- Interface controls (check boxes, push buttons, radio buttons, spin buttons, fields, folders, icons, list boxes, items inside list boxes, multicolumn lists, containers, menu choices, menu names, tabs, property sheets), labels (such as **Tip:**, and **Operating system considerations**:)
- Keywords and parameters in text

Italic

- Words defined in text
- Emphasis of words (words as words)
- New terms in text (except in a definition list)
- Variables and values you must provide

`Monospace`

- Examples and code examples
- File names, programming keywords, and other elements that are difficult to distinguish from surrounding text
- Message text and prompts addressed to the user
- Text that the user must type
- Values for arguments or command options

SNMP naming conventions

Table 1 lists definitions for terms that are commonly used when working with the SNMP features provided in Netcool/SSM.

Table 1. Common terms

Term	Meaning
MIB	An acronym for Management Information Base. The MIB provides a structured set of data variables, or objects, that represent the resources to be managed. The MIB consists of a set of MIB modules that add information to it.
object	A data variable.
OID	An object identifier that uniquely identifies an object within the MIB.
subagent	A module that implements a part of the network management and monitoring functions provided by Netcool/SSM.
trap	An unsolicited message sent by the subagent to notify the management station of an event. Traps are also known as notifications.
tree	Shows the hierarchical structure between related objects.

MIB object names

In this guide, MIB objects are referred to by an abbreviated object name or by the name of the function they provide. The object name usually has a prefix that specifies the MIB module in which the object appears, while the remainder of the object name usually describes its function.

For example, in the object name `sysObjectID`:

- The prefix is `sys`
- The abbreviated object name is `ObjectID`
- The function is object identifier

The function name has initial capitals in normal body type with a space between words, whereas the object name is shown as a single word in `monospace` font. Object names are taken directly from MIB module definition documents.

Operating system considerations

All command line formats and examples are provided for both the standard UNIX shell and the Windows command-line. UNIX is case-sensitive. You must type commands in the case shown in the book.

Chapter 1. Introduction

Netcool/SSM provides real-time monitoring for systems, applications, and networks. It is a key technology in maintaining overall performance and availability of business and application services.

Netcool/SSM is based on the Simple Network Management Protocol (SNMP) and implements the MIB-II, RMON and RMON2 network monitoring standards.

Netcool/SSM is a rich data source for network management systems such as Netcool/OMNIbus that provide performance evaluation, reporting, event management, and fault resolution tools. You can configure Netcool/SSM to send event-related data to management systems in the form of SNMP notifications, or use the management system to poll Netcool/SSM for performance data using SNMP requests.

Netcool/SSM supplements the RMON network monitoring standards, providing a range of additional features including:

- Network route-tracing and security monitoring
- System resource usage data including CPU, memory and disk utilization statistics
- System process monitoring, including facilities for starting and stopping processes as required
- Application availability, response and usage data, including automated client-server and application discovery
- Sophisticated performance thresholds and alarm generation facilities
- Scheduled job execution and scripting capabilities

Netcool/SSM includes the Netcool/ASM suite of monitors, which offer detailed monitoring capabilities for leading commercial server and database products. The Netcool/ASM suite installs and runs as an integral component of Netcool/SSM.

Netcool/SSM installs directly on the systems that you wish to monitor. It runs on a variety of UNIX, Linux and Windows platforms.

Netcool/SSM components

Netcool/SSM consists of a set of functional components.

The Netcool/SSM components are:

- The Netcool/SSM master agent.
- Netcool/SSM subagents for performance monitoring, including the Netcool/ASM suite of monitors.
- Administrative tools for configuring and controlling Netcool/SSM.

Netcool/SSM master agent

The Netcool/SSM master agent, known more simply as the *agent*, forms the basis of Netcool/SSM. It is a lightweight SNMP agent that installs on the workstations and servers that you wish to monitor in your enterprise.

Netcool/SSM subagents and MIB modules

Netcool/SSM subagents and their MIB modules implement the system service monitoring capabilities provided by Netcool/SSM.

Each MIB module defines the set of performance metrics gathered; subagents are binary implementations of the SNMP interface, control facilities and performance metrics defined in the associated MIB module. Netcool/SSM subagents load and run on the master agent.

Netcool/ASM suite of monitors

The Netcool/ASM suite of monitors provides performance monitoring for a variety of commercial server and database products.

Each Netcool/ASM is a subagent that loads and runs just like any Netcool/SSM subagent. Evaluation versions of the Netcool/ASM monitors are installed together with Netcool/SSM.

The information in this Administration Guide applies to both Netcool/SSM and Netcool/ASM.

Netcool/SSM administrative tools

Netcool/SSM includes a set of administrative tools that configure and control the agent.

Command console

The command console provides a command line interface to the Netcool/SSM agent. You can use the console to connect to any Netcool/SSM agent installed on your network.

Service Controller

The Service Controller is a simple control panel for starting and stopping the Netcool/SSM agent, and launching the command console. This component is optionally installed, allowing you to include it on your test systems, but exclude it from the target monitored systems. It is only available on Windows platforms.

MIB Explorer

MIB Explorer is a graphical tool that provides a subset of the control and configuration functions available in the command console. It allows you to connect to Netcool/SSM agents installed on network nodes, view MIB data in a hierarchical, tree-based graphical structure and create simple charts based on MIB data.

With this tool you can also monitor notifications generated by the Netcool/SSM agents on your network. It is only available on Windows platforms.

Service monitoring with Netcool/SSM

Netcool/SSM acts a key data source in the service monitoring process, providing performance metrics and alarms for key system and applications on the monitored host.

Figure 1 demonstrates a typical service monitoring application, showing the operation of Netcool/SSM and its interaction with a management station.

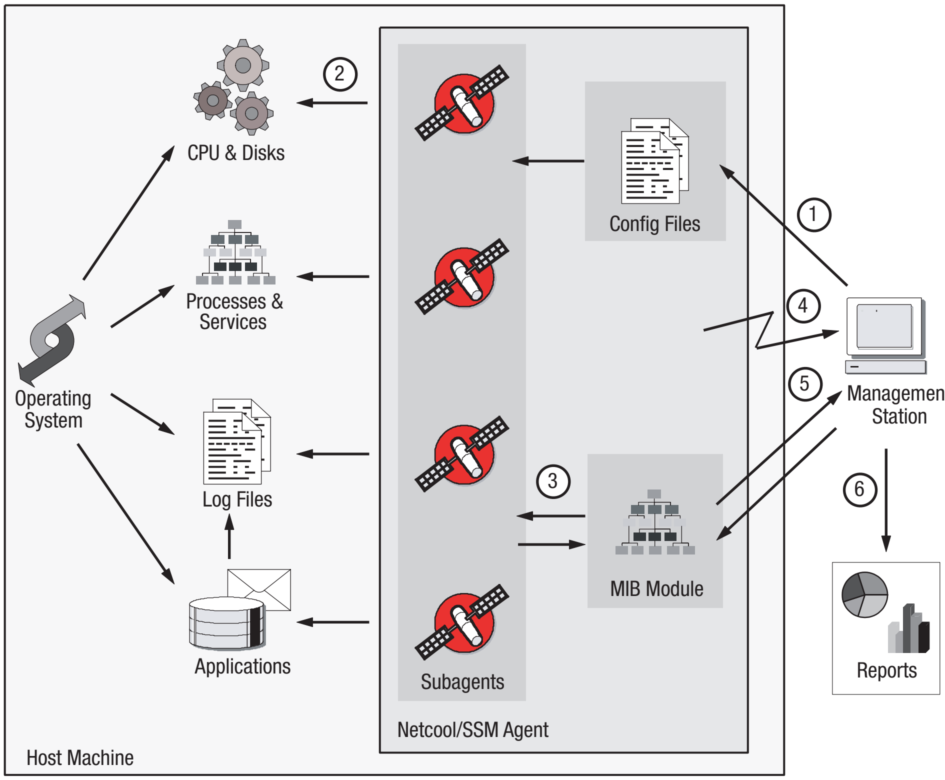


Figure 1. Service monitoring using Netcool/SSM

A typical service monitoring process using Netcool/SSM involves the following operations:

1. A Management Station distributes configuration and state files to the Netcool/SSM agent. These files load and configure the appropriate subagents for the service monitoring tasks.
2. Netcool/SSM subagents monitor system components such as CPUs, disks, processes, services, log files, and applications according to the defined service monitoring tasks.
3. Netcool/SSM subagents update their MIB modules in the Netcool/SSM MIB with metrics obtained from the monitored system components.
4. Netcool/SSM agent and subagents send notifications to the Management Station in response to system events, performance threshold violations and other alarm conditions.
5. The Management Station polls the Netcool/SSM agent using SNMP requests to obtain performance data and metrics from the Netcool/SSM MIB.
6. The Management Station, or a related tool, produces reports and event management data for evaluation and actioning.

Chapter 2. Installation

Read this information before deploying Netcool/SSM on your enterprise networks. Separate installation procedures are provided for installing Netcool/SSM on Windows and UNIX platforms.

Deploying Netcool/SSM

This section presents a number of important topics that you may wish to consider before deploying Netcool/SSM on the servers and workstations in your enterprise.

Target systems

Netcool/SSM installs directly on target systems. Install Netcool/SSM individually on each workstation or server machine that you wish to monitor.

Netcool/SSM runs on a range of Microsoft Windows and UNIX operating systems; for details of these, see “Installing Netcool/SSM on Windows” on page 8 and “Installing Netcool/SSM on UNIX” on page 20.

Installation methods

The Netcool/SSM installer provides both interactive and unattended installation methods.

- Interactive installation

  In interactive mode, the installer guides you through the installation and you provide data when prompted. This option requires your active input throughout the installation process.
- Unattended (or silent) installation

  In unattended mode, you pass parameters to the installer through command line switches or an installation response file. This method is particularly useful if you wish to install Netcool/SSM on multiple hosts using the same configuration.

Installation response files

Installation response files specify installation parameters used by the Netcool/SSM installer.

In an unattended installation, the contents of the installation response file provide the parameters for the installation. In an interactive installation, the contents of the installation response file provide the default parameters presented by the installer in each step of the installation process.

You record installation response files using the Netcool/SSM installer. These files are ASCII text files; if necessary you can edit a response file to customize responses to installation steps as required.

Tip: To perform a silent installation on a Windows server cluster using an installation response file, record the response file on one of the cluster nodes.

Upgrading an existing Netcool/SSM installation

If you wish to upgrade the version of Netcool/SSM currently installed on a machine, it is not necessary to uninstall the existing version.

The installer automatically detects the existing Netcool/SSM version and modifies only those files required to upgrade Netcool/SSM.

During installation, the installer checks whether Netcool/SSM is already installed on the host machine. If it finds an existing installation, it shuts down any Netcool/SSM processes running before performing the new installation.

Attention: If the installer stops the processes on an existing Netcool/SSM installation, all MIB data collected by that installation will be lost.

Retaining existing configuration

When upgrading an existing installation, you can retain the configuration of that installation.

To retain the configuration of the existing installation during upgrade, run the installer with the `NoReconfig` parameter on Windows systems, or the `INST_RETAIN` parameter on UNIX systems. This parameter instructs the installer to use the configuration of the existing installation.

Tip: You cannot use the installer to revert to a previous version of Netcool/SSM. To do so, you must uninstall the current version then reinstall the old version.

Remote application monitoring

Some monitors in the Netcool/ASM suite support remote application monitoring, which enables you to use the Netcool/SSM installed on one machine to monitor an application running another machine.

This capability is not provided by all Netcool/ASMs, so if you are deploying Netcool/SSM to perform remote application monitoring, check the Netcool/ASM documentation for details about its capabilities. If remote monitoring is not supported, you must install Netcool/SSM on the machine running the application that you wish to monitor.

Port usage

Netcool/SSM uses a UDP port for SNMP communication. You must select the port number for Netcool/SSM during installation.

If another SNMP agent on the host machine is already using the port, do one of the following:

- Disable the other SNMP agent before installing Netcool/SSM.
- Configure the other SNMP agent to use a different port.
- During installation, configure Netcool/SSM to use a different port.

Firewall configuration

If a firewall separates a management station from the machines on which Netcool/SSM agents are running, you must ensure that the firewall is configured to permit communication between the management station and the agents.

The firewall configuration required is:

- Allow incoming and outgoing traffic on the UDP port on which the agent receives and responds to SNMP requests. By default this is port 161.
- Allow outgoing traffic on the UDP port used by Netcool/SSM to send notifications. Destination UDP ports are specified in notification target definitions.
- Allow incoming and outgoing traffic on any TCP port if you use the Netcool/SSM command console to remotely connect to Netcool/SSM agents. You may block any incoming SYN connections because TCP connections may only be created by the agent.

  Note: When you open the command console, it sends an initial connection request to the agent on the UDP port used for SNMP requests. If the agent accepts the connection request, it opens a TCP connection for the console session on any available TCP port. For this reason it is not necessary to allow incoming TCP SYN connections.

Figure 2 summarizes the firewall configuration requirements.

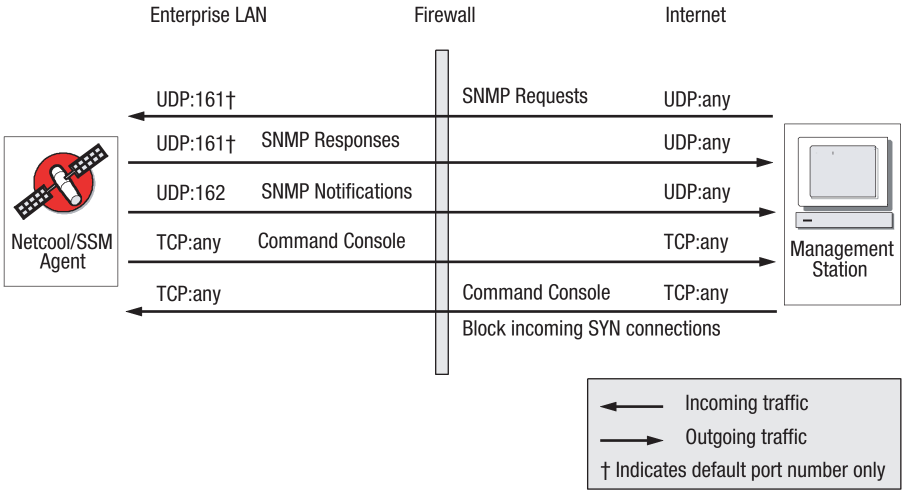


Figure 2. Firewall configuration requirements

Installing Netcool/SSM on Windows server clusters

To install Netcool/SSM on a Windows server cluster, run the installation process on each node (that is, on each machine in the cluster).

Security privileges on Windows

On Windows platforms, the types of operation performed by Netcool/SSM require security privileges that permit access to low-level information on the host machine. By default, the installer configures Netcool/SSM to run under the *Local System* account, which provides the required privileges.

The following functional aspects of Netcool/SSM demand this level of security:

- Promiscuous packet capture, which requires the installation of low-level packet capture drivers.
- Collection of data from hardware devices such as CPUs, disk drives, network interfaces, and file systems via IOCTL calls.
- Process monitoring and control, which requires the ability to terminate arbitrary processes.
- Service control, which requires the ability to start and stop services on the host machine and to restart services when stopped.

Netcool/SSM creates the following registry key for the collection of network statistics if necessary:

```
HKLM\SYSTEM\CurrentControlSet\Services\SNMP\Parameters\RFC1156Agent\
sysServices
```

Some of the privileges required by Netcool/SSM cannot be granted to non-administrative users, which means that it must run as a user with privileges at least equivalent to those of the *Local System* account in order to function properly.

Installing Netcool/SSM on Windows

Read this section for information about how to install, upgrade and uninstall Netcool/SSM on machines running Windows operating systems. It also provides detailed information about hardware and software required to successfully install and run Netcool/SSM.

System requirements

Before installing Netcool/SSM, you need to ensure that the host machine satisfies the minimum hardware and software requirements.

Hardware

The hardware requirements for Windows platforms are:

- Intel Pentium II processor (400 MHz or greater)
- 128 MB RAM or greater recommended
- Approximately 25 MB free disk space
- 50MB minimum swap space recommended
- Ethernet 10/100 network interface card capable of supporting promiscuous packet reception.

Software

The software requirements for Windows platforms are:

- Microsoft Windows 2008, x86 and x64 Editions
- Microsoft Windows Vista, x86 and x64 Editions
- Microsoft Windows 7, x86 and x64 Editions
- Microsoft Windows 8, x86 and x64 Editions
- Microsoft Windows 2008 R2, x86 and x64 Editions
- Microsoft Windows Server 2012, x64 Edition

Third-party SNMP services

If the target system is running a third-party SNMP service, it may be necessary to disable that service before installing Netcool/SSM or configure Netcool/SSM to operate alongside it.

Disabling Microsoft SNMP Service

If you wish to assign a port number to the Netcool/SSM agent that is already in use by Microsoft SNMP Service then you must disable this service or reconfigure it to use another port.

To disable a Microsoft SNMP Service from your Windows desktop:

1. Open **Services**.

   Select the SNMP service and click **Startup**.
2. The Startup dialog is displayed.

   Set the `Startup Type` to `Disabled`.
3. Click **OK**.

   The Microsoft SNMP Service is now disabled.

Integration with HP Systems Insight Manager

Normally, the Netcool/SSM installer does not attempt to modify or integrate with any other SNMP services already installed on the host machine, however you can instruct the installer to integrate Netcool/SSM with HP Systems Insight Manager on Windows platforms.

To integrate Netcool/SSM with HP Systems Insight Manager, use the `--OverrideSNMP` switch when running the installer. For more detailed information about the integration process see “Integrating with HP Systems Insight Manager (Windows)” on page 154.

Interactive installation

In an interactive installation, the Netcool/SSM installer guides you through the installation process. This process consists of a number of steps in which you enter details for the installation.

Installation procedure

Follow these steps to install Netcool/SSM.

Start the installer:
Procedure

Start the installer in either of the following ways:

- To start the installer from CD-ROM, place the Netcool/SSM CD-ROM in your CD-ROM drive. The Netcool/SSM installer starts automatically.
- To start the installer from the command line, run the following command: `netcool-ssm-4.0.1-xxx.win32.exe`. The `x` characters represent the build number of your version of the Netcool/SSM installer. The command line supports a number of switches. For more details on these switches, see "Installer command line switches" on page 14.

Welcome:
Procedure

The Welcome dialog does not require any input.

Click **Next** to proceed.

License Agreement:
Procedure

The License Agreement dialog displays the Netcool/SSM license agreement.

Read the license agreement. If you accept the terms of the license agreement and wish to install Netcool/SSM click **Yes**.

Note: If you do not wish to accept the license agreement you cannot install Netcool/SSM.

Important Information:
Procedure

The Important Information dialog provides important release information specific to your version of Netcool/SSM.

Read the information then click **Next**.

Registration information:
Procedure

The Registration Information dialog records the owner of the Netcool/SSM product.

Enter the name of the person and company to which your version of Netcool/SSM is licensed and click **Next**.

Setup type (not for upgrade):

Use the Setup Type dialog box to select the type of setup that you wish to perform. The Complete setup installs all the Netcool/SSM components listed in the table below. The Custom setup enables you to select the components that you wish to install.

Procedure

Select a setup type and click **Next**.

Feature selection (not for complete setup type or upgrade):

About this task

Use the Feature Selection dialog to select the Netcool/SSM components that you wish to install and, if required, to specify the Netcool/SSM installation directory. This dialog is only displayed if you chose the Custom setup in the step "Setup type (not for upgrade)."

Procedure

1. Select the components that you wish to install.

Table 2. Netcool/SSM components

Component	Description
Core Agent Files	Files that implement the core Netcool/SSM functions. This component is always installed.
MIB Explorer	Graphical tool for polling SNMP agents and viewing MIB data.
Text Console	Command console for querying and configuring SNMP agents via simple command line operations.
Service Controller	Control interface for running Netcool/SSM as a Windows service.
Desktop Help	Simple tool allowing users of the host machine to send queries to a help desk via SNMP.

   To change the installation folder for Netcool/SSM, click **Browse** and select a folder.
2. Click **Next**.

SNMP settings (SNMP v1/v2c):

Use the SNMP Settings dialog to specify the SNMPv1/v2c settings for Netcool/SSM.

Procedure

1. In the **Port** field, enter the UDP port on which Netcool/SSM listens for incoming SNMP requests. This port must be a number in the range 1-65535 inclusive and must not be in use by any other application. If you specify port 161 and the Microsoft SNMP service is running on the host machine, conflict may arise between the Netcool/SSM agent and the Microsoft SNMP service. In this case either disable the Microsoft SNMP service, reassign it to another port or specify another port for Netcool/SSM. You can change the port used by Netcool/SSM again later if necessary. For information about disabling the Microsoft SNMP service, see "Disabling Microsoft SNMP Service" on page 9.

2. Select or deselect the **Enable** check boxes to specify the type of SNMP requests to which Netcool/SSM will respond. Deselecting a check box indicates that Netcool/SSM will not respond to requests of that type.
3. Use the **v1/v2c Settings** group to assign default module access privileges for communities. For each community used in your network management environment, enter its name in the field and assign it an access privilege using the **Read Only** and **Read/Create** buttons. See "Access control using communities" on page 97 for more details about module access.
4. Click **Next**.

SNMP settings (SNMP v3):

Use the SNMP Settings dialog to specify the SNMPv3 settings for Netcool/SSM.

Procedure

1. Select or deselect the **Enable v3** check box to specify whether Netcool/SSM will respond to SNMPv3 requests. If the check box is cleared, Netcool/SSM will not respond to SNMPv3 requests.
2. If you enable SNMPv3, enter information in the fields in the **v3 Settings** group.

Table 3. SNMPv3 Settings

Field	Description	
Username	Sets the username used by Netcool/SSM in SNMPv3 communication.	
Authentication	Type	Sets the method of authentication used for the SNMP v3 protocol. Valid values are: `NONE` - no authentication `SHA` - Secure Hash algorithm `MD5` - MD5 message digest algorithm If you select a value other than `NONE`, you must also specify a password.
	Password	Sets the password used in SNMPv3 authentication. This password must be at least 8 characters in length and may contain spaces.
	Confirm	A confirmation field for the password. This field and the **Authentication Password** field must be identical for the password to be valid.
Privacy *If you specify these settings, you must also specify the Authentication settings.*	Privacy	Sets the method used for encrypting SNMP v3 protocol messages. Valid values are: `NONE` - no encryption used `DES` - Data Encryption Standard algorithm If you set this parameter to a value other than `NONE`, you must also specify a password.
	Password	Sets the password used in SNMPv3 encryption. This password must be at least 8 characters in length and may contain spaces.
	Confirm	A confirmation field for the password. This field and the **Privacy Password** field must be identical for the password to be valid.

3. Click **Next**.

Confirm installation:

The Start Copying Files dialog box displays a summary of the files that will be installed.

Procedure

- To begin installing these files on the host machine, click **Next**.
- To make changes to the installation options before beginning, click **Back**.

Installation complete:

The installer displays the Installation Complete dialog box after the installation process has finished.

Procedure

Click **Finish**.

Restart:

The Netcool/SSM installation is now complete.

Procedure

Restart the host machine if requested. This ensures that any shared files required by Netcool/SSM are updated on the host machine.

Unattended (or silent) installation

The unattended (or silent) installation method uses the same installer as the interactive installation method; however, no interaction with the installer is required on your part. In an unattended installation, you supply the installer with installation options using an installation response file.

Tip: The unattended installation mode is particularly useful for installing Netcool/SSM on a large number of machines using the same installation parameters on each machine.

Performing an unattended installation

Unattended installation runs silently from the command line.

About this task

To perform an unattended installation, run the installer with the `/s` and `/f1` switches:

```
netcool_ssm-4.0.1-xxxx-win32.exe /s /f1"resp_file"
```

The `/f1` switch specifies the name and path of the installation response file. For more details on the command line switches, refer to "Installer command line switches" on page 14.

Note: During an unattended installation, the installer suppresses error messages relating to the creation of the service entry and service user.

Installer command line switches

The Netcool/SSM installer supports a number of command line switches that control the way the installer operates.

The format for the command switches is as follows:

```
netcool_ssm-4.0.1-xxxx-win32.exe {/s /f1"resp_file"} | {/r /f1"resp_file"}
[/z"params"] [/f2"log_file"]
```

Table 4 describes each switch.

Table 4. Netcool/SSM installer switches

Switch	Function
/s	Selects silent (unattended) installation. Omitting this switch runs the interactive installation mode. If you use this switch, you must also use the /f1 switch to specify a response file.
/r	Selects installation record mode. Use this mode to record an installation response file. If you use this switch, you must also use the /f1 switch to specify a response file. For more details see "Recording an installation response file" on page 19.
/z	Specifies a list of installation parameters. For more details see "Installation parameters."
/f1	Specifies the full path and filename of the installation response file to be used in the installation process. In an unattended installation, the contents of the installation response file provide the parameters for the installation. In an interactive installation, the contents of the installation response file provide the default parameters presented by the installer in each step of the interactive installation process. For more details see "Recording an installation response file" on page 19.
/f2	Specifies the full path and filename of the file in which the installer logs information about the installation process. Installation log files are useful in diagnosing any errors that occur during installation.

The following example runs an unattended installation using the response file setup.iss from the directory c:\install and logs information about the installation process in the setup.log file:

```
netcool_ssm-4.0.1-xxxx-win32.exe /s /f1"c:\install\setup.iss"
/f2"c:\install\setup.log"
```

Installation parameters

To pass installation parameters to the installer from the command line, use the /z switch.

The format for parameters passed with this switch is as follows:

```
/z"--param[=value] ..."
```

For example, the following command shows how to specify the username and password for registering Netcool/SSM during installation:

```
netcool_ssm-4.0.1-xxxx-win32.exe /z"--SvcUser=agentUser --SvcUserPassword=password"
```

The following rules apply to parameters specified with the /z switch:

- The list of parameters must be enclosed in one set of double quote characters ("). For example:

  ```
  /z"--SvcUser=agentUser --SvcUsePassword=password"
  ```

- Parameter values may contain spaces but may not contain double quote characters (").
- If you specify a value for a parameter that does not require one, such as `OverrideSNMP`, the installer ignores the value but the parameter itself still has effect.
- Command line options are not case-sensitive.

Note: You can also use these parameters when running the installer in the interactive installation mode. Any parameters supplied in this way appear as the default values in the corresponding installer dialogs.

Table 5 lists the installation parameters available for use with the `/z` switch. You can change some of these parameters again after Netcool/SSM is installed: where appropriate, the table indicates the command or variable for changing a setting after installation.

Table 5. Installation parameters

Parameter	Type	Description	*Variable*/ **Command**
`SvcUser`	string	Sets the username used to register the Netcool/SSM service. If you specify this switch, you must also specify the `SvcUserPassword` parameter. On host machines with Active Directory installed (Windows domain controllers) this parameter is ignored and the local system account is used instead. Default: local system account.	n/a
`SvcUserPassword`	string	Sets the password used to register the Netcool/SSM service. If you specify this parameter, you must also specify the `SvcUser` switch. On host machines with Active Directory installed (Windows domain controllers) this parameter is ignored and the local system account is used instead.	n/a
`UDPPort`	integer	Sets the UDP port number on which Netcool/SSM listens for incoming SNMP requests. The port number must be a number in the range 1-65535 inclusive. The port specified must not be in use by any other application. If the Microsoft SNMP service is installed on the host machine and you set this parameter to 161 then the actions taken by the installer depend on the `OverrideSNMP` parameter. If `OverrideSNMP` is specified, the installer stops the Microsoft SNMP service, sets its startup mode to manual, installs the Netcool/SSM service and registers the Netcool/SSM service in automatic startup mode. If `OverrideSNMP` is not specified, the installer leaves the Microsoft SNMP service running and registers the Netcool/SSM service in manual startup mode. Default: `161`	*UdpPort*

Table 5. Installation parameters (continued)

Parameter	Type	Description	*Variable/* **Command**
`Community`	string	Sets the default community string for Netcool/SSM. The community string is only used if the agent has either SNMP v1 or SNMP v2c enabled. Default: `private`	`community`
`TargetDir`	string	Sets the directory in which Netcool/SSM will be installed. Default: On Windows 32-bit systems, *`drive:`*`\Program Files\Netcool\SSM` On Windows 64-bit systems, `\Program Files(x86)\Netcool\SSM`	n/a
`Force`	n/a	Forces the installation process. In a forced installation, the installer does not migrate the configuration of any Netcool/SSM currently installed on the host machine to the new version, destroying any existing configuration files. *The agent's unique ID is always preserved, regardless of the state of the Force parameter.* Default: Not forced. The installer preserves the configuration of any Netcool/SSM already installed on the host machine.	n/a
`Version`	n/a	Displays the product name and version of the installer. When you specify this parameter, the installer does not perform any other installation tasks.	n/a
`OverrideSNMP`	n/a	Disables the Microsoft SNMP service if it is currently installed on the host machine and will conflict with Netcool/SSM.This parameter only has effect if either of the following criteria are met: • Netcool/SSM is configured to use UDP port 161. In this case, the installer stops the Microsoft SNMP service, installs Netcool/SSM and starts the Netcool/SSM service. • The host machine is running HP Systems Insight Manager services and Netcool/SSM is configured to use UDP port 161. In this case, the installer modifies the Microsoft SNMP service to use Netcool/SSM instead of the Microsoft service. If the parameter is not specified and the host machine is currently running the Microsoft SNMP service, the installer does not modify it in any way. If Netcool/SSM is configured to use UDP port 161, the installer sets the startup mode of the Netcool/SSM service to manual. Otherwise it sets the startup mode to automatic and both Netcool/SSM and the Microsoft SNMP service will run simultaneously. Default: The Microsoft SNMP service is not modified in any way.	n/a

Table 5. Installation parameters (continued)

Parameter	**Type**	**Description**	*Variable*/ **Command**
`DisableV1`	n/a	Configures Netcool/SSM not to respond to SNMP v1 requests. This parameter and the `EnableV1` parameter are mutually exclusive. You may specify only one of these two parameters. Default: The agent responds to SNMP v1 requests.	*DisableV1*
`EnableV1`	n/a	Configures Netcool/SSM to respond to SNMP v1 requests. This parameter and the `DisableV1` parameter are mutually exclusive. You may specify only one of these two parameters. Default: The agent responds to SNMP v1 requests.	
`EnableV2`	n/a	Configures Netcool/SSM to respond to SNMP v2c requests. This parameter and the `DisableV2` parameter are mutually exclusive. You may specify only one of these two parameters. Default: The agent responds to SNMP v2c requests.	*DisableV2*
`DisableV2`	n/a	Configures Netcool/SSM not to respond to SNMP v2c requests. This parameter and the `EnableV2` parameter are mutually exclusive. You may specify only one of these two parameters. Default: The agent responds to SNMP v2c requests.	
`EnableV3`	n/a	Configures Netcool/SSM to respond to SNMP v3 requests. This parameter and the `DisableV3` parameter are mutually exclusive. You may specify only one of these two parameters. If you specify this parameter, you must also specify the `SecName` parameter. Default: The agent does not respond to SNMP v3 requests.	*DisableV3*
`DisableV3`	n/a	Configures Netcool/SSM not to respond to SNMP v3 requests. This parameter and the `EnableV3` parameter are mutually exclusive. You may specify only one of these two parameters. Default: The agent does not respond to SNMP v3 requests.	

Table 5. Installation parameters (continued)

Parameter	Type	Description	*Variable/* **Command**
SecName	string	Sets the username used by Netcool/SSM in SNMPv3 communication. This parameter is only valid when the EnableV3 parameter is specified.	user
AuthProto	NONE | SHA | MD5	Sets the method of authentication used for the SNMP v3 protocol. Valid values are: • NONE - no authentication • SHA - Secure Hash algorithm • MD5 - MD5 message digest algorithm If you set this parameter to a value other than NONE, you must also specify the AuthPass parameter. Default: NONE	
AuthPass	string	Specifies the password used in SNMPv3 authentication. This password must be at least 8 characters in length and may contain spaces; however, it must not contain the double quote character (") or two consecutive dashes (--).	
PrivProto	NONE|DES	Sets the method used for encrypting SNMP v3 protocol messages. Valid values are: • NONE - no encryption used • DES - Data Encryption Standard algorithm If you set this parameter to a value other than NONE, you must also specify the PrivPass parameter. Default: NONE	
PrivPass	string	Specifies the password used in SNMPv3 message encryption. This password must be at least 8 characters in length and may contain spaces; however, it must not contain the double quote character (") or two consecutive dashes (--).	
Minimal	n/a	Sets the selection of components installed to core agent files only.	n/a
SvcManual	n/a	Registers Netcool/SSM service in manual mode and does not start it after installation.	n/a
ClusterInstall	n/a	Selects the cluster-aware installation mode for installing Netcool/SSM on a Windows server cluster node.	n/a
ClusterGroup	string	Name of the cluster group on which to install Netcool/SSM.	n/a
ClusterStorage Resource	string	Name of the physical disk resource on which to install Netcool/SSM core configuration files	n/a
NoReconfig	n/a	Bypasses the installer configuration dialogs during an upgrade.	n/a

Recording an installation response file

Recording an installation response file requires you to run the interactive installation process and at each step enter the desired parameters for the installation. The installer records these parameters in an installation response file.

About this task

Attention: The host machine on which you record an installation response file must not have a version of Netcool/SSM currently installed. Attempting to record an installation on a machine that contains an installed version of Netcool/SSM will uninstall the existing version instead of recording an installation.

Procedure

To record an installation response file:

1. Enter the command:

   ```
   netcool_ssm-4.0.1-xxxx-win32.exe /r /f1"filename"
   ```

   where the x characters represent the build number of your version of the Netcool/SSM installer.
2. Follow the installation procedure through to completion. When the procedure is complete, the installation response file is stored in the file specified by the `/f1` switch.

Results

Tip: If you did not specify a file using the `/f1` switch, the installer records the responses in the file `setup.iss` located in the Windows directory of the host machine, which is `C:\WINDOWS` or `C:\WINNT`. If you specify a filename but no path, the installer creates the file in the Windows directory.

Uninstalling Netcool/SSM

To uninstall Netcool/SSM follow these steps.

Procedure

1. Select **Start** > **Settings** > **Control Panel**.
2. Double-click **Add/Remove Programs**.
3. From the list of programs, select **Netcool/SSM** and click **Change/Remove**.

   The Netcool/SSM uninstaller removes all files and directories in the Netcool/SSM installation.
4. Restart the host machine.

Server Clusters

To uninstall Netcool/SSM from a server cluster, perform the uninstallation process on each cluster node on which Netcool/SSM was installed.

About this task

Attention: Uninstall Netcool/SSM from all inactive nodes before you uninstall it from the active node. Failure to do so may cause failover of the cluster group on which Netcool/SSM is installed.

Installing Netcool/SSM on UNIX

Read this section for information about how to install, upgrade and uninstall Netcool/SSM on machines running UNIX operating systems. It also provides detailed information about hardware and software required to successfully install and run Netcool/SSM.

System requirements

You can install Netcool/SSM on the following UNIX platforms:

- Solaris
- AIX
- HP-UX
- Red Hat Linux
- SuSE Linux Enterprise Server

Before installing Netcool/SSM, you need to ensure that the host machine satisfies the minimum hardware and software requirements. The following sections describe these requirements for the supported UNIX platforms.

Solaris

The following system requirements apply to Solaris host machines.

Hardware

The minimum hardware requirements for running Netcool/SSM on Solaris platforms are:

- UltraSPARC IIi 400 MHz processor or AMD/Intel x64 processor
- 128 MB RAM or greater recommended
- Approximately 25 MB free disk space
- 50 MB minimum swap space recommended
- Ethernet 10/100 network interface card capable of supporting promiscuous packet reception

Software

Netcool/SSM is available on the following versions of Solaris:

- Solaris 10 (SunOS 5.10) SPARC or x86
- Solaris 11 (SunOS 5.11) SPARC or x86

Note: On Solaris SPARC systems, ensure 32-bit libraries are installed.

Latest library patches

Patches to the Solaris Operating Environment libraries are available. Ensure that the latest patch for the Solaris Operating Environment is installed and running on the host machine before you install Netcool/SSM.

Table 6 on page 21 lists the current patch IDs available at the time of writing. Before installing a patch, check that it is the latest version as patches are cumulative and always backward compatible.

Table 6. Patch IDs for Solaris Operating System Versions

Solaris	Patch ID
10	119963-08

For further information about patches to the Solaris Operating Environment, see http://developers.sun.com/sunstudio/downloads/patches/ss11_patches.html.

Limitations

- Packet capture cannot work in a Solaris zone
- If you are using Solaris 11 on Oracle 64-bit, ensure that you install the Oracle 32-bit libraries before installing Netcool/SSM

AIX

The following system requirements apply to AIX host machines.

Hardware

The minimum hardware requirements for running Netcool/SSM on AIX platforms are:

- IBM POWER4/5 64-bit processor
- 64 MB RAM or greater recommended
- Approximately 55 MB free disk space
- 50 MB minimum swap space recommended
- Ethernet 10/100 network interface card capable of supporting promiscuous packet reception

Software

Netcool/SSM is available on the following versions of AIX:

- AIX 6.1 (64-bit only)
- AIX 7.1 (64-bit only)

HP-UX

The following system requirements apply to HP-UX host machines.

Hardware

The minimum hardware requirements for running Netcool/SSM on HP-UX platforms are:

- Intel Itanium 2 processor
- 64 MB RAM or greater recommended
- 70 MB free disk space
- 50 MB minimum swap space recommended
- Ethernet 10/100 network interface card capable of supporting promiscuous packet reception

Software

Netcool/SSM is available for the following versions of HP-UX:

- HP-UX 11.23 (11i v2) on Intel Itanium

- HP-UX 11.31 (11i v3) on Intel Itanium

Latest Library Patches

The host machine may be running patches that have been recalled. The Netcool/SSM installer checks to see what operating system patches have been installed on the machine. If it detects a problem it provides a warning similar to the following:

```
# ./netcool-ssm-4.0.1-xxxx-platform-arch.installer
Netcool/SSM Installer
Patch PHCO_23919 (patch for printf) detected.
*** WARNING: THIS PATCH MAY CAUSE SETUP TO CRASH ***
Consult your Netcool/SSM documentation for details.
Do you want to continue? [n]
```

Note: Visit the Hewlett-Packard (HP) IT Resource Center online at http://www.itrc.hp.com for information on updating or removing this patch. The Netcool/SSM installer will not do this for you.

Red Hat Linux

The following system requirements apply to Red Hat Linux host machines.

Hardware

The minimum hardware requirements for running Netcool/SSM on Red Hat Linux platforms are:

- Intel Pentium II processor 400 MHz or greater
- 128 MB RAM or greater recommended
- Approximately 20 MB free disk space
- 50 MB minimum swap space recommended
- Ethernet 10/100 or Token Ring network interface card capable of supporting promiscuous packet reception

Software

Netcool/SSM is available for the following versions of Red Hat Linux:

- Red Hat Enterprise Linux 5 - x86, x64, ppc64
- Red Hat Enterprise Linux 6.x - x86, x64, ppc64

Note: Ensure that all applicable versions (32-bit or 64-bit) of the libstdc++-33-3.2.3 packages are installed.

SuSE Linux Enterprise Server

The following system requirements apply to SuSE Linux Enterprise Server (SLES) host machines.

Hardware

The minimum hardware requirements for running Netcool/SSM on SLES platforms are:

- Intel Pentium II processor 400 MHz or greater
- 128 MB RAM or greater recommended
- Approximately 20 MB free disk space
- 50 MB minimum swap space recommended

- Ethernet 10/100 or Token Ring network interface card capable of supporting promiscuous packet reception

Software

Netcool/SSM is available for the following versions of SLES:

- SuSE Linux Enterprise Server 10 - x86, x64, ppc64
- SuSE Linux Enterprise Server 11 - x86, x64, ppc64

Note: Ensure that all applicable versions (32-bit or 64-bit) of the libstdc++-33-3.2.3 packages are installed.

Installing Netcool/ASM for Oracle

If you do not specify the Oracle home directory during installation, the installer attempts to obtain it from information contained in the Oracle installation's `oratab` file, which it locates using the environment variable `ORATAB_LOCATION`.

If `ORATAB_LOCATION` does not exist, the installer then attempts to find the file using the standard pathnames `/etc/oratab` and `/var/opt/oracle/oratab`.

If the installer cannot locate the `oratab` file using either of these methods, it cannot determine the Oracle home directory. In this case, you must specify the Oracle home directory and `oratab` file by setting the `OracleHome` and `OratabLocation` inivars manually. See the*Netcool/SSM Reference Guide* for more details about configuring Netcool/ASM for Oracle.

Interactive installation

In an interactive installation, the Netcool/SSM installer provides you with a menu of installation options, which you use to configure the installation process.

Procedure

To install Netcool/SSM using the interactive installation method:

1. Log in as the super user (`root`).
2. Change to the directory containing the Netcool/SSM installation package and run the following command:

   `./netcool-ssm-4.0.1-xxxx-`*`platform-arch`*`.installer`

   where the *`x`* characters and the *`platform-arch`* string represent your version of the Netcool/SSM installation package.

   The license agreement is displayed.
3. Read the license agreement and indicate whether you accept the terms:
   a. If you wish to accept the terms of the license agreement and install Netcool/SSM, enter `Y.`The Setup menu is displayed.
   b. If you do not wish to accept the license agreement enter `N.` The installer exits.
4. From the Setup menu, make selections and follow the prompts.

   Table 7 on page 24 describes each installation option. You can change some of these settings again after Netcool/SSM is installed. Where appropriate, Table 7 on page 24 also indicates the command or variable for changing a setting after installation.
5. Press Enter.

The installer displays a summary of the actions it will perform, then prompts you for permission to proceed with the installation.

6. Press Enter.

   Installation begins using the options you selected.

Table 7. Installation options

<table>
<tr><th>Option</th><th colspan="2">Description</th><th><i>Variable</i>/ <b>Command</b></th></tr>
<tr><td>1</td><td colspan="2">Sets the directory in which Netcool/SSM is installed.</td><td>n/a</td></tr>
<tr><td>2</td><td colspan="2">Selects whether the command console is installed.</td><td>n/a</td></tr>
<tr><td>3</td><td colspan="2">Specifies whether the MIB definition files are installed.</td><td>n/a</td></tr>
<tr><td>4</td><td colspan="2">Sets the UDP port number on which the Netcool/SSM agent listens for incoming SNMP requests. The number must be in the range 1-65535 inclusive. The port specified must not be in use by any other application.</td><td><i>UdpPort</i></td></tr>
<tr><td>5</td><td colspan="2">Sets the default community string for the Netcool/SSM agent.</td><td>community set
community add</td></tr>
<tr><td rowspan="6">6</td><td>Secure username</td><td>Sets the name of the user for the Netcool/SSM agent to use in SNMPv3 communication. The installer creates this user during installation.</td><td rowspan="6">user add</td></tr>
<tr><td>Authorization protocol</td><td>Selects the method of authentication used for the SNMP v3 protocol. Valid values are:
NONE - No authentication
SHA - Secure Hash algorithm
MD5 - MD5 message digest algorithm
<b>Note:</b> If you select a value other than NONE, you must also specify a password.</td></tr>
<tr><td>Authorization password</td><td>Sets the password used in SNMPv3 authentication. This password must be at least 8 characters in length and may contain spaces.</td></tr>
<tr><td>Privacy protocol
<b>Note:</b> If you specify this setting, you must also specify the Authentication settings.</td><td>Selects the method used for encrypting SNMP v3 protocol messages. Valid values are:
NONE - No encryption used
DES - Data Encryption Standard algorithm
<b>Note:</b> If you set this parameter to a value other than NONE, you must also specify a password.</td></tr>
<tr><td>Privacy password</td><td>Sets the password used in SNMPv3 encryption. This password must be at least 8 characters in length and may contain spaces.</td></tr>
</table>

Table 7. Installation options (continued)

Option	Description	*Variable*/ **Command**
7	Specifies the format of notifications sent by the agent: 1- SNMP v1 Trap 2 - SNMPv2 Trap	n/a
8	Sets the path and filename of the agent log file.	*LogFile*
9	Specifies the location of the `oratab` file. This option is only used if Netcool/SSM includes the `oracle` subagent.	*OratabLocation*
10	Specifies the Oracle home directory. This option is only used if Netcool/SSM includes the `oracle` subagent.	*OraHome*

Unattended installation

Unattended installation runs silently from the command line.

About this task

To perform an unattended installation, use the following command from the `netcool-ssm` directory:

```
./netcool-ssm-4.0.1-xxxx-platform-arch.installer [install|upgrade]
  [silent|record] [param=value ...]
```

Table 8 describes the command line options.

Table 8. Netcool/SSM installer options

Option	Function
`install`	Installs Netcool/SSM.
`upgrade`	Upgrades an existing Netcool/SSM installation.
`silent`	Installs Netcool/SSM in unattended mode using the default installation parameters.
`record`	Records an installation response file.
param	Specifies installation parameters overriding the default value. See Table 4 for a list of installation parameters.

Table 4 lists the installation parameters that you can use with the installer. You can change some of these settings again after Netcool/SSM is installed. Where appropriate, the table also lists the command or variable for changing a setting after installation.

Table 9. Installation parameters

Parameter	Values	Description	*Variable*/ **Command**
`CONF_DEFCOM`	string	Specifies the SNMP community string for the agent. Default: `private`, `public`	n/a
`CONF_DISABLE_V1`	`y|n`	Configures handling of SNMP v1 requests: `y` - The agent does not respond to SNMP v1 requests `n` - The agent responds to SNMP v1 requests Default: `n`	*DisableV1*

Table 9. Installation parameters (continued)

Parameter	Values	Description	*Variable*/ **Command**
CONF_DISABLE_V2	y|n	Configures handling of SNMP v2 requests: y - The agent does not respond to SNMP v2 requests n - The agent responds to SNMP v2 requests Default: n	*DisableV2*
CONF_DISABLE_V3	y|n	Configures handling of SNMP v3 requests: y - The agent does not respond to SNMP v3 requests n - The agent responds to SNMP v3 requests Default: n	*DisableV3*
CONF_LOGFILE	string	Specifies the path and filename of the agent log file. Default: agent.log	*LogFile*
CONF_LOGSIZE	integer	Specifies the maximum size of the agent log file (in bytes). Default: 1000000	*LogSize*
CONF_TRAPVER	1|2	Specifies the format of notifications sent by the agent: 1- SNMP v1 Trap 2 - SNMPv2 Trap Default: 1	n/a
CONF_UDPPORT	integer	Sets the UDP port number on which the Netcool/SSM agent listens for incoming SNMP requests. The port-number must be an integer in the range 1-65535 inclusive. The port specified must not be in use by any other application. Default: 161	*UdpPort*
INST_CFGPATH	string	Sets the agent configuration directory. Default: /opt/netcool/ssm/config	n/a
INST_COEXIST	y|n	Specifies whether any existing Netcool/SSM installation in any path other than that indicated by INST_PATH is removed during installation. Use when testing multiple installations on one machine. If you want to install Netcool/SSM in the same directory as the existing installation, specify n. Default: y	n/a
INST_CONS	y|n	Selects whether the command console is installed: y - Command console installed n - Command console not installed Default: y	n/a

Table 9. Installation parameters (continued)

Parameter	Values	Description	*Variable*/ **Command**
INST_DLPI	y|n	*For AIX only.* Determines whether the /etc/pse.conf file is modified for automatic startup of the DLPI interface at boot-time: y - File modified n - File not modified	n/a
INST_MIBS	y|n	Specifies whether the MIB definition files are installed: y - MIB files installed n - MIB files not installed Default: y	n/a
INST_OPTIONS	string	Specifies the file and path name of the installation response file.	n/a
INST_PATH	string	Sets the Netcool/SSM installation directory. Default: /opt/netcool/ssm	n/a
INST_RETAIN	y|n	Specifies whether the configuration of current Netcool/SSM installation is retained in the new installation: y - Configuration retained n - Configuration not retained Default: y	n/a
INST_SAVESTATE	y|n	Specifies whether the state of any Netcool/SSM agent currently installed is saved then restored on the new installation: y - Save and restore state n - Do not save and restore state Default: y	n/a
INST_STARTAGENT	y|n	Specifies whether the Netcool/SSM agent is started after installation: y - Started n - Not started Default: y if the agent is running before installation, otherwise n.	n/a
INST_TMPDIR	string	Specifies a temporary directory for the installer to use. Ensure that the directory exists before you run the installer. If no value is specified, or the specified directory is invalid, the default /tmp directory is used. You may want to use this option if /tmp does not exist or if you can't run files from /tmp.	n/a
SECNAME	string	Sets the name of the SNMPv3 user created during installation. Default: ""	user add

Table 9. Installation parameters (continued)

Parameter	Values	Description	*Variable*/ **Command**
AUTHPROTO	SHA| MD5| NONE	Selects the method of authentication used for the SNMP v3 protocol. Valid values are: NONE - No authentication SHA - Secure Hash algorithm MD5 - MD5 message digest algorithm If you select a value other than NONE, you must also specify a password. Default: NONE	user add
AUTHPASS	string	Sets the password used in SNMPv3 authentication. This password must be at least 8 characters in length. Default: ""	user add
PRIVPROTO	string	Selects the method used for encrypting SNMP v3 protocol messages. Valid values are: NONE - No encryption used DES - Data Encryption Standard algorithm If you set this parameter to a value other than NONE, you must also specify a password. Default: DES	user add
PRIVPASS	string	Sets the password used in SNMPv3 encryption. This password must be at least 8 characters in length.	user add

The following command performs an unattended installation using the default installation parameters:

```
./netcool-ssm-4.0.1-xxxx-platform-arch.installer silent
```

The following example performs an unattended installation using the installation response file ssmagent-setup.opt:

```
./netcool-ssm-4.0.1-xxxx-platform-arch.installer silent
  INST_OPTIONS=/tmp/ssmagent-setup.opt
```

Recording an installation response file

Recording an installation response file requires you to run the interactive installation process and at each step enter the desired parameters for the installation. The installer records these parameters in an installation response file.

About this task

To record an installation response file, enter the following command:

```
./netcool-ssm-4.0.1-xxxx-platform-arch.installer record [INST_OPTIONS=file]
```

This command starts an interactive installation. Follow the installation procedure through to completion. The installer records the parameters that you enter during this process but does not actually install Netcool/SSM. The recorded parameters are stored in the installation response file specified by the INST_OPTIONS parameter. If you do not specify this parameter, the installer prompts you for a filename. The default filename is /tmp/ssmagent-setup.opt.

Uninstalling Netcool/SSM

To uninstall Netcool/SSM, follow these steps.

Procedure

1. As the root user, verify that the Netcool/SSM agent is not running.

   If the agent is running, stop it.
2. Change to the Netcool/SSM `bin` directory.
3. At the shell prompt, enter the command:

   ```
   ./remove-ssmagent
   ```

   Netcool/SSM and its directory tree are removed.

Chapter 3. Getting started

Getting started with Netcool/SSM after installation involves starting the agent, and using the Netcool/SSM command console.

Running Netcool/SSM on Windows

You can run Netcool/SSM as either an application or a service.

Running Netcool/SSM as an application

In application mode, you must manually start Netcool/SSM. Application mode is a suitable choice if you intend to run Netcool/SSM infrequently.

Starting Netcool/SSM in Application mode

Procedure

To run Netcool/SSM in application mode, complete one of the following:

- From the Windows desktop, select **Start** > **All Programs** > **Netcool** > **SSM** > **Application Mode**.
- From a command-line, change to the `bin` directory of the Netcool/SSM installation and execute the command `ssmagent`.

Results

Netcool/SSM starts and the Netcool/SSM icon is displayed in the Windows System Tray.

Stopping Netcool/SSM in Application mode

About this task

To stop the Netcool/SSM when it is running in application mode, right-click the Netcool/SSM icon in the System Tray and select **Exit**. Netcool/SSM stops running and its icon is no longer displayed in the Windows System Tray.

Running Netcool/SSM as a service

Running Netcool/SSM in service mode has the several advantages.

- It is not necessary for you to log on to the machine to start Netcool/SSM.
- Netcool/SSM operates transparently in the background.

In service mode, Netcool/SSM starts when the host machine starts, and is available whenever the machine is running. It is generally the preferred option because it does not require you to be logged on in order for Netcool/SSM to operate.

In service mode, Netcool/SSM is installed as a Windows service, so you must specify the account and password of an administrative user to run the service. By default, the installer configures Netcool/SSM to run under the *Local System* account.

Service Controller

The Service Controller manages the operation of Netcool/SSM on Windows.

Figure 3 shows the Service Controller.

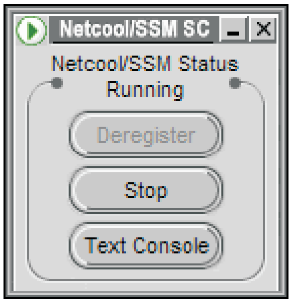


Figure 3. Netcool/SSM service controller

To open the Service Controller, select **Start** > **All Programs** > **Netcool** > **SSM** > **Service Controller**. When the Service Controller has started, its icon is displayed in the System Tray.

Tip: To display the Service Controller when it is running and minimized, right-click the Service Controller icon in the System Tray and select **Restore**.

The Service Controller provides toggle buttons to start, stop, register, and deregister the Netcool/SSM service.

Starting the Netcool/SSM service

To start the Netcool/SSM service from the Service Controller, click **Start**.

About this task

The Service Controller's Status display indicates that the service is starting by displaying the status `Starting`. After a short amount of time, the status changes to `Running` and the Service Controller icon in the System Tray changes to reflect this. The function of the Service Controller **Start** button changes to **Stop**. The Netcool/SSM service has now started.

Tip: If the status reverts back to `Not Running` state, check the Application Log in the Windows Event Viewer to determine the reason why Netcool/SSM did not start.

Note: Netcool/SSM must be registered as a service before you can start it using the Service Controller.

Alternatively, you can start the Netcool/SSM service using the standard Windows service control facility.

Stopping the Netcool/SSM service

To stop the Netcool/SSM service from the Service Controller, click **Stop**.

About this task

The Service Controller's Status display indicates that the service is stopping by displaying the status `Stopping`. After a short amount of time, the status changes to `Not Running` and the Service Controller icon in the System Tray changes to reflect this. The function of the Service Controller **Stop** button changes to **Start**. The Netcool/SSM service has now stopped.

Alternatively, you can stop the Netcool/SSM service using the standard Windows service control facility.

Registering Netcool/SSM as a service

During the installation process, the installer creates a service for Netcool/SSM and registers it under the service name `NCAgent` (with display name `Netcool/SSM`). If the installer does not create this service for some reason, you must manually register Netcool/SSM as a service.

Procedure

To register Netcool/SSM as a service:

1. In the Service Controller, click **Register**.
2. The Register Service dialog is displayed.
3. In the **Log On As** pane, select the account that you wish to run Netcool/SSM under.
4. If you select the **Specify Account** option, you must enter the account credentials for a Windows user with permissions sufficient to allow service logins.
5. In the **Startup Type** pane, select the **Automatic** option.
6. If you wish to select a different option, you must manually start the Netcool/SSM service.
7. Click **Register**.
8. The Register Service dialog closes and the service is registered. In the Service Controller, the function of the **Register** button changes to **Deregister**.

Deregistering the Netcool/SSM service

To deregister the Netcool/SSM service, follow these steps.

Procedure

1. In the Service Controller, click **Deregister**.
2. The Service Controller stops the Netcool/SSM service and deregisters it. The Netcool/SSM icon in the System Tray changes to reflect this. In the Service Controller, the function of the **Deregister** button changes to **Register**.

Running Netcool/SSM on Windows server clusters

On Windows server clusters, use the Windows Cluster Administrator application to control the Netcool/SSM service.

To open the Cluster Administrator, select **Start** > **Administrative Tools** > **Cluster Administrator**.

To start Netcool/SSM:

1. In the Cluster Administrator, locate the Group in which Netcool/SSM is installed.
2. Select the `SSM Agent` resource.
3. Select **File** > **Bring Online**.

To stop Netcool/SSM:

1. In the Cluster Administrator, locate the Group in which Netcool/SSM is installed.
2. Select the `SSM Agent` resource.
3. Select **File** > **Take Offline**.

Running Netcool/SSM on UNIX

Netcool/SSM runs as a daemon process, which you can configure to start either automatically or manually.

By default, the Netcool/SSM installer creates the necessary initialization parameters and system configuration to automatically start Netcool/SSM at boot time; however, you can configure the Netcool/SSM daemon to start manually.

The procedure for manually starting the Netcool/SSM daemon varies depending on the platform running on the host machine. The following sections explain how to start Netcool/SSM manually on the various supported platforms.

Note: If you specified SNMP port 161 when you installed Netcool/SSM, a conflict may arise if another SNMP service included is running on the host machine and is using this port. In this case, disable the other SNMP service or reassign it to another port. Alternatively specify another port for the Netcool/SSM agent. For more details about changing this port number, see “Initialization file (init.cfg)” on page 63.

AIX

About this task

To manually start the Netcool/SSM daemon, use the command:

```
startsrc -s ssmagent
```

Note: On AIX platforms, ensure that the DLPI module is loaded; otherwise Netcool/SSM will not run. See “Loading the DLPI module” on page 35 for details.

To manually stop the Netcool/SSM daemon, use the command:

```
stopsrc -c -s ssmagent
```

This command stops the Netcool/SSM daemon using `sig 2`. If this has no effect, it uses `sig -9` to kill the daemon.

Loading the DLPI module

Netcool/SSM requires the system DLPI module to allow it to promiscuously monitor network data. This module is disabled by default, so you must enable it.

About this task

To check whether DLPI is enabled, use the command:

```
strload -q -d dlpi
```

If the system does not respond with `dlpi: yes` then enable DLPI using the following command:

```
strload -f /etc/dlpi.conf
```

The host machine will revert back to default behavior after a reboot. The default behavior is defined in the file `/etc/pse.conf`. If you wish to permanently enable DLPI, uncomment any lines in this file that contain references to `dlpi`.

HP-UX

About this task

To manually start, stop, or restart the Netcool/SSM daemon, use the command:

```
/sbin/init.d/ssmagent {start|stop|restart}
```

Note: The startup script `init.ssmagent` kills the SNMP agent process `snmpd`.

Red Hat Linux

About this task

To manually start, stop, or restart the Netcool/SSM daemon, or view its status, use the command:

```
/etc/rc.d/init.d/ssmagent {start|stop|restart|status}
```

Note: The startup script `init.ssmagent` kills the SNMP agent process `snmpd`.

Solaris

Procedure

To manually start, stop, or restart the Netcool/SSM daemon on Solaris:

Run the command: `svcadm {enable|disable|restart} ssmagent`

What to do next

Tip: The **`svcadm`** command persists across system boots. To temporarily start or stop the agent, use the command:

```
svcadm {enable|disable} -t ssmagent
```

SuSE Linux Enterprise Server

Procedure

To manually start, stop, or restart the Netcool/SSM daemon, or view its status:

Run the command, `/etc/init.d/ssmagent {start|stop|restart|status}`

Note: The startup script `init.ssmagent` kills the SNMP agent process `snmpd`.

Netcool/SSM command-line options

Netcool/SSM provides several command-line options, which you may use when starting Netcool/SSM from the command-line.

The command format for using these options is:

```
ssmagent -option [argument]
```

Table 10 lists the available options.

Table 10. ssmagent command line options

Option	Argument	Description
f	string	Specifies the directory in which the Netcool/SSM configuration files are located. This location is interpreted relative to the Netcool/SSM `bin` directory.
h	N/A	Displays command line help. Using this option does not start Netcool/SSM.
p	integer	Specifies the UDP port on which the agent listens for SNMP requests. This option overrides any port set by the `UdpPort` inivar. Netcool/SSM uses port 161 by default.
v	N/A	Displays the version number, build number and build date of Netcool/SSM. Using this option does not start Netcool/SSM.

For example, to start Netcool/SSM on port `1161`, issue the command:

```
ssmagent -p 1161
```

Note: The command-line options are listed here for reference purposes only. When starting Netcool/SSM, use the procedures described in “Running Netcool/SSM on Windows” on page 31 and “Running Netcool/SSM on UNIX” on page 34.

Using the command console

The command console is a command line interface and interpreter that enables full control of the agent.

You can also use the MIB Explorer to perform some configuration functions (see Chapter 8, “MIB Explorer,” on page 109); however, a greater degree of control is available with the command console. This section describes how to start and use the command console.

Starting the command console

You can run command console commands using either the command console or the command-line.

About this task

Tip: To run command console commands using the command-line, use the -c option. For example, in the command-line run ssmcons -c "loglevel debug".

Procedure

To start the command console on Windows platforms, do any of the following:

- From the **Start** menu, select **Programs** > **Netcool** > **SSM** > **Text Console**.
- In the Service Controller, click **Text Console**.
- From the System Tray, when the Netcool/SSM icon is visible, right-click the icon and select **Text Console**.
- From the Netcool/SSM bin directory, enter the command ssmcons

Results

To start the command console on UNIX platforms, from the Netcool/SSM bin directory, enter the command ./ssmcons.

The syntax of the ssmcons command on all platforms is:

```
ssmcons [{-option [argument]} ...] [hostname]
```

The *hostname* parameter specifies the name or IP address of the agent that you wish to connect to. Table 11 describes the ssmcons command line options and arguments.

Table 11. ssmcons command line options

Option	Argument	Description
c	N/A	Executes the console command and prints the output to stdout. For example: ssmcons -c "subagent list" lists all loaded subagents. Some characters like $ need to be escaped in the shell before they are passed in. For example, ssmcons -c "snmp walk $hrDiskStorageTable" does not work, while ssmcons -c "snmp walk \$hrDiskStorageTable" does work. Character escaping is operating system and shell dependent.
e	N/A	Forces encryption of console commands, even on local connections. The console automatically negotiates the strongest mode available, either Blowfish, 3DES, or AES.
h	N/A	Displays the command line help for the ssmcons command.
l	integer	Specifies the TCP port on which the console listens for agent responses.
n	N/A	Prevents resolution of the hostname parameter.
p	integer	Specifies the UDP port on which the target agent receives and responds to SNMP requests. The valid range is 1 to 65535. Most agents use port 161 unless configured otherwise, and the command console uses port 161 by default.
r	integer	Sets the maximum number of retries when attempting to connect to the agent. The valid range is 0 to 9999.

Table 11. ssmcons command line options (continued)

Option	Argument	Description
t	integer	Sets the timeout period (in seconds) for each retry. The valid range is 1 to 3600.
w	string	Reads a password from a specified file to authenticate against the agent. That is, -w *filename*, where *filename* is the name of the file that contains the console password.

For example, to open the command console and connect to an agent located on the machine with IP address 187.12.3.4 using port 1161 and 3 retries, use the command:

```
ssmcons -r 3 - p 1161 187.12.3.4
```

Note: If you omit the *hostname* parameter, the console connects to the local agent (that is, the agent at IP address 127.0.0.1).

Connection format

The command console sends the initial connection request to the agent on the UDP port used by the agent for receiving and responding to SNMP requests. If the agent accepts this connection request, it opens a TCP connection with the command console using any available TCP port, and the rest of the session uses this port.

Command console passwords

If you are logged on to the agent's host machine as the Administrator or root user, the ssmcons command does not require a password; however, if you wish to connect to a remote agent or you are not an Administrator or root user then you must supply a password.

To create a password for securing access to an agent from the command console, enter the following command from the console:

```
password new_password
```

The password is saved in the file console.dat with administrator-only file permissions in the agent's configuration directory.

Note: On Windows platforms, you can enforce administrator-only file permissions on NTFS file systems but not on FAT file systems.

Tip: You can add the password command to an agent's agent.cfg file. For security reasons you should ensure that non-administrative users do not have access to the agent.cfg file if it contains password information.

Command input

The command console provides basic command editing and history features.

- The cursor up and cursor down keys move through the command history.
- The cursor left, cursor right, Home and End keys move the cursor along the command line.
- The Escape key deletes the current contents of the command line.
- The Delete and Backspace keys delete single characters in the command line.

For information about the actual configuration commands provided in Netcool/SSM, see "Agent configuration commands" on page 47.

Note: On Windows platforms, when the Netcool/SSM service is paused, the command console still appears to be functional and allows you to enter configuration commands, even though the Netcool/SSM agent itself is actually paused and not responding to SNMP queries. Any commands that you enter while Netcool/SSM service is paused have no effect.

Command help

The command console provides a help facility that displays basic syntax information for configuration commands.

About this task

To obtain general help, enter:

```
help
```

To obtain help about a particular console command, enter either of the following:

```
help command_name
command_name ?
```

where *command_name* is the name of the command for which you require information.

Note: Information about command usage for subagent configuration commands is only available if the associated subagent is loaded. For information about loading subagents, see "Loading a subagent" on page 65.

Closing the command console

Procedure

To close the command console:

Enter the following command:

```
quit
```

Note: This closes the command console but does not stop the Netcool/SSM agent.

Stopping the Netcool/SSM agent from the command console

You can terminate the agent from the command console.

Procedure

To stop the Netcool/SSM agent from the command console:

Run the `terminate` command.
This command also closes the command console.

Attention: Any data that the agent has collected is lost when the agent stops.

Chapter 4. Configuring the agent

Understanding how to configure the agent enables you to tailor its operation to the monitoring tasks that are appropriate to your enterprise's requirements and operating environment.

Netcool/SSM is highly configurable and adapts to a wide range of service monitoring applications. The exact set of tasks required to configure Netcool/SSM depends on the type of service monitoring application, however typical initial configuration steps include:

- Defining trap destinations, such as management stations.
- Selecting the supported SNMP version.
- Defining a standard agent configuration and setting the agent's state and configuration behavior.
- Loading and configuring the subagents required for your monitoring application.
- Assigning MIB module access privileges to communities or users.

Netcool/SSM provides configuration commands for this work. You configure the agent by issuing configuration commands from the command console or including them in configuration files.

Agent configuration and state

Agent configuration and state represent the setup of Netcool/SSM agent.

- Agent *configuration* defines the basic agent setup as well as information such as the subagents loaded.
- Agent *state* defines the state of all control rows created on the agent.

By default, configuration is saved in the `agent.cfg` file and state is saved in the `agent.state` file. These files are located in the Netcool/SSM configuration directory (`config`).

Saving agent configuration and state

An important step in configuring the agent is saving its current configuration and state. Saving the configuration and state before making any changes allows you to roll back to a known configuration if subsequent changes prove unsatisfactory. You can save the agent's current configuration and state at any time.

About this task

To save the configuration and state, use the following command:

```
config save [config file [state file]]
```

Procedure

- If no arguments are supplied, the command stores the configuration and state in the default files `agent.cfg` and `agent.state`.
- If one argument is supplied, the command uses this argument as the base path and filename in creating `.cfg` and `.state` files.

For example, the command `config save curr` stores the agent's configuration in the file `curr.cfg` and the agent's state in the file `curr.state`. If no path is specified as part of the name, both files are saved in the Netcool/SSM configuration directory (`config`).

- If two arguments are supplied, these form the path and filenames of the configuration and state files directly.

  For example, the command `config save myconfig.cfg mystate.txt` stores the agent's configuration in the file `myconfig.cfg` and the agent's state in the file `mystate.txt`. If no path is specified as part of a file's name, the file is saved in the Netcool/SSM configuration directory (`config`).

Results

Tip: Agent configuration and state files are useful in ensuring that all Netcool/SSM agents in your enterprise have the same setup. Once you have established the desired configuration and monitoring parameters on one agent, you can save the configuration and state of the agent then distribute it to other agents in your enterprise.

Restoring agent configuration and state

Restoring agent configuration or state loads reverts to the configuration or state stored in a file.

About this task

To restore the configuration or state from a file, use the command:

```
config execute file
```

The parameter *`file`* represents the path and name of the file in which the configuration or state is stored. Omitting the file's path from this parameter indicates that the file is located in the agent's configuration directory (`config`). Omitting the *`file`* parameter altogether restores the configuration saved in the file `agent.cfg`.

Setting agent state behavior

You can control the agent's state behavior by instructing it to save its state to the file `agent.state` each time it shuts down and to reload this state from `agent.state` when it starts up again. This ensures that the agent will start up in the same state it was in when it previously shut down.

About this task

To set the agent's state behavior, use the following command:

```
set restore on|off
```

Setting restore to `off` prevents the agent from saving and restoring its state. The current setting of the agent's state behavior is stored in the `RestoreState` inivar. The default value is `on`, meaning that the agent's state is persistent.

Agent startup, restart, and shutdown sequences

When the agent starts, restarts or shuts down, it performs a defined sequence of operations, including loading or saving its configuration and state. The agent sends a notification to indicate that it is performing a startup or shutdown sequence.

There are three sequences:

- Cold boot (both initial startup and restart)
- Warm boot
- Shutdown

Attention: When the agent restarts or shuts down, any data previously collected is lost.

Cold boot

The agent performs a cold boot on initial startup, or when explicitly forced to restart using the `coldboot` command.

Cold boot (initial startup)

When the agent starts up, having previously not been running, it performs the sequence of operations shown in Figure 4 on page 44. The agent sends a `coldStart` notification during cold boot.

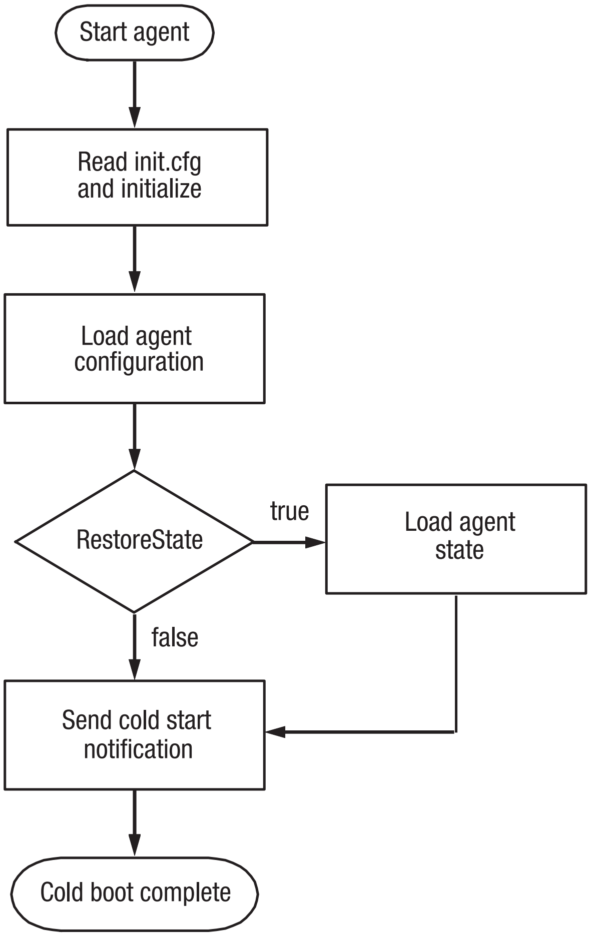


Figure 4. Agent cold boot sequence (initial startup)

Cold boot (restart)

When the agent receives the `coldboot` configuration command, or when its `probeResetControl` object is set to `coldBoot(3)`, it restarts by performing a cold boot and performs the sequence of operations shown in Figure 5 on page 45. The agent sends a `coldStart` notification during cold boot.

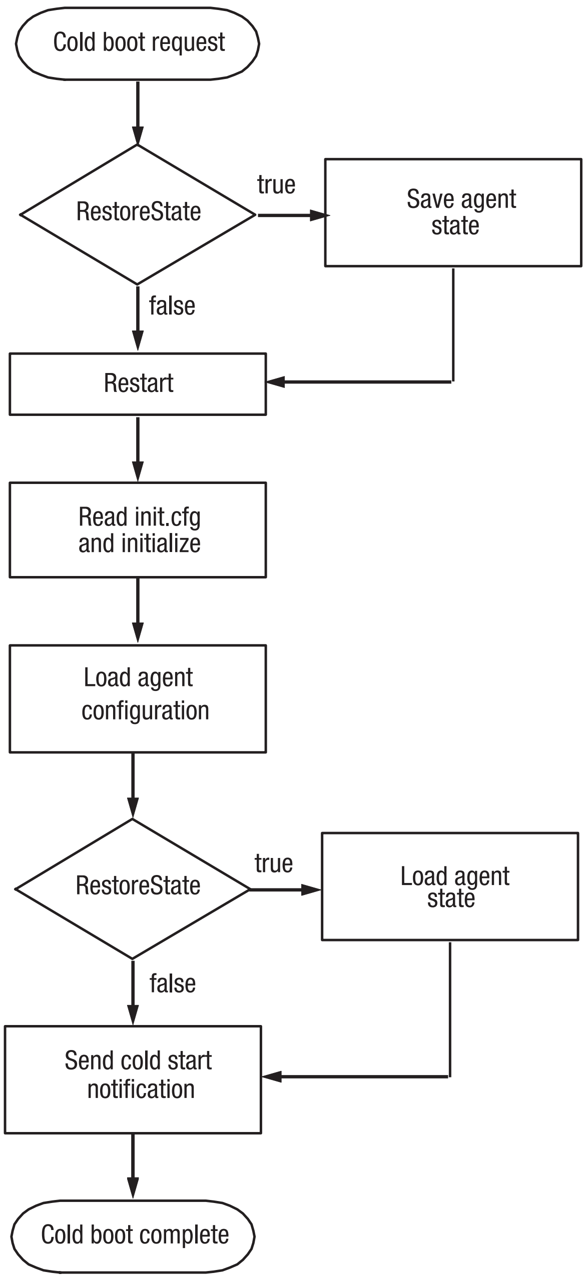


Figure 5. Agent cold boot sequence (restart)

Warm boot

The agent performs a warm boot when it receives the `warmboot` configuration command or when the `probeResetControl` object is set to `warmBoot(3)`.

During a warm boot, the agent performs the sequence of operations shown in Figure 6 on page 46. The agent sends a `warmStart` notification during warm boot.

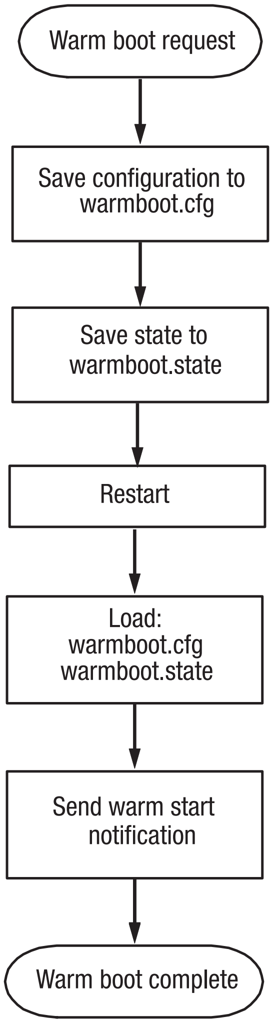

Warm boot request
Save configuration to warmboot.cfg
Save state to warmboot.state
Restart
Load: warmboot.cfg warmboot.state
Send warm start notification
Warm boot complete

Figure 6. Agent warm boot sequence

Shutdown

The agent shuts down in response to the `terminate` command.

When shutting down, it performs the sequence of operations shown in Figure 7 on page 47. The agent sends an `agentTermination` notification during shutdown.

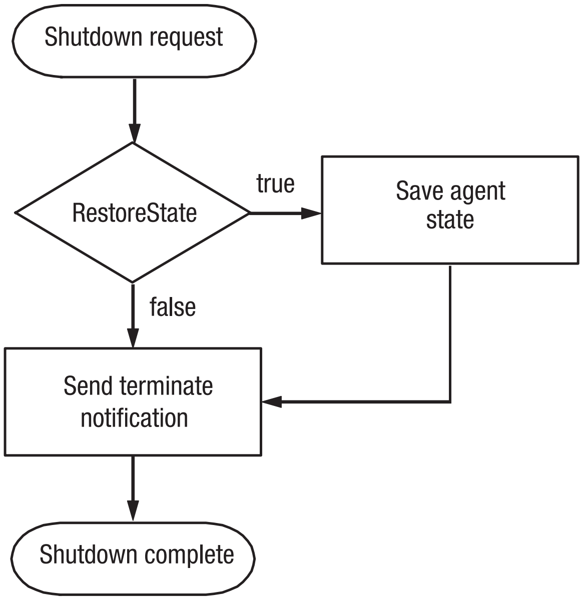


Figure 7. Agent shutdown sequence

Agent configuration commands

Configuration commands control the configuration and operation of the Netcool/SSM agent. These commands perform operations such as starting and stopping the agent, loading subagents and configuring notification destinations.

To execute configuration commands, issue them from the command console or include them in configuration files.

Command syntax

All configuration commands have the same general format. You can get help on configuration commands from the command console.

Configuration commands take form:

```
command [sub-command] [argument ...]
```

For a complete list of configuration commands, see Appendix A, “Configuration commands,” on page 157. Selected configuration commands are described in detail in the subsequent chapters of this guide.

To obtain help about a particular configuration command from the command console, enter either of the following:

```
help command_name
command_name ?
```

where `command_name` is the name of the command about which you require information.

Variables

Variables are useful for storing and re-using temporary values. Variables can have any name that is not a reserved keyword.

The syntax for defining variables is:

```
variable_name = value
```

Some examples of variable definitions are:

```
myentoid=.1.3.6.1.4.1.1977
mystring="testing"
myvalue=53
```

Note: Variable names are case-sensitive.

To expand a variable, prefix it with a $ character:

```
myentoid=.1.3.6.1.4.1.1977
mycopy=$myentoid
```

Variables are always expanded before a command is interpreted, so in the following sample the string value `newvar` is assigned to the variable `var`, then a new variable named `newvar` is created and set to the value `.1.3.6.1.4.1.1977`:

```
Agent> var=newvar
Agent> $var=1.3.6.1.4.1.1977
Agent> echo $newvar
1.3.6.1.4.1.1977
```

Storing returned values

Netcool/SSM stores the value returned by the most recently executed configuration command in the special variable, `$?`. This variable enables you to access the index of a row after its creation and use it as a parameter in another operation—a feature that is especially useful when creating event rows prior to creating control rows that require an event index.

For example, the following commands create an event and assign its index to the variable *normalevent*:

```
event reset
event descr="Threshold normal"
event create type=snmp-trap community=public
normalevent=$?
```

The `$?` variable is also useful in combination with the `snmp match` command for obtaining the index of an existing table row. For example, to obtain the index of a row in `hrStorageTable` that contains information about a ZIP drive, you could use the following configuration commands:

```
snmp match $hrStorageDescr s .*ZIP.*
zipDriveIndex=$?
```

Object identifiers and aliases

To enter object identifiers (OIDs) as numeric strings in configuration commands, precede the OID with a dot character (.).

For example, .1.3.6.1.2.1.1.5.0 is the OID for the RMON object sysname.

When entering OIDs in configuration commands you can use an OID alias instead of the OID numeric string. OID aliases are often easier to remember than OID numeric strings.

For example, the OID of the object srSystemProcessCount in the systemResources MIB is .1.3.6.1.4.1.1977.9.1.1 but you can substitute it with its OID alias $srSystemProcessCount.

OID configuration files map OID numeric strings to OID aliases. These files are located in the config\oid directory on Windows installations or in the config/oid directory on UNIX installations, relative to the Netcool/SSM root installation directory.

Note: OID aliases only represent objects. They do not contain the instance portion of the OID. To specify an instance, append the instance to the alias in the format: object_alias.*x* where *x* denotes the instance. For example, to specify an instance of the object $srSystemProcessCount use $srSystemProcessCount.0.

Writing text to the command console

The echo command writes text to the command console.

The format of this command is:

```
echo text
```

You can include variables in the display text by prefixing them with a dollar character ($). To specify a literal $ character, use $$.

Quote characters

Quote characters receive special handling in configuration commands.

The quote characters " and ' are removed from commands:

```
Agent> cmd="subagent load process"
Agent> echo $cmd
subagent load process
```

To embed quote characters in a string, enclose them in the other type of quote character:

```
Agent> string="This 'quoted' string"
Agent> echo "$string"
This 'quoted' string
```

Tip: To eliminate the possibility of quotes in strings being interpreted incorrectly, always enclose strings and string variables in double quotes.

A command may extend over multiple lines if it contains an unterminated quote:

```
Agent> event description="this is
> a multi-line command"
Agent> event create
```

Multiple space characters are only retained when enclosed in quotes.

Backslash character

The backslash character (\) has special meaning in commands.

- \%*nn* inserts a character, where *nn* is the ASCII value of the character, in hexadecimal.
- \' and \" insert literal quote characters.
- A line ending with a backslash extends onto the next line, for example:

```
Agent> subagent load \
> rmonc
Loaded rmonc
```

- \\ escapes the special meaning of the backslash character, inserting a single backslash.

Supported ASN types

Type identifiers in configuration commands correspond to supported ASN types.

Table 12 lists the ASN types allowed in agent commands. When entering ASN types in commands that accept more than one type, use the abbreviated type identifier indicated in the Type column.

Table 12. Supported ASN types

Type	ASN type	Description
a	IpAddress	An octet string of length 4, with a maximum value of 255.255.255.255.
d	Octet String	A decimal string.
i	Integer32	A signed 32-bit integer.
n	Null	The null value .0.0.
o	ObjID	An object identifier that conforms to BER encoding.
s	Octet String	A literal ASCII string, usually a DisplayString.
t	TimeTicks	Time ticks expressed in integer format.
u	Unsigned32	An unsigned 32-bit integer.
x	Octet String	A hexadecimal string.

Multi-byte character support

Netcool/SSM supports multi-byte character encodings in string objects. It reliably preserves the encoding format objects sent in notifications and responses to SNMP requests.

When creating string objects, ensure that you use the character encoding appropriate to your monitoring environment. For example, if you wish to configure Netcool/SSM to send notifications to an MT Trapd probe running on a system that uses GB 2312 encoding, ensure that you use GB 2312 to define the Netcool/SSM string objects that you wish to include as variable bindings in notifications sent to MT Trapd. Figure 8 on page 51 illustrates the situation.

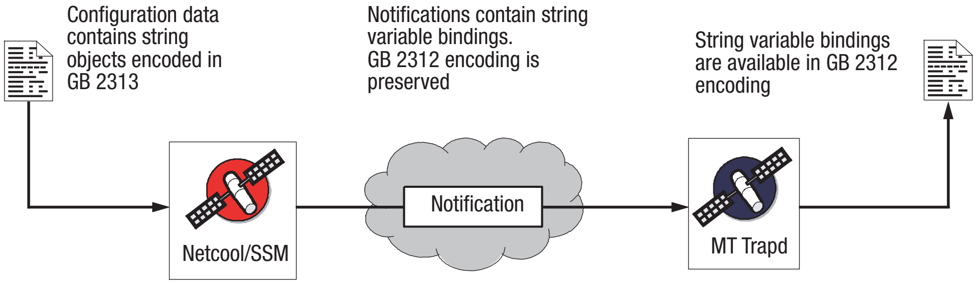

Configuration data contains string objects encoded in GB 2313
Notifications contain string variable bindings. GB 2312 encoding is preserved
String variable bindings are available in GB 2312 encoding
Netcool/SSM
Notification
MT Trapd

Figure 8. Multi-byte character encoding

Compatibility guidelines

Changes in the configuration command syntax between major releases of Netcool/SSM may affect the compatibility of configuration files that were created for use with earlier releases.

Configuration files that you have created for use with Netcool/SSM version 3.2 will generally operate correctly on Netcool/SSM version 4.0. When upgrading to version 4.0 from version 3.2, check configuration files for missing closing-quotes because they are now interpreted as multi-line strings. Also check for the character combinations `\\`, `\%`, `\"`, and `\'` as these are interpreted differently in Netcool/SSM 4.0.

If you have created and tested configuration files for use on Netcool/SSM version 3.1, test them carefully on Netcool/SSM version 3.2 or 4.0 installations to ensure that syntax changes do not affect their operation.

Changes introduced in Netcool/SSM version 4.0

Netcool/SSM version 4.0 introduces changes in the configuration command syntax that affect the handling of quote and backslash characters. Follow these compatibility guidelines to ensure that configuration files developed for earlier releases operate correctly on Netcool/SSM version 4.0.

Quote characters in configuration commands

Quote characters are interpreted and removed from all configuration commands. Table 13 shows the effects of the changes introduced in Netcool/SSM version 4.0.

Table 13. Syntax comparison: quote characters in config commands

Netcool/SSM version 3.2	Netcool/SSM version 4.0

`Agent> "host" "add"`
`Unknown command - type ? for help`
	`Agent> "host" "add"`
	`Syntax: host add <address> [mask]`
	`[Master agent]`

Additionally, strings containing unterminated quote characters may extend over multiple lines; see "Quote characters" on page 49 for more details. This syntax is also incompatible with earlier releases, so to ensure compatibility across versions, do not use unterminated quote characters or strings spanning multiple lines.

This syntax is not compatible with releases prior to version 4.0.

Backslash characters

The backslash character (\) has special meaning when followed by percent, quote, or newline characters. Table 13 on page 51 shows the effects of the changes introduced in Netcool/SSM version 4.0.

Table 14. Syntax comparison: backslash characters

Netcool/SSM version 3.2	Netcool/SSM version 4.0

`Agent> path="C:\Windows\System32"`
`Agent> echo "$path"`
`C:\Windows\System32`
`Agent> echo "C:\Window\%7a\System32"`
`C:\Window\%7a\System32`
`Agent> echo "C:\\Window\%7a\System32"`
`C:\\Window\%7a\System32`
`Agent> echo \\\\blah\\\\`
`\\\\blah\\\\`
`Agent> echo 'It\'s embedded'`
`It\s embedded`
`Agent> echo "%58\%58%6a\%6A"`
`%58\%58%6a\%6A`
	`Agent> path="C:\Windows\System32"`
	`Agent> echo $path`
	`C:\Windows\System32`
	`Agent> echo "C:\Window\%7a\System32"`
	`C:\Windowz\System32`
	`Agent> echo "C:\\Window\%7a\System32"`
	`C:\Windowz\System32`
	`Agent> echo \\\\blah\\\\`
	`\\blah\\`
	`Agent> echo 'It\'s embedded'`
	`It's embedded`
	`Agent> echo "%58\%58%6a\%6A"`
	`%58X%6aj`

Configuration commands containing Windows file paths should be interpreted correctly on all versions of Netcool/SSM. If a command contains any of the character combinations `\\`, `\%`, `\"`, or `\'`, incompatibilities may occur.

Additionally, commands containing a trailing backslash character extend onto the following line; see “Backslash character” on page 50 for more details. This syntax is incompatible with earlier releases.

Changes introduced in Netcool/SSM version 3.2

Changes in the configuration command syntax introduced in Netcool/SSM version 3.2 may affect configuration files that were developed for Netcool/SSM releases earlier than version 3.2.

Quote characters in assignment, echo and subagent configuration commands

Quote characters are interpreted and removed from variable assignment operations, **echo** commands, and subagent configuration commands. Table 15 on page 53 shows the effect of the changes introduced in Netcool/SSM version 3.2.

Table 15. Syntax comparison: quote characters in assignment, echo, and subagent config commands

Netcool/SSM version 3.1	Netcool/SSM version 3.2

`Agent> cmd="subagent load process"`
`Agent> $cmd`
`Unknown command - type ? for help`
`Agent> echo $cmd`
`"subagent load process"`
`Agent> foo='Hello world'`
`Agent> echo $foo`
`'Hello world'`
`Agent> subagent load process`
`Loaded process`
`Agent> d="My monitor"`
`Agent> process description=$d`
`Agent> process description="$d"`
`Unrecognized keyword 'My'`
	`Agent> cmd="subagent load process"`
	`Agent> $cmd`
	`Loaded process`
	`Agent> echo $cmd`
	`subagent load process`
	`Agent> foo='Hello world'`
	`Agent> echo $foo`
	`Hello world`
	`Agent> subagent load process`
	`Loaded process`
	`Agent> d="My monitor"`
	`Agent> process description=$d`
	`console: Unknown command "monitor".`
	`Agent> process description="$d"`

To ensure compatibility when using variables as command macros, never use double or single quotes:

```
Agent> cmd=subagent load process
Agent> $cmd
Loaded process
```

To ensure compatibility of subagent configuration commands, define property strings without using quote characters, and enclose the variable in quotes when passing it to the subagent configuration command:

```
Agent> subagent load process
Loaded process
Agent> d=My monitor
Agent> process description="$d"
```

Embedded quote characters

You can insert literal quote characters into a quote-delimited string by using the alternate quote character, that is, by using ' characters inside a "-delimited string, or a " characters inside a '-delimited string. Table 16 shows the effect of the changes introduced inNetcool/SSM version 3.2.

Table 16. Syntax comparison: embedded quote characters

Netcool/SSM version 3.1	Netcool/SSM version 3.2

`Agent> echo '"Alternate quotes"'`
`'"Alternate quotes"'`
`Agent> echo "It's embedded"`
`"It's embedded"`
	`Agent> echo '"Alternate quotes"'`
	`"Alternate quotes"`
	`Agent> echo "It's embedded"`
	`It's embedded`

To ensure compatibility, adopt the syntax introduced in Netcool/SSM version 3.2; the version 3.1 syntax is no longer supported.

Multiple space characters

Sequences of multiple space characters are only preserved when enclosed in quote characters. Table 17 on page 54 shows the effects of the changes introduced in Netcool/SSM version 3.2.

Table 17. Syntax comparison: multiple space characters

Netcool/SSM version 3.1	Netcool/SSM version 3.2
`Agent> echo   Hello   World   !`	`Agent> echo   Hello   World   !`
`Hello   World   !`	`Hello World !`
`Agent> foo="One Two  Three   !"`	`Agent> foo="One Two  Three   !"`
`Agent> echo $foo`	`Agent> echo $foo`
`"One Two  Three   !"`	`One Two Three !`
`Agent> echo "$foo"`	`Agent> echo "$foo"`
`""One Two  Three   !""`	`One Two  Three   !`
`Agent> bar=$foo`	`Agent> bar=$foo`
`Agent> echo "$bar"`	`Agent> echo "$bar"`
`""One Two  Three   !""`	`One Two Three !`
`Agent> bar="$foo"`	`Agent> bar="$foo"`
`Agent> echo "$bar"`	`Agent> echo "$bar"`
`"""One Two  Three   !"""`	`One Two  Three   !`
`Agent> foo=One Two Three !`	`Agent> foo=One Two  Three   !`
`Agent> echo "$foo"`	`Agent> echo "$foo"`
`"One Two  Three   !"`	`One Two Three !`

To ensure compatibility, adopt the syntax introduced in Netcool/SSM version 3.2; the version 3.1 syntax is no longer supported.

String concatenation

Adjacent strings not separated by space characters are concatenated. Table 18 shows the effects of the changes introduced in Netcool/SSM version 3.2.

Table 18. Syntax comparison: string concatenation

Netcool/SSM version 3.1	Netcool/SSM version 3.2
`Agent> echo Hell'o world'`	`Agent> echo Hell'o world'`
`Hell'o world'`	`Hello world`
`Agent> a="pro"`	`Agent> a="pro"`
`Agent> b="cess"`	`Agent> b="cess"`
`Agent> subagent load $a$b`	`Agent> subagent load $a$b`
`Failed to load SubAgent: pro`	`Loaded process`

To ensure the compatibility of space characters contained in quote characters, adopt the syntax introduced in Netcool/SSM version 3.2; the version 3.1 syntax is no longer supported.

Equals characters in subagent configuration commands

In subagent configuration commands, property assignments must use the equals character (=). Table 19 shows the effects of the changes introduced in Netcool/SSM version 3.2.

Table 19. Syntax comparison: equals characters in subagent config commands

Netcool/SSM version 3.1	Netcool/SSM version 3.2
`Agent> subagent load process`	`Agent> subagent load process`
`Loaded process`	`Loaded process`
`Agent> process reset`	`Agent> process reset`
`Agent> process description MyDescription`	`Agent> process description MyDescription`
`Agent> process description=MyDescription`	`console: Loose word "MyDescription" when`
	`command is already "description"`
	`Agent> process description=MyDescription`

To ensure compatibility of subagent commands, always use the equals character in property assignments.

Initialization variables

Initialization variables, also called *inivars*, modify the default behavior of the master agent and subagents.

To set the value of an inivar, use the command:

```
set inivar name=value
```

The agent stores initialization variable definitions in the initialization file `init.cfg`. See "Initialization file (init.cfg)" on page 63 for more information about this file.

Table 20 lists the inivars available for the master agent. Details about subagent inivars are available in the *Netcool/SSM Reference Guide*.

Note: When not explicitly defined, most inivars assume a default value.

Table 20. Inivar descriptions

Parameter	Argument	Description
`AgentCfg`	*string*	The name of the agent startup configuration file. Default: `agent.cfg`.
`AgentState`	*string*	The name of the agent state configuration file. Default: `agent.state`.
`ConfigRemoveOnFail`	`true|false`	When `true`, control rows created using configuration commands are removed if activation fails.
`ConsoleFilter`	`on|off`	By default, ConsoleFilter is `on` and messages sent to the log file are not sent to the console. To have messages sent to the console, set `ConsoleFilter=off`. **Tip:** To temporarily disable the console filter so that snmp trace messages can be seen in the console, set `snmptrace=on`.
`ConsoleIdleTimeout`	*integer*	The maximum time (in seconds) that a command console connection may remain idle (that is, with no commands sent) before it is automatically disconnected by the agent. Default: 300 seconds.
`ConsoleMaxPasswordRetries`	*integer*	The maximum number of retries that a command console user is allowed to correct the password entered before the agent disconnects. Default: 2, which represents a total of 3 password attempts.
`ConsolePassword`	*string*	The password for the command console in encrypted format.

Table 20. Inivar descriptions (continued)

Parameter	Argument	Description
ConsoleSendTimeout	*integer*	The time (in seconds) that the agent will wait for the command console to accept a line of output before it is discarded. Default: 5 seconds.
ControlRowTimeout	*integer*	The length of time (in seconds) that idle control rows can exist before being deleted. An idle control row is one in which no objects have been set. A value of 0 indicates no timeout.
DefaultAddress	*IP address*	Specifies a substitute IP address used when the agent reports its IP address and contained in the AgentAddr variable binding in SNMPv1 trap PDUs.
DefaultDataSource	*string*	Specifies a data source to use by default based on its interface description (ifDesc). If this inivar is not set, the first monitored interface with or without a non-zero IP address is used as the data source.
DisableV1	true|false	Controls the agent's response to incoming SNMPv1 requests: false - The agent responds true - The agent does not respond **Tip:** This inivar is dynamic. It is not necessary to reboot the agent for changes in its value to take effect.
DisableV2	true|false	Controls the agent's response to incoming SNMPv2c requests: false - The agent responds true - The agent does not respond **Tip:** This inivar is dynamic. It is not necessary to reboot the agent for changes in its value to take effect.
DisableV3	true|false	Controls the agent's response to incoming SNMPv3 requests: false - The agent responds true - The agent does not respond **Tip:** This inivar is dynamic. It is not necessary to reboot the agent for changes in its value to take effect.
DisableVACM	true|false	When true, view-based access control is disabled.
EngineBoots	*integer*	Stores the number of times that the Netcool/SSM agent has initialized itself since snmpEngineID was last configured.

Table 20. Inivar descriptions (continued)

Parameter	Argument	Description
EventLogLimit	*integer*	Specifies the maximum number of event log entries for an RMON eventRow. If this value is non-zero and the number of log entries for a control row exceeds this number, the oldest entry is removed before a new one is added.
ExtendedAuthTraps	true|false	Enables inclusion of additional information in authentication-failure traps. The additional information is defined in the notification type agentAuthenticationFailure. **Tip:** To enable authentication-failure traps, set the MIB object snmpEnableAuthenTraps to enabled(1).
ForceIndexesReadable	true|false	Columnar index objects are by default not-accessible. Some NMS applications do not handle this very well. Setting this variable to true makes all index objects read-only.
LogCount	*string*	Controls the maximum number of rolled log files. By default, log files are rolled to ssmagent.log.1, ssmagent.log.2, and so on. Use LogFile to change the name of the rolled log files.
LogFile	*string*	The name of the Netcool/SSM log file.
LogLevel	*integer*	Sets the amount of data logged in the log file: 1 - fatal, least verbose 2 - warning 3 - information 4 - verbose 5 - debug, most verbose
LogSize	*integer*	Sets the maximum log file size (in bytes) before log rotation occurs.
MonitorRemoteFS	true|false	Enables remote file system monitoring.
NetifDisable	true|false	Disables packet capture: true - Packet capture is disabled false - Packet capture is enabled
NotificationQueueLimit	*integer*	Sets the maximum number of notifications allowed in the agent's notification processing system at any time.
ProtocolDescCfg	*string*	The name of the Netcool/SSM protocol description file. Default: pdesc.cfg.
ProtocolDirCfg	*string*	The name of the Netcool/SSM protocol directory configuration file. Default: pdir.cfg.

Table 20. Inivar descriptions (continued)

Parameter	Argument	Description
ProtocolDirMode	wildcard | encaps	The protocol directory can be configured to run in one of two modes: wildcard - Only wildcard protocols are visible in the protocol directory. Statistics for each protocol are aggregated and classified under the appropriate wildcard protocol. encaps - All encapsulations of all protocols are visible in the protocol directory; these protocols are used to populate MIB data tables (specifically, RMON2 tables). When this variable is set to encaps, the protocol directory contains a superset of those protocols visible in wildcard mode.
RestoreState	true|false	When true, the agent reloads its state from the file specified by the AgentState variable when preforming a cold boot.
SnmpPduSize	*integer*	Sets the size limit (in bytes) of SNMP PDUs sent by the agent.
SysDescrOverrideValue	*string*	If specified, overrides the MIB-2 system description (sysDescr) variable. To override sysDescr, stop the agent and edit this value.
SysObjId .1.3.6.1.4.1.*x*...		Defines the unique system ID, where x... represents a unique ID forNetcool/SSM.
TraceLogFile	*string*	Specifies the absolute path and filename of the log file to which SNMP trace data is output when trace data logging is enabled using the set snmptrace command. Default: trace.log in the Netcool/SSM log directory.
TrapDiscoverTimeout	*integer*	The amount of time (in seconds) after which the agent abandons attempts to determine the engine ID of a remote entity.
TrapLogFile	*string*	Specifies the absolute path and filename of the log file in which the agent stores details about generated traps. Traps are only logged if the value of the configuration variable TrapLogging is on. Default: traps.log in the Netcool/SSM log directory.
TrapLogging	on|off	Controls the trap logging function: on - The agent logs generated traps either in the file specified by the traplogfile configuration variable or, if this variable is not defined, in the default file traps.log located in the agent log directory. off - Trap logging is disabled. Default: off.

Table 20. Inivar descriptions (continued)

Parameter	Argument	Description
`TrapVersion`	`1 | 2`	Sets the default value of the `agentTrapDestVersion` object in for rows created in `agentTrapDestTable`: 1 - SNMP v1 traps 2 - SNMP v2 traps
`UdpPort`	*`integer`*	Specifies the UDP port on which Netcool/SSM agent listens for SNMP queries. Any valid port number may be used. Setting this inivar to the value `-1` completely disables SNMP communications.
Variable names are case-sensitive.		

Agent configuration files

Configuration files are text files that contain a set of variable definitions and configuration commands to be executed in order to control the way the agent operates.

Executing configuration files

You can execute configuration files manually from the command console.

About this task

To execute a configuration file, use the command:

```
config execute file
```

The parameter *`file`* represents the path and filename of the configuration file to be executed. Omitting the file's path from this parameter indicates that the file is located in the Netcool/SSM configuration directory (`config`). Omitting the *`file`* parameter altogether executes the configuration file `agent.cfg`.

Creating configuration files

You can create your own configuration files. They provide a convenient way to configure multiple agents identically. They are also useful in situations where you often have to change an agent's configuration to perform different tasks, because you can create a separate configuration file for each task then simply execute the appropriate file for each task.

The following guidelines apply to creating configuration files:

- Agent configuration files are normally located in the Netcool/SSM configuration directory (`config`); however, you can execute configuration files from any directory accessible to the agent. The configuration directory contains a number of configuration files, all of which can be activated concurrently on the agent.
- You can save a configuration file using any valid filename. Configuration files usually have the filename extension `.cfg`, however this is not mandatory.

- You can add comments to your configuration files. Any text in a configuration file that follows a hash symbol (#) is interpreted as a comment and is not executed. Hash symbols inside quote characters are not interpreted as comment delimiters.
- Configuration files may contain any of the commands described in Appendix A, "Configuration commands," on page 157. The commands in a configuration file are executed sequentially, in the order in which they appear in the file.

Files installed with Netcool/SSM

Netcool/SSM contains a number of standard application, system and configuration files. You can modify some of these files to control the operation and configuration of the agent.

Executable files

Executable file components are program executables and libraries installed by Netcool/SSM.

These files are located in the Netcool/SSM `bin` directory.

Table 21 lists the files installed on Windows systems. Table 22 lists those installed on UNIX systems.

Table 21. Executable Windows Files

Filename	**Description**
name.dll	Support libraries and subagents.
ssmagent.exe	Agent executable.
ssmagentsc.exe	Netcool/SSM Service Controller.
ssmcons.exe	Command console.
mibexplorer.exe	Netcool/SSM MIB Explorer.

Table 22. Executable UNIX Files

Filename	**Description**
lib*name*.so/sl	Support libraries and subagents.
ssmagent	Agent executable.
ssmcons	Netcool/SSM command console.

Log files

Log files record significant events related to the operation of Netcool/SSM. You can consult these files in the event of an error to help identify the cause.

By default, these files are located in the Netcool/SSM `log` directory.Table 23 on page 61 lists the installed log files.

Table 23. Log Files

Filename	Description
ssmagent.log	The file in which Netcool/SSM logs details about its operations. You can change the name of this file by setting the configuration variable LogFile. The amount of detail contained in the file is controlled by the configuration variable LogLevel. For more details about these two variables see Table 20 on page 55. The size of this file is limited to 1 MB. When the file's size reaches this limit, Netcool/SSM creates a new log file and renames the old file to *file_name*.log.old, where *file_name* represents the original name of the log file.
Trap log files	The file in which Netcool/SSM logs details of the traps it generates. See "Trap log files" for more details.

Trap log files

You can configure the agent to log details about each trap that it generates. The initialization variable (or *inivar*) TrapLogging controls the operation of this function. When this variable is set to on, the function is enabled.

By default, the agent logs trap details in the file traps.log, which is located in the log directory; however, you can set the path and filename of the log file using the TrapLogFile inivar.

The general format of each trap log is as follows:

```
Trap_date_and_time
Dest:address:port
Version: trap_version
Community: target_community
Pdu Type: PDU_type
Request-id: ID
Error-status: status
Error-index: index
num variable bindings
object_ID object_type value
```

The maximum allowed size of the trap log file is 1 MB. When the file's size reaches this limit, the agent creates a new, empty trap log file in which it logs any subsequent traps and renames the old file to *log_name*.x.old, where *log_name* is the name of the original trap log file and x is initially 1 and increments with each occurrence.

For example, if the trap log file has the name traps.log, when its size reaches the 1MB limit, the agent renames it to traps.log.1.old, creates a new file traps.log and logs any further traps in this new file. When the size of the new file reaches the limit, the agent renames that file to traps.log.2.old and creates a new traps.log file and so on.

Core configuration files

After installation, the Netcool/SSM configuration directory (`config`) contains a set of standard configuration files, known as *core* configuration files, which set up the agent and its subagents, and configure control rows. These files are necessary for the basic operation of the agent—they must exist in the `config` directory for the agent to operate correctly. You can modify some of the core configuration files so that the agent is set up according to your needs.

Attention: Core configuration files serve specific purposes. You should only modify them if you understand the effects of the changes you wish to make. Always create a backup copy of the original file before making any modifications to a core configuration file.

If Netcool/SSM is running on a Windows server cluster, the core configuration files are stored in the `ssm_cfg` directory located on the quorum drive of the cluster.

Table 24 lists the core configuration files.

Table 24. Core Configuration Files

Filename	Description
`agent.cfg`	The startup configuration file. This configuration file is executed when the Netcool/SSM agent starts or performs a coldboot. See "Agent configuration (agent.cfg)" on page 63 for more details.
`agent.state`	The agent state file. This file stores the state of all control rows. The agent uses this information if it is instructed to restore values.
`init.cfg`	The initialization file. This file defines a number of configuration parameters. This configuration file is executed when the Netcool/SSM agent starts or performs a coldboot. See "Initialization file (init.cfg)" on page 63 for more details.
`pdesc.cfg`	The protocol description file. This file maps ports to protocol descriptions used by various Netcool/SSM enterprise MIBs.
`pdir.cfg`	The protocol directory file. This file stores additional entries to the protocol directory to define new protocols. These additional entries are stored in `pdir.cfg` if requested, and are restored to the protocol directory (`protocolDirTable`) when the agent is restarted.
`warmboot.cfg`	The warmboot configuration file. This file stores the agent configuration when the warmboot occurred and is used to reinitialize the agent during warmboot.
`warmboot.state`	The warmboot state file. This file stores the agent state when the warmboot occurred and is used to reinitialize the agent during warmboot.
Configuration files usually have the extension `.cfg`, but you may omit or change this extension if desired.	

Agent configuration (agent.cfg)

The agent configuration file agent.cfg sets up the general configuration of the agent. It contains commands that perform operations such as setting community strings and loading subagents. The master agent reads this file during startup or when performing a coldboot, and executes the commands that it contains. These commands are global and affect the master agent and all subagents.

Agent configuration file

```
#
# Netcool/SSM configuration file
#
set verbose off
#
# Default Communities
#
community add default public RC
community add default private RC
#
# Subagent Section
#
subagent load agentreg
subagent load datactrl
subagent load agentconfig
subagent load rmonc
subagent load mib2
```

Initialization file (init.cfg)

The initialization file init.cfg contains inivars definitions. Each entry in the file is a name-value pair of format *name=value*, where *name* is the name of the inivar. The master agent reads and loads these definitions during startup or when performing a coldboot.

You may add definitions to this file using the set inivar command or by manually editing the file. For changes in inivar definitions to take effect, you must coldboot the agent or—if the inivar only applies to a particular subagent—unload and reload the subagent.

Attention: Always stop the agent before editing the init.cfg file. If you make changes to this file while the agent is running, the changes will be lost.

Sample initialization file

```
UdpPort=161
ProtocolDirMode=wildcard
TrapVersion=2
SysObjId=.1.3.6.1.4.1.1977.1.6.1977.1
AgentCfg=agent.cfg
AgentState=agent.state
ProtocolDirCfg=pdir.cfg
ProtocolDescCfg=pdesc.cfg
LogFile=..\Log\agent.log
LogLevel=debug
LogSize=1000000
CtrlRowTimeout=0
```

Storing inivars in other files

You can configure the agent to read the value of particular inivars from files other than init.cfg. This is useful when you wish to secure information such as the agent's unique ID in a separate file. To specify a different file, use the following declaration in init.cfg:

```
inivarName__file=otherfile.cfg
```

where *inivarName* is the name of the inivar that you wish to place in another file, and *otherfile*.cfg is the name of that file. For example, to place the definition of the AgentIdUniqueId inivar in the file /etc/ssmunique.cfg, add the following entry to init.cfg:

```
AgentIdUniqueId__file=/etc/ssmunique.cfg
```

Alternatively, you can specify the location of an inivar definition using the set inivar command:

```
set inivar inivarName__file=otherfile.cfg
```

State files (agent.state, warmboot.state)

The state files agent.state and warmboot.state store the state of all control rows configured on the agent and its subagents. The control rows' state is stored as a series of either snmp set commands or subagent configuration commands. When the agent starts up or restarts, it executes these commands, enabling to continue operating in the state it was in prior to its most recent shutdown.

See “Agent startup, restart, and shutdown sequences” on page 43 for full details about the way the agent uses state files.

Chapter 5. Subagents and MIB modules

The system service monitoring capabilities provided by Netcool/SSM are implemented by subagents. Each subagent performs a specific type of monitoring task and is directly coupled to a MIB module, which defines the control and data objects for the monitoring task.

Creating service monitoring applications using Netcool/SSM requires you to configure and utilize the particular subagents that provide the data and functions you need for your monitoring purposes. Your monitoring application performs the following tasks:

1. Load and configure the required subagents on Netcool/SSM.
2. Create and activate monitoring tasks for the subagents.
3. Gather the resulting performance data by polling the subagent's MIB module using SNMP requests.
4. Receive and process any notifications generated by Netcool/SSM and its subagents.

The Netcool/ASM suite of monitors provides additional features for monitoring leading commercial applications and databases. Each Netcool/ASM loads and runs as a Netcool/SSM subagent, and provides a MIB module containing metrics specifically designed for the target application or database.

The information presented here applies to both Netcool/SSM and the Netcool/ASM suite of monitors.

Subagents

Subagent management tasks include loading, unloading, configuring and initializing subagents. Netcool/SSM provides a set of configuration commands for performing these tasks. You can issue them from the command console or include them in configuration files.

Loading and unloading subagents

You can load or unload subagents according to your monitoring requirements. Unloading unused subagents helps prevent Netcool/SSM consuming host resources unnecessarily.

Loading a subagent

Load a subagent to use its features.

About this task

To load a subagent, use the command:

```
subagent load name
```

The subagent name is a shortened version of the loadable module name, which you can use in a non platform-specific manner. For example, the subagent name `genalarm` refers to the subagent called Generic Alarm, which implements the MIB defined in `genalarm-mib.mib`.

Unloading a subagent

Unload a subagent if its features are not required.

About this task

To unload a subagent, use the command:

```
subagent unload {name|id}
```

Listing loaded subagents

List subagents to obtain the name, unique ID, build number and description of all subagents currently loaded on the agent.

About this task

To list subagents, use the command:

```
subagent list
```

To obtain further information about a particular subagent, such as its description, vendor or build date, use the command:

```
subagent info {id|name}
```

The *id* and *name* parameters are those obtained using the `subagent list` command.

Subagent configuration commands

Subagent configuration commands enable you to create and configure control rows in the MIB associated with a subagent and set the value of MIB scalar objects. You can issue them from the command console or include them in configuration files.

Each command applies to a particular subagent and is only available when the subagent is loaded. Most subagents provide one or more of these commands.

Basic syntax

All subagent configuration commands have the same basic syntax.

The general syntax of these commands is:

```
keyword [property=value] ...
keyword create [property=value] ...
keyword createpersistent [property=value] ...
keyword store [property=value] ...
keyword reset
```

Table 25 describes these commands.

Table 25. Subagent configuration command syntax

Command	Description
keyword [*property*=*value*]	Specifies a value for the object identified by *property*.
keyword create [*property*=*value*]	Creates and activates a row in the table associated with *keyword*, setting all row objects for which a property-value pair has been specified since the last *keyword* reset command was executed.
keyword createpersistent [*property*=*value*]	Has the same function as the create command; however, any MIB table rows that you create with this command from within a configuration file are retained whenever the agent boots.

Table 25. Subagent configuration command syntax (continued)

Command	Description
keyword store [*property*=*value*]	Creates a row in a table that requires an association with a control row. The row created using this command is associated with the next control row created using a *keyword* create command.
keyword reset	Resets the property-value pairs associated with the *keyword* configuration command.

Tip: Enter help *command_name* or *command_name* ? from the command console to obtain a listing of the basic syntax for a subagent configuration command. This feature is only available when the subagent is loaded.

Command sequence for create commands

Create commands create and activate table rows.

The general sequence for using create commands is:

```
keyword reset
keyword property=value
keyword property=value
...
keyword create
```

This sequence also applies to createpersistent commands.

Command sequence for store commands

The store commands create rows when the next create or createpersistent command is issued, so you must use the store command prior to the create command.

The general sequence for using store commands is:

```
keyword_A reset
keyword_A property=value
keyword_A property=value
keyword_A store
...
keyword_B create
```

Tip: You can issue multiple store commands before using the create or createpersistent command.

Property-value pairs

Once set, property-value pairs keep their value until they are explicitly redefined; that is, they are persistent within the scope of the keyword originally used for the definition. This is useful when you have to create a number of similar control rows using the same command. Once you set the property-value pairs, you only need to redefine those pairs whose values change in each subsequent control row before issuing the next create command.

Executing the create or createpersistent command without first defining any properties creates a row in which all objects are assigned default values. In some subagent commands, properties are not provided for all row objects. Those objects for which there is no property are set to their default value.

For details about the properties supported in a particular subagent's configuration commands, see the *Netcool/SSM Reference Guide*. This guide also provides examples that show how to use each of the subagent's commands.

Index property

The `create` and `createpersistent` commands provide a special property named `index`. This property sets the index of the row created by a `create` or `createpersistent` command.

You can use the `index` property in commands in the same way you would use any other property:

```
keyword index=value
keyword create
```

If you do not explicitly specify a value for the `index` property, the `create` and `createpersistent` commands create rows using the next available index.

Row persistence

Any MIB table rows created by subagent `create` commands that are executed from within a configuration file are not persistent. That is, they will not be restored if the agent reboots. Netcool/SSM assumes that any configuration you wish to reload is stored in configuration files that are executed when the agent boots.

However, if you need to ensure that a table row created from within a configuration file is restored when the agent reboots, use the `createpersistent` configuration command in place of the `create` command. Any MIB table rows created this way are restored whenever the agent starts.

Note: Do not use the `createpersistent` command in configuration files that are executed each time the agent boots because any rows created by the command are duplicated each time the agent boots.

Subagent inivars

Subagent inivars set parameters for particular aspects of a subagent's operation.

To set the value of an inivar, use the following configuration command:

```
set inivar name=value
```

You may issue this command from the command console or include it in configuration files. Alternatively, you may add inivar definitions to the agent initialization file `init.cfg`. If you change a subagent inivar, you must unload and reload the subagent for the change to take effect.

Note: When not explicitly defined, most inivars assume a default value.

Details about each subagent's inivars are available in the *Netcool/SSM Reference Guide*.

MIB modules

Netcool/SSM MIB modules provide standard SNMP interfaces, however the structure of each MIB module and the objects it defines vary according to its particular application.

For full details about the tables and objects in each MIB module, see the *Netcool/SSM Reference Guide* or consult the MIB definition documents located in the Netcool/SSM `mibs` directory.

Common features

Netcool/SSM MIB modules commonly define control, data, and history tables.

Many MIB modules also define a set of notification types that specify the variable bindings, also known as *varbinds*, for notifications sent as a result of events generated by the subagent.

Control tables

Control tables specify the parameters used for configuring the monitoring operations performed by a subagent.

When you create control rows, you can use the default values for the objects in a control row or provide values explicitly according to the operations you require. Control tables use RMON-style semantics. Table 18 provides a list of common row objects defined in Netcool/SSM control tables.

Table 26. Common Control Row Objects

Row object	**Description**
`Index`	This object uniquely identifies a row in the control table.
`Status`	Sets the status of the control row (see “Creating, activating and destroying control rows” on page 70) [RFC 2578].
`SampleInterval`	Defines the sampling interval (in seconds). Data is collected at the beginning of each sampling interval and sent to the Data table.
`BucketsRequested`	Sets the number of rows (buckets) requested in the history table.
`BucketsGranted`	Indicates the number of rows (buckets) in the history table allocated by the subagent.
`HistoryInterval`	Sets the history interval (in seconds). At the end of each interval, data collected in the data table is transferred to the allocated buckets in the history table.
`Owner`	Indicates the owner of the control row.
`DataControl`	Controls the data collection mechanism. Set this value to `off` if data collection is to be triggered by an event specified by `TurnOnEventIndex` [RFC 1213]. Default value: `on`; data collection starts when Status is active.
`TurnOnEventIndex`	Identifies the external event that triggers the start of data collection by turning the `DataControl` value to on. Set this value to `0` if data collection is not started by an external event [RMON].
`TurnOffEventIndex`	Identifies the external event that stops data collection by turning the `DataControl` value to off. Set this value to `0` if data collection is not started by an external event [RMON].
`EventIndex`	Selects the event that is generated when monitoring fails, indicating that the application or network is not responding [RMON].

Table 26. Common Control Row Objects (continued)

Row object	Description
EventStatus	Controls the number of events generated [RMON]: • eventReady - The control row generates a single event when the event condition is satisfied, after which the value of this object changes to EventFired and must be manually reset to eventReady before another event can be generated. • eventAlwaysReady - Generates an event whenever the event condition is satisfied Default value: eventReady.
References to standards are denoted by square brackets [ ].	

Creating, activating and destroying control rows

Netcool/SSM provides a set of configuration commands for creating control rows, however you can create, activate or destroy a control row by setting its `Status` object directly:

- `createAndGo(4)` creates a control row and immediately activates it using the default values of all control row objects
- `createAndWait(5)` creates a control row but does not activate it. This allows you to modify values manually before activating the control row by setting the `Status` object to `active(1)`.

If a control row requires certain objects to be initialized before it may be activated, the value of the `Status` object is `notReady(3)`. Once all required objects are initialized, the value of `Status` changes to `notInService(2)` and may then be set to `active(1)`.

Active control rows may be deactivated by setting the `Status` object to `notInService(2)`. To remove the control row, set the `Status` to `Destroy(6)`.

Table 27 summarizes the operation of the `Status` object.

Table 27. Row Status

Action	Required Row Status
Activate a row	active (1)
Deactivate a row, but preserve the row parameters	notInService(2)
Indicate that a row requires more data before it can be activated	notReady(3) (this value is set by the agent)
Create a new row and immediately activate it	createAndGo(4)
Create a new row and wait until manual activation	createAndWait(5)
Delete an existing row	destroy(6)

Activating and deactivating control rows using events

Some control tables provide a facility for activating or deactivating a control row in response to an event. In such modules, set the *tableName*`TurnOnEventIndex` and *tableName*`TurnOffEventIndex` control row objects to the index of the events that you wish to activate and deactivate the control row.

Starting and stopping data collection using data control

Many control tables provide a data control object (with name format *tableName*DataControl) that enables you to start or stop the data collection operations defined by a control row without affecting any data already collected. If the value of the data control object is off(2), data collection pauses and data tables are 'frozen'. When the value is on(1), normal data collection takes place.

Note: The Status object of a control row must have the value active(1) before the data control object has any effect.

In some circumstances, you want data collection to start or stop in response to a specific event. Using the dataControlTable, you can configure event-driven activation of data control. This feature is implemented by the datactrl subagent, which is described in detail in the *Netcool/SSM Reference Guide*.

Data tables

Data tables provide storage for data collected by the subagent.

The specific objects contained in data tables vary from subagent to subagent. Each row in a data table has a one-to-one correspondence to a control row, which exclusively owns the data row. Data rows are indexed through the control row's index.

If the subagent collects data periodically, each new data sample overwrites the existing data in the data table. The interval at which new data is collected is usually controlled by a SampleInterval control table object or a subagent inivar.

History tables

History tables provide longer-term storage for data collected periodically by a subagent. The subagent transfers values from the data table to the history table at regular intervals. This allows historical data to be accessed after a data table has been updated.

The control table objects BucketsRequested and BucketsGranted determine the number of history table rows allocated to a control row. Control rows request history rows using BucketsRequested; the number of rows actually allocated by the agent is indicated by BucketsGranted. During operation, if all allocated history table rows are filled, the oldest row is removed from the history table each time a new row of data is transferred to it from the data table. The control row HistoryInterval object determines how often data is transferred from the data table to the history table.

Notification types

Notification types determine the data that is included in notifications sent in response to such events generated by a subagent. Each notification type defines a set of objects (called *variable bindings* or *varbinds*) that are included as part of the information sent in the notification.

Detailed information about the notification types implemented by a particular MIB module is available in the *Netcool/SSM Reference Guide*.

Manipulating MIB objects

Netcool/SSM MIB modules provide a standard SNMP interface, so you can use any SNMP command generator application to manipulate MIB objects. Alternatively, you can use MIB Explorer to do this work. Netcool/SSM also provides a set of configuration commands that implement the SNMP `GET`, `GETNEXT` and `SET` commands. You can issue these commands from the command console or include them in configuration files.

Getting a single object value

To get the value of a MIB object, use the following command:

```
snmp get object_id
```

This command performs an SNMP `GET` operation, displaying on the console the value of the MIB object indicated by the *`object_id`* parameter.

For example, to obtain the value of the object `agentBuildVersion`, use the command:

```
snmp get $agentBuildVersion.0
```

Tip: Use the `$?` variable to access the value returned by the `snmp get` command.

Alternatively, to retrieve the value of the next MIB object relative to a base OID, use the command:

```
snmp getnext object_id
```

This command performs an SNMP `GETNEXT` operation, displaying on the console the value of the next MIB object whose base OID is indicated by the *`object_id`* parameter.

Getting multiple object values

To retrieve the value of all the MIB objects in a sub-tree, use the following configuration command:

```
snmp walk object_id
```

This command performs a set of SNMP `GETNEXT` operations, displaying on the console the values of all MIB objects in the sub-tree whose base OID is indicated by the *`object_id`* parameter.

For example, to obtain values of the objects in the `agentLocation` group, use the command:

```
snmp walk $agentLocation
```

Setting object values

To set the value of one or more MIB objects, use the following command:

```
snmp set object_id type value
```

This command performs an SNMP `SET` operation, changing the value of the MIB object indicated by the *`object_id`* parameter to that specified by the *`value`* parameter. The *`type`* parameter indicates the value's ASN type; see Table 12 on page 50 for a list of allowed types.

You can set multiple MIB objects by including the wildcard (*) in the *object_id* parameter. Using a wildcard effectively specifies an OID sub-tree, in which the command attempts to set all objects to the value specified. This is useful if you wish to set all values in a particular table column, regardless of index value, for example:

```
snmp set $genAlarmControlStatus.* i 1
```

Getting object indexes

To get the row index of a MIB object that contains a particular value, use the following command:

```
snmp match object_id type value
```

This command returns the index of the row containing the MIB object indicated by the *object_id* parameter whose value matches that specified by the *value* parameter. The *type* parameter indicates the value's ASN type; see Table 12 on page 50 for a list of allowed types.

The *value* parameter may contain a regular expression. If this expression matches the value of more than one MIB object, the command returns the index of the first row containing a matching value.

Tip: Use the special variable $? to store and reuse the index value returned by this command, for example:

```
snmp match $hrStorageDescr s .*ZIP.*
zipDriveIndex=$?
...
genalarm var=$hrStorageUsed.$zipDriveIndex
genalarm create
...
```

Chapter 6. Events and notifications

In Netcool/SSM, *events* provide the mechanism for triggering actions in response to a particular situation or condition, such as a performance threshold violation. *Notifications* (also known as *traps*) pass event-related information to network management stations such as Netcool/OMNIbus.

Events

Netcool/SSM implements facilities defined in the RMON standard for providing event-handling capabilities.

Configuring events

Netcool/SSM provides the `event` configuration commands for defining events.

The general syntax of these commands is:

```
event property=value
event create [property=value ...]
event reset
```

Table 28 lists the properties supported in these commands. For general information about using subagent configuration commands, see "Subagent configuration commands" on page 66.

Table 28. Configuration command parameters - event

Property	**Type**	**Description**	**Sets MIB object**
`description`	string	A description of the event.	`eventDescription`

`type`	enum	Selects the action taken when this event is generated. The allowed values are:
		`log`
		`log-and-trap`
		`none`
		`snmp-trap`
			`eventType`

`community`	string	Sets the destination community of any notification sent as a result of this event.	`eventCommunity`

Tip: To manually trigger an event, use the command:

```
set event index
```

where *index* is the value of the event's `eventIndex` object.

MIB tables

The RMON event table (eventTable) is central to the operation of event-handling.

This table is part of the event group in the RMON MIB module, as shown in Figure 9.

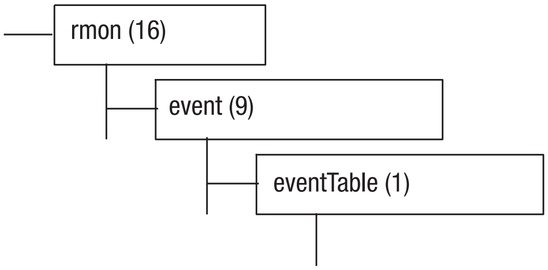


Figure 9. OID tree diagram for eventTable

Each row in eventTable represents an event and defines details about the event. Table 29 lists the row objects in eventTable.

Table 29. eventTable objects

Object	Description
eventIndex	Uniquely identifies a row in the table.
eventDescription	A description of the event.
eventType	Determines the action taken by Netcool/SSM when the event is generated: none(1) - No action. log(2) - Creates a log entry in the event log table logTable. SNMP-trap(3) - Sends a notification to the community indicated by the eventCommunity object. log-and-trap(4)- Creates a log entry in the event log table logTable and sends a notification to the community indicated by the eventCommunity object.
eventCommunity	Specifies the community to which Netcool/SSM sends a notification when this event is generated.
eventLastTimeSent	Value of sysUpTime when the event last occurred.
eventOwner	Owner of the row.
eventStatus	Status of the row.

RMON notification framework

In Netcool/SSM, the RMON notification facilities enable you to send notifications of type SNMPv1 Trap and SNMPv2 Trap over the UDP transport protocol.

Note: Netcool/SSM supports the transmission of SNMPv2 Traps over TCP; however, this is not possible using the RMON framework. To send SNMPv2 Traps over TCP, use the SNMPv3 message processing subsystem. See "SNMPv3 notifications" on page 81 for details.

How Netcool/SSM processes RMON notifications

When an event is generated, Netcool/SSM checks the community string associated with the event and sends a notification to any RMON notification destinations associated with that community.

Figure 10 presents a simplified view of RMON notification processing in Netcool/SSM. The relationship between `eventTable` and `trapDestTable` is one-to-many, enabling multiple trap destinations to be associated with a single community.

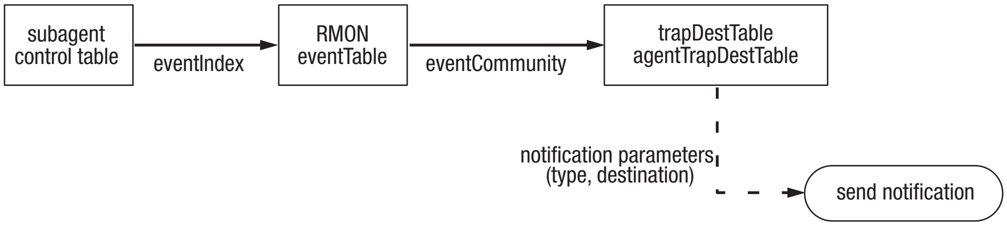


Figure 10. RMON notification processing

Configuring RMON notifications

To configure Netcool/SSM to send RMON notifications, create a notification destination for each IP address that you wish to send notifications to and specify the type of notification to be sent, either SNMPv1 Trap or SNMPv2 Trap.

Creating a notification destination

Netcool/SSM provides the `agenttrapdest` configuration commands for creating notification destinations.

About this task

The general syntax of these commands is:

```
agenttrapdest property=value
agenttrapdest create [property=value ...]
agenttrapdest reset
```

Table 30 lists the properties supported in these commands. For general information about using subagent configuration commands, see "Subagent configuration commands" on page 66.

Table 30. Configuration command parameters - agenttrapdest

Property	Type	Description	Sets MIB object
`community`	string	The destination community of the notification.	`trapDestCommunity` `agentTrapDestCommunity`

Table 30. Configuration command parameters - agenttrapdest (continued)

Property	Type	Description	Sets MIB object
address	string (dec)	The destination IP address and port number of the destination in network-byte order, that is, in a decimal string of format a.b.c.d.e.f where: a.b.c.d represents the IP address of the trap destination .e.f represents the destination port encoded as: .(port div 256).(port mod 256)	trapDestAddress agentTrapDestAddress
transport	enum	Selects the transport protocol used in sending the notification. The allowed values are: tcp - not supported. To send SNMPv2 Traps over TCP, use the SNMPv3 message processing subsystem. See "SNMPv3 notifications" on page 81 for more details. udp - UDP	agentTrapDestTransport
version	enum	Selects the type of notification to be sent: 1 - SNMPv1 Trap 2 - SNMPv2 Trap	agentTrapDestVersion

For example, to create a destination for the IP address 127.0.0.1 and port 8162, for notifications of type SNMPv2 Trap sent over UDP to the community public, use the command sequence:

```
agenttrapdest reset
agenttrapdest community=public
agenttrapdest address=127.0.0.1.31.226
agenttrapdest transport=udp
agenttrapdest create
```

Tip: (8162 div 256) = 31 and (8162 mod 256) = 226

Alternatively, you can use the trapdest add command to create notification destinations, however this command only creates them for SNMPv1 Traps—you cannot send SNMPv2 Traps to destinations created this way. The general syntax of the command is:

```
trapdest add {hostname|ip_address}[:port] community
```

This command creates a row in the RMON trap destination table (trapDestTable). Table 31 on page 79 describes the parameters for this command.

Table 31. Configuration command parameters - trapdest

Parameter	Type	Description	Sets MIB object
{*hostname* |*ip_address*} [:*port*]	string	The IP address or host name and, optionally the port number, of the notification's destination.	trapDestAddress agentTrapDestAddress
community	string	The destination community of the notification.	trapDestCommunity agentTrapDestCommunity

Removing a notification destination

About this task

To remove a notification destination use the command:

```
trapdest remove index
```

This command removes the row from both trapDestTable and agentTrapDestTable whose index (the trapDestIndex or agentTrapDestIndex object, respectively) matches the value of *index*.

MIB tables

The RMON MIB module defines the trap destination table (trapDestTable) for configuring notification destinations.

This table is part of the probeConfig group, as shown in Figure 11.

Trap destination table (trapDestTable)

Each row in trapDestTable defines a destination for notifications generated by Netcool/SSM. Each destination is identified by an IP address and, optionally, a port number.

Table 32 lists the row objects in trapDestTable.

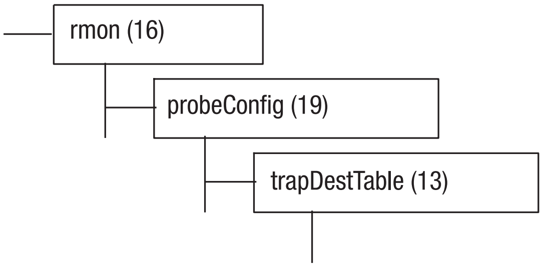


Figure 11. OID tree diagram for trapDestTable

Table 32. trapDestTable objects

Object	Description
trapDestIndex	Unique index for the row.
trapDestCommunity	Community to which these destination IP addresses belong; each time an associated event entry sends a trap due to an event, that trap will be sent to each address in the trapDestTable with a trapDestCommunity value equal to eventCommunity.

Table 32. trapDestTable objects (continued)

Object	Description
trapDestProtocol	Protocol with which to send this trap: ip(1) - IP ipx(2) - *Not supported*
trapDestAddress	IP address (plus port number) to which to send traps on behalf of this entry. If the associated trapDestProtocol object is equal to ip(1), the encoding of this object is the same as the snmpUDPAddress textual convention in RFC1906: For an snmpUDPAddress of length 6, octets 1-4 specify the IP address encoded in network-byte order and octets 5-6 specify the UDP port encoded in network-byte order. That is, the value takes the format: #.#.#.#.p.p, where #.#.#.# is the IP address and p.p is the port address. The standard SNMP port is port 162, which is encoded as 0.162. This object may not be modified if the associated trapDestStatus object is equal to active(1).
trapDestOwner	The owner of the row.
trapDestStatus	The status of the row.

Agent trap destination table (agentTrapDestTable)

Netcool/SSM provides an augmented trap destination table (agentTrapDestTable) that enables you to specify the version of SNMP supported by the destination node. In effect, this table determines the type of Trap—either SNMPv1 or SNMPv2—sent to a notification destination.

This table is part of the Agent MIB module, as shown in Figure 12.

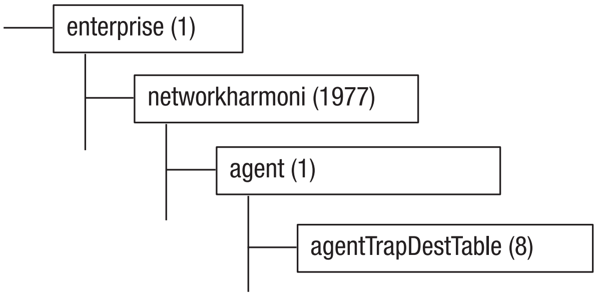


Figure 12. OID tree diagram for agentTrapDestTable

Each entry in agentTrapDestTable also exists in trapDestTable. agentTrapDestTable effectively augments trapDestTable, but for implementation reasons is coupled only by description, rather than by definition. Table 33 on page 81 lists the row objects in agentTrapDestTable.

Table 33. Augmented trap destination table (agentTrapDestTable)

Object	Description
agentTrapDestIndex	This object corresponds exactly to the trapDestIndex in trapDestTable.
agentTrapDestStatus	The status of the row.
agentTrapDestOwner	The entity that configured this entry and is therefore using the resources assigned to it.
agentTrapDestCommunity	This object has the same format and semantics as trapDestCommunity in the RMON2 Trap Destination Table.
agentTrapDestAddress	This object has the same format and semantics as trapDestAddress in the RMON2 Trap Destination Table.
agentTrapDestTransport	This object is deprecated. Its value is always udp(1). To send traps using TCP, use the SNMPv3 framework. See "SNMPv3 notifications" for more details.
agentTrapDestVersion	Selects the type of notification sent: snmpv1(1) - SNMPv1 Trap snmpv2(2) - SNMPv2 Trap

SNMPv3 notifications

Netcool/SSM implements the SNMPv3 message processing subsystem, which you can use to send notifications of type SNMPv3 Trap and SNMPv3 Inform.

Tip: The SNMPv3 message processing subsystem also enables you to send SNMPv2 Traps over the TCP transport protocol. See "Sending SNMPv2 Trap PDUs over TCP" on page 87 for details.

How Netcool/SSM processes SNMPv3 notifications

When an event is generated, Netcool/SSM checks the notification table (snmpNotifyTable) for a row matching the event's community string. The matching row, through a combination of objects in snmpNofityTable and the target address table (snmpTargetAddrTable), defines the notification destination, the type of notification to be sent and the parameters used to send it.

Figure 13 on page 82 presents a simplified view of SNMPv3 notification processing in Netcool/SSM. The relationship between snmpNotifyTable and snmpTargetAddrTable is one-to-many, enabling multiple trap destinations to be associated with a single notification tag.

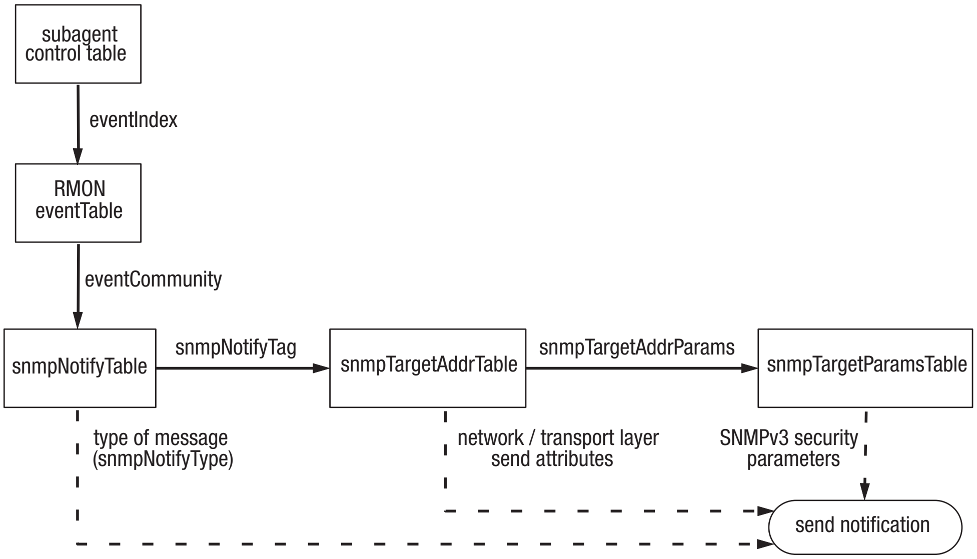


Figure 13. Processing SNMPv3 notifications

Configuring SNMPv3 notifications

The `trapdest` configuration command provides a series of sub-commands specifically for configuring SNMPv3 notifications.

About this task

The general syntax of these commands is:

```
trapdest {addnotify|addtarget|addparam|list|flush|removenotfiy|
removetarget|removeparam} arguments
```

To configure SNMPv3 notifications on Netcool/SSM:

Procedure

1. Use the `trapdest addnotify` command to create a *notification entry*. This associates a community string with a notification tag and specifies the type of message to be sent.
2. *(Optional)* Use the `trapdest set` command to specify the timeout and retry behavior for notification transmission.
3. Use the `trapdest addparam` command to create a set of *notification parameters*. This specifies the message processing model, security model, and security level used when sending the notification.
4. Use the `trapdest addtarget` command to create a *notification target*. This associates the notification tag with a target address for the notification, specifies a transport protocol and indexes a set of parameters for message transmission.

Results

Note: For correct operation of notifications, both the originator (sender) and target (receiver) entities must be configured to use the same SNMPv3 user. For information about creating SNMPv3 users, see “Users” on page 91.

Creating a notification entry

To create a notification entry use the `trapdest addnotify` command.

About this task

The syntax of the command is:

```
trapdest addnotify name tag type
```

This command creates a row in the notify table (`snmpNotifyTable`). Table 34 describes the command's parameters.

Table 34. Configuration command parameters - trapdest addnotify

Parameter	Description
name	The value of a community string to be matched. This parameter sets the value of `snmpNotifyName`.
tag	The tag used to reference the target address table. This parameter sets the value of `snmpNotifyTag`.
type	The type of notification to be generated: `trap` - Trap PDU `inform` - Inform PDU This parameter sets the value of `snmpNotifyType`.

For example, to create a notification entry with the tag `notifyMyStation` that sends Trap PDUs in response to events with a community string `public` use the following command:

```
trapdest addnotify public notifyMyStation trap
```

Tip: To generate Informs instead of Traps, configure a remote user for the target entity with the `user remote add` configuration command. See "Creating a remote user" on page 94 for details.

Setting notification transmission parameters

To set the transmission parameters for notification transmission use the `trapdest setcommand`.

About this task

The syntax of the command is:

```
trapdest set {retries|timeout} value
```

This command sets the transmission parameters for all notification targets created after the command is issued. Table 35 describes the command's parameters.

Table 35. Configuration command parameters - trapdest set

Parameter	Description
`retries`	Sets the number of attempts the agent makes to send the notification before abandoning transmission and registering a failure. This parameter sets the value of the `snmpTargetAddrRetryCount` object of any notification target created subsequently. Default: 3

Table 35. Configuration command parameters - trapdest set (continued)

Parameter	Description
timeout	Sets the transmission timeout period (in ticks). This parameter sets the value of the snmpTargetAddrTimeout object of any notification target created subsequently. Default: 1500

For example, to set a timeout value of 20 seconds for any notification targets created subsequently, use the following command:

```
trapdest set timeout 2000
```

Note: The retry and timeout values of any existing notification targets are not affected by trapdest set commands issued after the notification targets were created. To change the retry or timeout value of an existing notification target, use the snmp set command to set the snmpTargetAddrRetryCount or snmpTargetAddrTimeout objects accordingly.

Creating notification target parameters

To create parameters for a notification target, use the trapdest addparam command.

About this task

The syntax of the command is:

```
trapdest addparam param_tag version sec_model sec_name sec_level
```

This command creates a row in the target parameters table (snmpTargetParamsTable). Table 36 describes the command's parameters.

Table 36. Configuration command parameters - trapdest addparam

Parameter	Description
param_tag	A unique identifier for the parameter table row. This parameter sets the value of snmpTargetParamsName.
version	The message processing model (version) used when generating the notification: 2 - SNMPv2c 3 - SNMPv3 This parameter sets the value of snmpTargetParamsMPModel.
sec_model	The security model used when generating the notification: 2c - The SNMPv2c model (community strings) usm - The user-based security model (USM) any - The agent automatically selects the appropriate model This parameter sets the value of snmpTargetParamsSecurityModel.

Table 36. Configuration command parameters - trapdest addparam (continued)

Parameter	Description
sec_name	The user associated with the notification. If you set the value of the sec_model parameter to 2c, set this parameter to the community string of the trap destination; otherwise set it to the name of the user under which the notifications are sent. This parameter sets the value of snmpTargetParamsSecurityName. For details about creating users, see “Users” on page 91.
sec_level	The security level used when generating the notification: noAuthNoPriv authNoPriv authPriv If you set the value of the sec model parameter to 2c, the only value permitted is noAuthNoPriv. This parameter sets the value of snmpTargetParamsSecurityLevel.

For example, to create the parameters myStationParams that select the SNMPv3 message processing model, the USM security model, the user myStationUser and a security level of authNoPriv, use the following command:

```
trapdest addparam myStationParams 3 usm myStationUser authNoPriv
```

Creating a notification target

To create a notification target use the trapdest addtarget command.

About this task

The syntax of the command is:

```
trapdest addtarget name {ip-address|hostname}[:port] param transport tag [tag ...]
```

This command creates a row in the target address table (snmpTargetAddrTable). Table 37 describes the command's parameters.

Table 37. Configuration command parameters - trapdest addtarget

Parameter	Description
name	A unique identifier for the target address table row. This parameter sets the value of snmpTargetAddressName.
{*ip-address*|*hostname*}[:*port*]	The IP address or host name and, optionally, port number of the notification destination. This parameter sets the value of snmpTargetAddrTAddress.
param	The index of the row in the target parameters table that contains the SNMPv3 parameters used when sending the notification. This parameter sets the value of snmpTargetAddrParams.

Table 37. Configuration command parameters - trapdest addtarget (continued)

Parameter	Description
transport	Sets the transport domain for the target. Specify the domain using one of the following OIDs: udp | 1.3.6.1.6.1.1 - UDP tcp | 1.3.6.1.4.1.1977.200.1 - TCP tcp_persist | 1.3.6.1.4.1.1977.200.2 - TCP persistent This parameter sets the value of snmpTargetAddrTDomain.
tag	A list of tags, separated by space characters, that provide the correlation between the target address table and the notify table. You must supply at least one tag. This parameter sets the value of snmpTargetAddrTagList.

For example, to create a notification target for the management station with IP address and port 127.0.0.1:162 that will receive notifications via the UDP transport domain sent using the message parameters myStationParams for events associated with the notification tag notifyMyStation, use the following command:

```
trapdest addtarget myStation 127.0.0.1:162 myStationParams udp notifyMyStation
```

Removing a notification entry

To remove a notification entry use the trapdest removenotify command.

About this task

The syntax of the command is:

```
trapdest removenotify name
```

This command removes the notify table row whose snmpNotifyName object matches that specified by the *name* parameter.

Removing a notification target

To remove a notification target use the trapdest removetarget command.

About this task

The syntax of the command is:

```
trapdest removetarget name
```

This command removes the target address table row whose snmpTargetAddressName object matches that specified by the *name* parameter.

Removing target parameters

To remove target parameters use the trapdest removeparam command.

About this task

The syntax of the command is:

```
trapdest removeparam name
```

The command removes the target parameters table row whose snmpTargetParamsName object matches that specified by the *name* parameter.

Sending SNMPv2 Trap PDUs over TCP

The SNMPv3 notification framework also enables you to send SNMPv2 Traps over TCP.

Procedure

To configure this feature:

1. Create a notification entry specifying the destination community string, for example:
   ```
   trapdest addnotify public v2TCP_tag trap
   ```
2. Create a notification target corresponding to this notification entry, setting the transport layer to TCP (or TCP persistent), for example:
   ```
   trapdest addtarget TCPstation 127.0.0.1:162 v2TCP_params tcp v2TCP_tag
   ```
3. Create notification parameters for the target. Set the message processing model and security model to SNMPv2c, specify the destination community name, and select the `noAuthnoPriv` security level, for example:
   ```
   trapdest addparam v2TCP_params 2 2c public noAuthNoPriv
   ```

Transmission failover

Netcool/SSM provides a failover mechanism for handling any transmission failures that occur when sending notifications. Normally, when the number of unsuccessful attempts to send an Inform over UDP or a Trap over TCP exceeds the value set by the notification target's `snmpTargetAddrRetryCount` object, the agent abandons further attempts to send the notification. However, using the transmission failover mechanism, you can specify a failover target to which the agent will send the notification in case of transmission failure. The failover target is an alternative destination for the notification.

A failover target is identified by a notification tag of format *`tag_name`*`.FAILOVER.n`, where *`tag_name`* is the name of the notification tag (stored in the `snmpTargetAddrTagList` object of the original target) and `n` commences with the value 1 and increments with each failover tag defined for the target.

To define a failover target, use the `trapdest addtarget` command to create a notification target (a row in `snmpTargetAddrTable`) that has the address and port of the alternative destination and the failover tag formed from the original target's notification tag. For example, to create transmission failover for a target with notification tag `notifyMyStation`, create another notification target with the notification tag `notifyMyStation.FAILOVER.1`.

Where multiple failover tags exist for a particular target, the agent processes the failover tags in ascending order; it processes *`tag_name`*`.FAILOVER.1`, then *`tag_name`*`.FAILOVER.2` and so on. When the list of failover tags is exhausted, the agent abandons transmission altogether.

Note: Transmission failover is not supported for Traps sent over UDP.

MIB tables

SNMPv3 defines a set of MIB tables for configuring the notification mechanism.

Figure 14 shows the sub-tree related to notification configuration in SNMPv3.

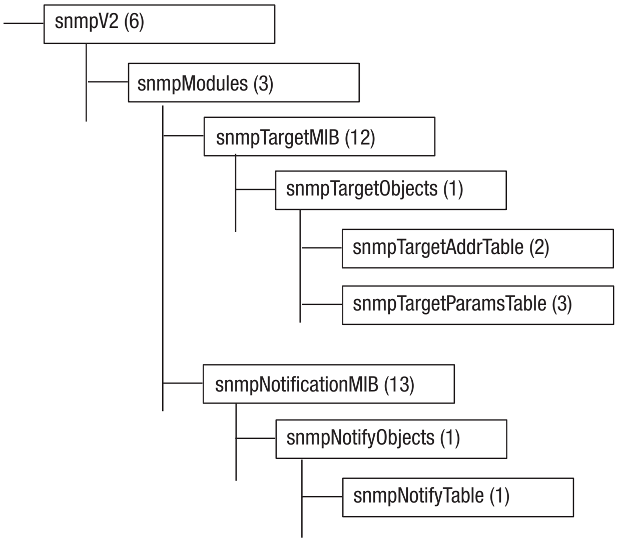


Figure 14. Sub-tree diagram for SNMPv3 Trap-related objects

Notification table (snmpNotifyTable)

The notification table configures the notification generation mechanism.

The `snmpNotfiyName` object corresponds to a community name defined by the `eventCommunity` object in the RMON `eventTable`. When an event is fired, the agent retrieves the event community value from the event's entry in `eventTable` and searches `snmpNotifyTable` for a row whose `snmpNotifyName` object contains a value matching the event community. If a matching value is found, a notification is generated.

Table 38 describes the row objects in `snmpNotifyTable`.

Table 38. snmpNotifyTable Objects

Row Object	Description
`snmpNotifyName`	A unique identifier used to index this table.
`snmpNotifyTag`	A tag value used to reference one or more entries in `snmpTargetAddrTable`.
`snmpNotifyType`	Selects the type of notification to be generated for the entries in the `snmpTargetAddrTable` referenced by `snmpNotifyTag`: `trap(1)` - Generates an SNMPv2c Trap PDU `inform(2)` - Generates an InformRequest PDU
`snmpNotifyStorageType`	Specifies how the row should be stored. The default value is `nonVolatile`.
`snmpNotifyRowStatus`	Controls row creation and deletion.

Target address table (snmpTargetAddrTable)

The target address table specifies the network and transport layer attributes of notification destinations.

Table rows are referenced by the `snmpNotifyTag` object defined in `snmpNotifyTable`. Any number of `snmpTargetAddrTable` rows may be associated with a `snmpNotifyTag` value. Table 39 describes the row objects in `snmpTargetAddrTable`.

Table 39. snmpTargetAddrTable objects

Row Object	**Description**
`snmpTargetAddrName`	A unique identifier used to index this table.
`snmpTargetAddrTDomain`	Specifies the transport domain of the address defined by `snmpTargetAddrTAddress`. The allowed values of this object are: `1.3.6.1.6.1.1` - UDP `1.3.6.1.4.1.1977.200.1` - TCP `1.3.6.1.4.1.1977.200.2` - TCP persistent
`snmpTargetAddrTAddress`	Specifies the target address, which consists of an IP address followed by a UDP port number; for example, `127.0.0.1.0.162`.
`snmpTargetAddrTimeout`	Sets a timeout value (in ticks) for the transmission of InformRequest PDUs or when using TCP-based connections. The agent will wait this amount of time for a response to an InformRequest PDU or TCP connection before attempting again.
`snmpTargetAddrRetryCount`	Sets the number of times that the agent will resend an InformRequest PDU or attempt to establish a TCP connection before abandoning further attempts and logging an error in the agent log file.
`snmpTargetAddrTagList`	A list that provides the correlation between `snmpTargetAddrTable` and `snmpNotifyTable`. When generating a notification, the agent searches this list for the value contained in `snmpNotifyTag`. If the list contains this value, then the agent uses the information in this row to create a destination for the notification. This object may also contain a tag for transmission failover. See "Transmission failover" on page 87 for details.
`snmpTargetAddrParams`	Indexes the row in `snmpTargetParamsTable` that describes the SNMPv3 parameters to be used when sending the notification. If the row specified does not exist, the notification will not be sent.
`snmpTargetAddrStorageType`	Specifies how the row should be stored. The default value is `nonVolatile`.
`snmpTargetAddrRowStatus`	Controls row creation and deletion.

Target parameters table (snmpTargetParamsTable)

The target parameters table specifies the parameters used in sending a notification.

Rows in this table are referenced by the snmpTargetAddrParams object defined in snmpTargetAddrTable. Table 40 describes the row objects in snmpTargetParamsTable.

Table 40. snmpTargetParamsTable objects

Row Object	Description
snmpTargetParamsName	A unique identifier used to index this table.
snmpTargetParamsMPModel	Selects the message processing model used when generating the notification: 1 - SNMPv2c 3 - SNMPv3
snmpTargetParamsSecurityModel	Selects the security model used when generating the notification: 2c - the SNMPv2c model (community strings) USM - the user-based security model (USM) any - the agent automatically selects the appropriate model
snmpTargetParamsSecurityName	Identifies the SNMPv3 user credentials to be associated with the notification. If the user does not exist, the notification will not be sent.
snmpTargetParamsSecurityLevel	Selects the security level used when generating the notification: authPriv - authorization and privacy authNoPriv - authorization, no privacy noAuthNoPriv - no authorization, no privacy See "Users" on page 91 for more details about these security levels.
snmpTargetParamsStorageType	Specifies how the row should be stored. The default value is nonVolatile.
snmpTargetParamsRowStatus	Controls row creation and deletion.

Chapter 7. Security features

Netcool/SSM provides a number of facilities for securing aspects its operation such as message transfer and MIB access, including methods defined in the SNMPv3 standard.

System password encryption

Many Netcool/SSM subagents and Netcool/ASMs require user credentials to access the systems that they are designed to monitor. When configuring control rows in these monitors, you must supply a username and password for the monitored system. To keep those passwords secure, use the Netcool/SSM password encryption facility rather than storing passwords in clear text.

About this task

To encrypt a password, open a Netcool/SSM command console and enter the command:

```
encrypt password
```

where *password* is the password string that you wish to encrypt. In response, Netcool/SSM displays the encrypted password on the command console. Make note of the encrypted password and use it in place of the original password.

To pass an encrypted password as a property in a configuration command use the password assignment operator (`@=`) in place of the normal assignment operator (`=`) as follows:

```
command password_property@=encrypted_password
```

For example, to supply the encrypted password `WMQIa0j0LeVHA` when configuring a control row in the Netcool/ASM for Sybase ASE, use the command:

```
sybasemda password@=WMQIa0j0LeVHA
```

Users

Users provide a mechanism for securing message transfer between SNMP entities. SNMPv3 provides the user-based security model (USM) for securing transactions.

Background

The SNMPv3 standard introduces new features to enhance the security and reliability of SNMP management environments. Two concepts, *entities* and *authoritative engines*, are important in understanding the SNMPv3 features supported in Netcool/SSM.

SNMPv3 entities

An important concept underlying the SNMPv3 features is the modular approach to relationships between SNMPv3 entities such as agents and management stations. Each role performed by an entity, for example sending a trap or replying to a `GET` request, is associated with a particular module. In combination, these discrete modules form a complete entity. Each entity is uniquely identified by its engine ID, which is defined by the object `snmpEngineId`.

The modular architecture of SNMPv3 entities is generally transparent, however it is helpful to keep it in mind when working with SNMPv3 because it differs from the agent-manager relationship typically associated with SNMPv1 and SNMPv2.

Authoritative engines

In SNMPv3 transactions, if an entity receives a message requiring a response, it becomes the *authoritative engine* in that transaction. The authoritative engine in any transaction between entities changes according to the role performed by each entity.

For example, in typical communication between an agent and a management station where the agent receives SNMP queries such as `GET` or `SET` that require a response, the agent is the authoritative engine because it must reply to the message. However, if the agent sends the management station an Inform, the management station becomes the authoritative engine because it must reply to the Inform.

The engine ID of the authoritative engine is sent as part of an SNMPv3 message.

User-based security model

Netcool/SSM supports the USM defined in SNMPv3. The USM secures message transfer by verifying the message's sender and timeliness, a process called *authentication*, and encrypting the messages being sent, a process called *privacy*.

When an SNMP entity sends a message, it does so as a particular user. Associated with the user are two pass phrases, one for authentication and one for privacy.

During transmission of a message, unique identification is formed from a combination of the entity's engine ID and the name of its user. This identification is then coupled with a set of security keys formed by combining the two pass phrases with the engine ID of the authoritative engine. The security keys are used during both the generation and verification stages of message processing.

For message transmission to be successful, the user under which the message is sent must be known to the entity receiving the message. This means that you must configure the same user on both the Netcool/SSM agent and any other entity, such as a management station, that sends requests to the agent or receives notifications from it.

Security levels

A security level is associated with each user. It determines the type of security applied during message transfer. Each security level represents a different combination of the authentication and privacy security measures.

The supported combinations of authentication and privacy are:

- No authentication, no privacy (`noAuthNoPriv`)

  This level is the least secure. It is equivalent to that provided in SNMPv1/v2c.
- Authentication, no privacy (`authNoPriv`)

  This level provides medium security. On packet receipt, the sender is verified.
- Authentication and privacy (`authPriv`)

  This level provides the greatest security. On packet receipt, the sender is verified and packets are sent in encrypted form.

The SNMPv3 standard allows for multiple methods of ensuring authentication and privacy. Netcool/SSM supports the use of either HMAC-MD5-96 (known as MD5) or HMAC-SHA-96 (known as SHA) algorithms for authentication and the CBC-DES Symmetric Encryption Protocol algorithm (known as DES) for privacy.

Configuring users

To connect with the Netcool/SSM agent using the SNMPv3 protocol, management stations must communicate as a user that has already been configured on the agent.

About this task

The `user` configuration command enables you to create, remove and list users on the Netcool/SSM agent. The general syntax of this command is as follows:

```
user {add¦remote add¦list¦remove¦remote remove} arguments
```

Creating a user

Create users with the `user add` command.

About this task

The general syntax of the command is:

```
user add username [{md5¦sha} authpass [des privpass]]
```

The command creates a row in `usmUserTable`. It uses the engine ID of the Netcool/SSM agent to generate the security information of the user. The parameter *`username`* sets the value of the `usmUserName` and `usmUserSecurityName` objects. The security level applied to the user depends on the parameters supplied.

Procedure

- To create a user with no authentication and no privacy (`noAuthnoPriv`), use the command:

  ```
  user add username
  ```
- To create a user with authentication and no privacy (`authnoPriv`), use the command:

  ```
  user add username {md5¦sha} authpass
  ```

Select either MD5 or SHA authentication and provide an authentication password using the authpass parameter. This password is used to generate the authentication key and must be at least 8 characters in length.

For example, to create a user named myAuthUser with SHA authentication and password myAuthPass, use the following command:

```
user add myAuthUser sha myAuthPass
```

- To create a user with authentication and privacy (authPriv), use the command:

  ```
  user add username {md5¦sha} authpass des privpass
  ```

  Supply the authentication parameters described previously. In addition, you must supply a privacy password using the *privpass* parameter. This password is used to generate the encryption key and must be at least 8 characters in length.

  Note: You cannot use the same password for both authentication and privacy.

  For example, to create a user named mySecureUser with MD5 authentication and authentication password myAuthPass, and with DES privacy and privacy password myPrivPass, use the following command:

  ```
  user add mySecureUser md5 myAuthPass des myPrivPass
  ```

User security levels and message transfer:

SNMP entities may send messages at any security level equal to or lower than the security level specified by their user.

For example, an entity communicating as the user mySecureUser created using the command `user add mySecureUser md5 myAuthPass des myPrivPass` is able to send messages using security levels authPriv, authnoPriv or noAuthnoPriv.

SNMP entities may not send messages at a security level higher than that specified by their user. For example, the user myAuthUser defined with the command `user add myAuthUser sha myAuthPass` cannot communicate using the security level authPriv.

Creating a remote user

When the Netcool/SSM agent sends an Inform, the remote entity that receives the Inform must send a response. In this situation, the agent is acting as the non-authoritative engine and the remote entity is the authoritative engine.

About this task

To generate the Inform, the Netcool/SSM agent must know the engine ID of the receiver. By creating a *remote user*, you enable the Netcool/SSM agent to discover the engine ID of the receiver.

The general syntax of the command for creating remote users is:

```
user remote add remote_addr:port username [{md5¦sha} authpass
[des privpass] [engineid]]
```

When you issue this command, the agent discovers the engine ID of the remote entity indicated by the *remote_addr* and *port* parameters, which specify the IP address and port of the remote entity. Alternatively, you may specify its engine ID using the *engineid* parameter. All other parameters and the command usage are identical to those of the user add command described in “Creating a user” on page 93.

By default, the Netcool/SSM agent abandons attempts discover the engine ID of the remote entity after 2 seconds, however you can configure this period by setting the value of the `TrapDiscoverTimeout` variable defined in the configuration file `init.cfg`.

Note: For the `user remote add` command to be successful, the remote entity must be running, or you must specify its engine ID explicitly.

For example, to create a remote user named `myRemoteUser` for a remote entity with IP address `207.53.64.110` on port `162`, supporting SHA authentication with the password `myAuthPass`, use the following command:

```
user remote add myRemoteUser 207.53.64.110:162 sha myAuthPass
```

Listing users

List users with the `user list` command.

About this task

The command lists all the users defined in `usmUserTable`.

Removing users

Remove users with the `user remove` command.

About this task

The syntax of the command is:

```
user remove username
```

The command removes from `usmUserTable` the user whose `usmUserName` object matches that indicated by the *`username`* parameter.

For example, to remove the user named `myAuthUser`, use the following command:

```
user remove myAuthUser
```

Removing remote users

Remove remote users with the `user remote remove`command.

About this task

The syntax of the command is:

```
user remote remove remote_addr:port username
```

The command removes from `usmUserTable` the remote user whose `usmUserName` object matches that indicated by the *`username`* parameter defined for the remote entity with IP address and port indicated by the *`remote_addr`* and *`port`* parameters.

For example, to remove the remote user named `myRemoteUser` defined for the remote entity `207.53.64.110:162`, use the following command:

```
user remote remove 207.53.64.110:162 myRemoteUser
```

MIB tables

The user table (`usmUserTable`) maintains a list of users for the USM. Each row in the table is indexed by a combination of the entity's engine ID and a username. The table also provides facilities for changing the authentication and privacy keys.

Figure 15 shows the MIB sub-tree diagram of objects related to USM.

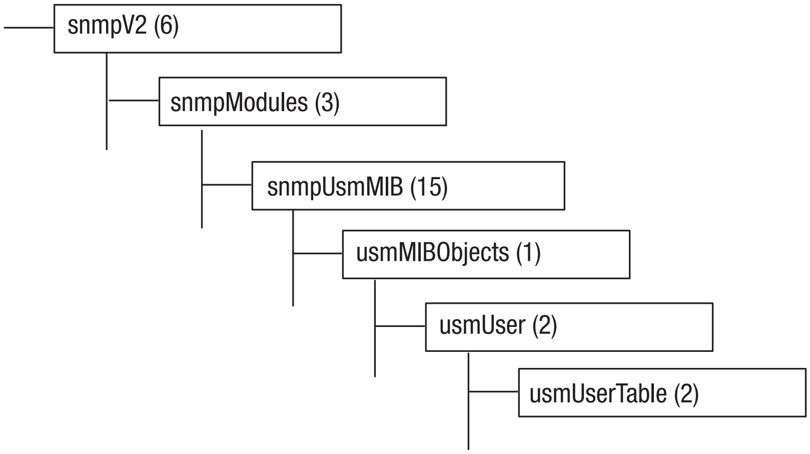


Figure 15. Sub-tree diagram of the USM MIB

Table 41 describes the row objects in `usmUserTable`.

Table 41. usmUserTable Objects

Row object	Description
`usmUserUserEngineID`	The value of the `snmpEngineID` object of the authoritative entity that this user is associated with.
`usmUserName`	The name of a user. This object, together with `usmUserUserEngineID`, indexes rows in this table.
`usmUserSecurityName`	The name of the user, independent of the security model. This object usually has the same value as `usmUserName`.
`usmUserCloneFrom`	Indexes the row from which this row was cloned. Privacy and authentication protocols and keys are copied from that row.
`usmUserAuthProtocol`	Sets the authentication protocol used by this user. The object identifiers of the available authentication options are: • `none - 1.3.6.1.6.3.10.1.1.1` • `MD5 - 1.3.6.1.6.3.10.1.1.2` • `SHA - 1.3.6.1.6.3.10.1.1.3`
`usmUserAuthKeyChange`	Used to change the user's secret authentication key. Reading this object returns an empty string.
`usmUserOwnAuthKeyChange`	Used to change the user's secret authentication key. This object can only be set if the username associated with the message matches `usmUserName`.
`usmUserPrivProtocol`	Sets the privacy protocol used by this user. The object identifiers of the available privacy options are: • `none - 1.3.6.1.6.3.10.1.2.1` • `DES - 1.3.6.1.6.3.10.1.2.2`

Table 41. usmUserTable Objects (continued)

Row object	Description
`usmUserPrivKeyChange`	Used to change the user's secret privacy (encryption) key. Reading this object returns an empty string.
`usmUserOwnPrivKeyChange`	Used to change the user's secret privacy (encryption) key. This object can only be set if the username associated with the message matches `usmUserName`.
`usmUserPublic`	Can be used to validate whether a change to the authentication or privacy (encryption) keys was successful. Modifying this object in the same operation as the key change object provides a means of determining whether the key change operation was successful.
`usmUserStorageType`	Specifies how the row should be stored. The default value is `nonVolatile`.
`usmUserStatus`	Controls row creation and deletion.

Access control using communities

Netcool/SSM enables you to control access to MIB objects by assigning access privileges to communities on a per-module basis. Modules are elements within Netcool/SSM that map a set of MIB objects to an alias; each subagent contains one or more modules.

By setting the module access privileges for a community, you control the type of operations that an entity belonging to that community can perform on the module's MIB objects.

Configuring module access

Netcool/SSM provides a set of commands for configuring a community's access privileges. When you assign access privileges to communities, you do so at the module level rather than at the subagent level.

Three levels of access are available: read-only (`ro`), read/write (`rw`) and read/create (`rc`).

Listing modules

To obtain a list of all modules on Netcool/SSM use the `module list` command.

About this task

The console displays a list of all modules and their IDs.

Listing module access privileges

To obtain a list of the communities that have access to a particular module, and the level of access assigned to each, use the `community list` command.

About this task

The syntax of the command is:

```
community list [module_name¦module_id¦all]
```

The console displays a list of a communities that have access privileges for the specified modules and the level of privileges assigned to each. You may specify the module either by name or ID. Using the keyword `all` lists access privileges for all modules in the subagents currently loaded.

Setting module access privileges

To set a community's access privileges for a particular module, use the `community set` command.

About this task

The syntax of the command is:

```
community set {module_name¦module_id¦all} community [ro¦rw¦rc]
```

If the specified community already has access privileges for the module, the command changes those privileges to the new setting, regardless of the previous level of privileges. You may specify the module either by name or ID. Using the keyword `all` applies the changes to all modules in the subagents currently loaded.

Adding module access privileges

To add access privileges for a particular module, use the `community add` command.

About this task

The syntax of the command is:

```
community add {module_name¦module_id¦all} community [ro¦rw¦rc]
```

If the specified community already has access privileges for the module, the command only alters those privileges if the level of access is greater than that already assigned; it does not decrease the level of access. For example, it does not change an existing `rc` access privilege to `rw`.

You may specify the module either by name or ID. Using the keyword `all` applies the changes to all modules in the subagents currently loaded.

Tip: Use the `community add` command instead of the `community set` command in situations where you wish to ensure that you do not inadvertently reduce the level of module access assigned to a community.

Removing module access privileges

To remove a community's module access privileges, use the `community remove` command.

About this task

The syntax of the command is:

```
community remove {module¦module_id¦all} community
```

You may specify the module either by name or ID. Using the keyword `all` removes the community's access privileges for all modules in the subagents currently loaded.

The default module

Netcool/SSM defines a special module, the `default` module. When you assign a community access privilege for this module, the community automatically obtains that access privilege for all modules loaded thereafter.

Tip: Assigning privileges to the `default` module allows you to define the default access privilege for a community.

View-based access control

View-based access control (VACM) is an SNMPv3 mechanism that regulates access to MIB objects by providing a fine-grained access control mechanism associating users with MIB views. The VACM facilities are essential in ensuring a completely secure agent. *Using SNMPv3 without VACM leaves open a security hole because no restrictions are placed on the level of security that a client must use when accessing MIB objects.*

Attention: Never enable SNMPv3 without first configuring and enabling VACM.

VACM gathers user and security model pairs into *security groups*, which provide a convenient means of identification. Each security group is associated with an *access entry*. Access entries define the access privileges afforded to a security group and they specify the security level that the security group must use in order to access MIB objects. Access entries also list the *MIB views* associated with read, write and notify scenarios. Each MIB view defines a set of *MIB sub-trees* to which a particular access entry is either granted or denied access. MIB sub-trees each consist of a node in the MIB tree hierarchy and all of the node's subordinate elements. You can use bit masks and wildcards in sub-tree definitions.

To enable VACM, set the `DisableVACM` inivar to `false`. VACM is only available with SNMPv3.

Configuring access control

Netcool/SSM provides a set of commands for configuring access control using VACM.

The general syntax for these configuration commands is:

```
vacm {add group¦add entry¦add view¦list¦remove group¦remove
group¦remove view} arguments
```

General procedure

The general procedure for using VACM is to create groups, access entries, and MIB views.

To set up view-based access using VACM:

- Create a security group for a user and security model pair with the `vacm group add` command.

  See “Creating a security group” on page 100 for details.
- Create an access entry using the `vacm add entry` command.

  See “Creating an access entry” on page 100 for details.
- Define the MIB views for the read, write and notify scenarios using `vacm add view` commands.

  See “Creating a MIB view” on page 101 for details.

Creating a security group

To create a security group for a username and security model pair use the `vacm add group` command.

About this task

The syntax of the command is:

```
vacm add group security_group security_name security_model
```

The command creates a row in `vacmSecurityToGroupTable`. Table 42 describes the command's parameters.

Table 42. vacm add group command parameters

Parameter	Description
security_group	The name of the security group, which must be unique among security groups. This parameter sets the value of `vacmGroupName`.
security_name	The name of the user. The value of this parameter is usually the value of a `usmUserName` object.
security_model	The name of the security model or models that this security group applies to. Allowed values are: `usm` - USM is currently the only supported security model.

For example, to associate the user `myUser` and security model USM with the group `myGroup`, use the following command:

```
vacm add group myGroup myUser usm
```

Creating an access entry

To create an access entry that associates a security group with a MIB view, use the `vacm add entry` command.

About this task

The syntax of the command is:

```
vacm add entry security_group security_model {authPriv¦authNoPriv¦noAuthNoPriv}
{read_view¦none} {write_view¦none} {notify_view¦none}
```

This command creates a row in `vacmAccessTable`. Table 43 describes the command's parameters. You may create more than one access entry for a given security group.

Table 43. vacm add entry command parameters

Parameter	Description
security_group	The name of security group that indexes this row. This parameter must contain the value of a `vacmGroupName` object already defined in `vacmSecurityToGroupTable`.
security_model	Selects the security model or models that this mapping applies to: `usm` - USM is currently the only supported security model.
`{authPriv¦authNoPriv¦ noAuthNoPriv}`	Selects the security level that this mapping applies to. For security models 1 and 2c, set this parameter to `noAuthNoPriv`.

Table 43. vacm add entry command parameters (continued)

Parameter	Description
{*read_view*¦none}	Sets the MIB view for which this mapping provides read access. The read view applies Read class PDUs, that is, PDUs containing Get and GetNext requests. This parameter must contain the name of a MIB view (a vacmViewTreeFamilyViewName object) defined in vacmViewTreeFamilyTable. If the MIB view is not yet defined, set this parameter to none.
{*write_view*¦none}	Sets the MIB view for which this mapping provides write access. The write view applies to Write class PDUs, that is, PDUs containing Set requests. This parameter must contain the name of a MIB view (a vacmViewTreeFamilyViewName object) defined in vacmViewTreeFamilyTable. If the MIB view is not yet defined, set this parameter to none.
{*notify_view*¦none}	Sets the MIB view for which this mapping provides notify access. The notify view applies to Notify class PDUs, that is, Trapv2 and InformRequest PDUs. This parameter must contain the name of a MIB view (a vacmViewTreeFamilyViewName object) defined in vacmViewTreeFamilyTable. If the MIB view is not yet defined, set this parameter to none.

For example, to assign the group myGroup access to the MIB view named allAccess for all types of messages when using a security level of noAuthNoPriv, use the following command:

```
vacm add entry myGroup usm noAuthNoPriv allAccess allAccess allAccess
```

Creating a MIB view

To create a MIB view, use the vacm add view command.

About this task

The syntax of the command is:

```
vacm add view name subtree {included¦excluded} [mask]
```

The command creates a row in vacmViewTreeFamilyTable. Table 44 describes the command's parameters.

Table 44. vacm add view command parameters

Parameter	Description
name	The name of the MIB view. This parameter sets the value of vacmViewTreeFamilyViewName.
subtree	The object ID of the MIB subtree that this MIB view applies to. This parameter sets the value of vacmViewTreeFamilySubtree.

Table 44. vacm add view command parameters (continued)

Parameter	Description
{included¦ excluded}	Determines whether access is granted or denied: included - Access is granted excluded - Access is denied This parameter sets the value of vacmViewTreeFamilyType.
mask	The mask bits that in combination with the subtree indicated by *subtree*, determine which object IDs are either included or excluded from the MIB view. For more information about this parameter, see "Using bit masks in MIB view sub-trees" on page 103.

For example, to create a MIB view named allAccess that provides access to all objects, use the following command:

```
vacm add view allAccess .1 included
```

Listing VACM details

To view a, use the vacm list command.

About this task

The command lists the contents of vacmSecurityToGroupTable, vacmAccessTable, and vacmViewTreeFamilyTable.

Removing a security group

To remove a security group use the vacm remove group command.

About this task

The syntax of the command is:

```
vacm remove group security_group security_name security_model
```

The command removes a row from vacmSecurityToGroupTable.

Removing a MIB view

To remove a MIB view, use the vacm remove view command.

About this task

The syntax of the command is:

```
vacm remove view name subtree
```

The command removes a row from vacmViewTreeFamilyTable.

Removing an access entry

To remove a MIB view mapping, use the vacm remove entry command.

About this task

The syntax of the command is:

```
vacm remove entry security_group security_model {authPriv¦authNoPriv¦
noAuthNoPriv}
```

This command removes a row from vacmAccessTable.

Using bit masks in MIB view sub-trees

The bit mask vacmViewTreeFamilyMask, which in combination with vacmViewTreeFamilySubtree determines whether a MIB object lies within a view sub-tree, is a binary mask. The most significant bit of this mask corresponds to the first sub-identifier in vacmViewTreeFamilySubtree, the second most significant bit corresponds to the second sub-identifier in vacmViewTreeFamilySubtree and so on.

The mask bits indicate how an object identifier is compared to vacmViewTreeFamilySubtree when determining whether the object identifier lies within the sub-tree. A bit value of 1 indicates that the sub-identifier at that position must match the corresponding sub-identifier in vacmViewTreeFamilySubtree. A bit value of 0 is equivalent to a wildcard matching any sub-identifier at that position.

If the bit mask contains excess bits, that is, if the mask is longer than the sub-tree identifier, then the excess bits at the least significant end of the mask are ignored during evaluation. If the bit mask is smaller than the sub-tree identifier, the mask is padded with 1s appended to the least significant end of the mask. If vacmViewTreeFamilyMask contains an empty string, this is equivalent to a bit mask consisting entirely of 1s.

For example, consider a view that allows access to an entire row in the ifTable (sub-tree identifier is .1.3.6.1.2.1.2.2) but only when the table index ifIndex has the value 3 (which is equivalent to the sub-tree identifier .1.3.6.1.2.1.2.2.1.0.3). The bit mask required to achieve this is:

1 1 1 1 1 1 1 1 1 0 1 (with equivalent hexadecimal representation FFA0)

The bit corresponding to the tenth sub-identifier, which is the ifTable column object, has the value 0. This enables all column values in ifTable to match. The bit corresponding to the eleventh sub-identifier has the value 1, indicating that the sub-identifier must match the value 3.

To create this sub-tree view, giving it the name if3Access, use the following configuration command:

```
vacm add view if3Access .1.3.6.1.2.1.2.2.1.0.3 included FFA0
```

Ensuring security with SNMPv3

To ensure a completely secure agent, you should configure and enable VACM before enabling SNMPv3.

The following commands create a basic VACM configuration for an administrative user, *adminuser*, providing access to the entire MIB with a security level of authPriv:

```
vacm add group Admin adminuser usm
vacm add entry Admin usm authPriv allAccess allAccess allAccess
vacm add view allAccess .1 included
set inivar DisableVACM=false
set inivar DisableV3=false
config save
```

The following examples demonstrate how to use the vacm commands to configure view-based access control.

- Provide the user myUser read, write and notify privileges to all MIB objects:

```
vacm add group myGroup myUser usm
vacm add entry myGroup usm noAuthNoPriv allAccess allAccess allAccess
vacm add view allAccess .1 included
```

- Restrict the user myUser to certain subtrees if the security level is lowAccess:

```
vacm add group myGroup myUser usm
vacm add entry myGroup usm noAuthNoPriv lowAccess none none
vacm add view lowAccess .1.3.6.1.2.1 included
vacm add entry myGroup usm authNoPriv allAccess allAccess allAccess
vacm add view allAccess .1 included
```

- Restrict the user if3User to read/write access on all enterprise subtrees when the security level is authPriv, and to read access for row 3 of ifTable for any security level:

```
vacm add group if3Group if3User usm
vacm add entry if3Group usm authPriv entAccess entAccess none
vacm add view entAccess .1.3.6.1.4.1 included
vacm add entry if3Group usm noAuthnoPriv if3Access none none
vacm add view if3Access .1.3.6.1.2.1.2.2.1.0.3 included FFA0
```

- Concatenate multiple views in order to grant access to distant areas of a MIB. The user if3User only requires read/write access to the 1977 enterprise branch and the 511 enterprise branch:

```
vacm add group if3Group if3User usm
vacm add entry if3Group usm authPriv entAccess entAccess none
vacm add view entAccess .1.3.6.1.4.1.1977 included
vacm add view entAccess .1.3.6.1.4.1.511 included
```

MIB tables

The snmpVacmMIB implements view-based access control.

Figure 16 shows part of the VACM sub-tree.

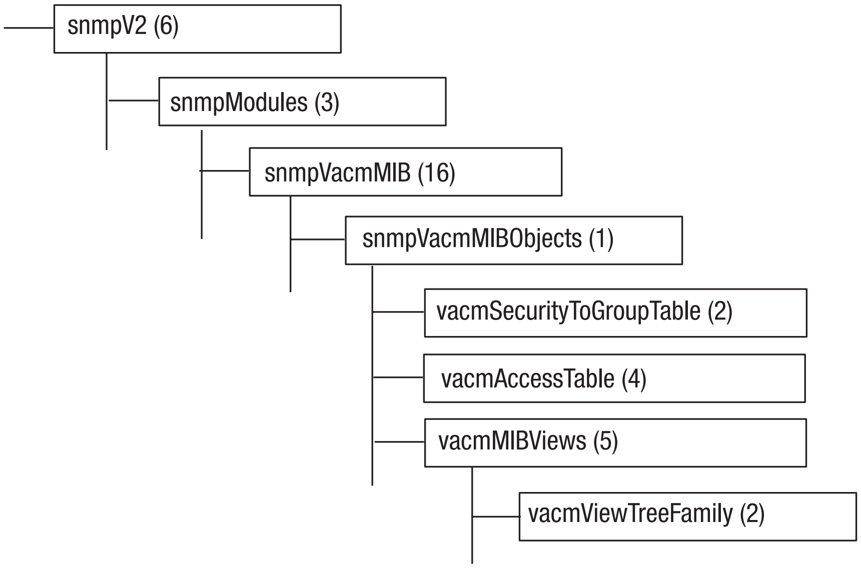


Figure 16. Sub-tree diagram showing a selection of VACM MIBs

The three tables vacmSecurityToGroupTable, vacmAccessTable and vacmViewTreeFamilyTable in combination provide control over access to MIB objects based on a number of parameters associated with the request for access.

- The vacmSecurityToGroupTable maps the message's security model and security name to a group name.
- The vacmAccessTable maps the group name, the security model and the security level to a MIB view according to the message type involved (either read, write or notification).
- The vacmViewTreeFamilyTable determines whether access to the MIB object is granted or denied in the view.

Figure 17 demonstrates the relationship between the VACM tables in controlling access to MIB objects.

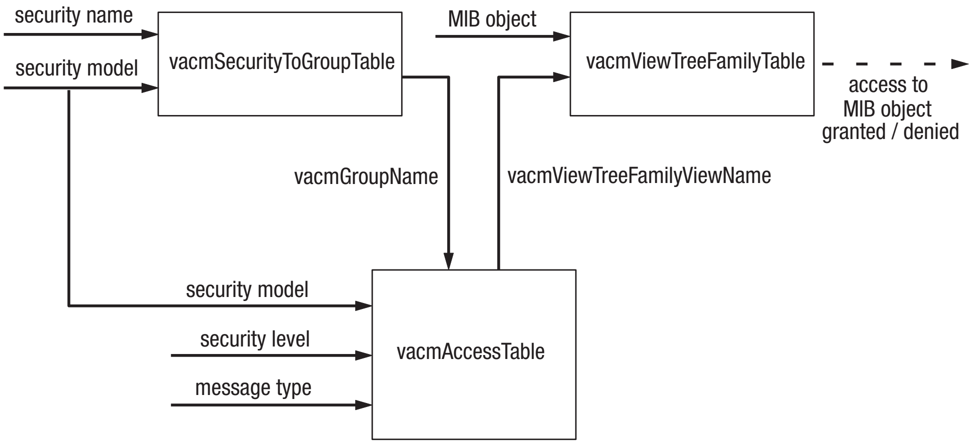


Figure 17. Relationship between VACM tables for access control

Security to group table (vacmSecurityToGroupTable)

vacmSecurityToGroupTable maps a security model and security name (in the USM security model this is a username) to a group name, which indexes entries in vacmAccessTable.

Table 45 describes the row objects in vacmSecurityToGroupTable.

Table 45. vacmSecurityToGroupTable objects

Row object	Description
vacmSecurityModel	The type of security model in use. In combination with vacmSecurityName, it references a row in this table.
vacmSecurityName	Identifies the user. This value is usually the same as usmUserName (see “Users” on page 91 for more details). In combination with vacmSecurityModel, it references a row in this table.
vacmGroupName	The group name. This object forms part of the index for entries in vacmAccessTable.
vacmSecurityToGroupStorageType	Specifies how the row should be stored. The default value is nonVolatile.
vacmSecurityToGroupStatus	Controls row creation and deletion.

Access table (vacmAccessTable)

vacmAccessTable maps a group name, security model, security level, message type and context to a MIB view defined in vacmViewTreeFamilyTable.

Table 46 describes the row objects in vacmAccessTable.

Table 46. vacmAccessTable objects

Row object	Description
ContextPrefix	Used to match against the contextName. The type of match is determined by vacmAccessContextMatch. Contexts are not currently supported.
SecurityModel	The security model to be matched against that specified by an incoming SNMPv3 request.
SecurityLevel	The security level to be matched against that specified by an incoming SNMPv3 request.
ContextMatch	Not supported.
ReadViewName	Identifies the MIB view for which this row authorizes read access. The value of this object references a row in vacmViewTreeFamilyTable via the vacmViewTreeFamilyViewName object. If its value is an empty string or does not match that of any vacmViewTreeFamilyViewName object, no access is granted.
WriteViewName	Identifies the MIB view for which this row authorizes write access. The value of this object references a row in vacmViewTreeFamilyTable via the vacmViewTreeFamilyViewName object. If its value is an empty string or does not match that of any vacmViewTreeFamilyViewName object, no access is granted.
NotifyViewName	Identifies the MIB view for which this row authorizes access for notifications. The value of this object references a row in vacmViewTreeFamilyTable via the vacmViewTreeFamilyViewName object. If its value is an empty string or does not match that of any vacmViewTreeFamilyViewName object, no access is granted.
StorageType	Specifies how the row should be stored. The default value is nonVolatile.
Status	Controls row creation and deletion.

View tree family table (vacmViewTreeFamilyTable)

vacmViewTreeFamilyTable determines whether an object can be accessed within a MIB view. Each row in the table specifies a MIB sub-tree and mask, and indicates whether access to MIB objects within that sub-tree/mask is to be granted or denied.

Table 47 describes the row objects in vacmViewTreeFamilyTable.

Table 47. vacmViewTreeFamilyTable objects

Row object	Description
ViewName	The name of a MIB view. It provides the association between this table and vacmAccessTable.
Subtree	The MIB sub-tree which in combination with vacmViewTreeFamilyMask defines a view sub-tree.

Table 47. vacmViewTreeFamilyTable objects (continued)

Row object	Description
`Mask`	A bit mask which in combination with `vacmViewTreeFamilySubtree` defines whether an object identifier is part of a view sub-tree. The most significant bit of the mask corresponds to the first sub-identifier position, the second bit corresponds to the second sub-identifier and so on. If a mask bit has the value 1, the sub-identifier in the object identifier must match the corresponding sub-identifier in `vacmViewTreeFamilySubtree` for that object identifier to be part of the sub-tree. Mask bits with value 0 are equivalent to wildcards. See “Using bit masks in MIB view sub-trees” on page 103 for more details.
`Type`	Determines the access to a MIB object: • `included(1)` - the MIB object can be accessed • `excluded(2)` - the MIB object cannot be accessed
`StorageType`	Specifies how the row should be stored. The default value is `nonVolatile`.
`Status`	Controls row creation and deletion.

Chapter 8. MIB Explorer

MIB Explorer provides basic tools for monitoring and debugging SNMP agents on your networks.

MIB Explorer enables you to:

- Display a visual representation of MIB and OID structures
- View and graph data from MIB tables
- View output from subagents
- Send SNMP queries to agents
- Receive SNMP notifications

Note: MIB Explorer is only available on Windows platforms.

Component files

MIB Explorer is comprised of a set of component files.

Table 48 lists the MIB Explorer component files.

Table 48. MIB Explorer component files

File	Location	Description
`mibexplorer.cfg`	`config`	MIB Explorer configuration file.
`mibexplorer.exe`	`bin`	MIB Explorer application executable file.
`mibexplorer.log`	`log`	MIB Explorer log file.

Starting MIB Explorer

To launch MIB Explorer select **Start** > **Programs** > **Netcool** > **SSM** > **MIB Explorer**.

About this task

MIB Explorer contains two panes. The information displayed in these panes and the functions provided in these panes depends on the view currently selected. The three available views are:

Procedure

- Host view

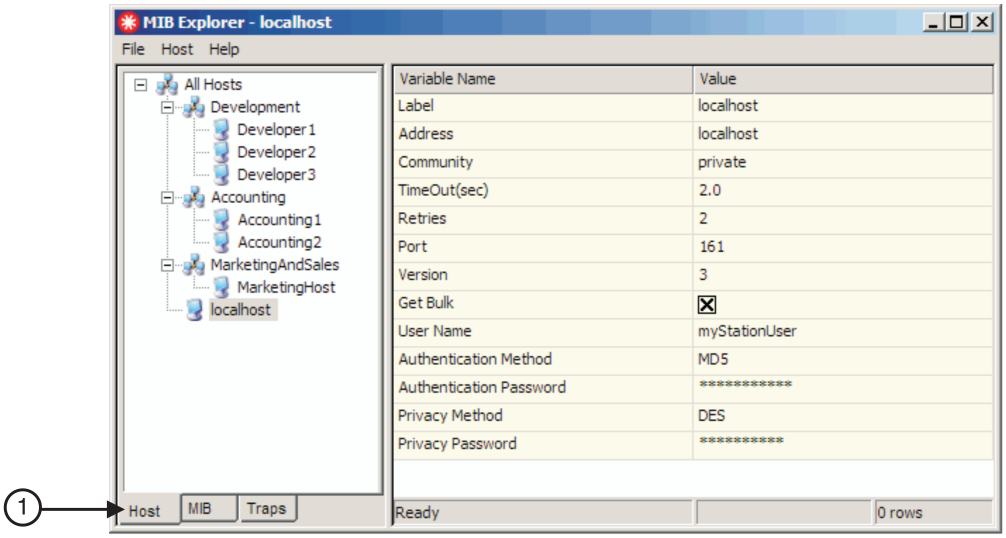
  Enables you to create connections to agents by creating hosts for those agents.
- MIB view

  Presents MIBs in a hierarchical tree structure, provides access to MIB data and graphing functions, and provides a facility for sending SNMP queries.
- Traps view

  Enables trap monitoring.

Results

You can change views by clicking the view tabs located at the base of the left pane (Figure 18).



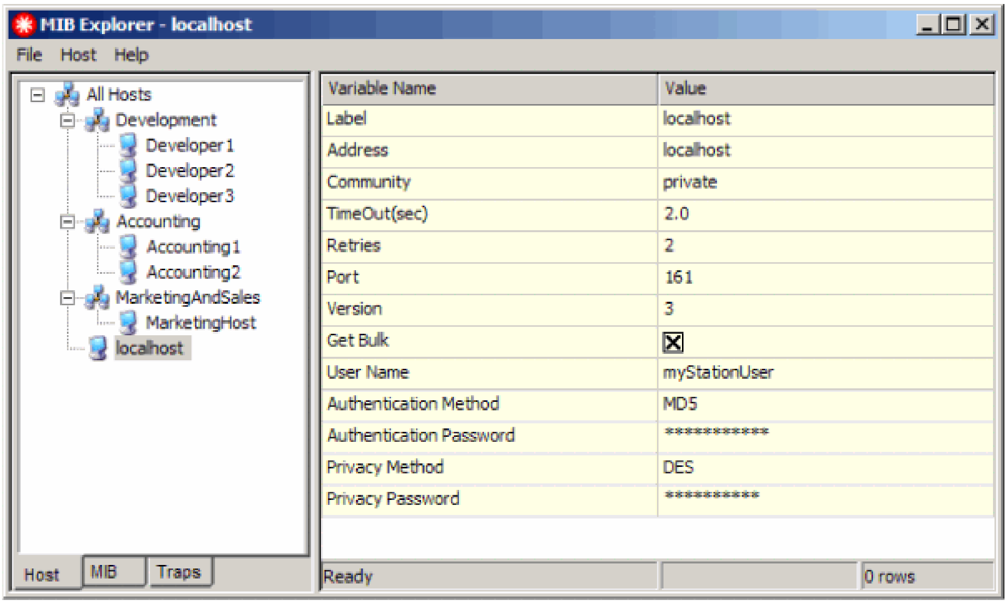
Variable Name	Value
Label	localhost
Address	localhost
Community	private
TimeOut(sec)	2.0
Retries	2
Port	161
Version	3
Get Bulk	☒
User Name	myStationUser
Authentication Method	MD5
Authentication Password	************
Privacy Method	DES
Privacy Password	***********

Figure 18. MIB Explorer view tabs

Using the Host view

Hosts provide access to SNMP agents on your networks. Each host represents a connection to one SNMP agent.

MIB Explorer can maintain multiple hosts, which you access through the Host view (Figure 19 on page 111). The Host view displays a folder tree in the left pane showing all available hosts. Initially, only the local host (that is, the machine on which Netcool/SSM is running) is available, but you can create as many hosts as you require and organize them using the folder tree.



Variable Name	Value
Label	localhost
Address	localhost
Community	private
TimeOut(sec)	2.0
Retries	2
Port	161
Version	3
Get Bulk	☒
User Name	myStationUser
Authentication Method	MD5
Authentication Password	************
Privacy Method	DES
Privacy Password	***********

Figure 19. Host view

Right-click the Host view to display the context menu (Figure 20). This menu provides access to the functions available in the Host view.

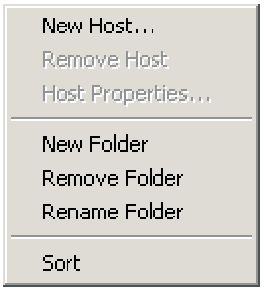


Figure 20. Host view context menu

Managing hosts

The folder tree in the left pane of the MIB Explorer provides a simple way of managing hosts.

About this task

To add a new folder:

Procedure

1. In the left pane, select the folder that is to hold the new subfolder.
2. Right-click and select **New Folder**.
3. The new folder is created.

Results

To rename a folder:

1. In the left pane, select the folder whose name you wish to change.

2. Right-click and select **Rename Folder**.
3. Enter a new name for the folder.

To remove a folder:

1. In the left pane, select the folder that you wish to remove.
2. Right-click and select **Remove Folder**.
3. The folder is deleted, together with any folders and hosts it contains.

To move folders and hosts:

1. In the left pane, select the folder or host that you wish to move.
2. Drag the folder and drop it in the new location.
3. If you move a folder, all the sub-folders and hosts it contains are moved automatically.

Hosts

Hosts define the connections to agents on your network.

To create a host:

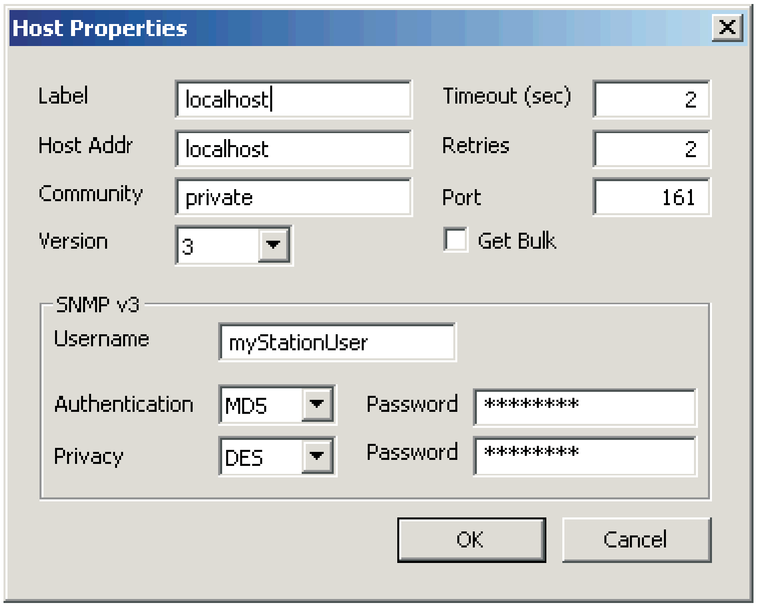
1. In the left pane, select the folder in which you wish to create the new host.
2. Right-click and select **New Host...**

   The Host Properties dialog is displayed.
3. Enter the properties for the new host. For more information on host properties, see Table 49 on page 113.
4. Click **OK**. The new host is created and is displayed in the folder tree in the left pane.

Host properties

Selecting a host in the left pane displays the host's properties in the right pane.

You can edit any host property by clicking the **Value** column of a property. Alternatively, select the host then right-click and select **Host Properties**. The Host Properties dialog (Figure 21 on page 113) is displayed, enabling you to edit the host's properties.



Label: localhost
Host Addr: localhost
Community: private
Version: 3
Timeout (sec): 2
Retries: 2
Port: 161
□ Get Bulk
Username: myStationUser
Authentication: MD5 Password: ********
Privacy: DES Password: ********

Figure 21. Host properties dialog

Table 49 describes the host properties.

Table 49. Host properties

Property	Description
Label	Label that identifies the host.
Host Addr	Textual name or the host IP address.
Community	SNMP community string used in queries to the agent.
Timeout	Sets the timeout period for SNMP requests sent by the MIB Explorer. If a response to a request is not received within this period, the MIB Explorer will retransmit the request a number of times before abandoning the request altogether.
Retries	Number of retransmissions the MIB Explorer performs before abandoning further attempts to send a request.
Port	UDP port to which queries are sent.
Version	The version of SNMP protocol used.
Get Bulk	Enables SNMPv2 `GetBulkRequest` operations.
SNMP v3 group (for complete information on SNMPv3 user settings, see “Users” on page 91).	
Username	Specifies the SNMPv3 username that MIB Explorer uses when communicating with an SNMPv3 agent. To communicate successfully with a particular agent, the username used by MIB Explorer must be the same as that used by the agent.
Authentication	The drop-down list specifies the type of authentication algorithm that MIB Explorer uses when communicating with an SNMPv3 agent. The password field specifies the password to be used with the authentication algorithm. This password must be at least 8 characters in length. To communicate successfully with a particular agent, the authentication algorithm and password used by MIB Explorer must be the same as that used by the agent.

Table 49. Host properties (continued)

Property	Description
Privacy	The drop-down list specifies the type of encryption algorithm that MIB Explorer uses when communicating with an SNMPv3 agent. The **password** field specifies the password to be used with the encryption algorithm. This password must be at least 8 characters in length. To communicate successfully with a particular agent, the privacy algorithm and password used by MIB Explorer must be the same as that used by the agent.

Using the MIB view

The MIB view displays MIB objects and their data values. Using the MIB view you can view, edit and graph the data contained in MIB objects, and you can send SNMP queries to agents.

The MIB view always operates on the agent that is connected to the host currently selected in the Host view. However, you can perform operations on other agents by selecting another host in the Host view then returning to the MIB view.

The menu bar in the MIB view provides access to all the functions available in the MIB view, and the context menu provides access to a selection of these functions. The following sections describe the MIB view functions.

Viewing MIB data

In the MIB view, the left pane displays the MIB tree structure. Each folder represents a node in the MIB tree, identified by name and sub-OID. You can expand or collapse the MIB sub-tree below any node by double-clicking the node.

About this task

Tip: You can quickly locate and view any loaded MIB module using the **Jump to** list in the left pane.

The right pane displays the objects associated with the node selected in the left pane. By default, editable fields (that is, objects that can be set using an SNMP `set` command) in the right pane are highlighted in yellow and read-only fields in blue.

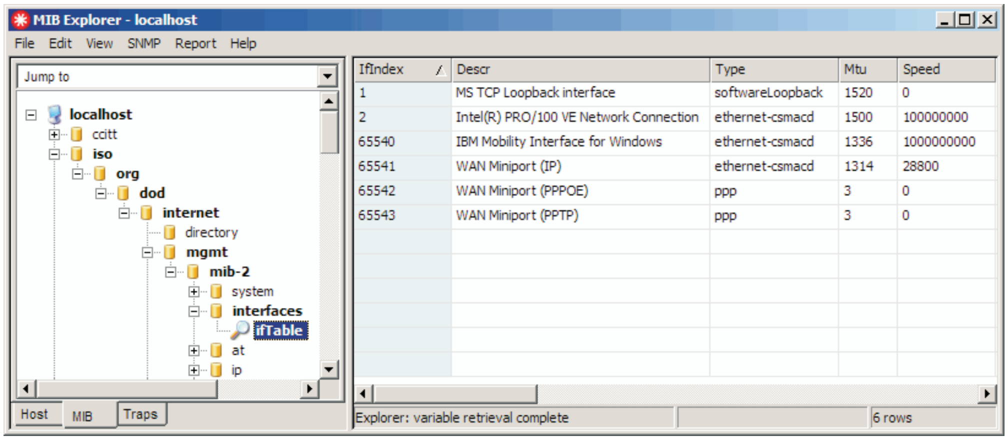
Tip: You can configure field highlighting. See "Viewing options" on page 131 for more details.

Display formats

The MIB view uses different formats to display tables and scalar objects in the right pane.

Tables

The right pane displays tables with a column for each table object and a row for each entry in the table. Figure 22 on page 115 shows the MIB view with the right pane in tabular mode.



IfIndex	Descr	Type	Mtu	Speed
1	MS TCP Loopback interface	softwareLoopback	1520	0
2	Intel(R) PRO/100 VE Network Connection	ethernet-csmacd	1500	100000000
65540	IBM Mobility Interface for Windows	ethernet-csmacd	1336	1000000000
65541	WAN Miniport (IP)	ethernet-csmacd	1314	28800
65542	WAN Miniport (PPPOE)	ppp	3	0
65543	WAN Miniport (PPTP)	ppp	3	0

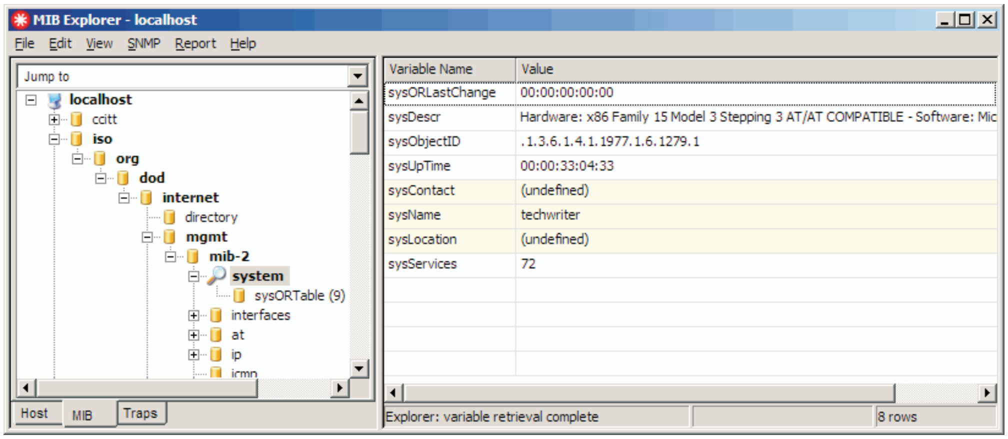
Explorer: variable retrieval complete

6 rows

Figure 22. MIB view with right pane in tabular mode

Scalars

The right pane displays scalar objects with one row per object. The **Variable Name** column displays the name of the object and the **Value** column displays contents of the object (the object's data). Figure 23 shows the MIB view with the right pane in scalar mode.



Variable Name	Value
sysORLastChange	00:00:00:00:00
sysDescr	Hardware: x86 Family 15 Model 3 Stepping 3 AT/AT COMPATIBLE - Software: Mic
sysObjectID	.1.3.6.1.4.1.1977.1.6.1279.1
sysUpTime	00:00:33:04:33
sysContact	(undefined)
sysName	techwriter
sysLocation	(undefined)
sysServices	72

Explorer: variable retrieval complete

8 rows

Figure 23. MIB view with right pane in scalar mode

Selecting the field display format

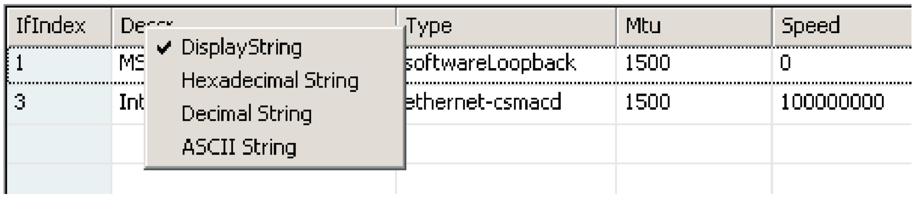
In the MIB view, each object's type determines the format of the data displayed. You can reformat the data for tables and scalars.

Procedure

To modify the display format of an object in tabular mode:

1. In the right pane, right-click in the column head of the table object whose display format you wish to change.
2. The format selection menu is displayed.
3. Select the output format that you require.

   Figure 24 on page 116 shows the display format selection menu in tabular mode.



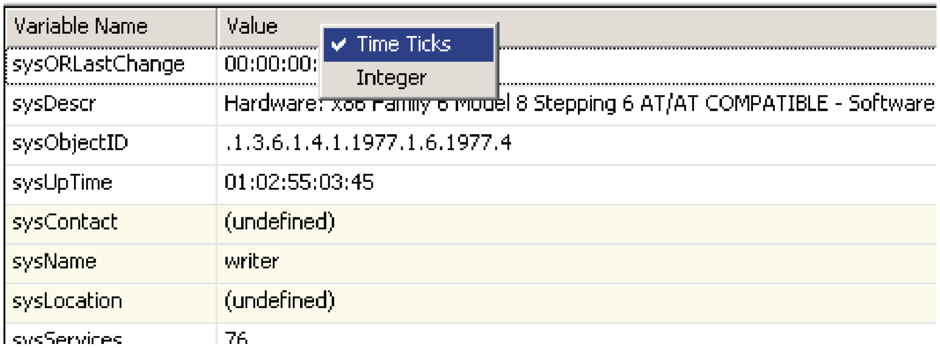
IfIndex	De	Type	Mtu	Speed
1	MS	softwareLoopback	1500	0
3	Int	ethernet-csmacd	1500	100000000

Figure 24. Field display format (tabular mode)

What to do next

To modify the display format of an object in scalar mode:

1. Right-click in the **Value** field of the object whose display format you wish to change.
2. The format menu is displayed.
3. Select the output format that you require.



Variable Name	Value
sysORLastChange	00:00:00:
sysDescr	Hardware: x86 Family 6 Model 8 Stepping 6 AT/AT COMPATIBLE - Software
sysObjectID	.1.3.6.1.4.1.1977.1.6.1977.4
sysUpTime	01:02:55:03:45
sysContact	(undefined)
sysName	writer
sysLocation	(undefined)
sysServices	76

Figure 25. Field display format (scalar mode)

> **Tip:** The field display format does not change the data stored in an object, however the field display format allows you to enter or view the data in the format of your choice.

Sorting tables

When viewing a table, you can sort the table's rows according to any column. By default, table data is sorted by the first column in ascending order. To sort the table rows according to a particular column, click the column's header. Clicking the column header again toggles the sort order between ascending and descending. If you change the sort order in a table, the order is indicated by an arrow in the header of the column by which the table is sorted. The direction of the arrow indicates ascending or descending sort.

Setting MIB objects

The MIB view enables you to set the value of MIB objects.

Procedure

To set an object's value:

1. Double-click the field of the object whose value you wish to set.
2. Enter a value or select a value from the drop-down list.
3. Click outside the cell or press Enter.
4. The new value is set.

Results

Note: Editable fields are displayed with a yellow highlight.

You can also use the SNMP Query function to set MIB objects. See "SNMP query" on page 132 for more details.

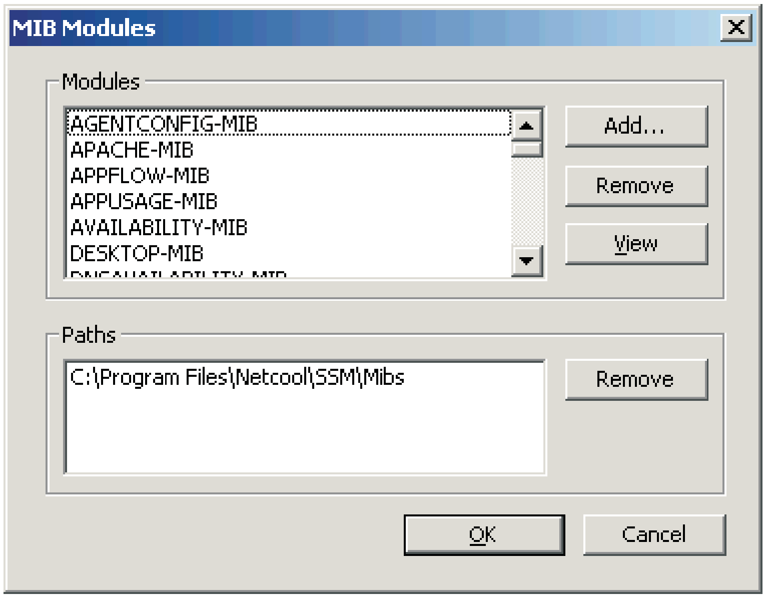
Tip: To change the value of objects in a control row, first set the row's `Status` object to `Not In Service`.

Managing MIB modules

The MIB Modules dialog enables you to add or remove MIBs in MIB Explorer.

About this task

To open the dialog, select **Edit** > **Add/Remove MIB**.



AGENTCONFIG-MIB
APACHE-MIB
APPFLOW-MIB
APPUSAGE-MIB
AVAILABILITY-MIB
DESKTOP-MIB

C:\Program Files\Netcool\SSM\Mibs

In the MIB Modules dialog box, the Modules pane displays the MIB currently loaded on the MIB Explorer. The Paths pane displays a list of paths from which the MIBs have been loaded.

Procedure

To add a MIB module:

1. In the Modules pane, click **Add**.
2. The Open dialog is displayed.
3. Locate the MIB file that you wish to add and click **Open**.
4. The MIB is added and is visible in the left pane of the MIB view.

What to do next

To view a MIB module:

1. In the Modules pane, select the MIB that you wish to view.
2. Click **View**.

3. The MIB file is displayed in a new window.

To remove a MIB module:

1. In the Modules pane, select the MIB that you wish to remove.
2. Click **Remove**.
3. The MIB is removed.

   Note: Removing a MIB does not delete the MIB file from the host system.

Reloading MIBs

If you modify a MIB file while MIB Explorer is running, you must then reload the MIB in MIB Explorer.

About this task

To reload the MIBs, select **Edit** > **Reload MIBs**.

Polling data

MIB Explorer obtains the values of the MIB objects displayed in the right pane by polling the Netcool/SSM agent. The data stored in MIB objects may change regularly, depending on the type of parameter that the MIB object represents. To update the displayed value of a MIB object in the right pane, select **View** > **Refresh** or press F5.

About this task

Refreshing the display prompts MIB Explorer to poll the Netcool/SSM agent and update the displayed values of the MIB objects in the right pane.

If the MIB object data that you wish to view changes regularly, you can set the MIB Explorer to poll the Netcool/SSM agent at regular intervals to ensure that the display in the right pane is updated appropriately. To enable polling at regular intervals, select **View** > **Polling**.

Procedure

To set the polling interval:

1. Select **View** > **Options**.
2. The Options dialog is displayed.
3. In the **Poll Time (secs)** field, enter a polling interval.
4. This value sets the polling interval in seconds.
5. Click **OK**.

Graphing MIB data

MIB Explorer provides a graphing utility for displaying data from a single MIB table.

Data may be displayed in the following graph formats:

- Label - The Label graph format presents data as an enumeration of table values.
- Plot - The Plot format presents data as value versus value graphs.
- Pie - The Pie format presents data in the form of pie charts.
- Real Time - The Real Time format displays data over a time period.

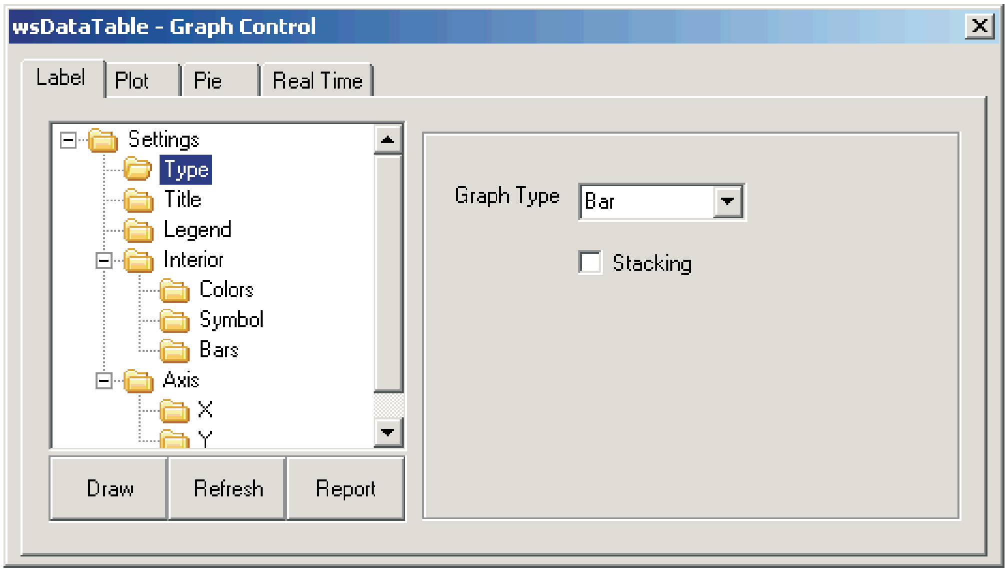
Creating graphs

Create graphs to display the values of tables or lists.

Procedure

To create a graph:

1. Select a folder in the MIB tree that is a table or list node.
2. Select **View** > **Graph...** or right-click and select **Graph...** .
3. The Graph Control dialog is displayed (Figure 26).



Graph Type: Bar

□ Stacking

Figure 26. Graph control dialog

4. Select the tab corresponding to the graph format that you wish to create.
5. In the **Settings** tree, specify the display settings for the graph.

   For details on the various types of settings see:

   a. “Common graph features” on page 120
   b. “Label graph” on page 121
   c. “Plot graph” on page 123
   d. “Pie Graph” on page 123
   e. “Real time graph” on page 124
6. Most settings have default values, however in some graph formats the **Axis** group settings do not have defaults and you must specify values for these settings before you can create a graph.

7. Click **Draw**.
8. The graph is displayed in a new window.

   Tip: When you create a graph, data polling ceases. To collect updated data before drawing the graph, click **Refresh Data**.

Saving graphs

When you have created a graph, you can save it as an image file.

Procedure

1. In the graph window, select **Graph** > **Save to image**.

   The **Save As**dialog is displayed.
2. Select a location for the image, enter a filename and select a graphics format for the saved graph.

   You can select Windows Bitmap, JPEG, or PNG graphics formats.
3. Click **Save**.

Printing graphs

When you have created a graph, you can print it.

Procedure

To print a graph:

1. In the graph window, select **Graph** > **Print**.
2. The Windows Print dialog is displayed.
3. Select the required print settings and click **OK**.

Common graph features

A number of the graph features provided in the **Graph Control** dialog are common to all four graph formats.

Table 50 and Table 51 list these features.

Table 50. Common graph features

Feature	Description
Type	Selects the type of graph: Plot, Bar, Stacking Bar or Area. Selecting the Stacking option shows the cumulative value of multiple series for each data point in the graph. The Type feature is not available in Pie format graphs.
Title	Sets the formatting of the header and footer title.
Legend	Sets the formatting of the graph legend, which indicates the colors and symbols used to denote each data series on the graph.
Interior	Specifies the colors of objects and symbols in the graph.
Axis	Sets the values for the X-axis and Y-axis on Label, Plot and Real Time formats. Specifies the Label and Values for Pie formats.

Table 51. Common interior settings

Setting	Description
Colors	Select the main colors for the foreground and background as well as the plot color for each element in the graph.

Table 51. Common interior settings (continued)

Setting	Description
Symbols	For each element in the graph, select the symbol type from the drop-down list then click on the color to display the color palette. Select the color you require for this element in the graph.

Label graph

The Label graph format presents data from rows in MIB tables, with objects from a row each grouped under one label. Labels, which group the data, are formed from one or more row objects and are displayed on the Y-axis. Data values of the objects in each group are plotted along the X-axis, and are represented as either total or average values.

Note: For formatting reasons, the X- and Y-axis assignments in the Label format are reversed. X-axis settings relate to the vertical axis and the Y-axis settings relate to the horizontal axis.

The **Axis** and **Bar** settings folders determine display settings specific to the Label graph format.

Axis settings

Use the **Axis** settings folder (**Settings** > **Axis**) to select the format for the information displayed on the X- and Y-axes. Table 52 describes these settings.

Table 52. Axis settings

Sub-folder	Setting	Description
X	Title	Sets the title of the graph's vertical axis. By default, the title is the concatenation of the names of the objects selected in the X axis values setting, however you can enter any title you wish.
	X axis values	Specifies the set of row objects by which MIB table rows are grouped. Each unique combination of these X-axis values forms a label.
	Equal Value Function	Selects the function used to determine the displayed data value of table rows that have the same label.
	Max Columns	Sets the maximum number of labels displayed on the graph.
Y	Title	Sets the title of the graph's horizontal axis. By default, the title is the concatenation of the names of the objects selected in the Y axis values setting, however you can enter any title you wish.
	Y axis values	Selects the MIB table objects displayed in the graph. For each Y axis value selected, a bar representing that data is displayed in each label.
	Ranges	Sets the display bounds of the horizontal axis. If the Auto option is enabled, these bounds are determined dynamically, based on the data displayed.
	Logarithmic	Selects a logarithmic scale for the data displayed on the horizontal axis.

Bar settings

Use the **Bars** settings folder (**Settings** > **Interior Bars**) and is only available for Label formats of type Bar or Stacking Bar. Table 45 describes these settings.

Table 53. Bar settings

Setting	Description
Cluster Overlap	Determines the spacing between the bars in a cluster (that is, the group of bars displayed for one label). The valid range is -100% to 100%: 100% - bars overlap each other completely. 0% - bars adjacent to each other. -100% - bars separated by the width of one bar.
Cluster Width	Determines the total width of the cluster (that is, the group of bars displayed for one label) expressed as a percentage of the space allocated on the graph for one cluster. The valid range is 0 to 100%.
Depth	Sets the visual depth of the bars, measured along the Z-axis. The valid range is 0 to 500.
Elevation	Sets the vertical viewing angle of the graph. The valid range is –45° and 45°.
Rotation	Sets the horizontal viewing angle of the graph. The valid range is –45° and 45°.
Shading	Sets the type of shading used for the 3D areas (the depth and height) of the bars.

Figure 27 shows an example of a graph using the Label format with 3D effect settings.

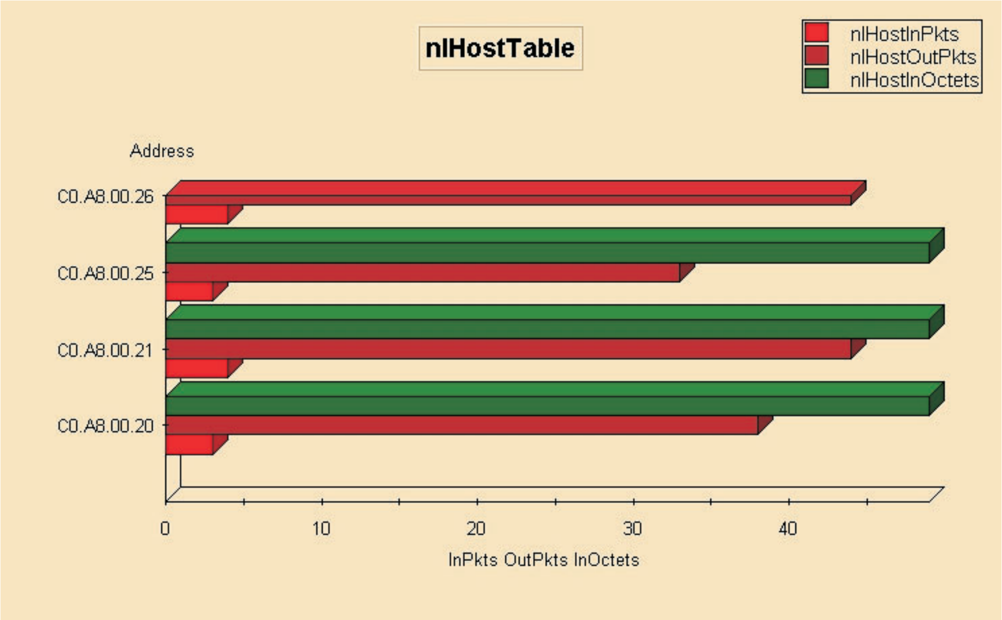


Figure 27. Graph in Label Format

Plot graph

The plot graph presents historical data. The time (or an interval of elapsed time) is usually displayed on the X-axis and values are plotted on the Y-axis. The displayed values can represent items such as packets or octets.

The **Axis** folder provides options for formatting the graph axes, as listed in Table 54.

Table 54. Axis parameters for plot graphs

Parameter	Description
X	The X-axis is a single column from the MIB table that has an integer value. The **Ranges** box contains an **Auto** check box, that makes the highest value on the axis equal to the highest data value. Deselect this box to set the upper and lower ranges. Select the **Logarithmic** box to use a logarithmic scale along the axis.
Y	The Y-axis can have any number of columns of values in the MIB table (shown in Y axis values). Each of these columns represents a different data series in the graph. The **Ranges** box contains an **Auto** check box, that makes the highest value on the axis equal to the highest data value. Uncheck this box to set the upper and lower ranges. Check the **Logarithmic** box to use a logarithmic scale along the axis.
Split Data	Select this option to split the data in the table into separate series based on different values in a selected column. For example, you might want to split data on the control row column to get different plot lines for two network interfaces.

Pie Graph

The pie graph is a type of label graph, presenting data that is related in some way. The key groups like items together into a common segment of the pie. The pie graph's other features are similar to those of the label graph. The pie graph is best used when you want to compare graphed values against the whole.

The **Axis** folder provides options for formatting the graph axes, as listed in Table 55.

Table 55. Axis parameters for pie graphs

Parameter	Description
Label	Rows in the MIB table correspond to labels in the pie graph. The label axis values are concatenated to form a label. When two rows in the MIB table have the same label the **Equal Value Function** determines the action taken.
Value	The size of the slices used in the pie chart. Each column selected represents another piece in the pie chart.

The display option **Pie**, located under **Settings** > **Interior**, is only available for pie graphs. Table 56 describes the parameters that it provides. For information on other display options, see “Common graph features” on page 120.

Table 56. Interior parameters for pie graphs

Parameter	Description
Threshold	The percentage of the smaller values placed in the “Other" slice.
Depth	Sets the visual depth of the bars, measured along the Z-axis. The valid range is 0 to 500.
Elevation	Sets the vertical viewing angle of the graph. The valid range is –45° and 45°.

Table 56. Interior parameters for pie graphs (continued)

Parameter	Description
Rotation	Sets the horizontal viewing angle of the graph. The valid range is –45° and 45°.
Shading	Sets the type of shading used for the 3D areas (the depth and height) of the bars.

Real time graph

The real time graph presents data against time. Time is measured on the X-axis in poll intervals; for example, if the graph poll time is 10 seconds then 100 on the X-axis indicates an interval of 1000 seconds. The Y-axis allows single values to be plotted against time. Single values will be scalars or single values in a table, but not a single row of a table.

The **Axis** folder provides options for formatting the graph axes, as listed in Table 57.

Table 57. Setting axis parameters for real time graphs

Parameter	Description
X	The X-axis represents the amount of time graphing. The poll interval is the time between points in the chart. The time range is the number of polls shown in the chart at once.
Y	The Y-axis is any scalar value chosen in the Series dialog. Plot delta shows the current value subtracted from the previous one. Select the **Auto** check box to scale the Y-axis to the maximum of the values read so far on the chart (this procedure occurs once every ten polls). It is normally better to use the **Auto** facility rather than try to determine the range of Y-axis values manually. Note that when **Draw Graph** is first selected, ten polls occur before the graph is redrawn so that it is displayed properly.
Series	This dialog is used to determine which values are to be graphed. In the case where a table node has been selected, two list boxes are shown. The left list box shows the scalar variables associated with the table node. The right list box shows a list of indexes representing rows from the table. Only by selecting an index in the right list box will a value be selected for graphing. The left list box shows different listings of indexes that can be selected. When a list node is selected, only the left list box is shown. In this case, select a scalar variable to select a value for graphing.

For information on other display options, see “Common graph features” on page 120.

Reports

Reports are pre-formatted graphs. They enable you to quickly display specific selections of MIB data.

Use the Report Manager (Figure 28 on page 125) to manage reports. To open the Report Manager, select **Report** > **Report Manager**.

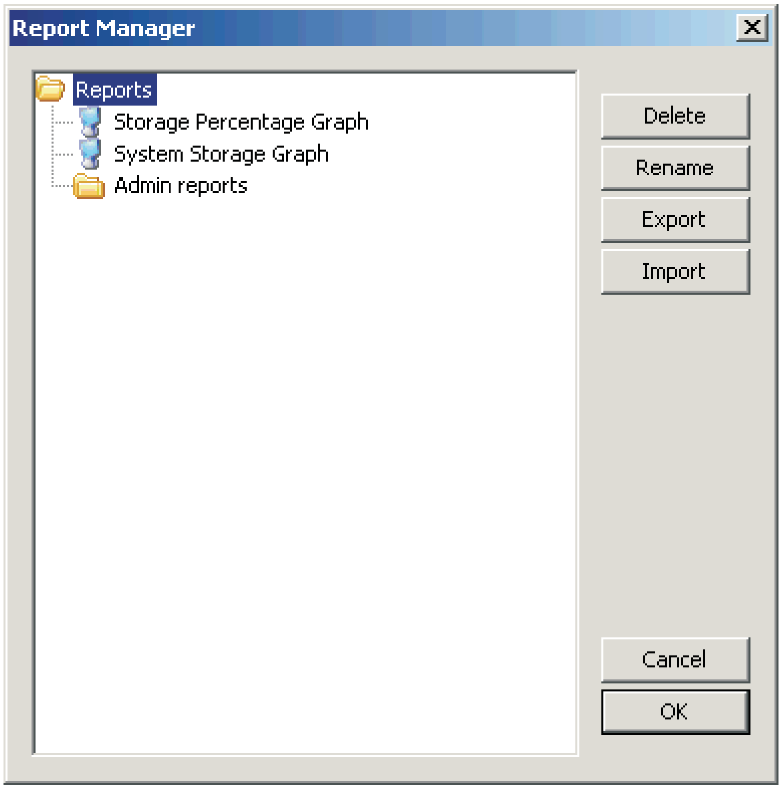


Figure 28. Report manager

Viewing reports

Use the Report Manager to view reports.

Procedure

To view an existing report:

1. In the **Report Manager**, select the report that you wish to view.
2. Right-click and select **View Report**.
3. The report is displayed in a new window.

Creating reports

Creating a report requires you to first create a graph upon which the report will be based.

Procedure

To create a report:

1. Create a graph.
2. For more details on creating graphs, see "Graphing MIB data" on page 119.
3. In the Graph Control dialog, click **Report**.
4. The Report Manager is displayed.
5. Select a location for the report and click **OK**.
6. The new report has the name `New Report` or `New Report(n)`; where `n` distinguishes between multiple reports of the same name.

Renaming reports and report folders

Use the Report Manager to rename reports and folders.

Procedure

To rename a report or a report folder:

1. In the **Report Manager**, select the item that you wish to rename.
2. Click **Rename** or right-click and select **Rename**.
3. Enter a new name for the item.

Organizing reports and report folders

You can create folders within the **Report Manager** or rearrange files in the `Report` folder.

Procedure

To create a new folder:

1. In the **Report Manager**, right-click on the folder in which you wish to create a new folder and select **Insert Folder**.
2. A new folder is created with the name `New Folder` or `New Folder(n)`; where `n` distinguishes between multiple folders of the same name. You can rename the folder as required.

What to do next

To move a report or report folder:

1. In the **Report Manager**, drag and drop the item to the desired location.
2. Moving a folder also moves any reports and folders that it contains.

Deleting reports or report folders

Use the Report Manager to delete reports and folders.

Procedure

To delete a report or a report folder:

1. In the Report Manager, select the item that you wish to delete.
2. Click **Delete** or right-click and select **Delete**.
3. The item is removed.

   Note: Deleting a report folder also deletes any reports or report folders that it contains.

Standard reports

MIB Explorer provides several standard reports.

Storage percentage

The Storage Percentage report (Figure 29 on page 127) includes statistics for physical memory and storage devices. The currently used system storage statistics are graphed as a percentage of the system storage capacity.

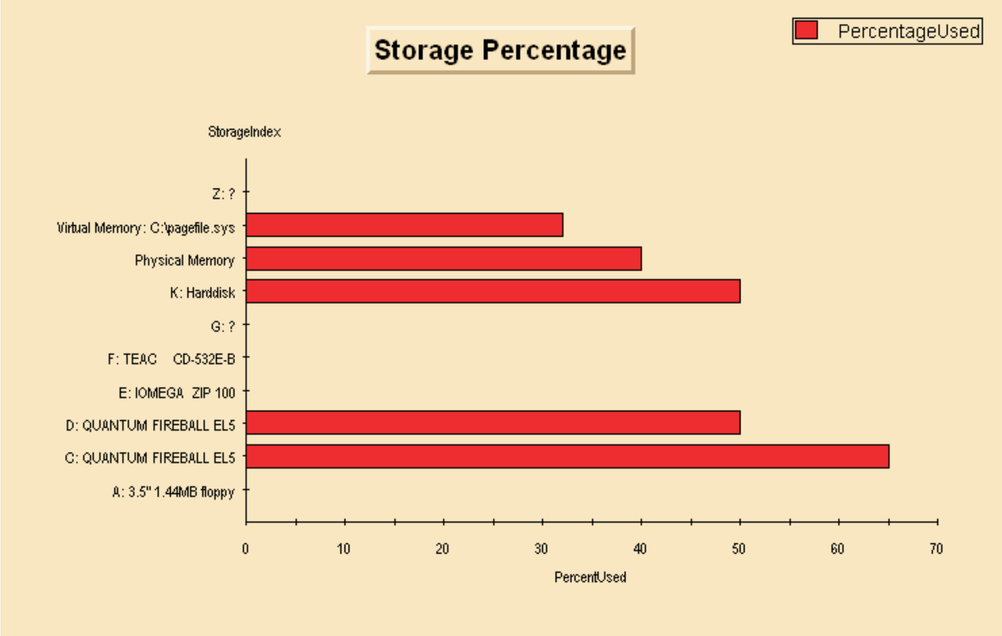


Figure 29. Storage percentage report

System storage

The System Storage report (Figure 30) presents system storage capacity (total) and currently used system storage statistics. It includes statistics for physical memory and storage devices.

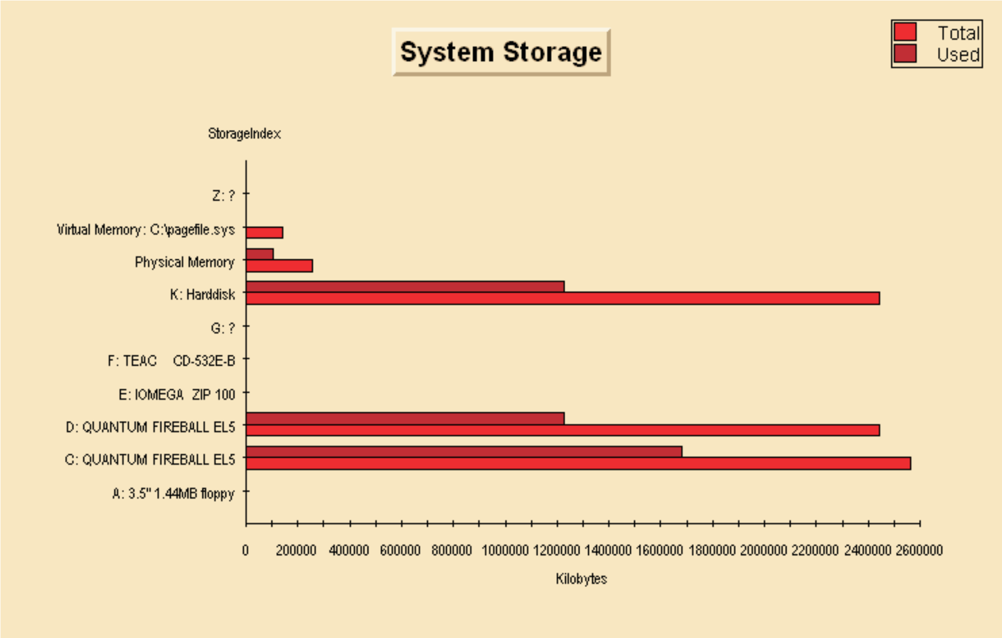


Figure 30. System storage report

Processor load

The Processor Load report (Figure 31) shows processor load as a percentage of processor capacity.

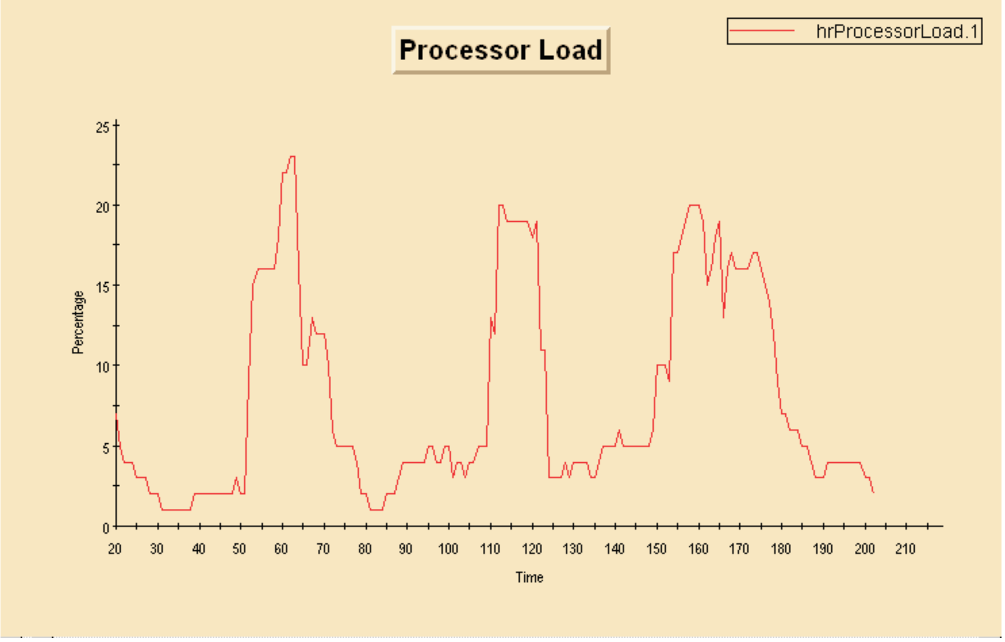


Figure 31. Processor load report

Processor allocation

The Processor Allocation report (Figure 32 on page 129) displays processor states (idle, user, sys, and wait) cumulatively as a percentage of the total processor activity.

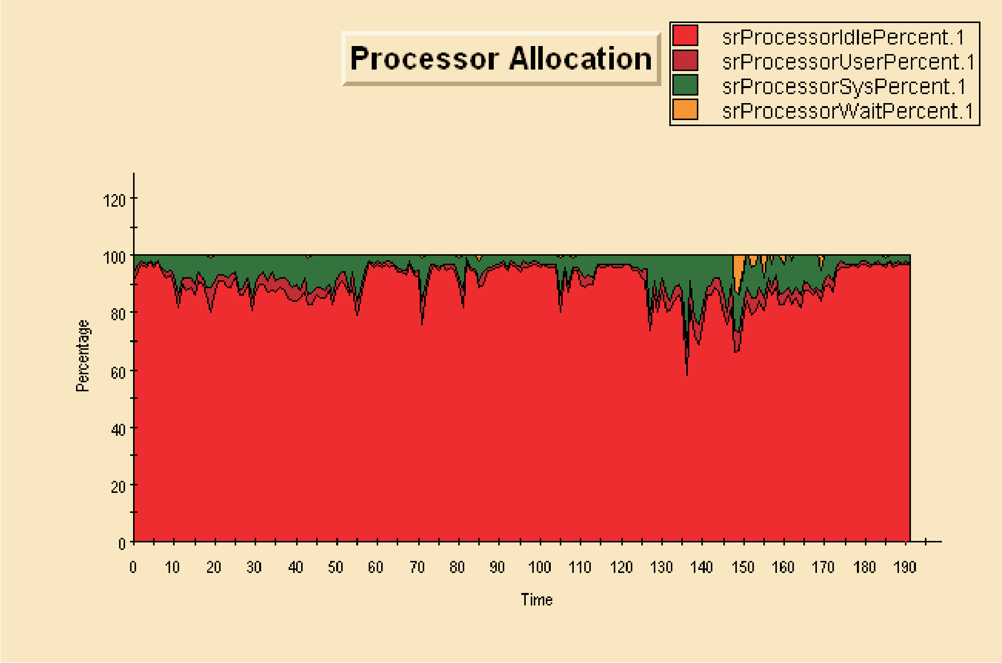


Figure 32. Processor allocation report

Process CPU usage

The Process CPU Usage report (Figure 33) shows the total process times separately for each process.

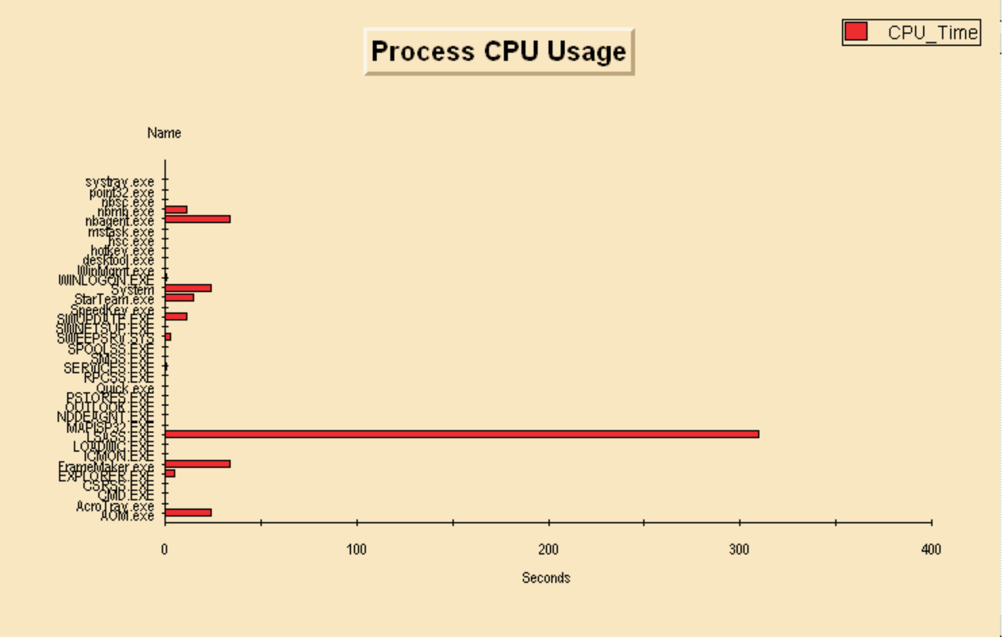


Figure 33. Process CPU usage report

Process memory usage

The Process Memory Usage report (Figure 34) displays the total process memory usage of each process.

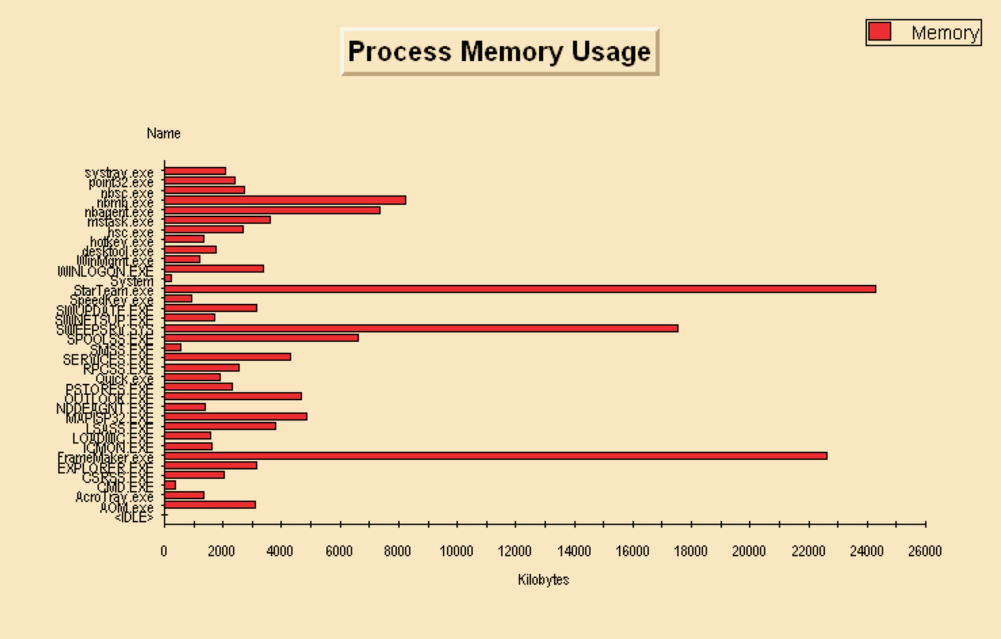


Figure 34. Process memory usage report

In/Out octets

The In/Out Octets report (Figure 35) graphs data transfers between the host machine and the network as total octets transmitted (out) and received (in).

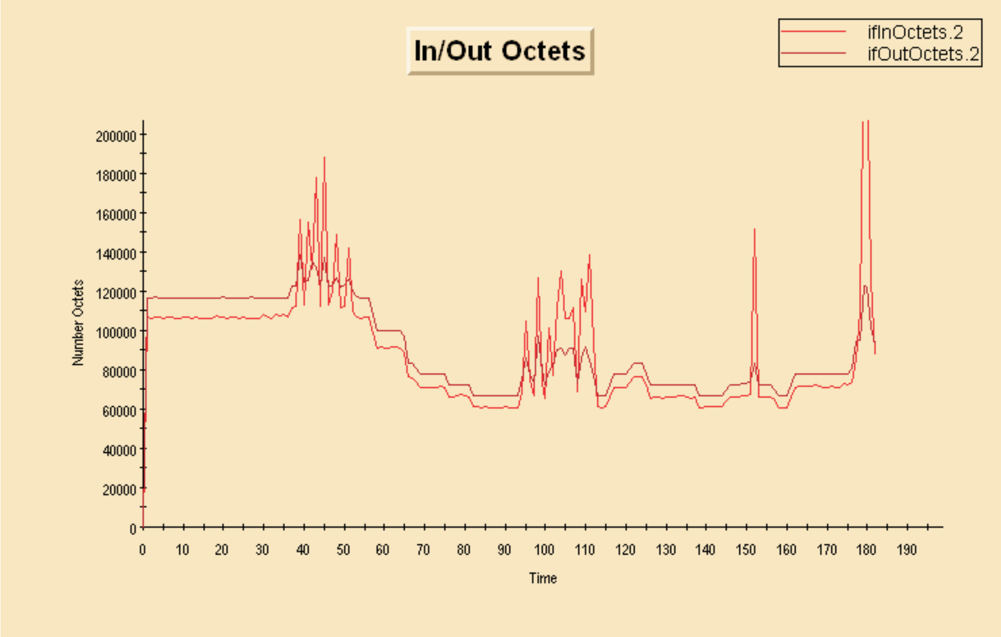


Figure 35. In/Out octets report

Viewing the MIB Explorer log file

The MIB Explorer log file (`log\mibexplorer.log`) stores any errors that occur when loading MIB files.

About this task

To view the log file to check for errors, select **View** > **View log file**. The log file is displayed in a new window.

Copying OIDs

You can copy the OID of any node to the clipboard for later use.

Procedure

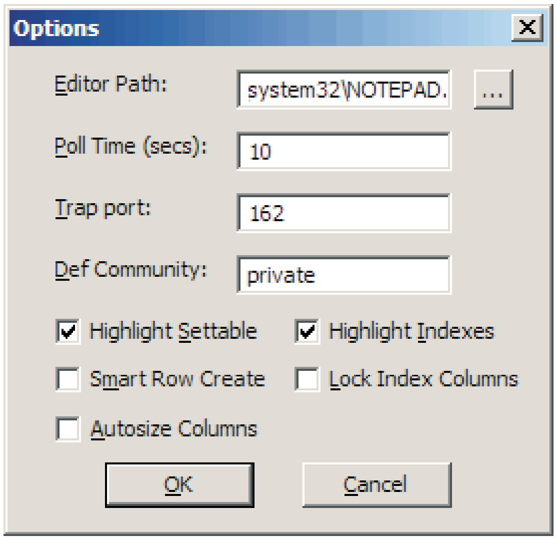
To copy an OID:

1. Select a node in the left pane.
2. Right-click and select **Copy OID**.
3. The OID for the selected node is placed on the clipboard.

Viewing options

MIB Explorer provides a number of settings that control the way MIB data is displayed.

These settings are defined in the Options dialog (Figure 36). To view or change these settings, select **View** > **Options...**.



Editor Path: system32\NOTEPAD.

Poll Time (secs): 10

Trap port: 162

Def Community: private

☑ Highlight Settable ☑ Highlight Indexes

□ Smart Row Create □ Lock Index Columns

□ Autosize Columns

Figure 36. Options dialog

Table 58 describes each of the settings in the Options dialog.

Table 58. Options settings

Setting	Description
Autosize Columns	Enables automatic adjustment of column widths to fit the width of display data and column headings.

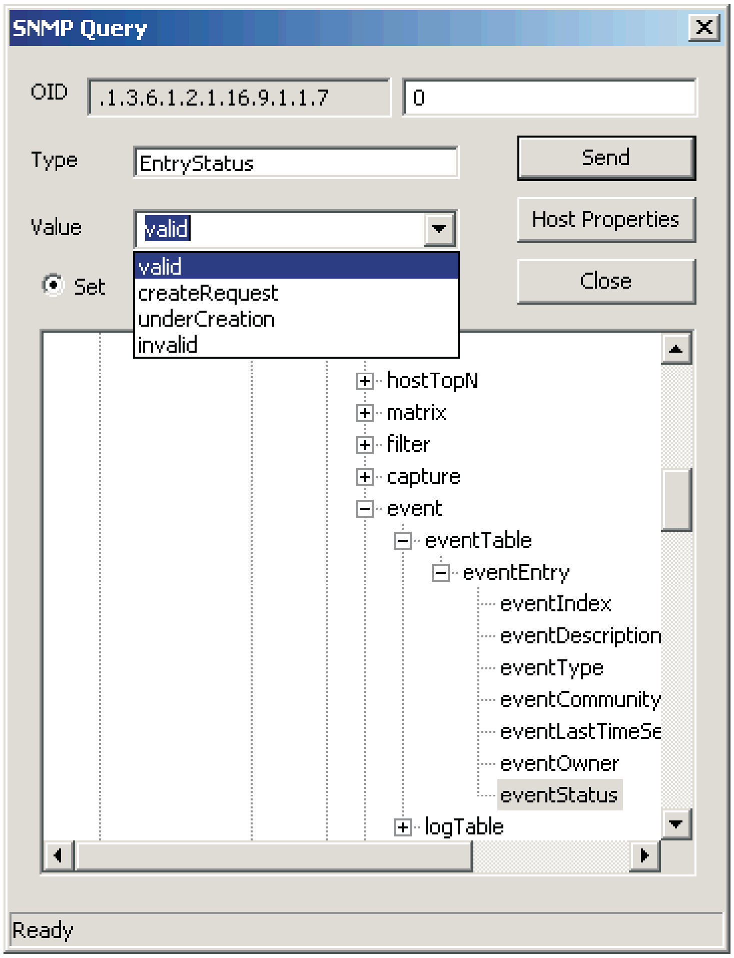
Table 58. Options settings (continued)

Setting	Description
Def Community	Sets the community string assigned to a host by default when the host is created.
Editor Path	Selects which editor MIB Explorer uses to view a text file.
Highlight Indexes	Enables highlighting of all index table columns in blue.
Highlight Settable	Enables highlighting of all editable MIB values (that is, objects that can be set using SNMP set commands) in yellow.
Lock Index Columns	Anchors index columns in the MIB view left pane, ensuring that they are always visible in the left pane, regardless of the horizontal scroll bar position. This setting is useful in ensuring that indexes in tables whose widths exceed that of the left pane are always visible.
Poll Time	Sets the interval (in seconds) at which MIB Explorer automatically collects and displays new MIB data.
Smart Row Create	Enables automatic incrementing of the row indexes when creating new rows in a table. Previously assigned indexes that have been freed by row deletion are used whenever available.
Trap Port	Sets the port on which the MIB Explorer listens for notifications.

SNMP query

The SNMP Query dialog enables you to query or set MIB objects individually by sending discrete SNMP requests.

To open the SNMP Query dialog select **SNMP** > **SNMP Query** or right-click and select **SNMP Query...**. Figure 37 on page 133 shows this dialog.


SNMP Query

OID .1.3.6.1.2.1.16.9.1.1.7 0

Type EntryStatus

Value valid

● Set

Ready

Figure 37. SNMP query dialog

Table 59 describes each field in the SNMP Query dialog.

Table 59. SNMP query dialog

Field	Description
OID	Displays the object identifier (OID) of the node selected in the lower pane of the SNMP Query dialog. The right field contains an index value, if applicable. The values of these two fields are concatenated to form a full OID.
Type	Displays the object syntax of the selected node.
Value	According to the type of operation performed, this field displays or sets the value of the selected MIB object. When setting values for objects of enumerated type, you can select a value from the drop-down list; for other types, the drop-down list presents a selection of values previously sent.
Set	Selects the SNMP set request, which sets an object to the value specified in the Value field. To perform a get request on a scalar object, the value of the index field must be 0.
Get	Selects the SNMP get request, which returns the value of an object. To perform a get request on a scalar object, the value of the index field must be 0. The value returned is shown in the Value field.

Table 59. SNMP query dialog (continued)

Field	Description
GetNext	Selects the SNMP get next request, which returns the value of the next object in a table. Successive get next requests step down a column, one row at a time, each time returning the value of the next object in that column. When the last entry in a column is reached, the following get next request returns the value of the object in the first row of the next column. The value returned is shown in the Value field.
Lower pane	Displays the MIB tree. Operations performed in the SNMP Query dialog relate to the node currently selected in this pane.

To send an SNMP query:

1. In the SNMP Query dialog, select the node in the MIB tree corresponding to the object you wish the query to operate on.
2. If the object you wish to query is part of a table row, set the index field to reference that row.
3. Select the type of SNMP request that you wish to send, either `Set`, `Get` or `GetNext`.
4. For more information about the different types of commands, see "Agent configuration commands" on page 47.
5. Click **Send**.
6. The fields of the SNMP Query dialog are updated to show the results of the operation and the status bar at the bottom of the dialog indicates the outcome of the query.

SNMP Walk

SNMP Walk enables you to view the OIDs and values of all the MIB objects below a selected node in the MIB tree.

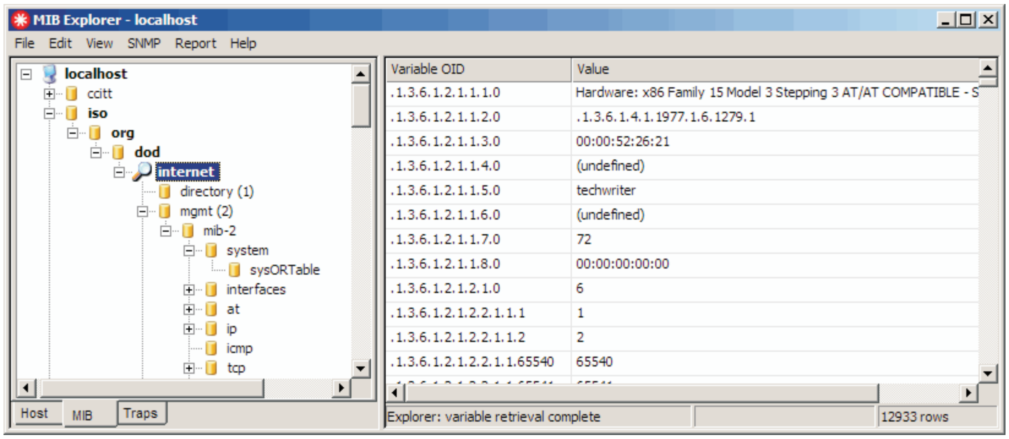
Procedure

To view the SNMP Walk for the MIB subtree below a node:

1. In the MIB view left pane, select the node in the MIB tree.
2. Select **SNMP** > **SNMP Walk** or right-click and select **SNMP Walk**.
3. Starting from the selected node, the MIB Explorer determines the OID of each object and displays them in a list in the right pane, together with those objects' values.

Results

Figure 38 on page 135 shows an example of the SNMP Walk. In this mode, the left column shows the numerical OID value by default. To display the OID value name, right-click on the column header and select **Object ID Named**.



Variable OID	Value
.1.3.6.1.2.1.1.1.0	Hardware: x86 Family 15 Model 3 Stepping 3 AT/AT COMPATIBLE - S
.1.3.6.1.2.1.1.2.0	.1.3.6.1.4.1.1977.1.6.1279.1
.1.3.6.1.2.1.1.3.0	00:00:52:26:21
.1.3.6.1.2.1.1.4.0	(undefined)
.1.3.6.1.2.1.1.5.0	techwriter
.1.3.6.1.2.1.1.6.0	(undefined)
.1.3.6.1.2.1.1.7.0	72
.1.3.6.1.2.1.1.8.0	00:00:00:00:00
.1.3.6.1.2.1.2.1.0	6
.1.3.6.1.2.1.2.2.1.1.1	1
.1.3.6.1.2.1.2.2.1.1.2	2
.1.3.6.1.2.1.2.2.1.1.65540	65540

Explorer: variable retrieval complete

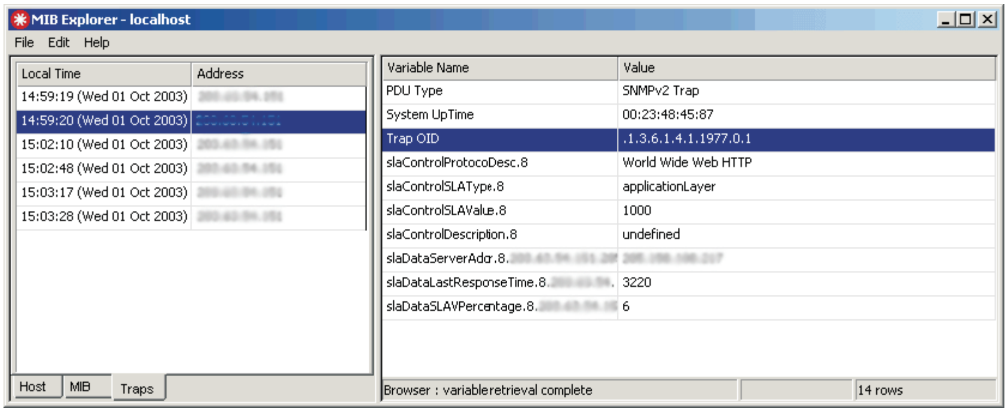
12933 rows

Figure 38. SNMP Walk

Using the Traps view

The Traps view provides a basic facility for monitoring SNMP notifications.

(Figure 39) MIB Explorer listens on port 162 for SNMP notifications and displays any notifications received in this view. Use the Traps view to monitor notifications generated by SNMP agents on your network.



Local Time	Address
14:59:19 (Wed 01 Oct 2003)	[redacted]
14:59:20 (Wed 01 Oct 2003)	[redacted]
15:02:10 (Wed 01 Oct 2003)	[redacted]
15:02:48 (Wed 01 Oct 2003)	[redacted]
15:03:17 (Wed 01 Oct 2003)	[redacted]
15:03:28 (Wed 01 Oct 2003)	[redacted]

Variable Name	Value
PDU Type	SNMPv2 Trap
System UpTime	00:23:48:45:87
Trap OID	.1.3.6.1.4.1.1977.0.1
slaControlProtocoDesc.8	World Wide Web HTTP
slaControlSLAType.8	applicationLayer
slaControlSLAValue.8	1000
slaControlDescription.8	undefined
slaDataServerAddr.8.[redacted]	[redacted]
slaDataLastResponseTime.8.[redacted]	3220
slaDataSLAVPercentage.8.[redacted]	6

Browser : variableretrieval complete

14 rows

Figure 39. Traps view

The left pane displays a list of the notifications received. It contains the following information:

- Date and time when the notification was received.
- IP address from which the notification was received.

The right pane lists further details about the notification currently selected in the left pane:

- Source of the notification
- Value for `sysUpTime` (specific to the agent) when the notification was received
- Enterprise object identifier of notification originator's vendor
- Variable bindings associated with the event that caused the notification to be sent

Right-click in the Traps view to display the context menu (Figure 40). This menu provides access to the functions available in the Traps view.

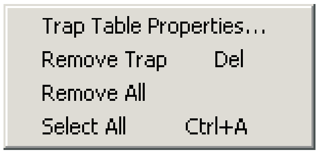


Figure 40. Traps view context menu

Creating notification destinations

The Traps view enables you to configure notification destinations on any SNMP agent on your network by adding notification destinations to the agent's trap destination table.

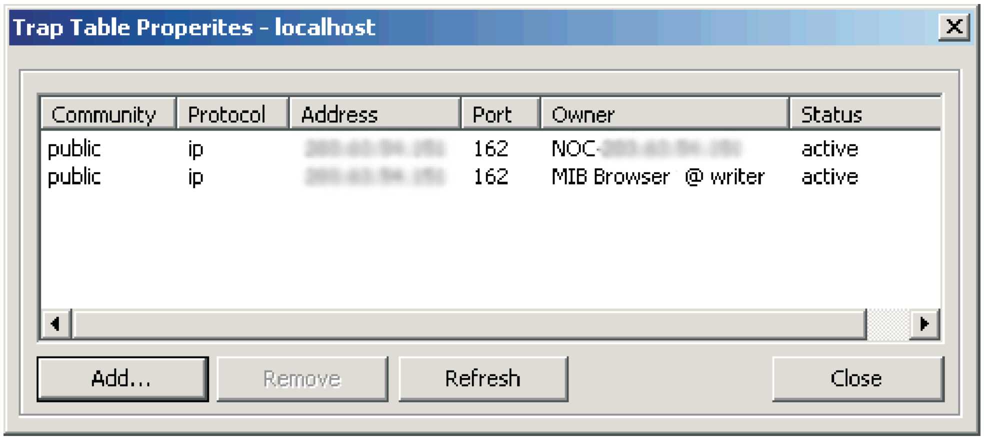
About this task

To configure a notification destination on an agent, first ensure that you have configured a host for the agent. For more details on configuring a host, see “Host properties” on page 112.

Procedure

To add a notification destination to an agent's trap destination table:

1. In the Host view, select the host for the agent whose trap destination table you wish to add a notification destination to.
2. In the Traps view, select **Edit** > **Trap Table Properties...** or right-click and select **Trap Table Properties...**.
3. The Trap Table Properties dialog is displayed (Figure 41).



Community	Protocol	Address	Port	Owner	Status
public	ip	[redacted]	162	NOC-[redacted]	active
public	ip	[redacted]	162	MIB Browser @ writer	active

Figure 41. Trap table properties dialog

4. Click **Add...**.
5. The Add Trap Destination dialog is displayed.
6. Enter the following information:
   a. **Host Name** - The IP address or DNS name of the notification destination.
   b. **Port** - The port on which the notification destination listens. This is usually port 162.

c. **Community** - The community string for the notification that you wish to receive.

7. Click **Add**.
8. The destination is added to the list of notification destinations displayed in the Add Trap Destination dialog. A row is created in the agent's RMON2 Probe Configuration Trap Destination table.

Receiving traps with MIB Explorer

To enable MIB Explorer to receive SNMP notifications, add the name or address of the machine on which it is running to the trap destination table of the agents from which the traps are sent.

Follow the procedure described in "Creating notification destinations" on page 136.

Ensure that you set the **Host Name** field to the IP address or host name of the machine on which MIB Explorer is running and the Port field to 162. In the Trap Table Properties dialog, the owner string of this entry is `MIB Browser @ hostname`, where `hostname` is the name of the machine on which MIB Explorer is running.

Removing traps

The left pane of the Traps view list all notifications received by MIB Explorer. You may wish to periodically remove notifications from this view when they are no longer of interest to you.

Procedure

- To remove a notification, select the trap then right-click and select **Remove**.
- To remove multiple notifications, press Shift and select the notifications you wish to remove, then right-click and select **Remove**.
- To remove all notifications, right-click and select **Remove All**.

MIB Explorer configuration file

Whenever you close MIB Explorer, it saves its current configuration in the file `mibexplorer.cfg`, which is located in the Netcool/SSM configuration directory.

This file stores settings such as:

- Field display formats
- Column widths of all MIB tables
- All host properties, as well as their position in the host tree
- Expansion of each host tree
- Final state of the MIB Explorer window

If the configuration file is deleted or becomes corrupted, MIB Explorer reverts to default values for these settings.

To save the MIB Explorer configuration, in any view select **File** > **Save Settings**. The current MIB Explorer configuration is written to the file `mibexplorer.cfg`.

Sample configuration file

```
========================= Sample mibexplorer.cfg file ===================
@main
showset trueshowind truepolltime 10
divrat 0.34
```

```
edpath "C:\WINNT\system32\NOTEPAD.EXE"
logpath "..\Config"
defcom "public"
lastoid .1.3.6.1.2.1.6.13
lasthost 1
startrect 184 160 1144 827
@mainend

@columnwidth
.1.3.6.1.2.1.16.11.2 80 80 80 81 80 80 80 80 80 80
@columnwidthend

@typemap
.1.3.6.1.2.1.16.11.2.1.1 103
.1.3.6.1.2.1.16.11.2.1.2 103
.1.3.6.1.2.1.16.11.2.1.4 102
.1.3.6.1.2.1.2.2.1.6 102
.1.3.6.1.2.1.3.1.1.2 102
.1.3.6.1.2.1.4.22.1.2 102
@typemapend

@hostlist
topindex 1

@host
index 1
label localhost
host localhost
community public
timeout 2000000
retries 2
port 161
version 1
getbulk false
username "fred"
auth@ "Lrx7EIWccj6qeSU"
hash 1
priv@ ""
crypt 0
@expandlist
.1
.1.3
.1.3.6
.1.3.6.1
.1.3.6.1.2
.1.3.6.1.2.1
.1.3.6.1.2.1.1
.1.3.6.1.2.1.16
.1.3.6.1.2.1.16.6
.1.3.6.1.2.1.2
.1.3.6.1.2.1.6
@expandlistend
@hostend
@hostlistend

@walkmap
@walkmapend

@grouplist
@group
1
@path
All Hosts
@pathend
```

```
expanded true
@groupend

@grouplistend
```

Main and hostlist configuration

Table 60 describes the settings in the `main` and `hostlist` sections of `mibexplorer.cfg`. You may view and edit these as required.

Table 60. mibexplorer.cfg parameter descriptions

Section	Setting	Description
`@main`		
	`showset`	Sets the yellow highlighting for editable fields in a MIB table.
	`showind`	Sets the blue highlighting of the index columns for tables.
	`polltime`	Interval in seconds between poll updates.
	`divrat`	Defines the relative sizes of the left and right panes as a ratio between 0 and 1.
	`edpath`	The full path name to the editor used by the MIB Explorer. If no path name is specified, the MIB Explorer uses auto discovery to determine the path name.
	`logpath`	The relative path name to the parsed log file.
	`lastoid`	Shows the last node selected in the left pane.
	`lasthost`	Shows the last host selected on the Host view.
	`startrect`	Specifies the dimensions of the MIB Explorer window.
	`trapport`	Specifies the port on which the MIB Explorer listens for notifications.
	`defcom`	Specifies the default community when a new host is created.
`@mainend`		
`@hostlist`		
	`topindex`	The highest index number of any host in this list.

Table 60. mibexplorer.cfg parameter descriptions (continued)

Section	Setting	Description
`@hostlist/host`		
	`index`	Each host must have a unique index number.
	`label`	A label given to identify the host.
	`host`	IP address or DNS name.
	`community public`	The SNMP community string used in queries to the agent.
	`timeout`	If a response is not received in `timeout` seconds, the browser will abort the request (or send one or more retries).
	`retries`	Number of retransmissions the browser performs before timing out. Default: `2`
	`port`	UDP port to which queries are sent.
	`version`	Sets the SNMP version used in queries.
	`getbulk`	SNMPv2 `GetBulkRequest`.
	`username`	The SNMPv3
	`auth@`	Encrypted SNMPv3 authentication password.
	`hash`	SNMPv3 authentication algorithm.
	`priv@`	Encrypted SNMPv3 privacy password.
	`crypt`	SNMPv3 privacy algorithm.
	`@expandlist`	Stores the current display state of the MIB tree.
`@hostlist/hostend`		
`@hostlistend`		

SNMP queries

MIB Explorer uses the default community string `public` when querying the agent on the local machine. Whenever you select a node in the MIB view containing accessible child objects, the MIB Explorer sends SNMP queries to the agent to obtain the data from those objects.

The type of SNMP query that MIB Explorer sends varies according to the type of MIB object queried:

- For tables, the queries comprise a series of `GetNext` requests, which continue until the end of the table is reached.
- For scalars, the queries comprise a series of `Get` requests which retrieve all OIDs associated with the selected node.

To alter the properties that MIB Explorer uses when making these queries, edit the host properties, as described in “Host properties” on page 112.

Other settings

Table 61 describes the settings contained in the other sections of `mibexplorer.cfg`.

Attention: The information presented in this table is for reference purposes only. Never change the settings contained in these sections of the `mibexplorer.cfg` file.

Table 61. mibexplorer.cfg section descriptions

Section	Description
`columnwidth`	Defines the table width.
`typemap`	Maps nodes in the MIB tree to field display formats.
`expandlist`	Defines nodes to expand in the MIB tree.
`walkmap`	Type mappings of SNMP Walk commands.
`grouplist`	Defines the structure of the host tree under the Host tab.

Chapter 9. Advanced configuration tasks

This information provides instructions for more advanced configuration tasks.

Adding and removing hosts

Netcool/SSM enables you to explicitly specify a list of hosts permitted to make queries to the agent. In the default state, in which no host list is configured, the agent will respond to queries from any host.

About this task

To retrieve a list of allowed hosts for an agent, use the command:

```
host list
```

To add an allowed host to an agent, use the command:

```
host add address [mask]
```

By default, the agent will respond to all hosts, however once you explicitly add an allowed host to an agent, the agent will only respond to those hosts that have been explicitly allowed using `host add` commands.

To remove an allowed host from an agent, use the command:

```
host remove address [mask]
```

Removing all allowed hosts returns the agent to the default state in which all hosts are allowed.

Using the optional mask field, you can add or remove entire subnets with a single command. For example, the command `host add 192.168.0.0 255.255.0.0` allows access by any host from the network `192.168.x.x`. If you omit the mask field, the default mask `255.255.255.255` is applied.

Setting the packet receive mode

The packet receive mode determines the type of packets received by the agent. Changing the packet receive mode affects subagents (such as SLA) that perform packet capture.

Procedure

To set the packet receive mode:

Use the following command:

```
set recvmode {promiscuous¦local¦directed}
```

The agent supports three modes for observing packets on the network:

- `promiscuous`

  All packets on the network.
- `local`

All packets with the localhost as the source or destination address, including broadcast/multicast packets.

- `directed`

  All packets with the localhost as the source or destination address, excluding broadcast/multicast packets.

Tip: In the command console, omitting the parameter from the `set recvmode` command displays the agent's current receive mode.

Using interface bindings

You can bind the agent to specific interfaces on a host machine. This can be useful in situations where security is of high concern. For example, many enterprise networks use a gateway to separate the internal network from the Internet. The gateway usually has two unique interfaces, and correspondingly two IP addresses. Binding the agent on the gateway only to the internal network's IP address ensures that SNMP statistics are only collected over the internal network and not over the Internet.

Creating an interface binding

To bind an agent to a particular interface, use the `interface bind` command.

About this task

The syntax of the command is:

```
interface bind address port
```

By default, the agent is bound to all interfaces on its host, listening on the port specified by the `UdpPort` variable in the file `init.cfg`. However, once you explicitly add an interface binding for an agent using the `interface bind` command, the agent will only listen on those interfaces that have been explicitly bound using `interface bind` commands.

The agent can also to listen on multiple ports on the same interface or on multiple ports on multiple interfaces. For example, an agent running on a host with only one interface could be bound to both ports 161 and 1161 if necessary.

Removing an interface binding

To remove an interface binding use the `interface unbind` command.

About this task

The syntax of the command is:

```
interface unbind address port
```

Unbinding all of an agent's interfaces returns the agent to the default state in which it is bound to all interfaces.

Attention: Ensure that no other processes are listening on any port bound to the agent. When another process listens on a port bound to the agent it may appear to use the port without error, however it is not possible to predict whether the agent or the other process will actually receive packets sent to that port.

Configuring interface blocking

Netcool/SSM maintains a list of known interfaces used for packet capture. If necessary you can modify the list of blocked interfaces.

About this task

Netcool/SSM stores the list of known interfaces in the Windows registry key `HKEY_LOCAL_MACHINE\SOFTWARE\Micromuse\SSM\4.0\pcap\interfaces`.

The registry key contains an entry for each known interface along with a rating for the interface. The format of these entries is:

```
ifName rating
```

When the agent opens an interface it stores a rating for each interface. The rating indicates the status of the interface. Table 62 describes the format of interface ratings.

Table 62. Interface registry entry format

Value	Description
ifName	The name of the interface as defined in the MIB object `ifDescr` in the RMON table `ifTable`.
rating	`0` - Not safe to open or enumerate the interface. `1` - Safe to enumerate the interface but it should not be opened for packet capture. `2` - Safe to access the interface.

Note: This facility is only provided on Windows platforms.

Blocking an interface manually

The agent provides a facility for manually blocking interfaces. The file `pcapblacklist.txt`, located in the agent's configuration directory, provides a list of blocked interfaces. It is formatted using one interface name per line. On startup, the agent reads this file and then sets the registry entry for each interface listed in the file to the value `0`, indicating that it should not be used.

Procedure

To block an interface:

1. Determine the name of the interface that you wish to block.
2. The interface name is defined in the MIB object `ifDescr` in the RMON table `ifTable`. Alternatively, you can obtain this name from the agent log file `agent.log` (interface names are contained in log file entries of type `Found interface x type x:`).
3. Open the file `pcapblacklist.txt` and add the name of the interface to the list.

   Tip: If the file `pcapblacklist.txt` does not exist in the agent configuration directory (`config`), create a new file with that name.

Recovering an interface manually

You can manually recover an interface that has been blocked.

Procedure

To recover an interface that has been blocked manually or by the agent:

1. Open the Windows registry key and change the rating of the interface to the value 2: `HKEY_LOCAL_MACHINE\SOFTWARE\Micromuse\SSM\4.0\pcap\interfaces`
2. Open the file `pcapblacklist.txt` and remove any entry for the interface.

Integrating Netcool/SSM with Netcool/OMNIbus

Netcool/SSM integrates with Netcool/OMNIbus through the Multi-Thread Trap Daemon probe (known as MT Trapd). MT Trapd receives notifications from Netcool/SSM, processes them and forwards them to the Netcool/OMNIbus Object Server.

Procedure

To integrate Netcool/SSM with Netcool/OMNIbus through MT Trapd:

1. Defining probe rules for MT Trapd that instruct it to process notifications sent by Netcool/SSM
2. Configuring Netcool/SSM to send notifications to MT Trapd

Defining rules for MT Trapd

To integrate Netcool/SSM with Netcool/OMNIbus, configure the Netcool/OMNIbus probe for SNMP as a trap destination in the SSM. The probe then receives traps and places them into Netcool/OMNIbus based on the contents of rules files. The Netcool Knowledge Library provides a predefined set of rules files for processing SNMP-based information.

About this task

IBM has produced a rules file that integrates with the Netcool Knowledge Library which handles notifications sent by Netcool/SSM. These rules files are available on request from IBM Support. They consist of look-up table definitions and a rules file. The files are named `netcool-ssm.include.snmptrap.lookup` and `netcool-ssm.include.snmptrap.rules`.

Procedure

To integrate Netcool/SSM and Netcool/OMNIbus:

1. Configure the Netcool/OMNIbus probe for SNMP to use the Netcool Knowledge Library. See Documentation for Netcool Knowledge Library(http://publib.boulder.ibm.com/infocenter/tivihelp/v8r1/topic/com.ibm.netcool_OMNIbus.doc/probes/nckl/nckl/wip/concept/nckl_intro.html) for further information.
2. Create a trap destination entry in Netcool/SSM that points to the server that is running the Netcool/OMNIbus probe for SNMP. See Chapter 6, “Events and notifications,” on page 75 for information about configuring Trap Destinations.
3. Copy the Netcool/SSM rules files to the `$NC_RULES_HOME/include-snmptrap/networkharmoni` directory.

4. Go to the Netcool Knowledge Library rules home directory. Edit `snmptrap.rules` to contain references to the `netcool-ssm` rules files:
   a. In the `snmptrap.rules` file, search for `lookup table Includes` to find the look-up table includes section.
   b. Add the line: `include "$NC_RULES_HOME/include-snmp/networkharmoni/netcool-ssm.include.snmptrap.lookup"`
   c. Search for `rules file Includes` to find therules files includes section.
   d. Add the line: `include "$NC_RULES_HOME/include-snmp/networkharmoni/netcool-ssm.include.snmptrap.rules"`
   e. Save the changes to `snmptrap.rules`.
5. Restart the Netcool/OMNIbus probe for SNMP.

Configuring notifications

Configure MT Trapd and Netcool/SSM according to the types of notification, SNMPv1, v2 or v3, used in your monitoring configuration.

SNMPv1 and SNMPv2 notifications

To configure Netcool/SSM to send SNMPv1 or SNMPv2 notifications to MT Trapd, use the `trapdest add` configuration command .

About this task

The appropriate syntax for using the command is:

```
trapdest add mttrapd_ip_address:mttrapd_port public
```

where *`mttrapd_ip_address`* represents the IP address of the machine on which the target MT Trapd probe is running and the and *`mttrapd_port`* is the port on which MT Trapd listens for SNMP traffic (this port is set by the MT Trapd `Port` configuration property defined in the `mttrapd.props` file).

Note: If you choose to add this command to the Netcool/SSM configuration file, `agent.cfg`, you must restart the agent for it to take effect.

SNMPv3 notifications

You can configure Netcool/SSM for sending SNMPv3 notifications to MT Trapd.

Procedure

1. Create an SNMPv3 user on both Netcool/SSM and MT Trapd

   You must create the same SNMPv3 user on both Netcool/SSM and MT Trapd, however the user creation process differs slightly according to the notification type that you wish Netcool/SSM to send, either Trap or Inform.
2. Define MT Trapd as a notification target on Netcool/SSM.

Configuring Traps:

When sending Traps to MT Trapd, Netcool/SSM acts as the authoritative engine and its engine ID is used to generate the security keys. MT Trapd requires the Netcool/SSM engine ID to create the keys necessary for decrypting and authenticating the Trap.

Procedure

To configure SNMPv3 Traps:

1. Obtain the Netcool/SSM engine ID from its `snmpEngineID` object (`snmp get .1.3.6.1.6.3.10.2.1.1.0`) or through the SNMPv3 discovery process.
2. Shut down the MT Trapd probe.
3. Create an SNMPv3 user with the desired security parameters on the MT Trapd probe by adding a `createUser` command to the MT Trapd configuration file, `mttrapd.conf`:

   ```
   createUser -e engineID v3user authtype authpass [privtype privpass]
   ```

   For more information about configuring MT Trapd, see the MT Trapd documentation.
4. Restart the MT Trapd probe.
5. Create the same user on Netcool/SSM with the command:

   ```
   user add v3user authtype authpass [privtype privpass]
   ```

   For more details on this command, see “Creating a user” on page 93
6. Configure the notification tables on Netcool/SSM with the commands:

   ```
   trapdest addnotify community targetTag trap
   trapdest addparam mttrapdParams 3 usm v3user securityLevel
   trapdest addtarget targetName mttrapd-addr:port mttrapdParams
       transport targetTag
   ```

   For more details on these commands, see “Configuring SNMPv3 notifications” on page 82.

Configuring SNMPv3 traps

Configure the MT Trapd probe with IP address `190.60.50.40` listening on port `162` to receive SNMPv3 Traps over UDP from a Netcool/SSM agent with engine ID `0x800007B905AABBCCDDEEFF`. Send and receive Traps using MD5 authentication as the user `MT-user`, with authentication password `B*o1zacd`.Table 63 summarizes the required configuration.

Table 63. Example configuration for SNMPv3 Traps

Property	Value
Notification type	`trap`
MT Trapd probe IP address	`190.60.50.40`
MT Trapd SNMP receive port	`162`
Netcool/SSM engine ID	`0x800007B905AABBCCDDEEFF`
Transport protocol	UDP
SNMPv3 username	`MT-user`
Security level	`authNoPriv`
Authentication type	MD5

Table 63. Example configuration for SNMPv3 Traps (continued)

Property	Value
Authentication password	B*o1zacd

MT Trapd command:

```
createUser -e 0x800007B905AABBCCDDEEFF MT-user MD5 B*o1zacd
```

Netcool/SSM configuration commands:

```
user add MT-user MD5 B*o1zacd
trapdest addnotify public MT-trapTag trap
trapdest addparam MT-authParams 3 usm MT-user authNoPriv
trapdest addtarget MT-trapTarget 190.60.50.40:162 MT-authParams udp MT-trapTag
```

Configuring informs:

When Netcool/SSM sends Informs to MT Trapd, MT Trapd is the authoritative engine and its engine ID is used to generate the security key. Netcool/SSM requires the engine ID of MT Trapd to create the security key, so you must create a *remote* user on Netcool/SSM.

Procedure

To configure SNMPv3 Informs:

1. Shut down the MT Trapd probe.
2. Create an SNMPv3 user with the desired security parameters on the MT Trapd probe by adding a `createUser` command to the MT Trapd configuration file, `mttrapd.conf`:

   ```
   createUser v3user authtype authpass [privtype privpass]
   ```

   Tip: It is not necessary to specify the engine ID of Netcool/SSM in this command.

   For more information about configuring MT Trapd, see the MT Trapd documentation.
3. Restart the MT Trapd probe.
4. Create the same user on Netcool/SSM with the command:

   ```
   user remote add remote_addr:port v3user authtype authpass [privtype privpass]
   ```

   For more details on this command, see "Creating a remote user" on page 94

   Note: MT Trapd must be running when you create the user on Netcool/SSM because Netcool/SSM performs a discovery on MT Trapd to determine its engine ID.
5. Configure the notification tables on Netcool/SSM with the commands:

   ```
   trapdest addnotify commmunity targetTag inform
   trapdest addparam mttrapdParams 3 usm v3user securityLevel
   trapdest addtarget targetName mttrapd-addr:port mttrapdParams
       transport targetTag
   ```

   For more details on these commands, see "Configuring SNMPv3 notifications" on page 82.

Example - Configuring SNMPv3 informs

Configure the MT Trapd probe with IP address `190.60.50.40` listening on port `1162` to receive SNMPv3 Informs over UDP from Netcool/SSM. Send and receive

Informs using MD5 authentication and DES encryption as the user MT-privUser, with authentication password stp*0ghf and privacy password b45h3r3d. Table 64 summarizes the required configuration.

Table 64. Example Configuration for SNMPv3 Informs

Property	Value
Notification type	inform
MT Trapd probe IP address	190.60.50.40
MT Trapd SNMP receive port	1162
Transport protocol	UDP
SNMPv3 username	MT-privUser
Security level	authPriv
Authentication type	MD5
Authentication password	stp*0ghf
Privacy type	DES
Privacy password	b45h3r3d

MT Trapd command:

```
createUser MT-privUser MD5 stp*0ghf DES b45h3r3d
```

Netcool/SSM configuration commands:

```
user remote add 190.60.50.40:1162 MT-privUser MD5 stp*0ghf des b45h3r3d
trapdest addnotify public MT-informTag inform
trapdest addparam MT-privParams 3 usm MT-privUser authPriv
trapdest addtarget MT-informTarget 190.60.50.40:1162 MT-privParams udp MT-informTag
```

Using third-party SNMP extensions

Netcool/SSM enables you to load and use third-party SNMP extensions that usually run on other SNMP services. On Windows platforms, you can load extensions for the Microsoft SNMP Service. On UNIX platforms, you can load extensions for the Net-SNMP snmpd agent.

When you load an extension on Netcool/SSM, you can use standard SNMP commands to access the functionality and MIB data that the extension provides.

Netcool/SSM can also import an SNMP extension's trap destinations, ensuring that any notifications that the extension generates are sent to the correct destinations.

Microsoft SNMP Service extensions

To load SNMP extensions that usually run on the Microsoft SNMP Service, use the Netcool/SSM ntext subagent.

About this task

To load this subagent, use the command:

```
subagent load ntext
```

The subagent provides a set of configuration commands for managing Microsoft SNMP Service extensions.

CAUTION:
Never attempt to load an SNMP extension while it is running on the Microsoft SNMP Service. Running an extension on Netcool/SSM and Microsoft SNMP Service at the same time may cause the system to hang or crash.

Loading an extension

To load an extension, use the `ntext load` command

The syntax of the command is:

```
ntext load ext
```

This command loads the extension identified by *ext* and creates a module of the same name. The access privileges assigned to this module are imported from the Microsoft SNMP Service.

Note: Only the access privileges `READ ONLY`, `READ WRITE`, and `READ CREATE` are imported. The privileges `NONE` and `NOTIFY` are not imported.

Listing extensions

To obtain a list of the extensions available on the host machine, use the `ntext list` command.

About this task

This command lists the name of the extension, a description of the extension and its vendor, and shows whether the extension is currently loaded on Netcool/SSM.

Unloading an extension

To unload an extension currently loaded on Netcool/SSM, use the `ntext unload` command.

The syntax of the command is:

```
ntext unload ext
```

This command unloads the extension identified by *ext* and removes the module associated with the extension.

Module access privileges

By default, the `ntext` subagent imports module access privileges from the Microsoft SNMP Service when loading an extension. If required you can prevent the import of access privileges.

If you do not wish to use the module's access privileges, use the following command when loading an extension:

```
ntext load ext nocomm
```

This command loads the extension identified by *ext*, creates a module of the same name and assigns it the default module access privileges defined in Netcool/SSM.

For more details on module access privileges, see “Access control using communities” on page 97.

Trap destinations

When you load the ntext subagent, it automatically imports any trap destinations defined in the Microsoft SNMP Service and creates rows for them in the trapDestTable. Netcool/SSM forwards any traps sent by the extension to those addresses.

Note: The ntext subagent automatically imports trap destinations whenever it loads. If you remove any of the imported destinations from trapDestTable, they will be reappear whenever ntext is reloaded unless you also remove them from Microsoft SNMP Service.

Net-SNMP extensions

To load and use SNMP extensions that usually run on the Net-SNMP snmpd agent, use the Netcool/SSM netsnmp subagent.

About this task

To load this subagent use the command:

```
subagent load netsnmp
```

The subagent provides a set of configuration commands for managing Net-SNMP extensions.

Note: Check the *Netcool/SSM Release Notes* for information about the extensions that you can load using netsnmp.

Loading an extension

To load and activate a Net-SNMP extension, use the netsnmp load and netsnmp activate commands.

About this task

The syntax of the commands is:

```
netsnmp load ext
netsnmp activate ext
```

The first command loads the extension identified by *ext*. The second command activates the extension, enabling its functionality (when you execute this command the netsnmp subagent calls the extension's init_*ext* initialization routine).

When you execute the netsnmp load command to load the extension *ext*, the subagent attempts to load the binary lib*ext*.so. If the name of the extension does not correspond to its binary in this way, or if the binary it not located in the path, load it using the command:

```
netsnmp load ext binary
```

where *ext* is the name of the extension, and *binary* is the full path and filename of the extension's binary.

To activate all loaded extensions, use the the command:

```
netsnmp activate all
```

Using Net-SNMP configuration files

The `netsnmp` subagent can read and use an existing `snmpd` configuration file to load and configure extensions.

About this task

Table 65 lists the `snmpd` configuration directives that the subagent can interpret.

Table 65. Supported snmpd configuration directives

Configuration directive	Action taken by the netsnmp subagent
`dlmod`	Load the specified Net-SNMP extension. **Note:** The subagent does not activate any loaded extensions. You must explicitly activate any loaded extensions using the `netsnmp activate` command.
`rocommunity`	Assign the read-only (`ro`) privilege to the specified OID tree.
`rwcommunity`	Assign the read/create (`rc`) privilege to the specified OID tree.
`trap2sink`	Create an `agentTrapDestTable` row for SNMPv2 Traps.
`trapcommunity`	Set the `agentTrapDestCommunity` to the community specified when creating `agentTrapDestTable` rows.
`trapsink`	Create an `agentTrapDestTable` row for SNMPv1 Traps.

To read an `snmpd` configuration file, use the command:

```
netsnmp configure config_file
```

where *`config_file`* is the path and filename of the `snmpd` configuration file.

Tip: You must activate any extensions loaded from an `snmpd` configuration file using the `netsnmp activate` command.

Managing extensions

Use the `netsnmp` commands to manage SNMP extensions.

About this task

To view a list of the Net-SNMP extensions currently loaded on Netcool/SSM, use the command:

```
netsnmp list
```

To suspend the operation of a loaded extension, use the command:

```
netsnmp deactivate {ext | all}
```

where *`ext`* is the name of the extension. When you execute this command, the `netsnmp` subagent calls the extension's `deinit_ext` deinitialization routine. To completely unload the extension from Netcool/SSM, use the command:

```
netsnmp unload {ext | all}
```

Monitoring clustered applications

You can use Netcool/SSM to monitor applications running on Microsoft Cluster Server (MSCS) server clusters. Monitoring clustered applications requires a configuration that detects the cluster node on which the target application is running, and is able to reconfigure automatically to start or stop monitoring when the application fails over to or from another node.

Netcool/SSM provides two configuration files, `msxmon-master.cfg` and `svctrack.cfg`, that demonstrate how to use this approach to monitor a clustered Microsoft Exchange server. You can use these files as a basis for creating your own monitoring solutions for clustered applications.

Integrating with HP Systems Insight Manager (Windows)

You can install Netcool/SSM to integrate with an existing HP Systems Insight Manager installation on Windows hosts.

HP Systems Insight Manager requires the Microsoft SNMP service to be installed and running on the host system for it to operate properly. Integrating the Netcool/SSM with HP Systems Insight Manager requires that the Netcool/SSM agent be inserted into the existing Microsoft SNMP services in such a way that the Microsoft SNMP services always appear available to HP Systems Insight Manager.

Normally, the Netcool/SSM installer does not attempt to modify or integrate with any SNMP services already installed on the host system. However you can instruct it to integrate the Netcool/SSM agent with HP Systems Insight Manager on Windows platforms. If you later uninstall Netcool/SSM, the uninstaller reverses these changes and restores the system to its previous state.

Installation

The `OverrideSnmp` installation parameter instucts the Netcool/SSM installer to integrate Netcool/SSM with HP Systems Insight Manager.

To integrate Netcool/SSM with HP Systems Insight Manager, run the installer using:

```
netcool_ssm-4.0.1-xxxx-win32.exe /z"--OverrideSNMP"
```

Effects

The effects of using the `OverrideSnmp` installation parameter depend on whether HP Systems Insight Manager agents are present on the host system.

If HP Systems Insight Manager agents are present on the host system, the Netcool/SSM installer substitutes the Microsoft SNMP services with the Netcool/SSM agent. The Netcool/SSM agent then has the service name `SNMP` and can be started and stopped using this name. The Netcool/SSM agent uses port 161 and the Microsoft SNMP service is no longer usable, unless Netcool/SSM is uninstalled again. The `agent.cfg` file contains `ntext load` entries, which ensure that the NT extensions for HP Systems Insight Manager are automatically loaded. The HP Systems Insight Manager agents are restarted after installation of the Netcool/SSM agent.

If HP Systems Insight Manager agents are not present on the host system, installing Netcool/SSM stops the Microsoft SNMP service and places it into the `manual` or `demand` mode. The Netcool/SSM agent service is placed in `automatic` mode and started. All other aspects of the installation and operation of the Netcool/SSM agent take place as if no Microsoft SNMP service was present.

Post-install configuration

After installing Netcool/SSM, check that the agentConfigTable in the agent MIB contains a row for the file cim.cfg, which loads the HP Systems Insight Manager agents.

About this task

The cim.cfg file itself contains the following commands:

```
echo loading Insight Manager agents
ntext load CPQMIB1K
ntext load CPQNIMIB
ntext load HOSTMIB
ntext load NICMIB
ntext load SERVMIB
ntext load STORMIB
```

Appendix A. Configuration commands

Netcool/SSM provides configuration commands for controlling its operation. To execute configuration commands, issue them from the command console, the command-line, or include them in configuration files.

Configuration commands take the general format:

```
command [sub-command] [argument ...]
```

Tip: To run command console commands using the command-line, use the `-c` option. For example, in the command-line run `ssmcons -c "loglevel debug"`.

Table 66 lists the configuration commands for the Netcool/SSM agent in alphabetical order.

Table 66. Configuration commands

Command & sub-command	Arguments	Description
`coldboot`		Forces the agent to perform a cold boot sequence.
`community add`	{*module*|*module_id*| `all`} *community* [`ro`|`rw`|`rc`]	Sets or modifies the module access privileges assigned to a community for the module identified by name or ID, or for all modules.The module access privileges available are read-only (`ro`), read-write (`rw`), and read-create (`rc`). *This command does not reduce the existing level of privileges assigned to a community.*
`community list`	{*module*|*module_id*| `all`}	Lists all communities and access privileges associated with the module identified by name or ID, or all modules.
`community remove`	{*module*|*module_id*| `all`} *community*	Removes the access privileges assigned to assigned to a community for the module identified by name or ID, or from all modules.
`community set`	{*module*|*module_id*| `all`} *community* [`ro`|`rw`|`rc`]	Sets or modifies the module access privileges assigned to a community for the module identified by name or ID, or for all modules. The module access privileges available are read-only (`ro`), read-write (`rw`), and read-create (`rc`).
`config execute`	*file*	Executes the configuration file indicated by *file*.

Table 66. Configuration commands (continued)

Command & sub-command	Arguments	Description
`config save`	[*config_file* [*state_file*]]	Saves the current agent configuration to the files indicated by *config_file* and *state_file*. Omitting the filenames saves this information in the default configuration and state files.
`encrypt`	*password*	Encrypts the password supplied in *password* and returns the encrypted password string.
`host add`	*address* [*mask*]	Adds the host identified by IP address *address* and, optionally, the address mask *mask* to the list of allowed hosts.
`host list`		Lists all allowed hosts.
`host remove`	*address* [*mask*]	Removes the host identified by IP address *address* and, optionally, the address mask *mask* from the list of allowed hosts.
`interface bind`	*address port*	Binds the agent's interface to the IP address *address* and port number *port*.
`interface list`		Lists the agent's interface bindings.
`interface unbind`	*address port*	Removes the agent's interface binding from the IP address *address* and port number *port*.
`loglevel`	[1-5 | fatal | warning | information | verbose | debug]	Sets the Netcool/SSM log level at run time.
`module list`		Lists all MIB modules.
`oid`	*name*=*OID*	Associates the identifier *name* with the object ID *OID*.
`password`	*password*	Sets the password for restricting access to an agent from the command console.
`set event`	*event_index*	Pulses the RMON event whose index is indicated by *event_index*.
`set heartbeat`	*interval*	Sets the agent's heartbeat interval (in seconds).
`set heartbeatstr`	*community*	Sets the community to which the agent's heartbeat notification is sent.
`set inivar`	[*name*=*value*]	Sets the value of the inivar identified by *name* and writes this definition to the file `init.cfg`. Omitting the name-value pair lists all inivars currently defined.
`set latitude`	[[+|-]*DDMMSS*[.*SS*]]	Sets the latitude value of the agent's geographical location.

Table 66. Configuration commands (continued)

Command & sub-command	**Arguments**	**Description**
`set location`	*`description`*	Sets the description of the agent's physical location.
`set longitude`	`[[+|-]`*`DDMMSS`*`[.`*`SS`*`]]`	Sets the longitude value of the agent's geographical location.
`set recvmode`	`[promiscuous|local|directed]`	Sets the agent's packet capture mode. Issuing the command without a value displays the agent's current setting.
`set restore`	`[on|off]`	Sets the agent's persistent state behavior. Issuing the command without a value displays the agent's current setting.
`set snmptrace`	`[on|off|log]`	Control the agent's SNMP debug trace output. Issuing the command without a value displays the agent's current setting: `off` - Turns debug trace off. `on` - Turns debug trace on. Trace data is output to the command console. `log` - Turns debug trace on. Trace data is output to a log file. The default log file is `trace.log`, located in the Netcool/SSM `log` directory, however you can specify another file using the `TraceLogFile` inivar.
`set uniqueid`	*`id_string`*	Sets the agent's unique identifier to the value indicated by *`id_string`*, which consists of a series of 8-bit hexadecimal values, with each value in the string separated by a space character.
`set verbose`	`[on|off]`	Selects the level of detail returned to the command console by the agent in response to configuration commands. Issuing the command without a value displays the agent's current setting.
`snmp get`	*`object_id`*	Gets the value of the MIB object indicated by *`object_id`*. This command performs a local SNMP `GET` operation.
`snmp getnext`	*`object_id`*	Gets the value of a MIB object relative to the base OID *`object_id`*. This command performs a local SNMP `GETNEXT` operation.

Table 66. Configuration commands (continued)

Command & sub-command	Arguments	Description
`snmp match`	*object_id type value*	Gets the index of the row containing the object indicated by *object_id* whose value matches the regular expression *value*. For details about the allowed values for *type*, see "Supported ASN types" on page 50.
`snmp set`	*object_id type value*	Sets the value of one or more MIB objects. This command performs local SNMP `SET` operations. For details about the allowed values for *type*, see "Supported ASN types" on page 50.
`snmp walk`	*object_id*	Gets the value of all MIB objects in the sub-tree relative to a base OID. This command performs local SNMP `GETNEXT` operations.
`subagent info`	{*id*|*name*}	Displays further information on the subagent indicated by *name* or *id*.
`subagent list`		Lists all subagents currently loaded on the agent.
`subagent load`	*name*	Loads the subagent identified by *name*.
`subagent unload`	{*id*|*name*}	Unloads the subagent indicated by *name* or *id*.
`terminate`		Forces the agent to shut down.
`trapdest add`	{*ip-address*|*hostname*} [:*port*] *community*	Creates an RMON trap destination. Default port value is 162.
`trapdest addnotify`	*name tag type*	Adds an SNMPv3 notification entry. See "SNMPv3 notifications" on page 81 for more details.
`trapdest addparam`	*param_tag version sec_model sec_name sec_level*	Adds an SNMPv3 notification parameter table entry. See "SNMPv3 notifications" on page 81 for more details.
`trapdest addtarget`	*name* {*ip-address*|*hostname*}[:*port*] *param transport tag* [*tag* ...]	Adds an SNMPv3 notification destination entry. See "SNMPv3 notifications" on page 81 for more details.
`trapdest flush`		Removes all notification, target address and target parameters table rows that were not created using either a configuration file or the `trapdest` command.
`trapdest list`		Lists all trap destinations (SNMPv1, v2c and v3).
`trapdest remove`	*index*	Removes the RMON trap destination identified by *index*.

Table 66. Configuration commands (continued)

Command & sub-command	Arguments	Description
trapdest removenotify	*name*	Deletes the notification entry identified by *name* (the value of snmpNotifyName in the entry to be deleted).
trapdest removeparam	*name*	Deletes the target parameters entry identified by *name* (the value of snmpTargetParamsName in the entry to be deleted).
trapdest removetarget	*name*	Deletes the notification entry identified by *name* (the value of snmpTargetAddressName in the entry to be deleted).
trapdest set	{retries|timeout} *value*	Sets notification transmission parameters: retries - Sets the number of attempts to send a notification timeout - Sets the transmission timeout period (in ticks) The valid range of *value* is 0-255.
user add	*username* [{md5|sha} *authpass* [des *privpass*]]	Creates an SNMPv3 user.
user list		Lists all SNMPv3 users.
user remote add	*remote_addr*:*port* *username* [{md5|sha} *authpass* [des *privpass*] [*engineid*]]	Creates an SNMPv3 using the engine ID of a remote entity.
user remote remove	*remote_addr*:*port* *username*	Removes an SNMPv3 user that was defined using the engine ID of a remote entity.
user remove	*username*	Removes the SNMPv3 user identified by *username*.

vacm add entry	*security_group*
	security_model
	security_level
	{*read_view*|none}
	{*write_view*|none}
	{*notify_view*|none}
		Creates a VACM access entry.

vacm add group	*security_group* *security_name* *security_model*	Creates a VACM security group.
vacm add view	*name* *subtree* {included|excluded} [*mask*]	Creates a VACM MIB view.
vacm list		Lists the current VACM configuration.
vacm remove entry	*group* *sec_model* *sec_level*	Removes a VACM access entry.
vacm remove group	*group* *sec_name* *sec_model*	Removes a VACM security group.
vacm remove view	*name* *subtree*	Removes a VACM MIB view.
version		Displays the agent version.

Table 66. Configuration commands (continued)

Command & sub-command	Arguments	Description
`warmboot`		Forces the agent to perform a warm boot.

Appendix B. Regular expressions

Netcool/SSM enables you to use regular expressions in many subagent configuration commands.

Regular expression syntax

Table 67 describes the syntax of the regular expression tokens supported by Netcool/SSM.

Table 67. Regular expression syntax

Token	Matches
`.`	Any character.
`^`	The start of a line (a zero-length string).
`$`	The end of a line; a new line or the end of the search buffer.
`\<`	The start of a word (where a word is a string of alphanumeric characters).
`\>`	The end of a word (the zero length string between an alphanumeric character and a non-alphanumeric character).
`\b`	Any word boundary (this is equivalent to `(\<¦\>)`).
`\d`	A digit character.
`\D`	Any non-digit character.
`\w`	A word character (alphanumeric or underscore).
`\W`	Any character that is not a word character (alphanumeric or underscore).
`\s`	A whitespace character.
`\S`	Any non-whitespace character.
`\c`	Special characters and escaping. The following characters are interpreted according to the C language conventions: `\0`, `\a`, `\f`, `\n`, `\r`, `\t`, `\v`. To specify a character in hexadecimal, use the `\xNN` syntax. For example, `\x41` is the ASCII character `A`.
`\`	All characters apart from those described above may be escaped using the backslash prefix. For example, to specify a plain left-bracket use `\[`.

Table 67. Regular expression syntax (continued)

Token	Matches
`[]`	Any one of the specified characters in a set. An explicit set of characters may be specified as in `[aeiou]` as well as character ranges, such as `[0-9A-Fa-f]`, which match any hexadecimal digit. The dash (`-`) loses its special meaning when escaped, such as in `[A\-Z]` or when it is the first or last character in a set, such as in `[-xyz0-9]`. All of the above backslash-escaping rules may be used within `[]`. For example, the expression `[\x41-\x45]` is equivalent to `[A-D]` in ASCII. To use a closing bracket in a set, either escape it using `[\]]` or use it as the first character in the set, such as `[]xyz]`. POSIX-style character classes are also allowed inside a character set. The syntax for character classes is `[:class:]`. The supported character classes are: • `[:alnum:]` - alphanumeric characters. • `[:alpha:]` - alphabetic characters. • `[:blank:]` - space and TAB characters. • `[:cntrl:]` - control characters. • `[:digit:]` - numeric characters. • `[:graph:]` - characters that are both printable and visible. • `[:lower:]` - lowercase alphabetic characters. • `[:print:]` - printable characters (characters that are not control characters). • `[:punct:]` - punctuation characters (characters that are not letters, digits, control characters, or spaces). • `[:space:]` - space characters (such as space, TAB and form feed). • `[:upper:]` - uppercase alphabetic characters. • `[:xdigit:]` - characters that are hexadecimal digits. Brackets are permitted within the set's brackets. For example, `[a-z0-9!]` is equivalent to `[[:lower:][:digit:]!]` in the C locale.
`[^]`	Inverts the behavior of a character set `[]` as described above. For example, `[^[:alpha:]]` matches any character that is not alphabetical. The ^ caret symbol only has this special meaning when it is the first character in a bracket set.
`{n}`	Exactly n occurrences of the previous expression, where 0 <= n <= 255. For example, `a{3}` matches `aaa`.
`{n,m}`	Between n and m occurrences of the previous expression, where 0 <= n <= m <= 255. For example, a 32-bit hexadecimal number can be described as `0x[[:xdigit:]]{1,8}`.
`{n,}`	At least n or more (up to infinity) occurrences of the previous expression.
`*`	Zero or more of the previous expression.
`+`	One or more of the previous expression.
`?`	Zero or one of the previous expression.
`(exp)`	Grouping; any series of expressions may be grouped in parentheses so as to apply a postfix or bar (`¦`) operator to a group of successive expressions. For example: • `ab+` matches all of `abbb` • `(ab)+` matches all of `ababab`
`¦`	Alternate expressions (logical `OR`). The vertical bar (`¦`) has the lowest precedence of all tokens in the regular expression language. This means that `ab¦cd` matches all of `cd` but does not match `abd` (in this case use `a(b¦c)d`).

Tip: When defining regular expressions to match multi-byte characters, enclose each multi-byte character in parentheses ().

Regular expression examples

These examples demonstrate how to use regular expressions to perform pattern matching.

Table 68 provides a set of regular expression examples, together with sample strings as well as the results of applying the regular expression to those strings.

There are two important cases in matching regular expressions with strings. A regular expression may match an entire string (a case known as a *string match*) or only a part of that string (a case known as a *sub-string match*). For example, the regular expression `\<int\>` will generate a *sub-string match* for the string `int x` but will not generate a *string match*. This distinction is important because some subagents do not support sub-string matching. Where applicable, the results listed in the examples differentiate between string and sub-string matches.

Table 68. Regular expression examples

This expression...	Applied to this string...	Results in...
`.`	`a`	String match
	`!`	String match
	`abcdef`	Sub-string match on `a`
	empty string	No match
`M..COUNT`	`MINCOUNT`	String match
	`MXXCOUNTY`	Sub-string match on `MXXCOUNT`
	`NONCOUNT`	No match
`.*`	empty string	String match
	`Animal`	String match
`.+`	Any non-empty string	String match
	empty string	No match
`^`	empty string	String match
	`hello`	Sub-string match of length `0` at position `0` (position `0` = first character in string)
`$`	empty string	String match
	`hello`	Sub-string match of length `0` at position `5` (position `0` = first character in string)
`^$`	empty string	String match
	`hello`	No match
`\bee`	`tee`	No match
	`Paid fee`	No match
	`feel`	No match
	`eel`	Sub-string match on `ee`

Table 68. Regular expression examples (continued)

This expression...	Applied to this string...	Results in...
.*thing.*	The thing is in here	String match
	there is a thing	String match
	it isn't here	No match
	thinxxx	No match
a*	empty string	String match
	aaaaaaaaa	String match
	a	String match
	aardvark	Sub-string match on aa
	this string	Sub-string match
((ab)*c)*	empty string	String match
	ccccccccc	String match
	ccccabcccabc	String match
a+	empty string	No match
	aaaaaaaaa	String match
	a	String match
	aardvark	Sub-string match on aa
	this string	No match
((ab)+c)*	empty string	String match
	ababababcabc	String match
(ab){2}	abab	String match
	cdabababab	Sub-string match on abab
	ab	No match
[0-9]{4,}	123	No match
	a1234	Sub-string match on 1234
a{0}	empty string	String match
	a	No match
	hello	Sub-string match of length 0 at position 0 (position 0 = first character in string)
[0-9]{1,8}	this is not a number	No match
	a=4238, b=4392876	Sub-string match on 4238
([aeiou][^aeiou])+	Hello	Sub-string match on el
	!!! Supacalafraglistic	Sub-string match on upacalaf
[+-]?1	1	String match
	+1	String match
	-1	String match
	.1	Sub-string match on 1
	value+1	Sub-string match on +1

Table 68. Regular expression examples (continued)

This expression...	Applied to this string...	Results in...
a¦b	a	String match
	b	String match
	c	No match
	Daniel	Sub-string match on a
abcd¦efgh	abcd	String match
	efgh	String match
	abcdfgh	Sub-string match on abcd
[0-9A-F]+	BAADF00D	String match
	C	String match
	baadF00D	Sub-string match on F00D
	c	No match
	G	No match
	g	No match
x = \d+	x = 1234	String match
	x = 0	String match
	x = 1234a	Sub-string match on x = 1234
	x = y	No match
	x^=^ where ^ represents a space character	No match
\D\d	a1	String match
	a11	Sub-string match on a1
	-9	String match
	a	No match
	8	No match
	aa	No match
	4t	No match
\s+	Hello_w0rld	No match
	Hello^^^world where ^ represents a space character	Sub-string match on ^^^ where ^ represents a space character
	Widget^ where ^ represents a space character	Sub-string match on ^ where ^ represents a space character
	^^^^^ where ^ represents a space character	String match

Table 68. Regular expression examples (continued)

This expression...	Applied to this string...	Results in...
\S+	Hello_w0rld	Sub-string match of length 11 on Hello_w0rld
	Hello^^^world where ^ represents a space character	Sub-string match on Hello
	Widget^ where ^ represents a space character	Sub-string match on Widget
	^^^^^ where ^ represents a space character	No match
\w+	D4n_v4n Vugt	Sub-string match on D4n_v4n
	^^^hello where ^ represents a space character	Sub-string match on hello
	blah	String match
	x#1	No match
	foo bar	No match
\W	Hello there	Sub-string match of length 1 on separating space character
	~	String match
	aa	No match
	a	No match
	_	No match
	^^^^444 == 5 where ^ represents a space character	Sub-string match of length 1 on first ^ where ^ represents a space character
\w+\s*=\s*\d+	x = 123	String match
	count0=555	String match
	my_var=66	String match
	0101010=0	String match
	xyz = e	No match
	delta=	No match
	==8	No match
[[:alnum:]]+	1234	String match
	...D4N13L	Sub-string match on D4N13L
[[:alpha:]]+	Bubble	String match
	...DANI3L	Sub-string match on DANI
	69	No match

Table 68. Regular expression examples (continued)

This expression...	Applied to this string...	Results in...
[[:blank:]]+	alpha^^^^and beta where ^ represents a space character	Sub-string match on ^^^^ where ^ represents a space character
	Animal	No match
	empty string	No match
[[:space:]]+	alpha^^^^and beta where ^ represents a space character	Sub-string match on ^^^^ where ^ represents a space character
	Animal	No match
	empty string	No match
[[:cntrl:]]+	...Hello W0rld!	No match
	empty string	No match
[[:graph:]]+	hello world	Sub-string match on hello
	^^^^^ where ^ represents a space character	No match
	^^^!? where ^ represents a space character	Sub-string match on !?
[[:lower:]]+	Animal	Sub-string match on nimal
	ABC	No match
	0123	No match
	foobar	String match
	^^^0blaH! where ^ represents a space character	Sub-string match on bla
[_[:lower:]]+	foo_bar	String match
	this_thinG!!!	Sub-string match on _thin
[[:upper:]]+	YES	String match
	#define MAX 100	Sub-string match on MAX
	f00 b4r	No match
[[:print:]]+	hello world	String match
	^^^^^ where ^ represents a space character	String match
[[:punct:]]+	didn't	Sub-string match on '
	Animal	No match
[[:xdigit:]]+	43298742432392187ffe	String match
	x = bAAdF00d	Sub-string match on bAAdF00d
	4327afeffegokpoj	Sub-string match on 4327afeffe
c:\\temp	c:\temp	String match

Notices and trademarks

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785 U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
958/NH04
IBM Centre, St Leonards
601 Pacific Hwy
St Leonards, NSW, 2069
Australia

IBM Corporation
896471/H128B
76 Upper Ground
London SE1 9PZ
United Kingdom

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758 U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at "Copyright and trademark information” at http://www.ibm.com/legal/copytrade.shtml.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

Index

Printed in USA