Managing NerveCenter

Version 6.2

LogMatrix, Inc.
230 N. Serenata Drive, Suite 711
Ponce Vedra Beach, FL 32082 USA

Toll Free +1 (800) 892-3646
Phone +1 (508) 597-5300
Fax +1 (774) 348-4953

info@logmatrix.com
http://www.logmatrix.com

Table of Contents

C

Introduction

1

Managing NerveCenter describes how NerveCenter works and how you can monitor your network most effectively. This book is written for users operating the NerveCenter Client.

NerveCenter Documentation

This section describes the available NerveCenter documentation, which explains important concepts in depth, describes how to use NerveCenter, and provides answers to specific questions.

The documentation set is provided in online (HTML) format, as well as PDF for printing or on-screen viewing.

Using the Online Help

You can view the documentation with Google Chrome, Mozilla Firefox, Apple Safari, or Microsoft Edge. Refer to the NerveCenter Release Notes for the browser versions supported with this release.

Note: For in-depth instructions on using the online documentation, click the Help button ? in the upper right of the Help window.

Printing the Documentation

The NerveCenter documentation is also available as Portable Document Format (PDF) files that you can open and print. All PDF files are located in your `installpath/doc` directory.

Note: You must have Adobe Acrobat Reader to open or print the PDF files. You can download the Reader free from Adobe's Web Site at www.adobe.com.

The NerveCenter Documentation Library

The following documents ship with NerveCenter.

Book Title	Description	Application	Audience	PDF for Print
NerveCenter Release Notes	Describes new NerveCenter features and includes late-breaking information, software support, corrections, and instructions.	All	All	relnotes.pdf
Installing NerveCenter	Helps you plan and carry out your NerveCenter upgrades and new installations. Use the *Release Notes* in conjunction with this book.	All	Installation team	install.pdf
Managing NerveCenter	Explains how to customize and tune NerveCenter after it has been installed.	NerveCenter Administrator	Administrator	managing_ nervecenter.pdf
NerveCenter Platform Integration Guide	Explains how to integrate NerveCenter with network management platforms.	NerveCenter Administrator	Administrator	integratingNC. pdf
Learning to Create Behavior Models	Provides step-by-step instructions and examples for creating behavior models.	NerveCenter Client	Users with administrative privileges	learningModel. pdf
Designing and Managing Behavior Models	Explains behavior models in depth, how to create or modify models, and how to manage your models.	NerveCenter Client	Users with administrative privileges	designingModels.pdf
Monitoring Your Network	Explains how NerveCenter works and how you can most effectively monitor your network.	NerveCenter Client	Users	monitoringNet. pdf

UNIX Systems

On UNIX systems, NerveCenter man pages provide command reference and usage information that you view from the UNIX shell as with other system man pages. When you specify documentation during NerveCenter installation, the script installs nroff-tagged man pages and updates your system's MANPATH environment variable to point to the NerveCenter man page directory.

Document Conventions

This document uses the following typographical conventions:

Element	Convention	Example
Key names, button names, menu names, command names, and user entries	**Bold**	Press **Tab** Enter **ovpa -pc**
■ A variable you substitute with a specific entry ■ Emphasis ■ Heading or Publication Title	*Italic*	Enter **./installdb -f** *IDBfile*
Code samples, code to enter, or application output	`Code`	`iifInOctets > 0`
Messages in application dialog boxes	*Message*	*Are you sure you want to delete?*
An arrow (>) indicates a menu selection	**>**	Choose **Start > Programs > NerveCenter**

Caution: A caution warns you if a procedure or description could lead to unexpected results, even data loss, or damage to your system. If you see a caution, proceed carefully.

Note: A note provides additional information that might help you avoid problems, offers advice, and provides general information related to the current topic.

Documentation Feedback

LogMatrix, Inc. is committed to providing quality documentation and to helping you use our products to the best advantage. If you have any comments or suggestions, please send your documentation feedback to:

Documentation
LogMatrix, Inc.
230 N. Serenata Drive, Suite 711
Ponce Vedra Beach, FL 32082 USA

support@logmatrix.com

LogMatrix Technical Support

LogMatrix is committed to offering the industry's best technical support to our customers and partners. You can quickly and easily obtain support for NerveCenter, our proactive IT management software.

Professional Services

LogMatrix offers professional services when customization of our software is the best solution for a customer. These services enable us, in collaboration with our partners, to focus on technology, staffing, and business processes as we address a specific need.

Educational Services

LogMatrix is committed to providing ongoing education and training in the use of our products. Through a combined set of resources, we can offer quality classroom style or tailored on-site training.

Contacting the Customer Support Center

For Telephone Support

Phone: Toll Free +1 (800) 892-3646 or Phone +1 (508) 597-5300

For E-mail Support

E-mail: support@logmatrix.com.

Understanding NerveCenter

2

NerveCenter™ is an application designed to monitor, manage, and automate events in complex heterogeneous networks. Through a process called event correlation, NerveCenter helps you track significant events, quickly identify the root cause of critical network problems, and initiate corrective actions.This chapter explains the core NerveCenter functionality, describes the main NerveCenter components, and explains how these they interact with each other.

What is NerveCenter?

There are many network management tools designed to identify network faults and send alerts, but in doing so they may often flood the event console with raw data. Each critical or warning message indicating a potential problem — a failed router, for example — is usually accompanied by many additional messages regarding devices downstream that cannot be contacted. As a result, there is a great deal of information to sift through before the real problem can be identified and corrected.

NerveCenter is a network management tool that helps automate the identification, tracking, management, and resolution of important network events to facilitate proactive isolation and response to key events. NerveCenter uses the Simple Network Management Protocol (SNMP) to acquire data about managed devices, as well as Internet Control Message Protocol (ICMP) messages from your network to provide basic information about unresponsive devices.

For each device being monitored, NerveCenter's event correlation engine creates one or more finite state machines — or *alarms* — that define operational states of interest and the transitions between those states. These state machines enable NerveCenter to correlate data from multiple sources over time before concluding that a problem exists. As a simple example, an interface link-down trap might not indicate a problem if a link-up trap is received within a given amount of time, allowing both traps to be ignored. If the link-up trap does not follow in the given time, then NerveCenter can then implement predefined actions such as notifying an administrator, executing a script to correct the problem, or notifying a network management platform.

In addition to being an advanced event automation solution, NerveCenter is also a highly scalable cross-platform client/server application. It can run co-resident with a network management platform (such as IBM Tivoli Netcool/OMNIBus) and manage thousands of nodes, or distributed as a background process at tens or even hundreds of remote offices.

How NerveCenter Correlates Events

NerveCenter obtains data from SNMP agents running on managed nodes by processing incoming SNMP traps and polling the nodes for specific MIB values using SNMP Get requests. When a predefined network condition is detected, NerveCenter stores the event information in a finite state machine called an alarm. The alarm continues to track the status of the interface, node, or enterprise being monitored. The alarm waits for subsequent events or issues polls to determine if the condition warrants further action.

For example, by tracking events based on their persistence, NerveCenter can detect when a router experiences a persistent link-down condition. Upon detecting a downed communication link, NerveCenter waits to see if the link is restored within a given amount of time. If the link comes back up within the specified time, NerveCenter acknowledges the situation and returns the alarm to its normal (ground) state. However, if the link remains down, NerveCenter can perform a variety of actions based on your management strategy. Also, NerveCenter can detect when a group of nodes appears to be down or unreachable and then poll their parent router. If the router is down, further polling of those nodes is disabled until the router is back up.

To correlate and filter this data, NerveCenter relies on configurable network and system behavior models for each type of managed resource. A behavior model is a group of NerveCenter objects that detect and handle a particular network or system behavior. A typical behavior model consists of an alarm with its supporting polls and masks, though behavior models can have multiple alarms. Any managed device can be associated with one or more behavior models. For a description of behavior models, see "How Behavior Models Work" on the facing page.

Once a NerveCenter behavior model has identified a problem, it can take automatic actions, including notifying an administrator or a network management platform, executing a program or script, modifying the node’s properties, changing SNMP values, and logging the critical data.

How NerveCenter Manages Nodes

To automate events, NerveCenter relies on the definition of *behavior models*. These models are constructed from NerveCenter objects and define:

- Which devices, or nodes, a behavior model will affect
- How NerveCenter will detect certain conditions on these nodes
- How NerveCenter will correlate the conditions it detects
- How NerveCenter will respond to network problems

NerveCenter can get the list of nodes to monitor from a network management platform, discover them on the network, or import the information from another NerveCenter database. NerveCenter most often obtains data from SNMP (Simple Network Management Protocol) agents running on managed nodes.

NerveCenter assigns a set of properties to each managed node that determine the behavior models that can be applied. Properties typically describe the type of the device—for example, a router—or are named after objects in the management information base (MIB) used to manage the node.

After gathering data, the NerveCenter correlation engine examines detected network conditions and determines how the conditions may be related, or the underlying cause that produced the conditions. Detected conditions can be correlated in many ways, which include detecting the persistence of a condition, finding a set of conditions, and looking for a sequence of conditions.
For instance, NerveCenter may scan a large number of events and identify a subset that relate to SNMP authentication failures on a managed node. NerveCenter may then determine that the authentication failures were far enough apart that no problem exists, or it may find that several failures occurred within a short period of time — which may indicate a possible security problem.

NerveCenter not only enables you to detect network and system problems, but is able to respond automatically to the conditions it detects. To set up these automated responses, you associate *actions* with state transitions. For example, NerveCenter can notify an administrator via email or pager, log the information to a file, trigger a transition in another alarm, or execute a script.

For more information on nodes, see "Managing Node Data" on page 59.

How Behavior Models Work

NerveCenter detects events and compares these events to conditions of interest defined in behavior models. To understand how behavior models work, you must understand the objects that make up behavior models and how they interact.

NerveCenter uses the objects listed in Table 1 to define behavior models:

Table 1: NerveCenter Objects Used with Behavior Models

Object	Description
Node	Represents either a workstation, server, or a network device such as a router, hub, or bridge.
Property	A text string that describes the type of node or one of the node's MIB objects. Polls and alarms use properties to target specific nodes.
Property group	A group of properties assigned to a node. The properties in the group allow behavior models to target the node.
Trigger	An internal event that is generated when a defined network event has been detected or a condition satisfied. Triggers are generated by polls and trap masks, as well as by other NerveCenter objects such as Perl subroutines and NerveCenter alarms.
Poll	Periodically solicits defined SNMP values from the agents running on targeted nodes and fires a trigger when specified conditions are met. Polls can fire multiple triggers, which can be detected by one or more alarms.
Trap mask	Detects a predefined type of SNMP trap. Trap masks can specify generic and specific trap numbers, enterprise OIDs, and variable binding data to match against incoming traps. A trap mask fires a trigger when the specified type of trap is detected.
Alarm	Detects triggers that cause the alarm to transition from one state to the next. Each transition is triggered by its own set of network data, which is defined in the poll or mask that generates the associated trigger. The alarm performs any actions assigned to a transition.

Behavior Model Operation

A behavior model is not a NerveCenter object. Rather, it is a collection of NerveCenter objects designed to monitor specific network devices, events, or behaviors. Figure 1 shows the interaction of objects in a simple behavior model.

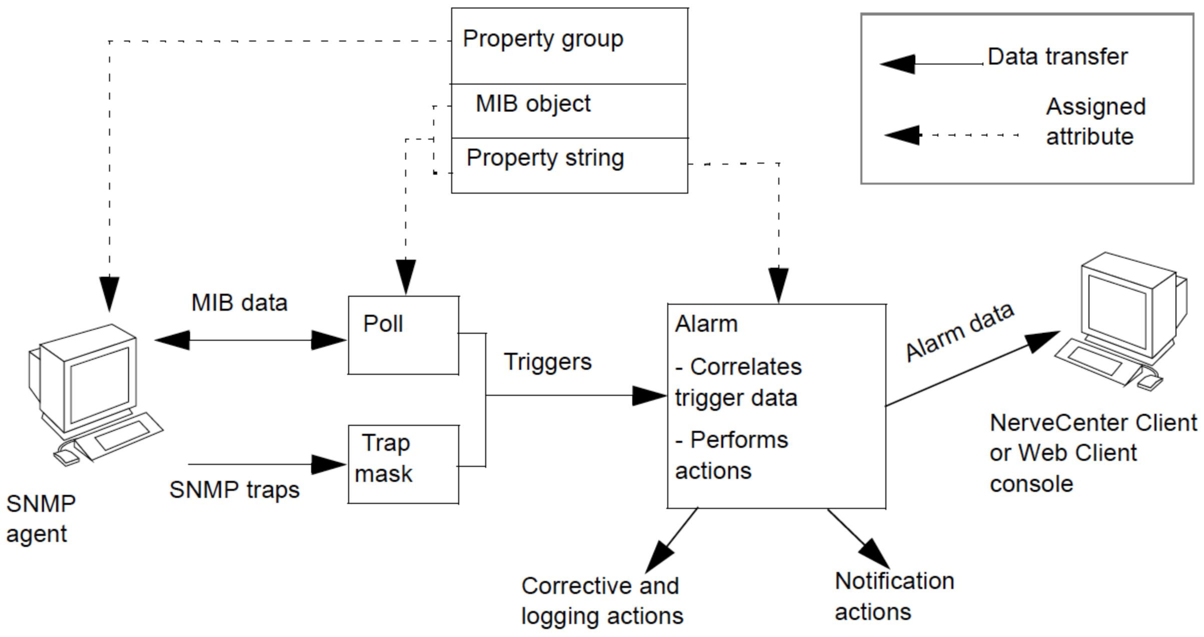


Figure 1: A Simple Behavior Model

When a trigger-generating object detects an event of interest on a node, it raises a trigger. This trigger generator might be a NerveCenter poll that looks for a specified network condition, or a trap mask that detects a certain type of SNMP trap. NerveCenter alarms internally subscribe for triggers based on alarm states. When an alarm detects its first trigger, the alarm transitions to a predefined state. Based on the state, the alarm ignores triggers that can no longer cause transitions and listens instead for those triggers that can. The alarm waits in that state until another trigger is received—either from the same or another trigger generator. This allows an alarm to react only to relevant events and poll only for relevant data. For more information about alarms, see "Alarms" on the facing page.

How Properties Affect Behavior Models

Properties allow you to limit excessive polling and control which nodes are targeted by which alarms. A property is text string that describes the type of node or one of the node’s MIB objects. A property group is a group of properties assigned to a node. Each node is targeted based on its assigned property group, which can contain many properties. Before an alarm can be applied to a node, the node's group of properties must include any property specified in the alarm or its associated polls. Additionally, before a node can be polled, the node’s property group must contain the MIB base object that the poll is designed to evaluate. Assigning a node to a property group with multiple properties allows the node to be targeted by multiple alarms.

Alarms

Alarms are key to behavior model operation. An alarm is a finite state machine that defines the operational states it wants to detect and the transitions from one state to the next when the proper trigger is received. Each transition is triggered by its own set of network data, which is defined in the NerveCenter object that generates the associated trigger. Transitions can also be driven by any alarm action that causes a trigger to be fired. If actions are associated with a transition, the server performs these actions each time the transition takes place.

Figure 2 shows the state diagram for an alarm called IfUpDownStatus. IfUpDownStatus monitors the operational status of interfaces on managed nodes. If an interface is down, an inform action notifies the network management platform.

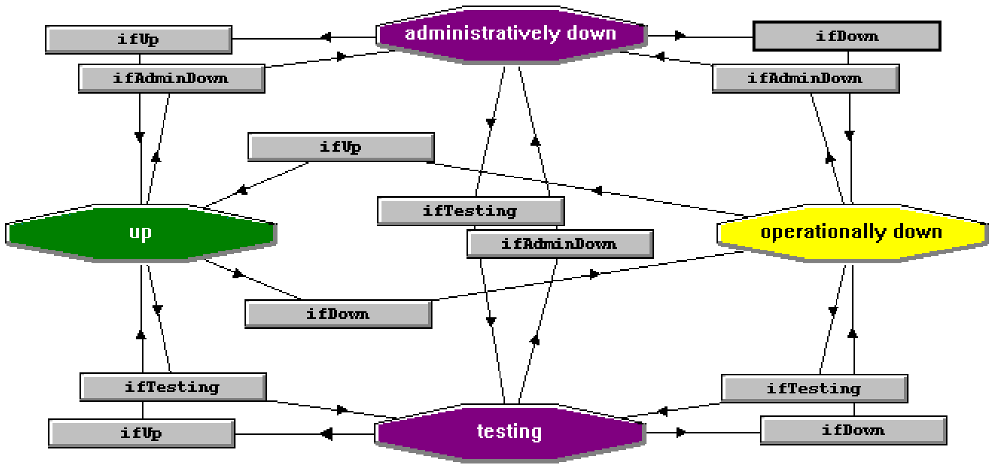


Figure 2: IfUpDownStatus Alarm

Each state is depicted as an icon in the diagram. The state icons are color coded to represent a particular level of severity.

The alarm contains four states related to the status of an interface:

- **Up**—The interface is up. The current (operational) and desired (administrative) statuses are set to 1 (defined in IF-MIB (RFC2863) as up). This state has a severity level of Normal.
- **Administratively down**—Both the current (operational) and the desired (administrative) statuses are set to 2, defined in IF-MIB (RFC2863) as down.
- **Testing**—Either the current (operational) or desired (administrative) status is set to 3, defined in IF-MIB (RFC2863)as test mode.
- **Operationally down**—The current (operational) status is down but the desired (administrative) status is up. This state has a high severity level of Major. All transitions leading to this state include an alarm action that sends an inform to NerveCenter or to the platform.

The ifStatus poll fires all the triggers that transition the IfUpDownStatus alarm. The poll is designed to evaluate the MIB base object attributes ifAdminStatus and ifOperStatus, which belong to the ifEntry base object in the interfaces MIB-II group. The poll fires four triggers—one for each of the four states previously described—based on the values returned for the two attributes. The alarm correlates the data it receives from the poll and transitions based on the combinations shown in Table 2.

Table 2: Correlation of IfEntry Attributes in the IfUpDownStatus Alarm

IfUpDownStatus Status	ifAdminStatus	ifOperStatus
Up	1 (down)	1 (down)
Administratively down	2 (up)	2 (up)
Testing	3 (testing)	3 (testing)
Operationally down	1 (down)	2 (up)

In this alarm, a trigger representing a condition other than up for either attribute—ifAdminStatus or ifOperStatus—transitions the alarm from the Up state. A subsequent trigger representing a condition other than up for the remaining attribute then transitions the alarm to a different state.
The state to which it transitions depends on the combination of the values for the two attributes.

To learn more about NerveCenter's behavior models, refer to the following sources:

- Learning to Create Behavior Models and Designing and Managing Behavior Modelscontain complete descriptions and procedures for behavior models.
- NerveCenter objects, such as alarms, include notes that describe the object and provide specific information related to that object. For example, the notes for an alarm include a list of triggers that transition the alarm, the objects that fire those triggers, and any actions associated with the alarm's transitions.

Alarm Actions

NerveCenter can implement a broad range of actions when an alarm is instantiated. A transition might cause NerveCenter to send a trap, email, or page. NerveCenter has its own type of message—an inform—that contains the variable bindings associated with the event that caused the alarm to transition. Informs can be sent to one or more destinations, including network management platforms and other NerveCenters. For each destination, you can specify a minimum severity level that your network conditions must meet before the informs are issued.

NerveCenter can call a Perl subroutine as an alarm action. The Perl subroutine might fire a trigger when certain conditions are met or execute other commands that effect changes to your network. Perl subroutines have access to associated alarm and trigger data for nodes that cause alarms to transition. Subroutines can then evaluate this information and conditionally take further action.

NerveCenter can also log alarm data and perform corrective actions such as setting SNMP attribute values. Finally, all actions can be performed conditionally using the NerveCenter Action Router. The Action Router performs actions based on user-defined criteria such as time of day, severity of the alarm, type of node, and so on.

Other Trigger Generators

Alarm transitions can trigger actions that, themselves, generate triggers. Trigger-generating actions can cause a transition in another alarm or transition yet another state in the same alarm. An alarm can also execute a Perl subroutine, which can fire a trigger when certain conditions are met. In both cases, the alarm serves as a trigger generator that serves an important function in a behavior model. These types of trigger generators provide an additional level of control over your alarms.

Figure 3 shows the interaction of various trigger generators in a behavior model.

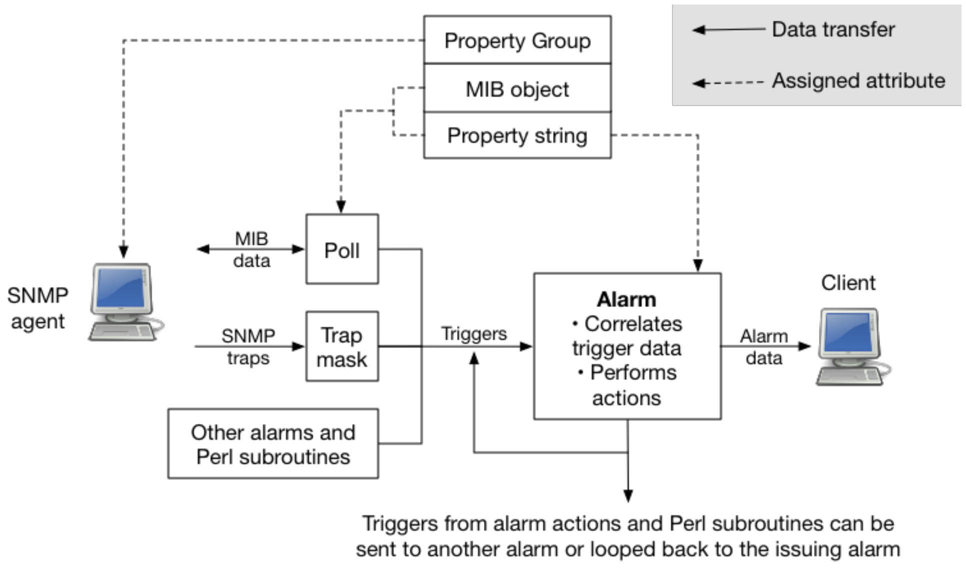

Property Group
MIB object
Property string
Data transfer
Assigned attribute
SNMP agent
MIB data
Poll
SNMP traps
Trap mask
Triggers
Alarm
• Correlates trigger data
• Performs actions
Alarm data
Client
Other alarms and Perl subroutines
Triggers from alarm actions and Perl subroutines can be sent to another alarm or looped back to the issuing alarm

Figure 3: A Behavior Model with Triggers Generated by Alarms

NerveCenter Components

This section introduces the main NerveCenter components:

NerveCenter Server

The NerveCenter Server is responsible for carrying out all of the major tasks that NerveCenter performs. For example, it handles the polling of SNMP agents, creates NerveCenter objects such as the finite alarms mentioned earlier, and makes sure that state transitions occur at the appropriate times. The server also performs all actions associated with state transitions.

The server can run as a daemon on UNIX systems. This capability to run in the background has important implications with regard to using NerveCenter at remote sites. You can install the server and database at a remote office and have that server manage the local network, yet control the server (via the NerveCenter Client) from a central location. Servers located at remote sites can forward noteworthy information to a server at the central location as required.

NerveCenter Database

The NerveCenter database is primarily a repository for the NerveCenter objects that make up a set of behavior models; the basic objects are introduced in "NerveCenter uses the objects listed in Table 1 to define behavior models:" on page 7. A set of objects that define many useful behavior models ships with NerveCenter and is available as soon as you've installed the product. The NerveCenter database is implemented as a flat file.

NerveCenter MIB

NerveCenter enables you to control which types of devices are managed based on their SNMP management information base (MIB) definitions. Figure 4 shows the NerveCenter MIB files along with the other NerveCenter components.

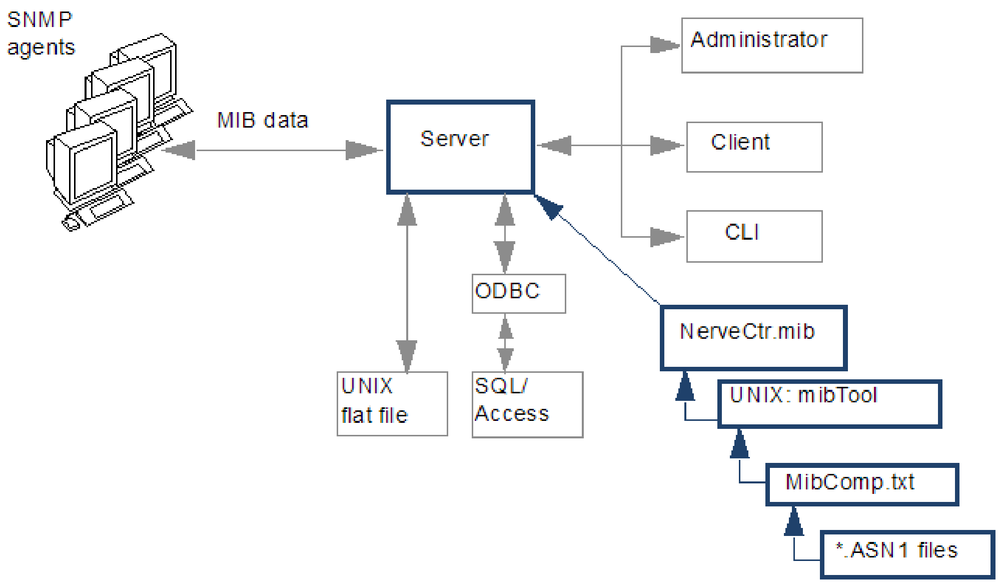


Figure 4: NerveCenter MIB Files

The compiled MIB file shipped with NerveCenter, nervectr.mib, contains definitions for Internet Standard RFC SNMP Versions 1, 2 and 3 agents as well as many vendor specific MIBs. nervectr.mib defines the management information available from an SNMP agent based on the MIBs the agent supports. This support information is contained in the .ASN1 file for each managed device. NerveCenter's MIB file enables you to construct SNMP requests for particular devices and to decode data received in SNMP traps.

nervectr.mib is compiled from definitions referenced in a text file, which by default is named MibComp.txt, which in turn points to the *.ASN1 files you are using. To add support for new devices, modify MibComp.txt to include the new *.ASN1 files, located within the directory /var/opt/NerveCenter/mib/. You must have the proper .ASN1 file for each device that is defined.

After modifying the text file, you recompile the MIB file. NerveCenter's MIB compiler, mibcomp, verifies each respective .ASN1 file, compiles the definitions in the text file, and creates a new version of nervectr.mib that can then be loaded into the NerveCenter Server.

NerveCenter Interfaces

This section introduces NerveCenter's principal user interfaces:

- "The NerveCenter Administrator" below
- "The NerveCenter Client Console" on page 15
- "The NerveCenter Client" on page 16
- "The Command Line Interface" on page 17

The NerveCenter Administrator

Users with administrator privileges can use the NerveCenter Administrator interface to:

- Configure NerveCenter discovery mechanisms
- Configure the number of SNMP polling retries and the retry interval
- Configure NerveCenter mail and paging actions
- Manage NerveCenter log files
- Configure NerveCenter to work with a network management platform

Figure 5 shows the NerveCenter Administrator.

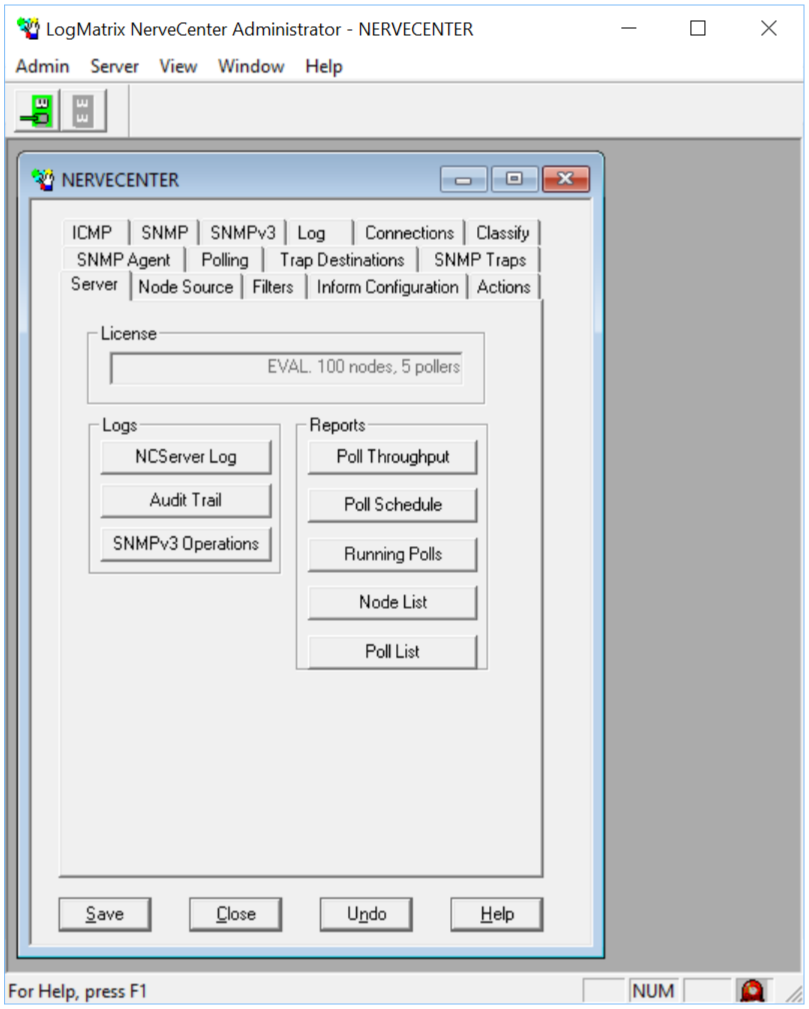


Figure 5: NerveCenter Administrator

The NerveCenter Client Console

The NerveCenter Client lets you monitor the network as well as create and modify the behavior models managed by the NerveCenter Server. You can access the MIBs, nodes, behavior models, alarm filters, and other NerveCenter objects associated with that Server. The Alarm Summary window displays all current alarms for the connected Server.

Figure 6 shows a sample Alarm Summary window.

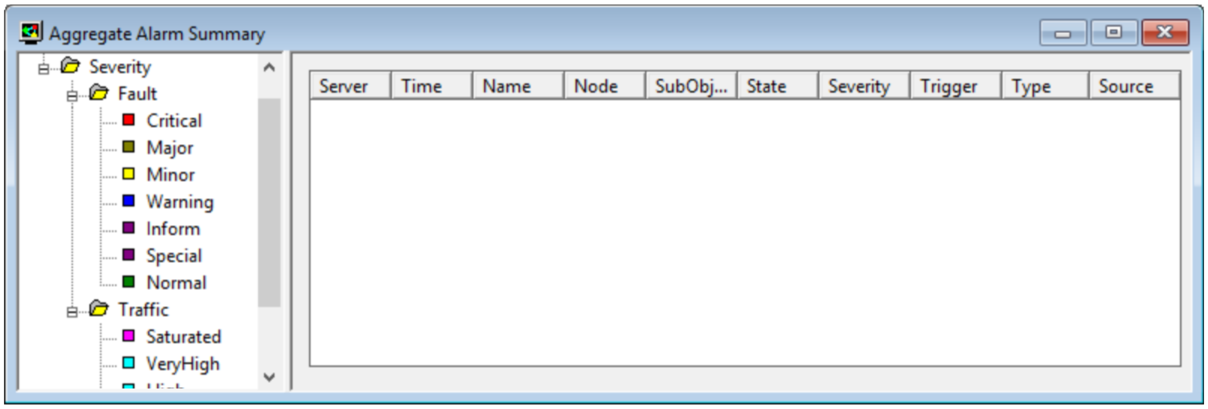


Figure 6: Alarm Summary Window

The left pane contains a tree that displays the total number of current alarm instances, the number of instances in each severity group (Fault and Traffic), and the number of instances of each severity. If there is no number next to a severity, there are no active alarm instances of that severity.

The right pane provides detailed information about current alarm instances for the folder or severity that is highlighted in the tree view.

The NerveCenter Client

The figure below shows the GUI for the NerveCenter Client.

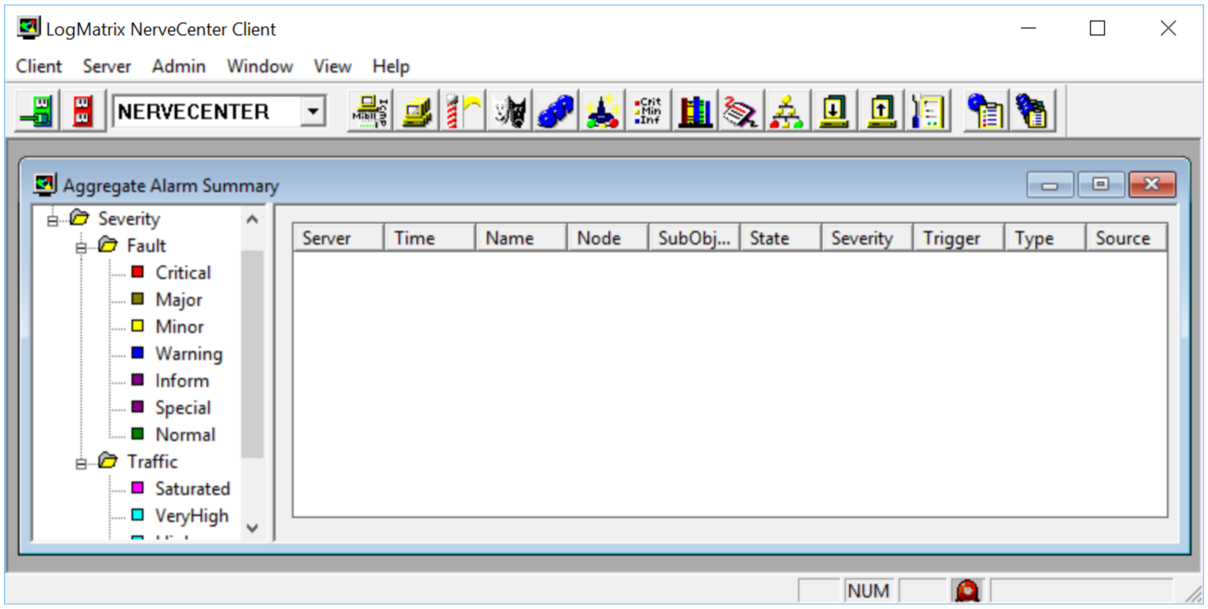


Figure 7: NerveCenter Client

Two types of users run the NerveCenter Client. Users with NerveCenter User privileges can run the client to:

- Monitor active alarms
- Filter alarms for the alarm summary windows
- View an alarm’s history
- Reset alarms
- Monitor the state of managed nodes
- Generate reports

For complete information on using the NerveCenter Client to perform the tasks listed above and others, see Monitoring Your Network.

Users with NerveCenter Administrator privileges can perform all the tasks that users with User privileges can. In addition, they can use the client to:

- Create new behavior models
- Customize the predefined behavior models
- Modify, copy, or delete any object in the NerveCenter database

The Command Line Interface

You can use NerveCenter's command line interface (CLI) to delete, list, or set (enable or disable) alarms, trap masks, nodes, and polls from a Windows Command Prompt or a UNIX shell. You can also connect to, display the status of, and disconnect from NerveCenter servers using the CLI. You can issue commands manually or from a script.

See the appendix "Controlling NerveCenter from the Command Line" on page 195 for command descriptions.

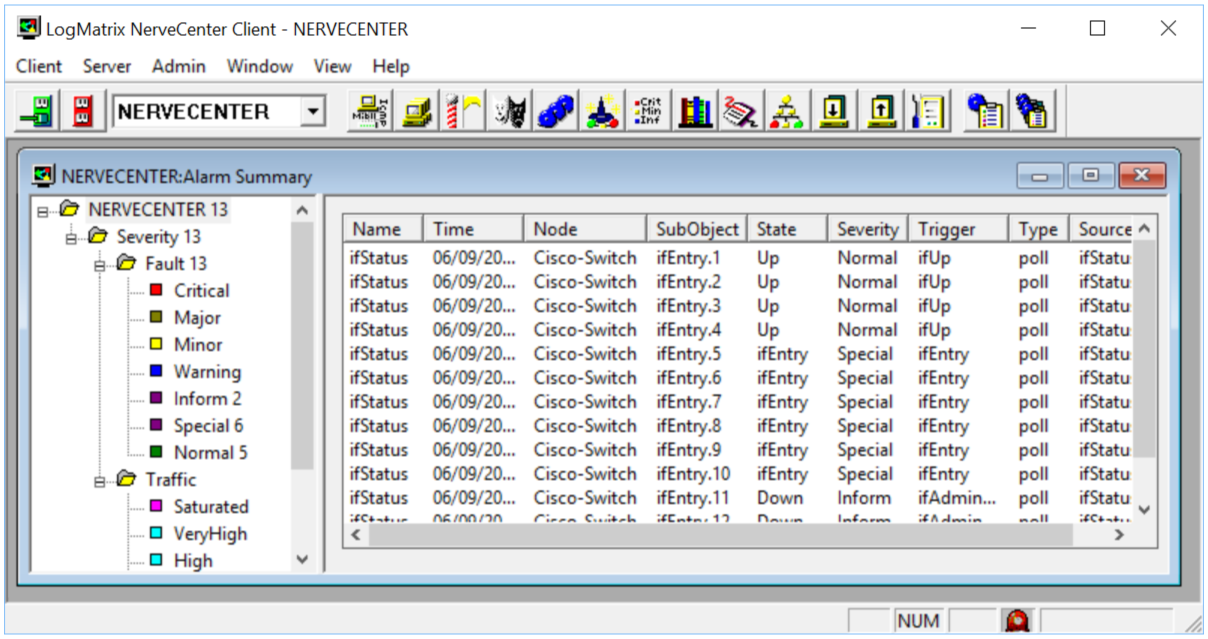
Role in Network Management Strategy

NerveCenter can play a variety of roles in an overall network management strategy. The role that NerveCenter plays in your strategy depends largely on the size of your network and on what other products you are using to manage your network and systems:

- If you are managing a small network, NerveCenter can be used as a standalone system. It can discover the workstations and network devices on the network, detect and correlate network conditions, respond automatically to conditions, and display information about active alarms. See the section "Standalone Operation" on the next page for further information.
- For larger networks, multiple NerveCenters can be used in concert. Local NerveCenter systems could be set up to manage remote sites, and the local NerveCenter servers could forward important information to the NerveCenter server at the central site. See the section "Using Multiple NerveCenter Servers" on page 19 for further information.
- NerveCenter can be used in conjunction with a network management platform such as IBM Tivoli Netcool/OMNIbus which manages systems systems, networks, intranets, and databases. NerveCenter can be configured to receive messages from or send messages to this and similar network management platforms. See the section "Integration with Network Management Platforms" on page 19 for further information.

Standalone Operation

At smaller sites, you can use NerveCenter alone for your network management tasks. As we've seen, NerveCenter is very strong in the areas of event correlation and automated actions. In addition, NerveCenter includes an alarm console, as shown in Figure 8.



Name	Time	Node	SubObject	State	Severity	Trigger	Type	Source
ifStatus	06/09/20...	Cisco-Switch	ifEntry.1	Up	Normal	ifUp	poll	ifStatu
ifStatus	06/09/20...	Cisco-Switch	ifEntry.2	Up	Normal	ifUp	poll	ifStatu
ifStatus	06/09/20...	Cisco-Switch	ifEntry.3	Up	Normal	ifUp	poll	ifStatu
ifStatus	06/09/20...	Cisco-Switch	ifEntry.4	Up	Normal	ifUp	poll	ifStatu
ifStatus	06/09/20...	Cisco-Switch	ifEntry.5	ifEntry	Special	ifEntry	poll	ifStatu
ifStatus	06/09/20...	Cisco-Switch	ifEntry.6	ifEntry	Special	ifEntry	poll	ifStatu
ifStatus	06/09/20...	Cisco-Switch	ifEntry.7	ifEntry	Special	ifEntry	poll	ifStatu
ifStatus	06/09/20...	Cisco-Switch	ifEntry.8	ifEntry	Special	ifEntry	poll	ifStatu
ifStatus	06/09/20...	Cisco-Switch	ifEntry.9	ifEntry	Special	ifEntry	poll	ifStatu
ifStatus	06/09/20...	Cisco-Switch	ifEntry.10	ifEntry	Special	ifEntry	poll	ifStatu
ifStatus	06/09/20...	Cisco-Switch	ifEntry.11	Down	Inform	ifAdmin...	poll	ifStatu

Figure 8: NerveCenter's Alarm Console

This console displays information about every current alarm instance. In addition, if you double-click on a line in the event console, you are taken to an Alarm History window that displays information about all of the alarm transitions that have occurred for the alarm instance you selected.

At small installations, no discovery mechanism is necessary; you can add nodes to NerveCenter manually. At somewhat larger sites, however, such a mechanism is helpful, and NerveCenter provides one in its Discovery behavior model.

Using Multiple NerveCenter Servers

Because a NerveCenter server can inform another NerveCenter server or management platform of a network condition, it's possible to set up NerveCenter servers at remote sites that notify a centrally located NerveCenter server or management platform (Figure 9).

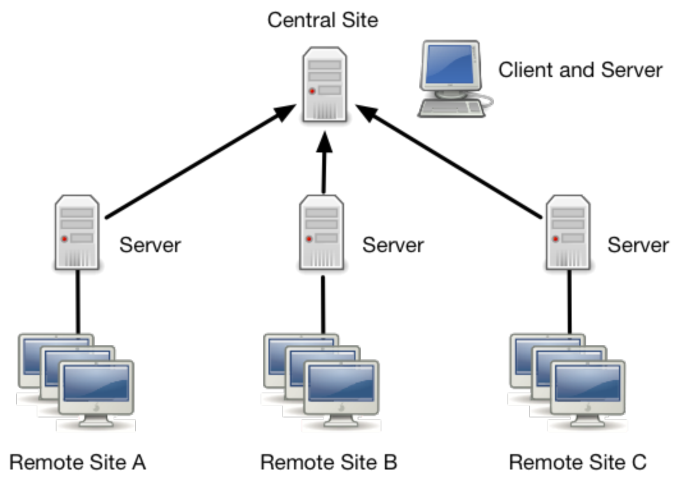


Figure 9: Distributed NerveCenter Servers

This is a reliable solution because the remote NerveCenter servers use TCP/IP to notify the central NerveCenter server and retransmit messages as necessary to ensure their delivery. There are a few advantages to this type of setup:

- Only a small amount of data is transmitted over the WAN. Any bandwidth intensive monitoring is conducted on a LAN and is managed by a remote NerveCenter server.
- Remote NerveCenter servers can be run in lights-out mode, which means that:
  - NerveCenter runs as a UNIX daemon
  - You can monitor and configure NerveCenter from a remote location
  - You can modify all NerveCenter parameters without shutting NerveCenter down
  - No display or operators are required at a site
- The central NerveCenter can further correlate and filter conditions across remote NerveCenter Server domains

Integration with Network Management Platforms

A network management platform (NMP) is an operations and problem-management solution for use in a distributed multi-vendor environment. Intelligent distributed agents on managed nodes monitor system and application log files and SNMP data. The agents apply filters and thresholds to monitored data and forward messages about conditions of interest to a central management station. When the management station receives messages, it can automatically take corrective action—such as broadcasting a command to a set of systems—or an operator can initiate a response.

You can integrate your NerveCenter installation with the NMP so that the NMP can send messages to NerveCenter for correlation or processing. After the messages arrive, NerveCenter correlates the conditions described in these messages with related conditions—from the NMP or from other sources—and can respond with any of its alarm actions, as appropriate. In addition, NerveCenter can send a message to an NMP in response to any network condition, whether the condition was originally detected by the NMP or not.

NMPs alone can detect a condition and invoke an action in response. However, you must integrate the NMP with NerveCenter if you want to:

- Correlate conditions detected by the NMP on different devices
- Correlate different types of conditions detected by the NMP on the same device
- Correlate conditions detected by the NMP with other types of events or conditions on the same device or across different devices

See the NerveCenter Platform Integration Guide for more information.

Administering NerveCenter

NerveCenter is designed to run in the background, quietly monitoring your network. Fortunately, it does not require a great deal of administrative attention. There are, however, some initial settings as well as ongoing maintenance issues that must be attended to. A person designated as a NerveCenter administrator usually handles these tasks.

Who Uses NerveCenter?

Although not necessary, you may find it helpful to think of three different groups who use NerveCenter:

- **basic users** – monitor their environment using NerveCenter Client. They have user login rights and should read Monitoring Your Network.
- **power users** – design and manage behavior models using the NerveCenter Client. They have administrator login rights and should read Designing and Managing Behavior Models and Learning to Create Behavior Models.
- **administrators** – manage the administrative tasks related to NerveCenter. They have administrator login rights and use the NerveCenter Administrator tool. A NerveCenter administrator should read this book.

Table 3 illustrates the differences between the three.

Table 3: The Differences between Three Groups of NerveCenter Users

	Primary Task	Primary Tool	Rights*	Book
Basic user	Monitoring NerveCenter	NerveCenter Client	User	Monitoring Your Network
Power user	Designing and managing behavior models	NerveCenter Client	Administrator	Designing and Managing Behavior Models Learning to Create Behavior Models
Administrator	Administering NerveCenter	NerveCenter Administrator	Administrator	Managing NerveCenter

NerveCenter Login Rights

By default, during installation NerveCenter establishes two groups with different login rights — Administrators and Users. Only those with NerveCenter administrator login rights can start the NerveCenter Administrator application. Those with either NerveCenter administrators and users login rights can start the NerveCenter Client application. However, opening under NerveCenter Users login rights limits what you can do with the Client.

Table 4 illustrates the difference between NerveCenter administrators and NerveCenter users login rights.

Table 4: User and Administrator Login Rights in NerveCenter Client

	User	Administrator
Monitor active alarms	✓	✓
View an alarm's history	✓	✓
Reset alarms	✓	✓
Monitor the state of managed nodes	✓	✓
Generate reports	✓	✓
Create new behavior models		✓
Customize the predefined behavior models		✓
Modify, copy, or delete an object in the NerveCenter database		✓

The Role of a NerveCenter Administrator

Because NerveCenter is intended to run in the background, quietly monitoring your network, it does not require a great deal of administrative maintenance. There are, however, a few regular habits a NerveCenter administrator should develop. These include:

- Backing up the NerveCenter database.
- Checking to see that there is available disk space for each log file.
- Checking log files to make sure they are truncating to the correct size. See "Specifying Settings for Log Management" on page 149.
- Checking for logged errors or messages in the UNIX system log files. These errors or messages could indicate problems that could be currently affecting NerveCenter performance or could later develop into a problem.
- Checking all your NerveCenter Server's Server Status pages to make sure all necessary connections are available. See "Viewing NerveCenter Server Status" on page 28. You will want to monitor information on the following tabs:
  - Node Source: Is the connection with the node data source still available?
  - Inform Configuration: Is the inform recipient list current?
  - Connected NerveCenter: Does this list show all the NerveCenter Servers that may potentially send the active server an inform packet?
- Updating any relevant changes in IP addresses, such as the location of a node data source or inform recipients.
- Keeping various NerveCenter login rights current to reflect changes in personnel. See "Managing NerveCenter Security" on page 141.
- Maintaining your network management platform with good procedures.

Connecting to a NerveCenter Server

3

For you to administer one or more NerveCenter Servers, you will need to make a connection between the NerveCenter Server and a NerveCenter Administrator application.

Running the NerveCenter Server

Before a client can connect to a NerveCenter Server, the Server must be installed and running. See Installing NerveCenter for instructions on installing the NerveCenter Server and other applications.

If NerveCenter is not running as a UNIX daemon, you can start it manually. NerveCenter Server can be started by either the superuser account or by an account that has been established as being an account under which the NerveCenter Server is allowed to run.

Caution: The NerveCenter server will not start without a valid license file, as described in Installing NerveCenter.

To start NerveCenter Server

- Run the following command:

```
/opt/OSInc/bin/ncstart
```

Note: If you want to disable all alarms when you start the server (for example, if you create an alarm that crashes the server), add -off as an argument to the ncstart command.

To stop NerveCenter Server

- Run the following command:

```
/opt/OSInc/bin/ncstop
```

For other command line switches, see "Controlling NerveCenter from the Command Line" on page 195.

Troubleshooting: Running the NerveCenter Server

This section contains some common problems users may have when running the NerveCenter Server.

IS THE NERVECENTER SERVICE RUNNING?

To test whether the NerveCenter service running, use `/opt/OSInc/bin/ncstatus`.

If your system does not have ‘nstatus’ command, you get it from http://docs.logmatrix.com/NerveCenter/6.2.00/index.html#tools-ncstatus

You must be logged into the same host where the NerveCenter service should be running; however you do not need to be the ‘root’ account to use ‘ncstatus’. If you have set up your shell environment to include `/opt/OSInc/bin` in the `$PATH` setting, then just `ncstatus` can be used.

Examples:

```
$ ncstatus
  bash: ncstatus: command not found
$ source /opt/OSInc/userfiles/ncenv.bash
$ ncstatus
  nervecenter service is stopped.
$ ncstart
$ ncstatus
  nervecenter service is running (pid 5799)
```

NERVECENTER KEEPS STOPPING OR EXITS SUDDENLY

If ncserver is not running, look at its log file (`/var/opt/NerveCenter/log/ncserver.log`) for statements on why it is exiting.

```
# ps -ef | grep ncserver
# ncstatus
  nervecenter service is stopped.
# tail /var/opt/NerveCenter/ncserver.log
```

Also check the log file(s) updated by the host’s syslog service. NerveCenter writes message to syslog and the cause for an non-running ncserver might be found in the syslog output.

COMMON REASONS FOR THE NERVECENTER SERVER TO EXIT

- The license is expired or is missing
  - Check for the file `/opt/OSInc/conf/`*`<hostname>`*`.dat`, where <hostname> is the value returned by `/bin/hostname`. (e.g., if the hostname is “nms01” , then look for /opt/OSInc/conf/nms01.dat)
  - If the license is expired this will be stated in `/var/opt/NerveCenter/log/ncserver.log`
- The group ‘ncadmins’ has been removed from `/etc/group`
  - Check that ‘ncadmins’ and ‘ncusers’ are in `/etc/group`
- The account used by NerveCenter is disabled or has been removed
  - If the files `ncstart-user` or `ncstart-authorized` exist in `/opt/OSInc/conf/`, check that the accounts given in these files exist and are explicitly listed as members of `/etc/group` of `ncadmins`.
- A file ncserver writes to has grown to the max allowed for a 32-bit process on Linux/UNIX.
  - Scan for large files under `/var/opt/NerveCenter`, `/opt/OSInc`, and `/tmp`

AN ALARM CAUSES THE NERVECENTER SERVER TO CRASH EVERY TIME I START IT

Problem: Any alarm that is enabled when a NerveCenter Server stops is still enabled when the NerveCenter Server starts again.

Solution: Start the NerveCenter Server with all its alarms turned off by typing at the command prompt **/opt/OSInc/bin/ncstart -off**.

Starting NerveCenter Administrator

The NerveCenter Administrator provides an administrator information about one or more active NerveCenter Servers. An administrator also uses the application to configure settings on a NerveCenter Server.

- From the Windows Start button, select **LogMatrix NerveCenter 6.2 > Administrator**.

  A NerveCenter Administrator window opens.

Note: It is also possible to start the NerveCenter Administrator from a command prompt (or **Start > Run**) by typing **ncadmin**.

Connecting Administrator to a NerveCenter Server

Before you can use the NerveCenter Administrator, you must connect to a NerveCenter Server. The NerveCenter Server provides the functionality required to manage your network. Also, it gives the NerveCenter Administrator access to the settings found in the NerveCenter database.

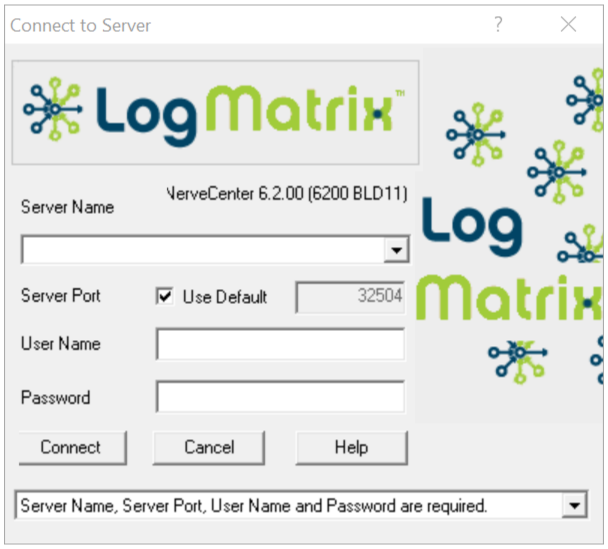
You can connect your NerveCenter Administrator to more than one server at one time. However, only one NerveCenter Server can be the *active server* at a time. The active server determines which NerveCenter database is used when you are configuring settings.

To connect with NerveCenter Administrator to a server, you must be able to access an account on the server host where the account is a member of the NerveCenter Administrators group (ncadmins).

To connect a NerveCenter Administrator to a NerveCenter Server

1. Open NerveCenter Administrator. See "Starting NerveCenter Administrator" on the previous page.
2. From the **Server** menu, select **Connect Server**.

   The Connect to Server window is displayed.



NerveCenter 6.2.00 (6200 BLD11)

Server Name

Server Port ☑ Use Default 32504

User Name

Password

Server Name, Server Port, User Name and Password are required.

3. Do one of the following:
   a. If this is the first time connecting to the NerveCenter Server, type in the Server Name field the hostname or IP address of the machine on which the NerveCenter Server is running.
   b. If you have connected to the server before, select the name or IP address from the Server Name list.

   The first time you connect to a server, the list is empty. After that, it contains a list of the machines to which you've connected, or attempted to connect, in the past. The list will not display the names of machines to which this NerveCenter Administrator is already connected. (For information on removing an entry from the Server Name list, see the section "Deleting a Name from the Administrator Server List" on the facing page.)

4. Type a user name and password in the **User ID** and **Password** fields.

   You must enter a user name and password. The user whose name you enter here must be a member of the NerveCenter Administrators group ncadmins.

5. Select **Connect**.

   If the machine to which you try to connect is not running a NerveCenter Server, you will see the message: **The server did not respond.** You must first start the NerveCenter Server and then try to connect again. See "Running the NerveCenter Server" on page 23 for more details.

When the NerveCenter Administrator connects to a NerveCenter Server, the Administrator window for that server appears.

Connecting Administrator to Multiple NerveCenter Servers

It is possible to use a NerveCenter Administrator to connect to more than one NerveCenter Server at the same time. However, only one NerveCenter Server can be the active server at a time. The active server determines which NerveCenter database is used when you are configuring settings.

To connect a NerveCenter Administrator to more than one NerveCenter Server, see "Connecting Administrator to a NerveCenter Server" on page 25. NerveCenter Administrator displays a window for each connected NerveCenter Server. The name of the active server always appears in the title bar of the main NerveCenter Administrator window.

Deleting a Name from the Administrator Server List

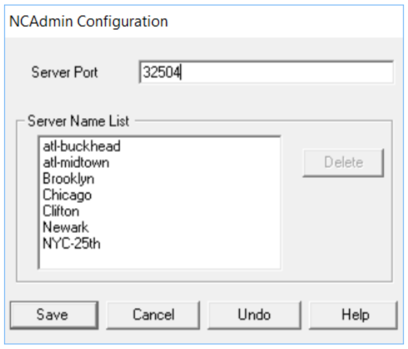
NerveCenter maintains a list of servers that the NerveCenter Administrator has attempted to connect to in the past. This list is used in the Connect to Server window, which you use to connect the administrator to a server.

As helpful as this list is, there may be times when you need to delete a server from this list. It may be the name of a server that you will never connect to again. It could also be the misspelled name of a server you were unable to connect to because of the misspelling.

To delete a server from the Administrator's server name list

1. Open NerveCenter Administrator. See "Starting NerveCenter Administrator" on page 25.
2. From the **Admin** menu, select **Configuration**.

   The NCAdmin Configuration window is displayed.



Server Port 32504

Server Name List
atl-buckhead
atl-midtown
Brooklyn
Chicago
Clifton
Newark
NYC-25th

   The Server Name List contains the names of all the machines this NerveCenter Administrator has attempted to connect to in the past.

3. In the Server Name List, highlight the machine name that you wish to delete.

4. Select **Delete**.

   The machine name is removed from the Server Name List.

5. Select **Save**.

The server's name is permanently deleted from the Server Name List for this NerveCenter Administrator.

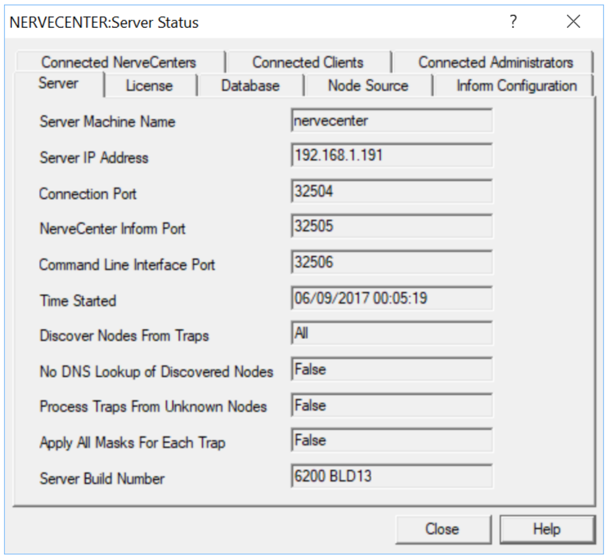
Viewing NerveCenter Server Status

The Server Status window of NerveCenter Administrator and Client allow you to see current information pertaining to an active NerveCenter Server.

To view information about a NerveCenter Server

1. Open NerveCenter Administrator and connect to a NerveCenter Server See "Connecting Administrator to a NerveCenter Server" on page 25.
2. From the **Server** menu, select **Server Status**.

   NerveCenter displays the Server Status window for the active NerveCenter Server.



Server Machine Name	nervecenter
Server IP Address	192.168.1.191
Connection Port	32504
NerveCenter Inform Port	32505
Command Line Interface Port	32506
Time Started	06/09/2017 00:05:19
Discover Nodes From Traps	All
No DNS Lookup of Discovered Nodes	False
Process Traps From Unknown Nodes	False
Apply All Masks For Each Trap	False
Server Build Number	6200 BLD13

3. To close the window, select **Close**.

The Server tab presents information about the machine the NerveCenter server is running on, the communication ports being used by NerveCenter, and the server's node-discovery settings. Table 5 describes the information on the Server page:

Table 5: Fields on Server Tab

Label	Explanation
Server Machine Name	The name of the host running the NerveCenter Server.
Server IP Address	The IP address of the server.
Connection Port	The port used by the server to communicate with the client.
NerveCenter Inform Port	The port used by the server to receive Inform actions from other NerveCenter servers.
Command Line Interface Port	The port number on the server used for the command line interface.
Time Started	The time that the server was started (in the server's time zone).
Discover Nodes from Traps	How nodes are to be discovered. If NerveCenter is set up to discover nodes, it adds nodes to the database based on this setting: ■ **None** - Never add an unknown node (NerveCenter discovery is disabled). ■ **All** - Add all unknown nodes. ■ **IP Filter** - Add the node if its IP address matches the IP filter criteria specified in NerveCenter Administrator.
No DNS Lookup of Discovered Nodes	■ **False**, the default, indicates that NerveCenter will attempt a DNS lookup for any IP address discovered from a trap. ■ **True** indicates that NerveCenter will not attempt a DNS lookup of any node discovered by a trap. Nodes are added to NerveCenter as an IP address. **Note:** No DNS Lookup of Discovered Nodes is only valid if Discover Nodes from Traps is selected.
Process Traps from Unknown Nodes	**True** indicates that NerveCenter processes traps from all nodes, regardless of filters imposed by NerveCenter or a network management platform. If **False**, NerveCenter discards traps coming from nodes outside your filters. This feature does not change the effect your filters have on discovery. While traps from any node can be processed, nodes are added to the NerveCenter database only if they meet your filter criteria.

Label	Explanation
Apply All Masks for Each Trap	**True** indicates that NerveCenter processes every incoming SNMP trap against all defined trap masks that are currently enabled. A mask processes traps even when its associated alarm is turned off or is not in a state that can be transitioned by the mask's trigger.
Server Build Number	The NerveCenter software version running on the server.

The other tabs in Server Status window display information about the active NerveCenter Server. Table 6 describes the other tabs.

Table 6: Service Status Tabs

Tab	Description
License	Displays license information for the active NerveCenter Server.
Database	Displays information about the database this NerveCenter Server is using. Includes statistics on its current behavior models.
Node Source	Displays the node filters as well as node source, if applicable, this NerveCenter Server is using to maintain its node list.
Inform Configuration	Displays the settings involved in forwarding event information to one or more inform recipients.
Connected NerveCenters	Displays information about each NerveCenter Server that may potentially send an inform packet to the active NerveCenter Server.
Connected Clients	Displays information about each host running a NerveCenter Client connected to the active NerveCenter Server.
Connected Administrators	Displays information about each host running a NerveCenter Administrator connected to the active NerveCenter Server.

Disconnecting the Administrator from a NerveCenter Server

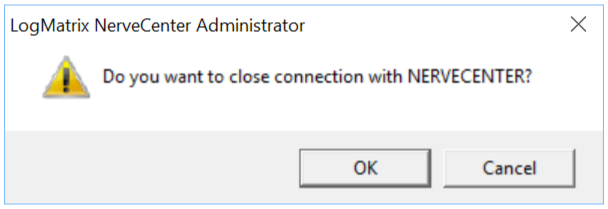
When you no longer need to configure settings in NerveCenter Administrator, you can disconnect from a NerveCenter Server. If you are connected

When you close a NerveCenter Administrator, all connections to NerveCenter servers are broken. However, you may also want to disconnect the administrator from a server without stopping a NerveCenter Administrator.

To disconnect a NerveCenter Administrator from a server

1. In NerveCenter Administrator, select the window of the server from which you want to disconnect.
2. From the **Server** menu, choose **Disconnect Server**.

   A warning message appears asking if you are sure you want to disconnect from the active server.


LogMatrix NerveCenter Administrator

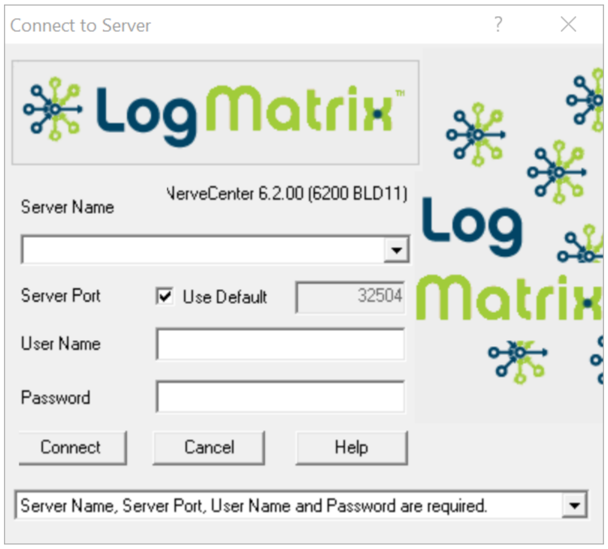
Do you want to close connection with NERVECENTER?

3. Select **OK**.

NerveCenter Administrator remains open but is no longer connected to the NerveCenter Server.

Troubleshooting: Connecting to the NerveCenter Server

The following list contains some common problems users have when starting the NerveCenter Administrator or Client and connecting to the NerveCenter Server.

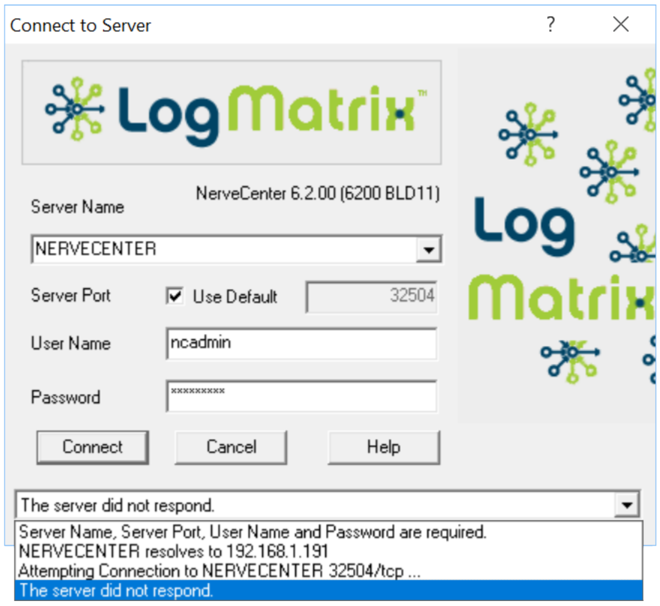


Server Name, Server Port, User Name and Password are required.

WHEN ATTEMPTING TO CONNECT WITH NCADMIN AND CLIENT, THE OUTCOME IS "THE SERVER DID NOT RESPOND."

1. Verify that the Server Name field is valid for the connection you are attempting to make:
   - If you are entering a hostname, verify that your Windows host can resolve that name to the correct IP address.
   - If you are entering an IP address, verify that it is correct.

   Click the area where the "There server did not respond." message appears (see below) and review the connection attempt steps; this might reveal clues, such as difficulty with name resolution. You can also open a command prompt and try running `nslookup.exe` or `ping.exe`.



Server Name NerveCenter 6.2.00 (6200 BLD11)
NERVECENTER
Server Port ✓ Use Default 32504
User Name ncadmin
Password *********

```
The server did not respond.
Server Name, Server Port, User Name and Password are required.
NERVECENTER resolves to 192.168.1.191
Attempting Connection to NERVECENTER 32504/tcp ...
The server did not respond.
```

2. Verify that the NerveCenter Server host is reachable. Routing or gateway issues between your Windows host and the NerveCenter Server host might be preventing the connection attempt from reaching the destination, or preventing a response from being returned back to the Windows station. Windows tools such as `ping.exe` and `tracert.exe` might help in diagnosing basic connectivity issues.
3. Verify that the firewall on the NerveCenter Server host, if present and running, is allowing communication via the port configured for the NerveCenter Administrator and Client TCP port 32504 by default). See the instructions for your firewall service if you need help evaluating its configuration.
4. Log into the NerveCenter Server host and verify whether the service is running. For example:

```
    # /etc/init.d/ncservice status      <--
    RHEL6/CentOS6/OracleLinux6, Solaris
    # systemctl status nervecenter      <-- RHEL7/CentOS7/OracleLinux7
    # ps -ef | grep ncserver
    # /opt/OSInc/bin/ncstatus
```

   If NerveCenter is not running, enable it or work with your administrators to get it running. For example:

```
    # /etc/init/ncservice start
    # systemctl start nervecenter
    # /opt/OSInc/bin/ncstart
```

5. Review the NerveCenter Server's log file to see whether it received the connection request.

   The NerveCenter Server (ncserver) writes immediately to its log file at `/var/opt/NerveCenter/log/ncserver.log` upon receiving a connection request, and would contain an entry similar to the following:

```
[2017-07-30 15:49:42] Login: Connection initiated from
[192.168.1.184]:59476/tcp
```

- If `ncserver.log` does not contain an entry of this type, recheck the routing to the NerveCenter Server host, the NerveCenter Server host's firewall, and that NerveCenter Server is running.
- If `ncserver.log` *does* contain an entry of this type, recheck the routing from the NerveCenter Server host back to the your Windows host.

WHEN ATTEMPTING TO CONNECT WITH NCADMIN AND CLIENT, THE OUTCOME IS "USER DOES NOT HAVE NERVECENTER ADMINISTRATOR OR USER PERMISSIONS."

The account in the **User Name** field is recognized by the NerveCenter Server and its host operating system; however, that account is not named as a member of either the NerveCenter Admin (ncadmins) or User (ncusers) groups. On the NerveCenter Server host, verify that the ncadmins and ncusers groups are defined and that the account is in the appropriate group.

Group information is typically found `/etc/group`, though the location may vary based on your Linux or Solaris host configuration. Also, services such as NIS/NIS+, LDAP, or ActiveDirectory may be involved.

WHEN ATTEMPTING TO CONNECT WITH NCADMIN AND CLIENT, THE OUTCOME IS "THE ACCOUNT MUST BE A MEMBER OF NERVECENTER ADMINS GROUP TO RUN ADMINISTRATOR."

This response might occur when attempting to connect with the Administrator application (not with the Client).

The account in the **User Name** field is recognized by the NerveCenter Server and its host operating system; however, that account is not named as a member of the NerveCenter Admin (ncadmins) group. On the NerveCenter Server host, verify that the ncadmins group is defined and that the account is a member.

WHEN ATTEMPTING TO CONNECT WITH NCADMIN AND CLIENT, THE OUTCOME IS "THE USER NAME AND PASSWORD ARE NOT ACCEPTED."

This message is the generic response for most other types of connection failures. It is meant to signal that the connection was not permitted yet reveals little to the source.

Diagnosing this issue requires checking the validity of the account in the **User Name** field — verify that the account exists, is enabled, and that the password is correct. For systems using password aging, ensure the most recent password assignment is within the allowed time span.

Running the `/opt/OSInc/nc/bin/nctestlogin` utility on the NerveCenter host may provide additional information. Run the command directly though a shell prompt, elevated to the super-user account. The utility will prompt for an account name and password and then proceed through the authentication steps.

The following is a sample session showing a test where the account is named "ncadmin" and the password is "ncadminssecretpassword".

```
# /opt/OSInc/nc/bin/nctestlogin
```

```
Reading /opt/OSInc/conf/nervecenter.xml
PAM Registration: default, servicename{nervecenter}, authcheck
{enabled}, accountcheck{enabled}
nctestlogin Version 6.2.00 (6200 BLD13), Linux(el6)/x86-64 (64-
bit) Copyright (C) 1989-2015 LogMatrix Inc.
Enter user name and password. Or press Enter to quit.
login: ncadmin
password: [entry will not be shown] ncadminssecretpassword
[2017-06-28 16:28:05]  Begin authentication check for
ncadmin@localhost
1. Check for User account
2. Begin PAM access. Using service name "nervecenter"
3. PAM Authentication Check
4. PAM Account Management Check
5. End PAM access
6. Checking Group Membership
- Searching group 'root'.  Members: (no match)
- Searching group 'ncadmins'.  Members: "ncadmin" (match)
- Success. Granted 'ncadmins'-level access
[2017-06-28 16:28:05]   "ncadmin" login granted with "ncadmins"
access.
[2017-06-28 16:28:05]  End authentication check for
ncadmin@localhost
Enter user name and password. Or press Enter to quit.
login:
```

If you run `nctestlogin` successfully but Administrator/Client access still fails, see "Troubleshooting: NerveCenter Connection Issues II" on page 143.

UNIX WILL NOT START THE NERVECENTER ADMINISTRATOR

Problem: The environment variables are not set.

Solution: Execute one of the following shell scripts:

- `/opt/OSInc/userfiles/ncenv.sh`
- `/opt/OSInc/userfiles/ncenv.csh`
- `/opt/OSInc/userfiles/ncenv.ksh`

See "Running the NerveCenter Server" on page 23.

I MISSPELLED A SERVER NAME WHILE TRYING TO CONNECT AND NOW THE MISSPELLED NAME APPEARS IN THE NERVECENTER ADMINISTRATOR SERVER NAME LIST

Problem: NerveCenter Administrator remembers the name of every server you attempted to connect to.

Solution: Remove the misspelled name by deleting it from the Administrator’s configuration window.

See "Deleting a Name from the Administrator Server List" on page 27.

Managing the NerveCenter Server

4

NerveCenter is based on a client/server architecture. This means the user views information and interacts with NerveCenter through client applications while most of NerveCenter's actions are carried out by the NerveCenter Server.

Much of the information NerveCenter needs to detect and correlate events are provided when behavior models are created. However, a NerveCenter administrator must configure and maintain certain foundational settings.

Note: To change the default port settings used by NerveCenter for various communications, see "Managing NerveCenter Port Settings" on page 47.

Managing the NerveCenter Trap Source

During installation, you indicated which trap source, or trap engine, you wanted to use for capturing SNMP traps. You can later change this trap source. Keep in mind that neither MSTrap nor OVTrapD support SNMPv3.

Trap Source Overview

During installation, you indicated whether NerveCenter should be configured to support SNMPv3. If you answered yes, NerveCenter is your trap sourceand it uses the port specified on the Ports tab for capturing traps. This value, SNMP Trap port, is 162 by default. If you want NerveCenter to capture traps, this port must be free for NerveCenter to capture the traps.

NerveCenter Trap Utilities

NerveCenter includes the following trap-related utilities:

- **Trapgen** — Generates a trap or inform on a specified node.
- **Traprcv** — Receives traps and shows the traps in standard output.

See "Controlling NerveCenter from the Command Line" on page 195 for more information.

Trapgen

Trapgen is a standalone utility that allows you to send an SNMP trap or inform to a particular device. From a command line or shell prompt, you can set the generic and specific trap numbers, provide the sender and recipient IP addresses, include an enterprise identifier, and specify the variable bindings.

Generating a trap is useful for testing trap masks and behavior models. When you send an SNMP trap to a NerveCenter Server, any trap mask configured to detect the specified trap values fires its trigger. If your trap mask and alarm are working properly, you see the alarm listed in your console.

Traprcv

The traprcv command displays the SNMP Trap messages received by the NerveCenter Trap service. This utility can be useful when debugging behavior models. When you run the traprcv command from a command line or shell prompt, any trap you receive is shown on the screen.

Traprcv can receive the following traps and informs:

- SNMPv1 Traps
- SNMPv2c Traps
- SNMPv2c Informs
- SNMPv3 Traps
- SNMPv3 Informs

Changing the SNMP Trap Source

When NerveCenter is your trap source, NerveCenter uses the port specified on the Ports tab for capturing traps. This value, SNMP Trap port, is 162 by default. If you want NerveCenter to capture traps, this port must be free for NerveCenter to capture the traps. This may require that you shut down MS Trap service and/or OVTrapD.

Follow the steps below to change the trap source used to process traps. The sequence is important because the current trap source has to release the port before the new trap source can use that port.

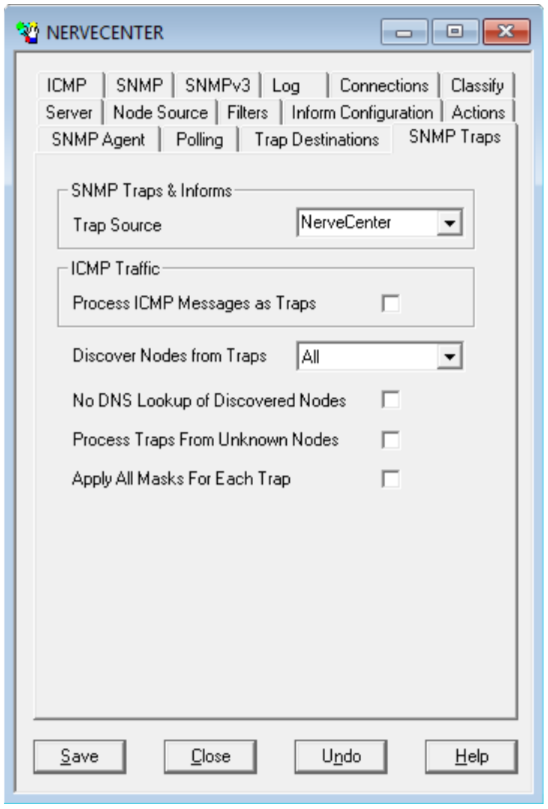
To change the NerveCenter trap source

1. Open NerveCenter Administrator and connect to the appropriate Server.

   See "Connecting Administrator to a NerveCenter Server" on page 25.

2. Select the **SNMP Agent** tab.

   The SNMP Agent page is displayed.



SNMP Traps & Informs

Trap Source: NerveCenter

ICMP Traffic

Process ICMP Messages as Traps: ☐

Discover Nodes from Traps: All

No DNS Lookup of Discovered Nodes: ☐

Process Traps From Unknown Nodes: ☐

Apply All Masks For Each Trap: ☐

3. Select the trap engine you want to use from the Trap Source drop-down.
4. Select **Save**.
5. Shut down the NerveCenter Server.
6. If applicable, stop the old trap service and then start the new one.

   Refer to Table 7 for details.

7. Restart the NerveCenter Server.

Table 7 summarizes what to do when changing the trap source.

Table 7: Changing the Trap Source

Switching From	Switching To	Action To Take...
OVTrapD or MS Trap	NerveCenter	Shut down OVTrapD and MS Trap service. If both are on your system, shut down both. Disable the MS Trap Service. If SNMP Trap port is other than 162 in NerveCenter, you don't need to shut down OVTrapd or MSTrap service.
NerveCenter	OVTrapD	Start OVTrapD.
NerveCenter	MS Trap	Start MS Trap service. Disable the NerveCenter SNMP Trap service.
OVTrapD	MS Trap	Stop OVTrapD. Start MS Trap service.
MS Trap	OVTrapD	Stop MS Trap service. Start OVTrapD.

TO CHANGE FROM THE NERVECENTER TRAP SERVICE TO MSTRAP

1. Select MSTRAP as your trap source.
   a. Open NerveCenter Administrator and connect to the appropriate Server.
   b. Select the **SNMP Agent** tab.
   c. Select **MSTRAP** from the Trap Source drop-down listbox.
   d. Select **Save**.
   A dialog box displays stating Data saved successfully.
2. Disconnect from the Server and shut down NerveCenter Administrator.
3. Stop the following services in the order given:
   a. NerveCenter service
   b. NerveCenter SNMP service
4. Set Startup type of the SNMP Trap Service to Automatic.
   Follow the instructions of "To disable the MS Trap service" on page 43 but select **Automatic** instead of **Disabled**.
5. Start the following service in the order given:
   a. SNMP Trap Service service
   b. NerveCenter SNMP service
   c. NerveCenter service

If you have snmputil installed, you can test that MS Trap service is receiving traps.

To test whether the MS Trap service is receiving traps

1. Run snmputil trap at a command prompt.

   The utility returns `snmputil: listening for traps...`

2. Enter the following command:

   ```
   trapgen -v1 your_ip-address 1.3.6.1.4.78 your_ip-address 6 101 132
   ```

   snmputil should return the following text:

   ```
   Incoming Trap:
   generic       = 6
   specific      = 101
   enterprise    = .iso.org.dod.internet.private.enterprises.78
   agent         = your ip-address
   source IP     = your ip-address
   community     = public
   ```

   If the AllTraps_LogToFile alarm is enabled, the alltraps.log should contain the following lines.

   ```
   Time=10/22/2002 13:11:02 Tue; LogId=1; DestStateSev=Normal;
   NodePropertyGroup=NC
   DefaultGroup; NodeName=mycomputer;
   AlarmName=AllTraps_LogToFile;
   OrigState=Ground;
   TriggerName=allTraps; DestState=Logging; TrapPduTime=132;
   TrapPduGenericNumber=6
   ; TrapPduEnterprise=1.3.6.1.4.1.78;
   TrapPduSpecificNumber=101;
   TriggerInstance=;
   TriggerBaseObject=
   ```

To change from the MSTRAP Service to NerveCenter Trap Service

1. Select NerveCenter as your trap source.

   a. Open NerveCenter Administrator and connect to the appropriate Server.

      For instructions, see "Connecting to a NerveCenter Server" on page 23.

   b. Select the **SNMP Agent** tab.

   c. Select NerveCenter from the Trap Source drop-down.

   d. Select **Save**.

      A dialog box displays stating **Data saved successfully**.

2. Disconnect from the Server and shut down NerveCenter Administrator.

   For instructions, see "Disconnecting the Administrator from a NerveCenter Server" on page 30.

3. Stop the following services in the order given:
   a. NerveCenter service
   b. NerveCenter SNMP service
   c. SNMP Trap Service service
4. Set Startup type of the SNMP Trap Service to Disabled.

   Follow the instructions of "To disable the MS Trap service" on the facing page.

5. Start the following services in the order given:
   a. NerveCenter SNMP service
   b. NerveCenter service

You can test that the NerveCenter Trap service is running.

TO TEST WHETHER THE NERVECENTER TRAP SERVICE IS RECEIVING TRAPS

1. Run traprcv in a command window.

   The utility returns Waiting for traps.

2. Enter the following command:

   ```
   trapgen -v1 your_ip-address 1.3.6.1.4.78 your_ip-address 6 101 132
   ```

   traprcv should return the following text:

   ```
   Received SNMPv1 Trap:
   Community: public
   From: your ip-address:2057
   Enterprise: enterprises.7
   Agent-addr: your ip-address
   Enterprise Specific trap: 101
   Time Ticks: 132
   ```

If the AllTraps_LogToFile alarm is enabled, the alltraps.log should contain the following lines.

```
Time=10/22/2002 13:38:02 Tue; LogId=1; DestStateSev=Normal;
NodePropertyGroup=NC
DefaultGroup; NodeName=mycomputer;
AlarmName=AllTraps_LogToFile;
OrigState=Ground;
TriggerName=allTraps; DestState=Logging; TrapPduTime=132;
TrapPduGenericNumber=6
; TrapPduEnterprise=1.3.6.1.4.1.78;
TrapPduSpecificNumber=101;
TriggerInstance=;
TriggerBaseObject=
```

To disable the MS Trap service

1. Open the Services control panel.
2. Right-click on the **SNMP Trap Service** and select **Properties** from the menu.
3. From the **Startup Type** list, select **Disabled**.
4. Select **OK**.

Reconfiguring a NerveCenter Server from the Command Line

An alternative to using the NerveCenter Administrator to configure a NerveCenter Server is to use the utility ImportUtil. ImportUtil is a utility that does the following from the command line:

- Imports behavior models and nodes to a NerveCenter Server.
- Imports configuration settings to a NerveCenter Server.

Using this utility allows you to reconfigure a setting on more than one NerveCenter Server at a time by changing one file and importing it to all the relevant servers. Figure 10 shows ImportUtil along with the other NerveCenter components.

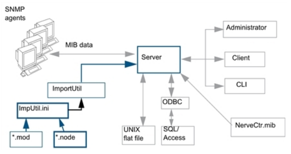


Figure 10: NerveCenter ImportUtil Utility

ImportUtil imports the data referenced in the editable file ImpUtil.ini, which is located in the *installation/*Sms directory (Windows) or the *installation/*userfiles directory (UNIX).

You can edit NerveCenter configuration settings in ImpUtil.ini as well as specify the behavior models and nodes you want to import. You must have the proper *.mod or *.node file for the behavior models or nodes you want to import. When you have made the changes you want to ImpUtil.ini, run ImportUtil to import the specified information or files.

See "Controlling NerveCenter from the Command Line" on page 195 for more information.

To reconfigure a NerveCenter Server from the command line

1. Find the file imputil.ini.
   - In a typical NerveCenter installation on Windows, this file can be found in the folder **Sms**, in the NerveCenter folder.
   - In a typical NerveCenter installation on UNIX, this file can be found in the directory *install_ path*/userfiles, where *install_path* is typically /opt/OSInc.

   The file imputil.ini is made of a number of sections that include a section header and keys.

2. Before making any changes, create a backup copy of the file imputil.ini.

   If you do not edit imputil.ini correctly, you can misconfigure or corrupt your NerveCenter Server. You may want to test your changes to imputil.ini in a test environment before running importutil in an production environment.

   You will not be able to restore the original imputil.ini after making changes to the file, unless you first make a backup copy.

3. Delete all but the relevant sections to be changed.

   See "importutil" on page 209 for all imputil.ini section descriptions. All sections in the file are optional. If you remove a section, including the section header and keys, ImportUtil does not change or delete any values in the NerveCenter configuration settings for that key.

   For example, if you are changing a value found only in the section **[CONFIG_SERVER]**, you delete all sections except the section header and the values in the **[CONFIG_SERVER]** section. ImportUtil will only change the values pertaining to that section.

4. Within the remaining sections, delete everything but the section headers and the relevant keys.

   Any new values left in imputil.ini will overwrite the old values. To avoid having placeholders overwrite legitimate values, delete any unnecessary keys before running ImportUtil.

   For example, if within the **[CONFIG_SERVER]** section you only want to change the value of the key InformNCListenPort, delete all but the following:

   ```
   [CONFIG_SERVER]

   InformNCListenPort = port
   ```

   If you are configuring the **[CONFIG_PLATFORM_NETNODENOTIFY]** or **[CONFIG_PLATFORM_MAPSUBNETS]** sections, you must include all values, including old values. ImportUtil deletes values from the NerveCenter configuration settings that are not included in these sections. Read the comments before each section in this file for more information.

5. Change the values by replacing the placeholders after the equal sign (=) with valid values.

   Unless otherwise noted, you may not leave the value after a key blank. For example, if you want to change the value of the key InformNCListenPort to 6024, change the file to read:

   ```
   [CONFIG_SERVER]

   InformNCListenPort = 6024
   ```

   See "importutil" on page 209 for details.

6. Save the changed file.
7. While the NerveCenter Server is running, run the utility ImportUtil from a UNIX shell or a Windows command prompt by typing:

   ```
   importutil imputil.ini
   ```

   You must either be in the same directory as the imputil.ini file or include the full pathname of the imputil.ini file. NerveCenter notifies you upon successful completion of the reconfiguration.

Troubleshooting: Managing the NerveCenter Server

The following list contains some common problems users may have when managing the NerveCenter Server.

I NEED TO MAKE THE SAME CHANGES TO SEVERAL NERVECENTER SERVERS

Problem: It would be time consuming to configure each NerveCenter Server manually from the NerveCenter Administrator.

Solution: Create one imputil.ini file and import it to all the NerveCenter Servers.

See "Reconfiguring a NerveCenter Server from the Command Line" on page 43.

IMPORTING IMPUTIL.INI CAUSED UNWANTED CHANGES TO A NERVECENTER SERVER

Problem: ImportUtil imports all values found in imputil.ini, including the defaults.

Solution: When using ImportUtil, delete all but the relevant keys.

See "Reconfiguring a NerveCenter Server from the Command Line" on page 43.

I CANNOT LOG IN TO A UNIX NERVECENTER SERVER.

Problem: The login, as defined on the UNIX system running NerveCenter server, might not exist or else might be blocked or not associated with the correct groups.

Solution: You need to work with the administrators for the UNIX system. The login needs to be a defined, valid account and must be in the 'ncadmins' or 'ncusers' groups. As well, the issue might lie in the UNIX system's PAM configuration.

See "Managing Security on UNIX" on page 141 for more on UNIX logins, groups and PAM.

Managing NerveCenter Port Settings 5

NerveCenter is based on a client/server architecture. This means that while the NerveCenter Server does the work, you make changes to the server using a client application, such as NerveCenter Client and NerveCenter Administrator. For these applications and a NerveCenter Server to communicate, they must use the same port connection. The Port tab enables you to specify various ports used by the NerveCenter Server.

Client and Server Communication Ports

NerveCenter is based on a client/server architecture. This means that while the NerveCenter Server does the work, you make changes to the server using a client application, such as NerveCenter Client and NerveCenter Administrator. For these applications and a NerveCenter Server to communicate, they must use the same port connection.

By default, NerveCenter Server communicate on the special port 32504. If for some reason—such as this port is being used by another application—it becomes necessary to change NerveCenter's communication port, an administrator may do so. However, the administrator must follow *all* of these procedures:

- "Configuring the NerveCenter Server Connection Port" on the next page
- "Changing the NerveCenter Administrator Server Port" on page 49
- "Changing the NerveCenter Client Server Port" on page 50

Caution: Any application attempting to communicate with a NerveCenter Server must have a matching communication port number.

Configuring the NerveCenter Server Connection Port

The NerveCenter Server communicates with other applications, such as NerveCenter Client and NerveCenter Administrator, on a special connection port (32504 by default).

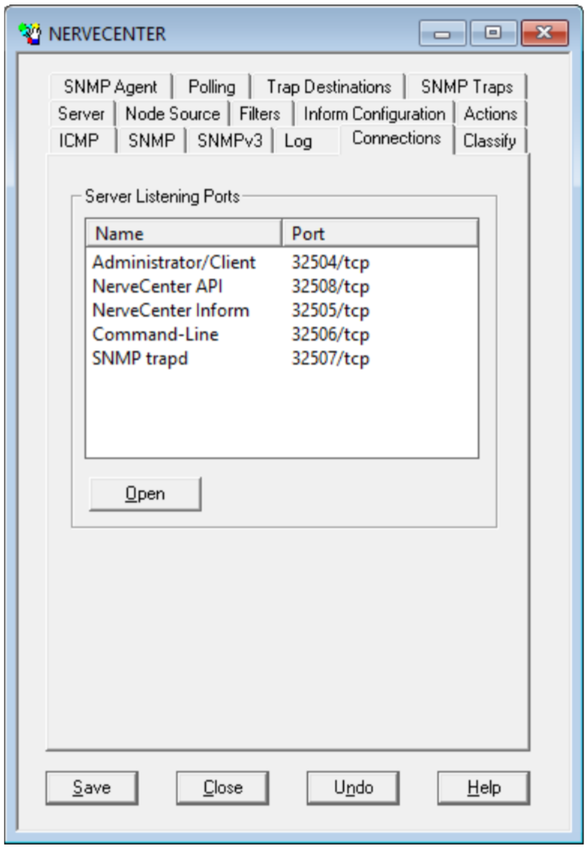
TO CHANGE THE NERVECENTER SERVER CONNECTION PORT

1. Open NerveCenter Administrator and connect to the appropriate Server.

   See "Connecting Administrator to a NerveCenter Server" on page 25.

2. Select the **Connections** tab.

   The Connections tab is displayed.



Server Listening Ports

Name	Port
Administrator/Client	32504/tcp
NerveCenter API	32508/tcp
NerveCenter Inform	32505/tcp
Command-Line	32506/tcp
SNMP trapd	32507/tcp

3. In the **Client Connection** field, enter the port number you want the NerveCenter Server to use for client/server communication.

Caution: If you change the port number here, you must make the same changes to the NerveCenter Client and NerveCenter Administrator.

4. Select **Save**.

5. Stop and start the NerveCenter Server.

   Changes to the communication port will not be complete until the NerveCenter Server is restarted. See "Running the NerveCenter Server" on page 23.

NerveCenter uses this port when connecting with a NerveCenter Administrator or Client.

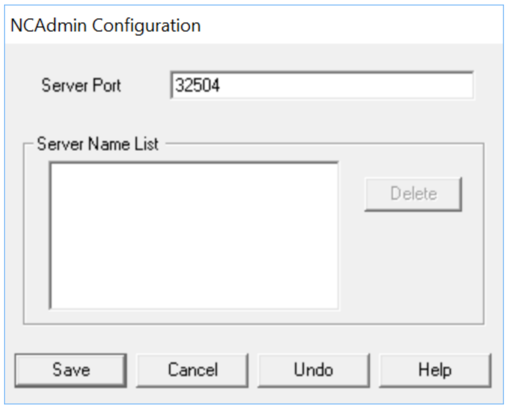
Changing the NerveCenter Administrator Server Port

If you change a NerveCenter Server's communication port number, be sure all applications, such as NerveCenter Administrator, have a matching port number.

TO CHANGE THE NERVECENTER ADMINISTRATOR'S SERVER PORT

1. Open NerveCenter Administrator. See "Starting NerveCenter Administrator" on page 25.
2. From the **Admin** menu, choose **Configuration**.

   The NCAdmin Configuration window appears.



NCAdmin Configuration

Server Port: 32504

   The NCAdmin Configuration window displays the Server port the NerveCenter Administrator application uses to communicate with a NerveCenter Server. It also includes a list of all the servers this application has connected to in the past.

3. In the Server Port field, type the connection port number of the NerveCenter Server you want the NerveCenter Administrator application to connect to.

Caution: Remember the port number. If you change it here, you must make corresponding changes to the NerveCenter Server.

4. Select **Save**.

The NerveCenter Administrator application will now communicate with the NerveCenter Server at the new port you specified in the Server Port field.

Changing the NerveCenter Client Server Port

If you change a NerveCenter Server's communication port number, be sure all applications, such as NerveCenter Client, have a matching port number.

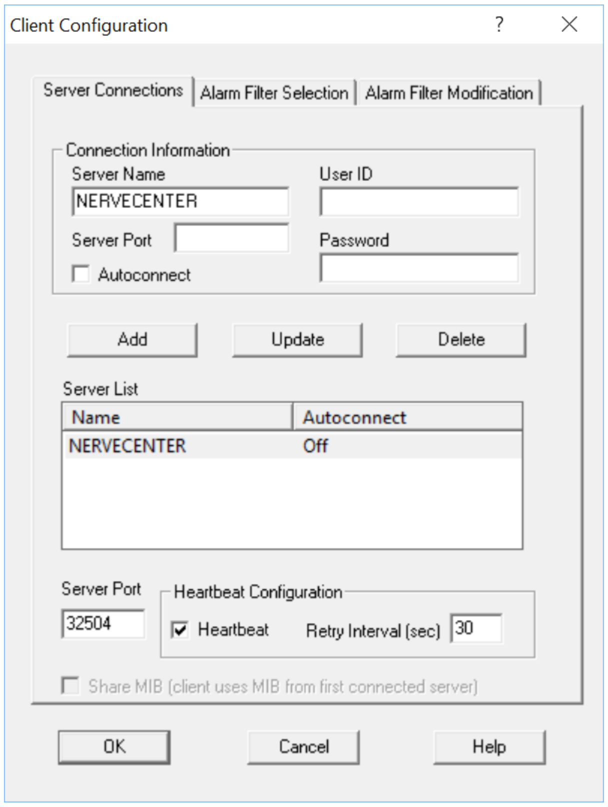
TO CHANGE THE NERVECENTER CLIENT'S SERVER PORT

1. Open NerveCenter Client.

   See Starting the NerveCenter Client for more details.

2. In the **Client** menu, choose **Configuration**.

   The Client Configuration window appears.



Server Name: NERVECENTER
User ID:
Server Port:
Password:
☐ Autoconnect

Server List

Name	Autoconnect
NERVECENTER	Off

Server Port: 32504
☑ Heartbeat Retry Interval (sec): 30
☐ Share MIB (client uses MIB from first connected server)

3. Select the **Server Connections** tab.
4. From the server list, select the appropriate server.

   The server's current port number appears in the Server Port field.

5. In the Server Port field, type the connection port number of the NerveCenter Server you want the NerveCenter Client application to connect to.

Caution: Remember the port number. If you change it here, you must make corresponding changes to the NerveCenter Server.

6. Select **Save**.

The NerveCenter Client application will communicate with the NerveCenter Server at the new port you specified in the Server Port field.

Configuring NerveCenter to Receive Inform Actions

Once a NerveCenter behavior model detects a problem, it can notify a network management platform of the problem using the Inform action. This action sends an inform packet to its recipients, indicating the nature of the problem.

Note: Although the message that the Inform action sends to its recipients contains the same information as a trap, the message is not sent via UDP. An inform message is sent via TCP to make sure the delivery mechanism is reliable.

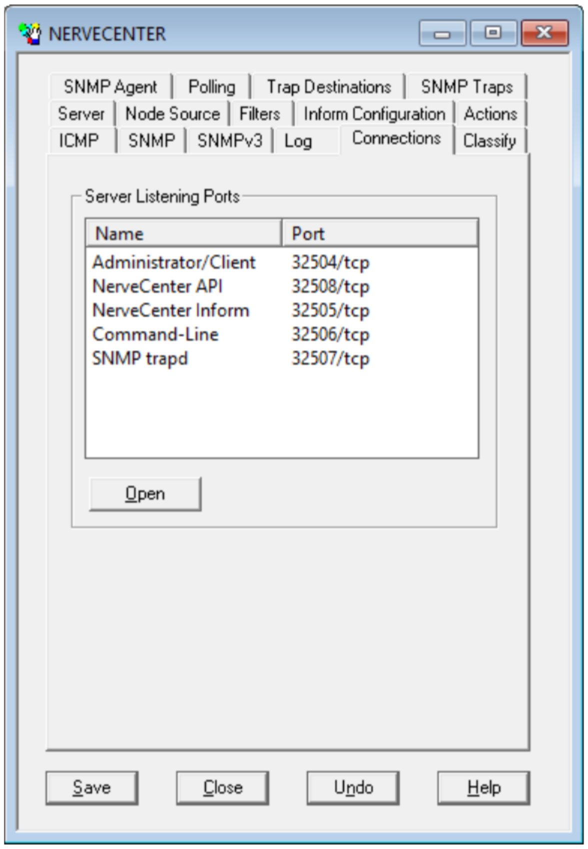
In addition to sending an Inform to network management platforms, a NerveCenter Server can send an Inform to another NerveCenter Server or itself.

Before a NerveCenter Server can receive a trap it must be configured to have a listening port number. By default, this port number is 32505.

To specify the port NerveCenter Server uses to receive an Inform packet

1. Open NerveCenter Administrator and connect to an appropriate NerveCenter Server. See "Connecting Administrator to a NerveCenter Server" on page 25.
2. Select the **Connections** tab.

   The Connections tab is displayed.



Name	Port
Administrator/Client	32504/tcp
NerveCenter API	32508/tcp
NerveCenter Inform	32505/tcp
Command-Line	32506/tcp
SNMP trapd	32507/tcp

3. In the NerveCenter **Inform** field, type the number of the port through which this NerveCenter Server should receive inform packets sent by a NerveCenter Server.
4. Select **Save**.
5. Stop and start the NerveCenter Server.

   Changes to the Inform port will not be complete until the NerveCenter Server is restarted. See "Running the NerveCenter Server" on page 23.

The current NerveCenter Server will receive Inform packets from other NerveCenter Servers at the port specified.

TO VIEW NERVECENTER SERVERS CONFIGURED TO SEND INFORMS TO THE CURRENT SERVER

1. From the **Server** menu in the Administrator or Client, choose **Server Status**.
2. Select the **Connected** NerveCenter**s** tab.

Any NerveCenter Server configured to send an inform packet to the current NerveCenter Server at the port specified in step 3 appears in the Inform NC Name list.

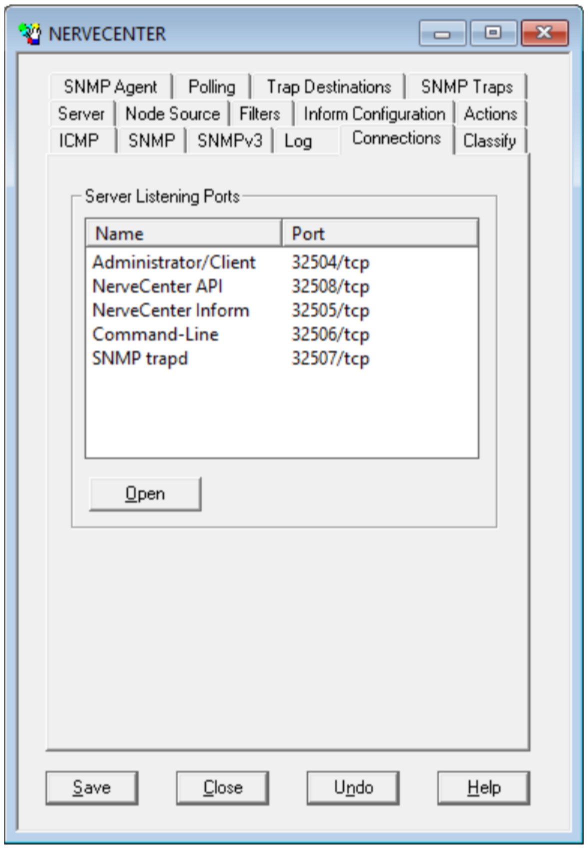
Changing the Command Line Interface Port

You can use NerveCenter's command line interface to delete, list, or set (enable or disable) alarms, trap masks, nodes, and polls from a Windows Command Prompt or a UNIX shell. You can also connect to, display the status of, and disconnect from NerveCenter servers using the CLI. You can issue commands manually or from a script.

To specify the NerveCenter command line interface port

1. Open NerveCenter Administrator and connect to an appropriate NerveCenter Server. See "Connecting Administrator to a NerveCenter Server" on page 25.
2. Select the **Connections** tab.

   The Connections tab is displayed.



Name	Port
Administrator/Client	32504/tcp
NerveCenter API	32508/tcp
NerveCenter Inform	32505/tcp
Command-Line	32506/tcp
SNMP trapd	32507/tcp

3. In the Command Line Interface field, enter the number of the port through which this NerveCenter Server should receive commands from its Command Line Interface.
4. Select **Save**.

The current NerveCenter Server will receive commands from the port specified.

Specifying SNMP Ports for NerveCenter

NerveCenter has two primary sources of information about network conditions:

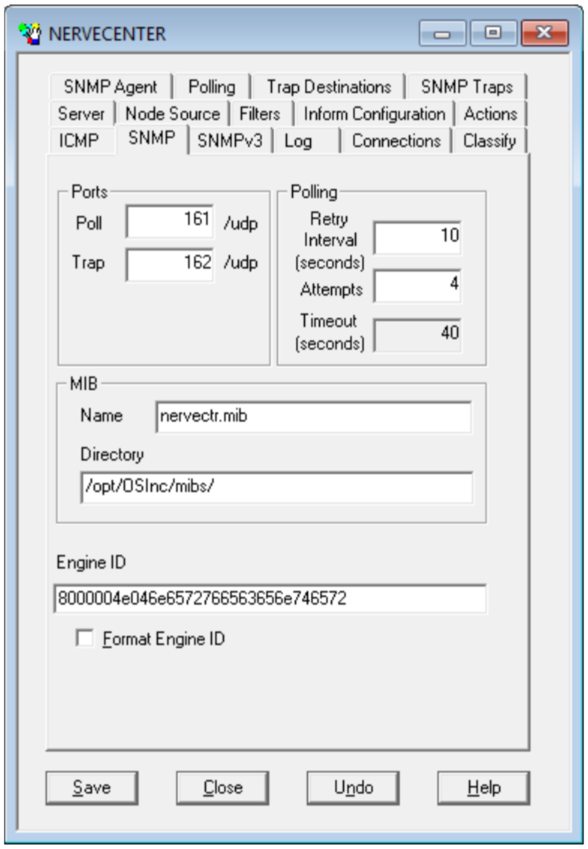
- NerveCenter listens passively for SNMP traps sent by a managed device.
- NerveCenter actively polls the SNMP agents on a managed device.

You can change the ports that NerveCenter uses for receiving traps and sending polls.

TO SPECIFY NERVECENTER SNMP PORT SETTINGS

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server. See "Connecting Administrator to a NerveCenter Server" on page 25.
2. Select the **SNMP** tab.

   The SNMP tab is displayed.



Ports
Poll 161 /udp
Trap 162 /udp

Polling
Retry Interval (seconds) 10
Attempts 4
Timeout (seconds) 40

MIB
Name nervectr.mib
Directory /opt/OSInc/mibs/

Engine ID
8000004e046e6572766563656e746572

3. In the SNMP Poll field, enter the number of the port you want NerveCenter to use to communicate with SNMP agents. This port is used to get or set SNMP information.

   The value entered here specifies the port on the node to which NerveCenter sends SNMP polls. You can change the port for any particular node in the node's definition window in NerveCenter Client.

   LogMatrix recommends that you keep the default port number 161.

4. In the SNMP Trap field, enter the number of the port you want NerveCenter to use for receiving SNMP traps. This setting has no effect if NerveCenter is co-resident with HP OpenView Network Node Manager. OpenView has the trap port and an internal NerveCenter process enables NerveCenter to receive a copy of the trap. The default port number is 162.

Note: You must shut down and restart the NerveCenter Server before the SNMP Trap port change takes effect. If you want to capture SNMPv3 traps, the SNMP Trap port must be free for NerveCenter to capture the traps. This requires that you shut down MS Trap service or OVTrapD. If both are on your system, shut down both.

5. Select **Save**.

NerveCenter handles SNMP data according to these settings.

Troubleshooting: Managing NerveCenter Port Connections

The following list contains some common problems users have when managing the NerveCenter Server port connections.

THE NERVECENTER SERVER'S COMMUNICATION PORT DOES NOT CHANGE WHEN I SELECTED SAVE IN NERVECENTER ADMINISTRATOR

Problem: NerveCenter must register the changes.

Solution: Stop and start the NerveCenter Server.

See "Running the NerveCenter Server" on page 23.

A NERVECENTER SERVER DOES NOT RECEIVE INFORMS SENT BY ANOTHER NERVECENTER SERVER

Problem: The inform packet is being sent to an incorrect port.

Solution: Make sure the receiving NerveCenter Server's inform port is the same as the port in the sending NerveCenter's inform configuration.

See "Configuring NerveCenter to Receive Inform Actions" on page 51.

THE NERVECENTER SERVER'S INFORM PORT DID NOT CHANGE WHEN I SELECTED SAVE IN NERVECENTER ADMINISTRATOR

Problem: NerveCenter must register the changes.

Solution: Stop and start the NerveCenter Server.

See "Running the NerveCenter Server" on page 23.

Designing and Managing Behavior Models

Managing Node Data

6

To detect and correlate network events, NerveCenter must have basic information about each of the managed nodes it is monitoring. It stores this data in a node list in NerveCenter's database. As an administrator you will need to determine which nodes belong in NerveCenter's node list. You will also be responsible for keeping the node list complete and current. This chapter describes the NerveCenter tools you can use to carry out these tasks.

How NerveCenter Manages Nodes

The following sections describe the tasks that NerveCenter performs to automate event handling:

- "Defining a Set of Nodes" below
- "Detecting Conditions" on the next page
- "Correlating Conditions" on the next page
- "Responding to Conditions" on page 65

Defining a Set of Nodes

NerveCenter can get the list of devices to monitor through a wide set of options. This includes, for example, learning a set of nodes from a network management platform, by discovering them on the network, or by importing this information from another NerveCenter Server.

NerveCenter represents each device that you want to manage, or at least have NerveCenter aware of, as a *node*. Each node represents a device and the set of nodes known to a NerveCenter Server is called the *Node List*. Managing NerveCenter's node data, thus, centers on managing what nodes are held in the Node List and the attributes defined for each node.

There are four primary attributes of each node: its name, its set of IP Addresses, whether it is to be actively managed, and its Property Group assignment.

- The *name* for any node must be unique across the set of nodes held in the Node List; no two nodes may have the same name. The name for a device is any name you choose. The name does not have to be the node's hostname or DNS name or WINS name. The naming of each node is entirely up to you to set, monitor and change.
- The set of *IP Addresses* for a node are similarly entirely under your control. A node must have at least one IP Address and there is no limit on how many IP Addresses can be assigned to a node. The set of IP Addresses defined for a node can be any mix of IPv4 and IPv6 addresses. Further, there is no restriction on having multiple nodes contain the same IP Address.
- The *'Managed' attribute* of each node is a simple flag. Nodes that are 'Managed' are actively polled, as applicable, and are monitored through the user of NerveCenter's Alarm Modeling framework. Nodes that are not 'Managed' may be held in the Node List but are not subject to active monitoring.

- The *Property Group* setting is the primary determiner of how NerveCenter will monitor a node. NerveCenter determines which alarm models to apply against a node by matching the property on which any given alarm model is associated to the properties listed within a Property Group. Only if there is a match does NerveCenter monitor the node using the alarm model. For example, if alarm model 'IfStatus' is configured to be associated with property 'ifEntry', then NerveCenter will use this alarm model to monitor a node if (and only if) its designated Property Group setting is one that includes 'ifEntry'. Property Group assignment, then, is the key driver signaling NerveCenter to monitor a given node with the appropriate set of alarm models.

Once NerveCenter assigns a set of properties to a node, NerveCenter automatically applies to that node all of the models that refer to those properties. If NerveCenter detects that a node has been deleted or that its properties have changed, the product immediately retires or updates the set of models that are actively managing that node. This dynamic process enables NerveCenter to adapt at once to changes in network configuration reported by the management platform or by NerveCenter's own discovery mechanism.

It is also possible to assign properties to nodes manually to further refine the set of models that NerveCenter uses to manage a node. For example, you may want to distinguish a backbone router from a campus router to regulate how much and how often status information is collected.

Detecting Conditions

NerveCenter can collect network and system data from a variety of sources. However, NerveCenter most frequently obtains data from Simple Network Management Protocol (SNMP) agents running on managed nodes. This means that NerveCenter detects most conditions by:

- Receiving and interpreting an SNMP trap
- Polling an SNMP agent for data and analyzing that data

One of the criticisms of SNMP-based enterprise management platforms over the years has been that, because SNMP trap delivery is unreliable, the platform must poll agents and this polling generates too much network traffic. NerveCenter helps alleviate this problem by enabling you to determine the interval at which a poll is sent and to turn a poll off. Even more important is NerveCenter's *smart polling* feature. NerveCenter sends a poll to a node only if the poll:

- Is part of a behavior model designed to manage that node
- Can cause a change in the alarm's state

Also, because of NerveCenter's client/server architecture, NerveCenter servers can be distributed so that all polling is done on LANs, and not across a WAN. Furthermore, use of SNMPv2c and v3 features allow SNMP to be utilized both reliably and securely.

Correlating Conditions

Event correlation involves taking a number of detected network conditions and determining:

- How these conditions, or some subset of them, are related
- The underlying cause, or the problem to which these conditions have led

For instance, NerveCenter may look at a large number of events and identify a subset of events that relate to SNMP authentication failures on a managed node. NerveCenter may then determine that the authentication failures were far enough apart that no problem exists, or it may find that several failures occurred within a short period of time, indicating a possible security problem. In the latter case, NerveCenter might notify administrators of the potential problem. In this way, administrators receive one notice about a potential security problem rather than having to browse through a long list of detected conditions and identify the problem themselves.

Detected conditions can be correlated in many ways. In fact, once you start working with NerveCenter, you will help determine how these conditions are correlated yourself. However, there are some typical ways in which NerveCenter finds relationships between conditions.

Several of these methods are discussed in the following sections.

Detecting the Persistence of a Condition

Probably the simplest method of correlating detected conditions is to search for the persistence of a problem. For example, a network administrator might want to know if an SNMP agent sends a link-down trap and that trap is not followed within three minutes by a link-up trap. NerveCenter can track such a link-down condition using a state diagram similar to the one shown in Figure 11.

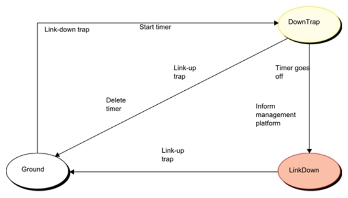


Figure 11: State Diagram for Detecting a Link-Down Condition

If NerveCenter has this state diagram in memory and is tracking a particular interface for a link-down condition.

- The first time NerveCenter sees a link-down trap concerning that interface, the current state becomes DownTrap, and NerveCenter starts a three-minute timer.
- If NerveCenter receives a link-up trap within three minutes of the link-down trap, the current state reverts to Ground (normal) because NerveCenter is looking for a *persistent* link-down condition. In addition, NerveCenter stops the timer. However, if three minutes expire before a link-up trap arrives, the current state becomes LinkDown, and NerveCenter informs a network management platform that the link is down.
- The current state remains LinkDown until a link-up trap does arrive. At that point, the current state reverts to Ground, and the process begins again.

Finding a Set of Conditions

Another common type of event correlation is identifying a set of conditions. For example, when monitoring the interfaces on a router, you might use the state diagram in Figure 12 to be notified when a low-speed or high-speed interface goes down.

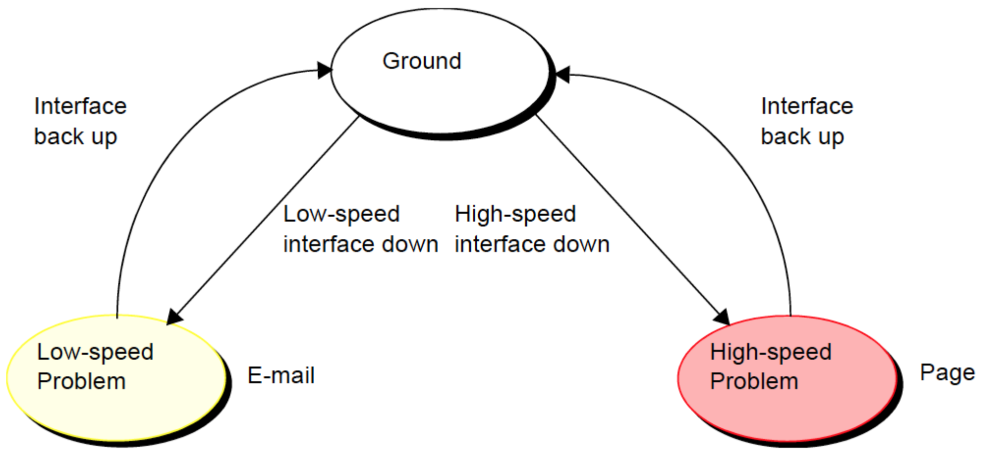


Figure 12: State Diagram for Detecting a Router Interface Problem

What causes state transitions in this situation? NerveCenter can poll the SNMP agent on the router for the values of the following interface attributes: ifOperStatus, ifAdminStatus, ifSpeed, ifInOctets, and ifOutOctets.

If the poll successfully returns values for these attributes, NerveCenter can then evaluate the expression shown below in pseudocode:

```
if ifOperStatus is down && ifAdminStatus is up &&
(ifInOctets > 0 || ifOutOctets > 0)
                if ifSpeed < 56K
                        move to lowSpeedProblem state
                else
                        move to highSpeedProblem state
                else
                move to ground state
```

This code is looking for two sets of conditions. The first set is:

- The operational state of the interface is down.
- The administrative status of the interface is up.
- Traffic has been passed on this interface. (If no traffic has been passed, the interface is just coming up.)
- The interface's current bandwidth is less than 56K.

If these conditions are met, a problem exists on an interface that is likely used for dial-up connections. The second set of conditions is the same except for the last condition, which checks whether the interface's current bandwidth is greater than or equal to 56K. If this second set of conditions is met, a problem exists on a higher speed interface. If neither of these sets of conditions is met, the current state should return to, or remain at, Ground.

NerveCenter may detect many conditions concerning an interface before it finds the set of conditions it is looking for. The administrator need not see information about each of these conditions. He or she will be emailed or paged if the interface goes down.

Looking for a Sequence of Conditions

NerveCenter also enables correlation by looking for sequences of conditions. This type of correlation is possible because, each state in a state diagram can look for a different set of conditions. For instance, let’s look at a state diagram that NerveCenter uses to track the status of a node and its SNMP agent. The diagram (Figure 13) includes states for the following conditions:

- The node and its SNMP agent are up.
- The node is up, but its agent is down.
- The node is unreachable.
- The node is down.

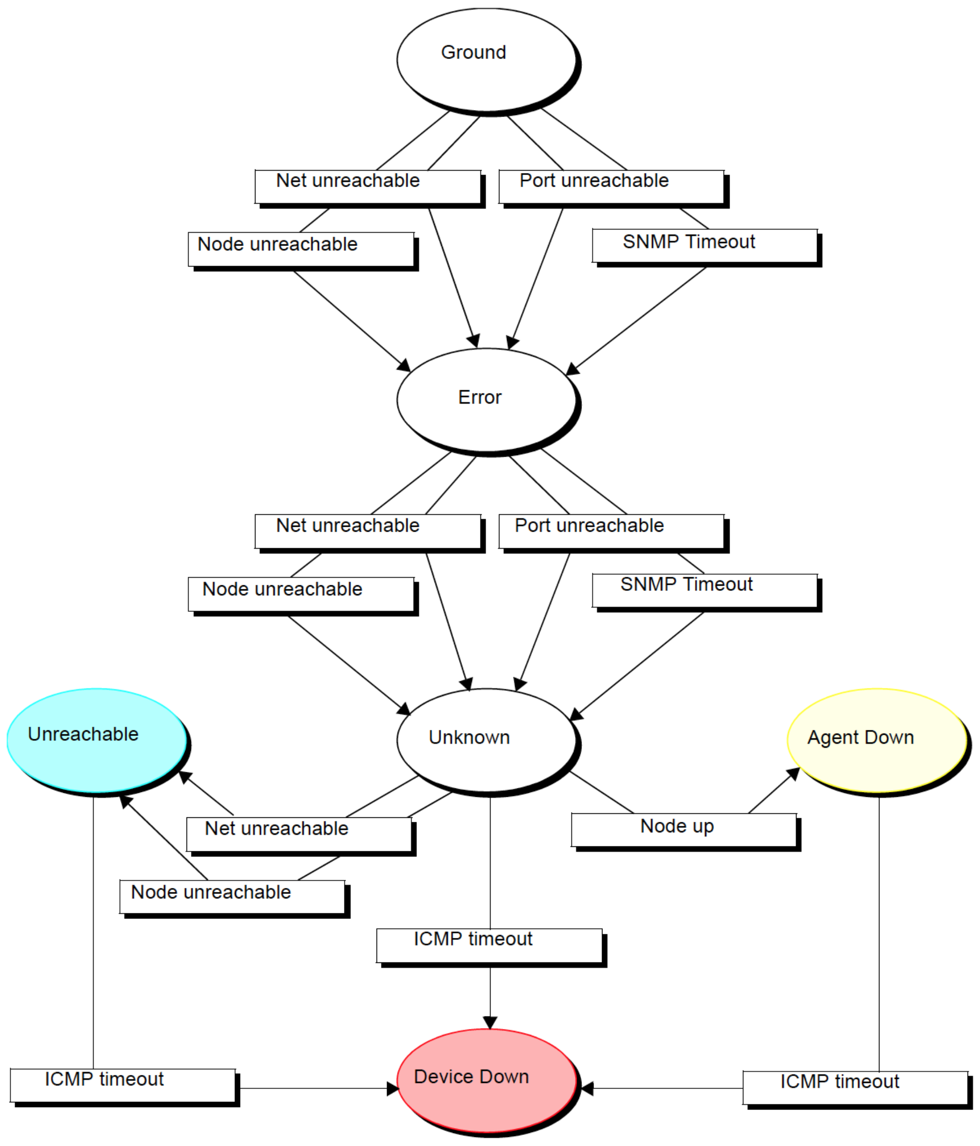


Figure 13: State Diagram for Determining Node Status

Note: A more realistic state diagram for tracking the status of a node would include transitions from the terminal problem states back to Ground.

When checking the status of a node and its SNMP agent, NerveCenter begins by polling the node to see if the node's SNMP agent will return the value of the MIB attribute sysObjectID. If the agent returns this value, the current state remains Ground. However, NerveCenter makes Error the current state if:

- The node, or the network the node is on, is unreachable
- The node is reachable, but the SNMP agent doesn't respond

Similarly, NerveCenter changes the current state to Unknown if it detects for a second time that the node is unreachable or the node's SNMP agent isn't responding.

Once the current state becomes Unknown, though, NerveCenter begins looking for a different set of conditions. NerveCenter checks to see whether the node will respond to an ICMP ping. If it will, NerveCenter knows that the node is up, but its SNMP agent is down. If it receives another network- or node-unreachable message, NerveCenter knows that the node is unreachable. And if the ping times out, NerveCenter knows that the node is down.

This ability of different states to monitor different conditions gives you the ability to correlate *sequences* of conditions. That is, a sequence of two SNMP timeouts followed by a Node up indicates that the node is up but its agent is down. And a sequence of two Node unreachables followed by an ICMP timeout indicates that the node is down.

Responding to Conditions

NerveCenter not only enables you to detect network and system problems, but is able to respond automatically to the conditions it detects. To set up these automated responses, you associate *actions* with state transitions.

The possible actions you can define are discussed in the following sections:

- "Notification" below
- "Logging" on the next page
- "Causing State Transitions" on the next page
- "Corrective Actions" on the next page
- "Action Router" on page 67

Notification

If a particular network or system condition requires the attention of an administrator, the best action to take in response to that condition is to notify the appropriate person. NerveCenter lets you notify administrators of events in the following ways:

- You can send an audible alarm (a beep) to workstations running the NerveCenter Client.
- You can send email to an administrator using either a Microsoft Exchange Server client or SMTP mail.
- You can send SMS messages.
- You can send information about a network or system condition to another NerveCenter server. This capability is useful if you have a number of NerveCenter servers at different sites and want these servers to forward information about important events to a central server.
- You can send information about a network or system condition to a network management platform such as IBM Tivoli Netcool/OMNIbus. Administrators can then be notified of a problem found by NerveCenter using the other management tool’s console.

For more information on integrating NerveCenter with other network management products, see the section "Role in Network Management Strategy" on page 17.

Logging

If you want to keep a record of an event that takes place on your network, you must explicitly log information about the event at the time it occurs. NerveCenter provides three actions that provide for such logging:

- Log to File
- EventLog

Log to File writes information about an event to a file. Log to Database writes information about an event to the NerveCenter database. The EventLog action writes information about an event to the system log on the host running NerveCenter Server.

When you assign a logging action to a behavior model, you have the choice of logging default data or customizing what data you deem relevant. This saves disk space and streamlines information used later for analysis and reporting.

Causing State Transitions

In some behavior models, one alarm needs to cause a transition in another. The action that enables such communication between alarms is called Fire Trigger. This action creates a NerveCenter object called a trigger that can cause a state transition in the alarm from which it was fired or in another alarm.

The Fire Trigger action also lets you specify a delay, so you can request that a trigger be fired in one minute or five hours. This feature is especially useful when you're looking for the persistence of a condition. Let's say that you want to look for three intervals of high traffic on an interface within a two-minute period. When your poll detects the first instance of high traffic, and your alarm moves out of the Ground state, you can fire a trigger with a two-minute delay that will return your alarm to the Ground state—unless a second and third instance of high traffic are detected.

If a third instance of high traffic is detected, you should cancel the trigger you fired on a delayed basis. You do this by adding the Clear Trigger action to the transition from the second high-traffic state to the third.

NerveCenter also includes a Send Trap action. You define the trap to be sent, including the variable bindings, and associate the action with a state transition. When the transition occurs, the trap is sent. The trap can be caught by a NerveCenter trap mask—in which case you can use Send Trap somewhat like Fire Trigger, to generate a trigger—or by any application with SNMP traps.

Corrective Actions

There are a number of NerveCenter actions that you can use to take corrective actions when a particular state transition occurs. These are:

- Command
- Perl Subroutine
- Set Attribute
- Delete Node
- SNMP Set

The Command action enables you to run a script or executable when a particular transition occurs.

The Perl Subroutine action enables you to execute a Perl script as a state-transition action. You first define a collection of Perl scripts and store them in the NerveCenter database; then, you choose one of your stored scripts for execution during a state transition.

The Set Attribute action enables you to set selected attributes of the NerveCenter objects used to build behavior models.

The Delete Node action deletes the node associated with the current state machine from the NerveCenter database. This action is useful if you use a behavior model to determine which nodes you want to monitor and manage.

The SNMP Set alarm action changes the value of a MIB attribute when an alarm transition occurs.

Action Router

The Action Router enables you to specify actions that should be performed when a state transition occurs *and other conditions are met*. To set up these conditional actions, you add the Action Router action to your state transition. Then, you use the Action Router tool to define rules and their associated actions.

For example, let’s assume that you want to be notified about a state transition only if the transition puts the alarm in a critical state. You can define the following rule:

```
        $DestStateSev eq ‘Critical’
```

Then define the action you want taken if the severity of the destination state is Critical, for example, a page. You will be paged if:

- The Action Router action is associated with the current state transition
- The destination state for the transition is Critical

Action Router rules can be constructed using many variables that NerveCenter maintains; for instance, you can also construct rules based on:

- The name of the alarm
- The day of the week
- The time of day
- The name or IP address or group property of the node being monitored
- The name of the trigger that caused the state transition
- The name of the alarm’s property
- The name or severity of the origin state
- The contents of a trap
- The contents of the varbind data associated with a trap or a poll

The NerveCenter Node List

To detect and correlate network events, NerveCenter must have basic information about each managed node it monitors. NerveCenter obtains this data through several means.

NerveCenter can obtain node information from any or all of the following sources:

- A network management platform
- The NerveCenter Discovery behavior model
- An administrator’s manual entries

Note: Though NerveCenter supports SNMPv1, v2c, and v3, when NerveCenter obtains nodes from a platform, the platform does not provide SNMP version information. By default, NerveCenter deems the SNMP agents on these nodes to be SNMPv1.

If you want NerveCenter to attempt SNMP version classification automatically for the nodes it receives from your platform, you must enable auto-classification. Then, NerveCenter can classify the correct SNMP version for each node with each resynchronization. Refer to "Managing SNMP Settings" on page 95 for more information about SNMP auto-classification.

When NerveCenter uses information obtained by a network management platform, it does not use the platform’s database as its repository for managed nodes. Instead, NerveCenter imports and stores a copy of the node information in its own database.

There are a few reasons for NerveCenter maintaining a node list in its own database:

- There may be a considerable distance between the platform’s database and NerveCenter, making frequent access time-consuming and costly.
- NerveCenter adds configuration data to the node data that the management platform does not necessarily provided.
- Administrators have the option of adding nodes not in the platform’s node database to the node list in NerveCenter’s database.

Filtering Nodes

Before populating the node list in NerveCenter’s database, it is important to decide which nodes NerveCenter will manage. You may want to include or exclude devices based on their type (printer, switch, workstation, server, etc), role, location, ownership, administrative control, and similar. There are several automated methods for restricting or allowing which nodes will be placed in NerveCenter’s node list:

- "Filtering Using Node Capabilities" on the facing page
- "Filtering Using a Node System Object Identifier" on the facing page
- "Filtering Nodes by IP Address" on page 70
- "Filtering Nodes by Hostnames" on page 74

Filtering Using Node Capabilities

You can monitor nodes with particular capabilities, which your network management system typically assigns to a node to determine the applicable activities such as isRouter, isHub, and isIP.

Note: Filtering by capabilities is available only when your network management platform has assigned specific capabilities to a node.

TO FILTER USING A NODE'S CAPABILITIES

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server.

   See the section "Connecting Administrator to a NerveCenter Server" on page 25.

2. Select the **Node Source** tab.
3. In the **Wanted Capabilities** field, type the name of the specific capability desired.

   If this field is left blank, NerveCenter will not filter using a node's capability.

4. To enter multiple capabilities, separate each with a space.

   NerveCenter will monitor any node that matches at least *one* of the capabilities in the list.

5. Select **Save**.

   The NerveCenter Server adds the new capabilities filter. It also closes and opens a new connection with the platform adapter.NerveCenter automatically resynchronizes with your network management platform database. New nodes will be added. Any node that is marked Autodelete (the default) will be deleted.

Filtering Using a Node System Object Identifier

NerveCenter allows you to monitor managed nodes according to their particular system object identifiers (OIDs). A node's System Object ID is an SNMP MIB-II object in the system group. It identifies the SNMP agent software running on the device. It is, however, commonly used to identify the type and vendor of the device because a particular vendor's agent usually runs on that vendor's devices.

TO FILTER USING A NODE'S SYSTEM OBJECT IDENTIFIER

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server.

   See "Connecting Administrator to a NerveCenter Server" on page 25.

2. Select the **Node Source** tab.
3. In the **System Object Ids** field, type the name of the system OID wanted.

   If this field is left blank, NerveCenter will not filter using a node's system OID.

4. To enter multiple OIDs, separate each with a space.

   NerveCenter will monitor any node that matches at least *one* of the OID in the list.

   For example, an administrator may want to restrict NerveCenter to nodes running SNMP agents from either Cisco or Hewlett-Packard nodes by typing the following:

   ```
   1.3.6.1.4.1.9 1.3.6.1.4.1.11
   ```

   Any device with an OID matching either of these numbers will be included in the NerveCenter nodes database.

5. Select **Save**.

   The NerveCenter Server adds the new OID filter. It also closes and opens a new connection with the platform adapter.NerveCenter automatically resynchronizes with your network management platform's database. New nodes will be added. Any node that is marked Autodelete (the default) will be deleted.

Filtering Nodes by IP Address

In addition to filtering nodes by OIDs and capabilities, NerveCenter allows you to filter out all nodes that do not belong to one or more subnets.NerveCenter determines the subnet by combining a specific IP address with a subnet mask.NerveCenter can filter by subnets of both Class B and Class C networks. In Class B networks, the first two octets specify the network while in Class C networks the first three octets identify the network.

Table 8 and Table 9 illustrate some filter configurations and their results:

Table 8: Sample Subnet Filters and Their Results for a Class C Network

IP address	Subnet mask	Result
134.204.179.0	255.255.255.0	All nodes on subnet 134.204.179.0 are included. For example, 134.204.179.7 is included.
197.22.44.0	255.255.255.240	All nodes 1-15 on subnet 197.22.44.0 are included. For example, 197.22.44.5 is included but 197.22.44.35 is excluded.
134.204.179.0 197.22.44.0	255.255.255.0 255.255.255.240	All nodes on subnets 134.204.179.0 and nodes 1-15 on 197.22.44.0 are included. For example, both 134.204.179.7 and 197.22.44.5 are included.

Table 9: Sample Subnet Filters and Their Results for a Class B Network

IP address	Subnet mask	Result
132.45.0.0	255.255.0.0	All nodes on subnet 132.45.0.0 are included. For example, 132.45.174.7 is included.

IP address	Subnet mask	Result
132.45.0.0	255.255.240.0	Nodes 1.0 - 15.255 are included. For example 132.45.14.231 is included but 132.45.174.7 is excluded.

In addition to filtering out all but an entire subnet, NerveCenter allows you to exclude a specific node or range of nodes within the remaining subnet.

Table 10 illustrates some filter configurations with exclusions and their results:

Table 10: Sample Subnet Filters with Exclusions and Their Results

IP address	Subnet mask	Excluded node(s)	Result
134.204.179.0	255.255.255.0	40	All nodes on subnet 134.204.179.0 except node 40 are include. For example, 134.204.179.7 is included but 134.204.179.40 is excluded.
134.204.179.0	255.255.255.0	40-55	All nodes on subnet 134.204.179.0 except nodes 40-55 are include. For example, 134.204.179.7 is included but 134.204.179.40 and 134.204.179.52 are excluded.
132.45.0.0	255.255.0.0	63.5	All nodes on subnet 132.45.0.0 except node 63.5 are included.

Note: You can filter nodes that have been discovered by NerveCenter, provided by the platform node source, or imported from a node file.

NerveCenter can automatically or manually determine the subnet criteria used to filter nodes by IP address.

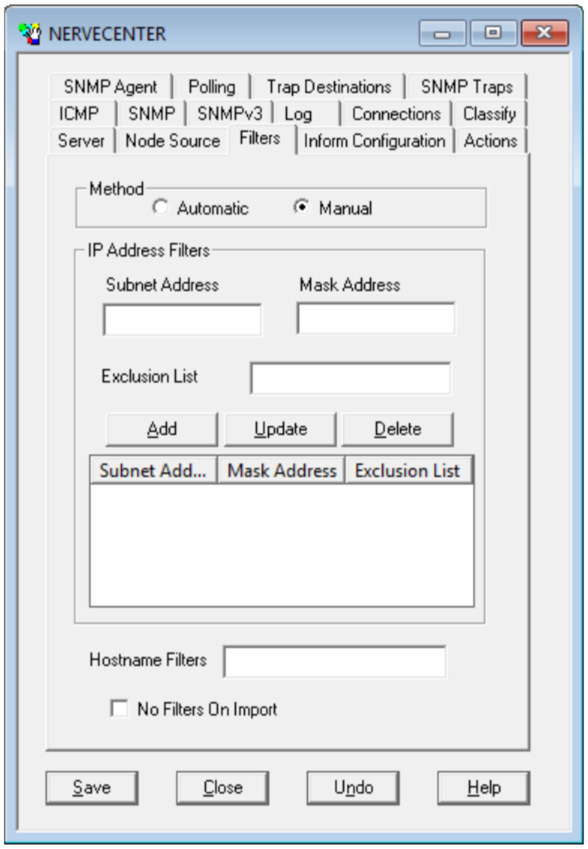
To Configure NerveCenter to Determine Subnet Criteria Automatically

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server.

   See "Connecting Administrator to a NerveCenter Server" on page 25.

2. Select the **Filters** tab.

   The Filters tab is displayed.



Method: Automatic [✓ Manual]

3. In the Method area, select **Automatic**.

   Setting the method to automatic tells NerveCenter to ignore any address filters you enter and use instead the server's masks as a filter.NerveCenter calculates the subnet address and mask using the IP address and mask of each network interface card on the server.

4. Select **Save**.

   NerveCenter will now automatically use the server's masks as a filter.

To Set the Subnet Criteria Manually

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server.

   See "Connecting Administrator to a NerveCenter Server" on page 25.

2. Select the **Filters** tab.

   The Filters tab is displayed.

3. In the Method area, select **Manual**.

   When the method is set to manual, NerveCenter will only use the subnet addresses listed in IP Address Filters area. If the IP Address Filters list is empty, NerveCenter ignores a node's subnet when determining if the node will be part of the node database.

4. In the **Subnet Address** field, type the appropriate subnet. In the **Mask Address** field, type the appropriate subnet mask.

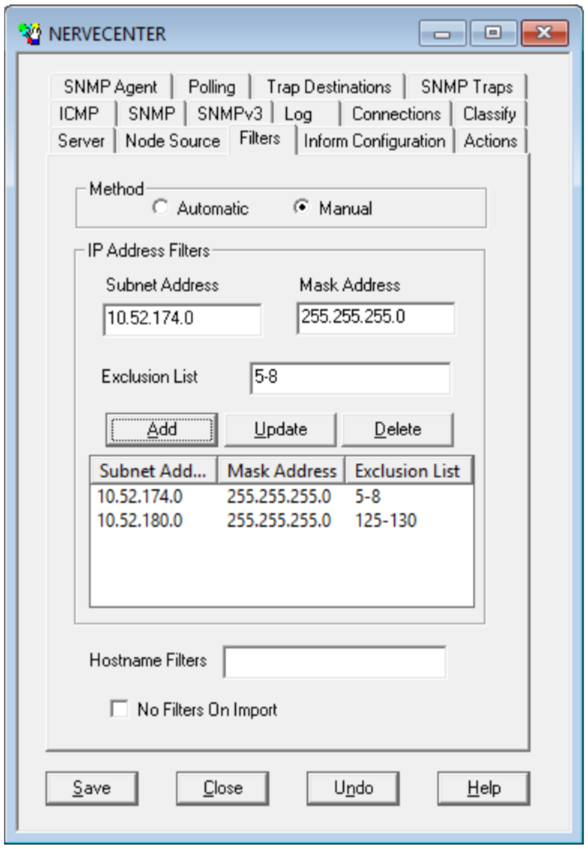
   A node's subnet address combines the node's IP address with the subnet mask.

5. In the Exclusion List field, enter all the nodes you want excluded from the subnet address. To exclude more than one node, separate each number with a comma without a space. To exclude a continuous range of nodes, use a hyphen to separate the minimum and maximum number by a hyphen.

Caution: Once you have added a node using the IPSweep behavior model, you cannot use an IP exclusion to delete it from the database. Once the node is in the database, the IP exclusion filter is not applied to it. The IP exclusion applies only to new nodes discovered after the filter is established.

6. Select **Add**.

   The subnet address and mask address will be added to the IP Address Filters list.


NERVECENTER
SNMP Agent | Polling | Trap Destinations | SNMP Traps
ICMP | SNMP | SNMPv3 | Log | Connections | Classify
Server | Node Source | Filters | Inform Configuration | Actions
Add
Update
Delete
Save
Close
Undo
Help

Method: ( ) Automatic (•) Manual

IP Address Filters

Subnet Address: 10.52.174.0

Mask Address: 255.255.255.0

Exclusion List: 5-8

Subnet Add...	Mask Address	Exclusion List
10.52.174.0	255.255.255.0	5-8
10.52.180.0	255.255.255.0	125-130

Hostname Filters:

☐ No Filters On Import

7. Select **Save**.
8. To filter by additional IP addresses and masks, repeat steps 4 and 7.

NerveCenter monitors any address falling within the subnet and not excluded by the filter.

Filtering Nodes by Hostnames

In addition to filtering nodes by OIDs, capabilities, and IP Addresses, NerveCenter allows you to filter out nodes by hostname. Filtering by hostname works in conjunction with IP Filters. If a node matches on either an IP filter or a hostname filter, the node will be filtered.

The hostname filter restricts imported node lists. Nodes discovered by SNMP traps are not affected by hostname filters because discovered nodes do not always have hostname information.

To configure hostname filters

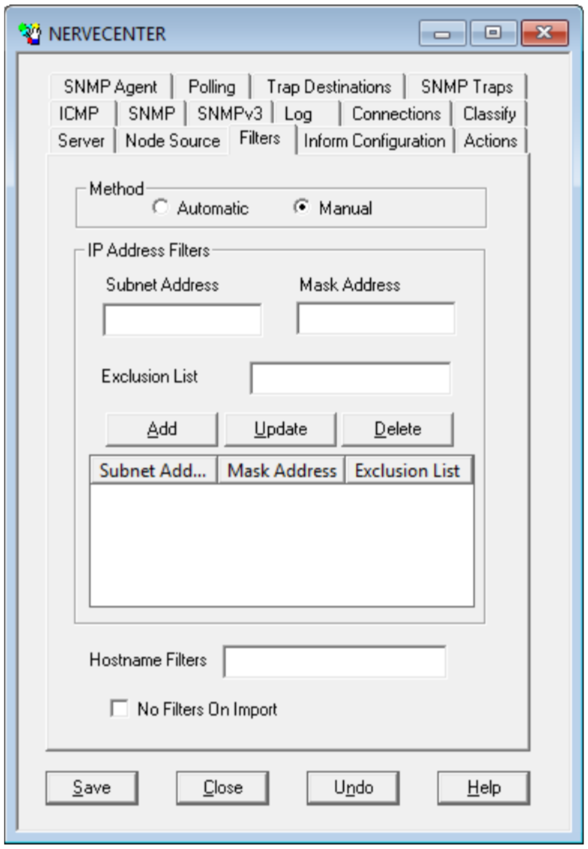
1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server.

   See "Connecting Administrator to a NerveCenter Server" on page 25.

2. Select the **Filters** tab.

   The Filters tab is displayed.

review - make this a link instead of showing the image again?



Method: Automatic [✓ Manual]

IP Address Filters

Subnet Address:

Mask Address:

Exclusion List:

Subnet Add...	Mask Address	Exclusion List

Hostname Filters:

☐ No Filters On Import

3. In the Hostname Filters field, enter the hostnames you want to exclude, separating names with a space.

   You can use an asterisk (*) as a wild card. For example, ***server** filters out all hostnames ending with server and ***router*** filters out all hostnames containing the word router.

4. Select **Save**.

   NerveCenter now filters by hostnames.

Enabling and Disabling IP and Hostname Filters

You can define IP Address and hostname filters to limit what nodes you add to your NerveCenter node list. This is important when using discovery methods such as IPSweep, so that you do not try to include the entire internet. However, if you have a node list you want to import of known hosts you want to monitor, you may not want to apply the IP and Hostname filters.

IP filters apply to the following:

- Importing node lists
- Populating the node lists with a network management platform
- Discovering nodes from traps, if you select IP Filter from the Discover Nodes from Traps list on the Server tab.

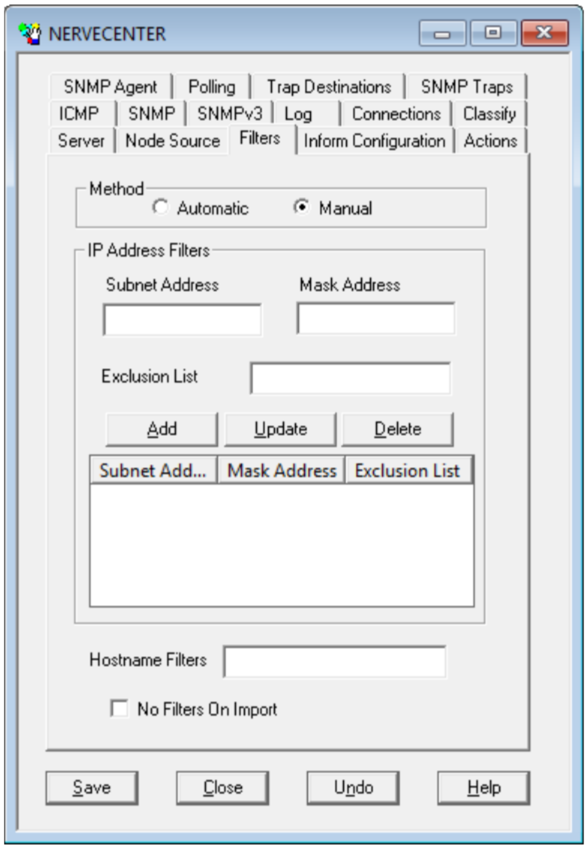
Hostname filters apply to the following:

- Importing node lists
- Populating the node lists with a network management platform

You can disable IP and hostname filters if you do not want to use them for an import from a node list or management platform.

To enable or disable IP and hostname filters

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server.
   See "Connecting Administrator to a NerveCenter Server" on page 25.
2. Select the **Filters** tab.
   The Filters tab is displayed.



Method: Automatic [✓ Manual]

IP Address Filters

Subnet Address

Mask Address

Exclusion List

Subnet Add...	Mask Address	Exclusion List

Hostname Filters

☐ No Filters On Import

3. Select **No Filters On Import** to disable IP and hostname filters; clear it to enable IP and hostname filters.
4. Click **Save**.

Processing Traps from Unknown Devices

NerveCenter offers a lot of flexibility in filtering the managed nodes it monitors. By setting node filters, you are telling NerveCenter to ignore any traps sent by nodes not stored in the node list. This keeps the node list to a size that is manageable and acceptable under your license agreement.

However, there may be times you will want a NerveCenter Server to process a trap from a node not found in its node list, but in the node list of another NerveCenter Server. For example, in the following diagram, a NerveCenter Server responsible for monitoring a LAN (Server A) needs to pass a trap along to a WAN-level NerveCenter Server (Server B).

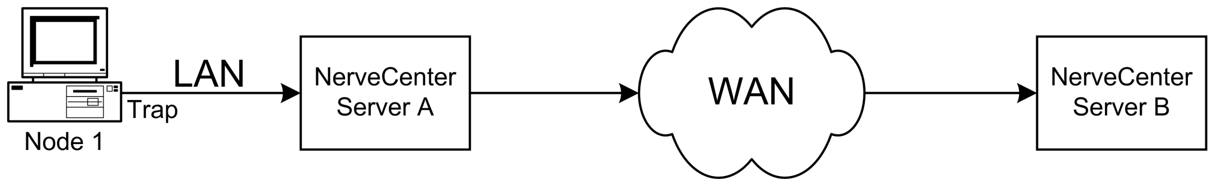


Figure 14: An Example of a Trap Being Processed from an Unknown Node

In this case, Node 1 is in the node list of Server A. Node 1 sends a trap to Server A, which in turn is passed along to Server B. Because Node 1 is being managed by Server A, it will not appear in Server B's node list. Server B needs to process the trap from Node 1. Server B could be set up to add unknown nodes to its node list whenever it receives a trap, but then both Server A and Server B will be monitoring Node 1. How can Server B process the trap from Node 1 without first adding the node to its node list?

The Process traps from unknown nodes feature allows a NerveCenter Server receiving a trap from an unknown node to process that trap if the trap is associated with an Enterprise scope alarm.

Note: The Process traps from unknown nodes feature will only process traps associated with Enterprise scope alarms.

At the same time, the Process traps from unknown nodes feature will keep the NerveCenter Server from adding the node to its node list. This ensures that the NerveCenter Server does not exceed the node limit allowed by its license. It also avoids a situation in which two NerveCenter Servers are responsible for the same node.

Note: To learn how a NerveCenter Server adds an unknown node to its node list when receiving a trap from that node, see "Adding Nodes Discovered from Traps" on page 88.

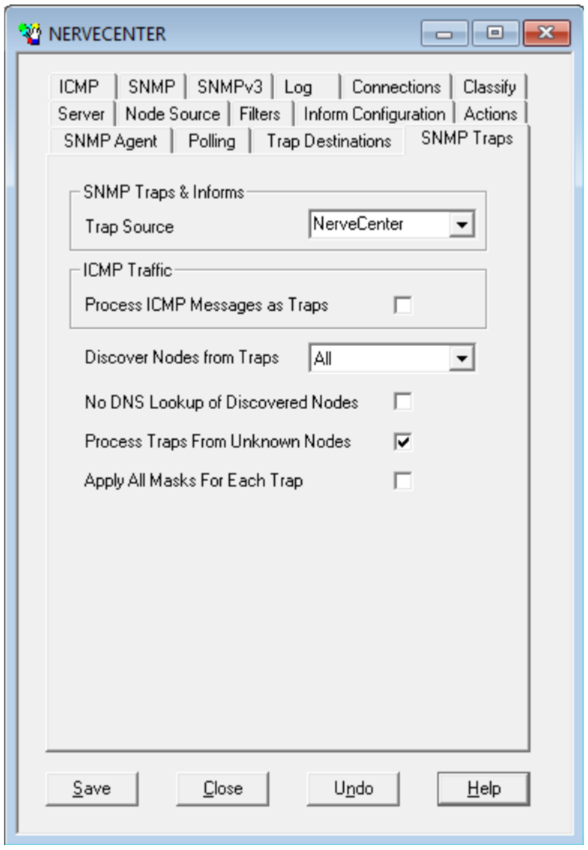
To Process Traps from Unknown Nodes

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server.

   See "Connecting Administrator to a NerveCenter Server" on page 25.

2. Select the **SNMP Traps** tab.

   NerveCenter displays the SNMP Traps tab.



SNMP Traps & Informs
Trap Source: NerveCenter
ICMP Traffic
Process ICMP Messages as Traps ☐
Discover Nodes from Traps: All
No DNS Lookup of Discovered Nodes ☐
Process Traps From Unknown Nodes ☑
Apply All Masks For Each Trap ☐

3. Select the **Process Traps From Unknown Nodes** box.

   If this box is not selected, NerveCenter ignores any traps from a node not in its node list.

4. Select **Save**.

The NerveCenter Server will now process traps associated with Enterprise scope alarms even when those traps are received from nodes not appearing in the server's node list. You can also use a Perl function, AddNode(), to add nodes to the NerveCenter database. See AddNode() Function in Designing and Managing Behavior Models for details.

Initially Populating the Node List

To detect and correlate network events, NerveCenter must have basic information about each of the managed nodes it monitors. Once you have configured the proper node filters (see "Filtering Nodes" on page 68), you must tell NerveCenter where to look for the initial data it will use to populate its node list.

Though you can manually add nodes to NerveCenter, it is easy to configure NerveCenter to discover nodes (Figure 15). When NerveCenter discovery is enabled, if the database does not already contain a node that sends it an SNMP trap or NerveCenter inform, NerveCenter adds that node to the database.

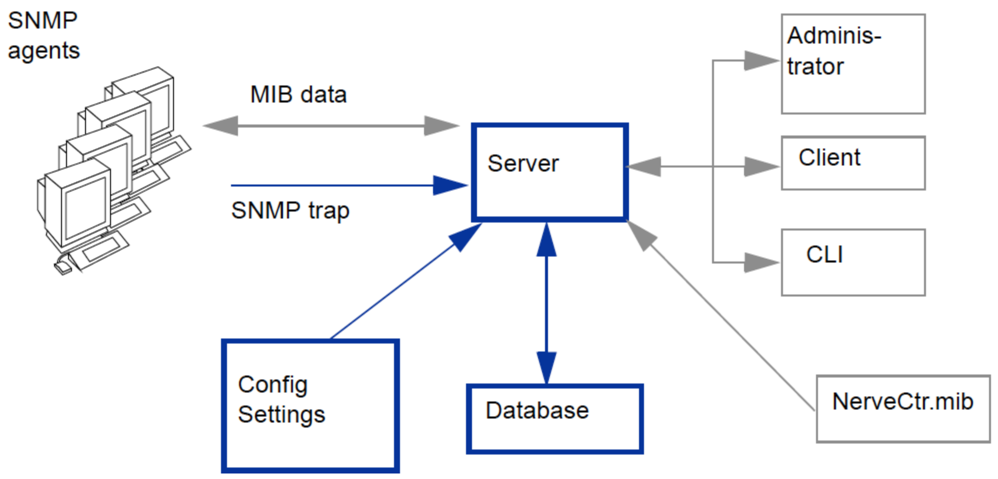


Figure 15: NerveCenter Configured To Discover Nodes

After installing NerveCenter, you can use the NerveCenter Administrator to define subnet IP filters that limit the sets of nodes NerveCenter can monitor. These filter values are stored with the NerveCenter configuration settings.

When a trap is received from a node, NerveCenter compares the node against those that are already in its database and confirms whether the node falls within the subnet IP filters you defined. Nodes that fall within the subnet range but are not in the database are added to the database.

Note: NerveCenter can be configured to process SNMP traps from nodes residing outside the defined subnet filters. These nodes, however, are not added to the database. See "Filtering Nodes by IP Address" on page 70 for more details.

There are three main methods for populating NerveCenter's node list:

- "Populating Using a Network Management Platform" below
- "Populating Using the IPSweep Behavior Model" on page 83
- "Populating the Node List Manually" on page 86

Populating Using a Network Management Platform

NerveCenter is able to receive information about some or all of the nodes managed by your network management platform.NerveCenter is able to retrieve data about managed nodes from the following HP OpenView Network Node Manager.

Note: To use your network management platform to populate NerveCenter's node database, you must have the NerveCenterOpenView Platform Adapter (OVPA) installed and running.

To populate NerveCenter's node list using your network management platform, you must specify it as a source for the node data. Each NerveCenter database populates its node list from just one network management platform database. Depending on your filtering, the database may contain all the nodes or just a subset. In either case, there is just one source of the information.

Caution: If you wish to map system Object Identifiers (OID) to NerveCenter property groups, you must make the necessary configurations in the NerveCenter Client before naming the node data source. (See Using OID to Property Group Mappings in Designing and Managing Behavior Models.) After NerveCenter initially populates its node list, any subsequent mapping of OIDs to property groups affect only new nodes added to the node list.

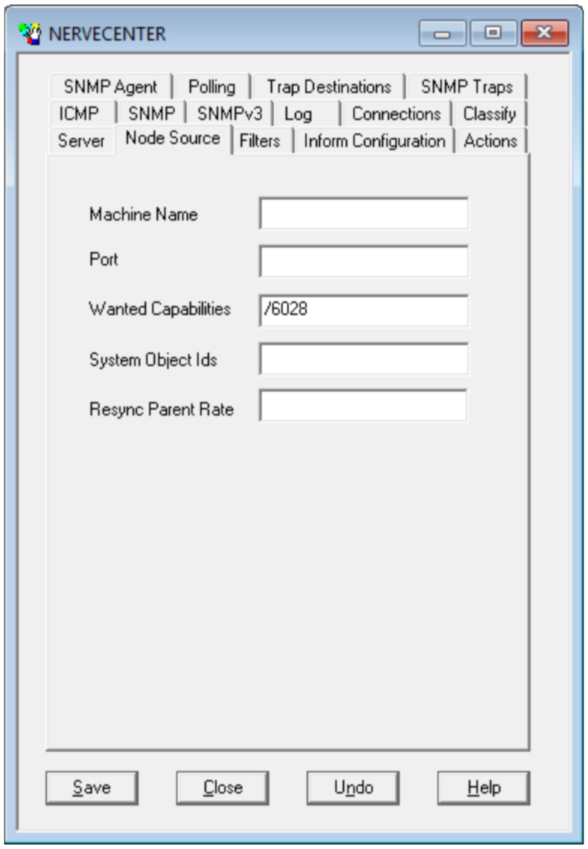
To Use Your Network Management Platform as a Node Data Source

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server.

   See "Connecting Administrator to a NerveCenter Server" on page 25.

2. Select the **Node Source** tab.

   NerveCenter displays the Node Source tab.



Machine Name	
Port	
Wanted Capabilities	/6028
System Object Ids	
Resync Parent Rate	

3. In the **Machine Name** field, type the name or IP address of a host that runs your network management platform.

   If the Machine Name field is left blank, NerveCenter does not retrieve nodes.

4. In the **Port** field, type the number of the port used to communicate with the platform adapter process on the host. The default is 6024.

   The platform adapter must be configured to listen on the same port specified in this field.

5. Select **Save**.

NerveCenter will now retrieve its initial node data from your network management platform's database. IP and hostname filters can apply to what node data is imported from the network management platform. See "Enabling and Disabling IP and Hostname Filters" on page 76 for details about using filters with node lists.

Populating Using the IPSweep Behavior Model

There may be situations in which you will not want to use a network management platform to populate NerveCenter's node list initially. Such situations could include:

- Your network does not contain an applicable network management platform or you are using NerveCenter as a stand-alone application.
- A large distance or a costly link separates the platform from the nodes that NerveCenter will be monitoring locally.

The IPSweep behavior model is intended to handle these situations. IPSweep allows NerveCenter to detect unknown nodes and add them to the node database, provided they fall within the set filters. Figure 16 shows how IPSweep fits in with the other NerveCenter components.

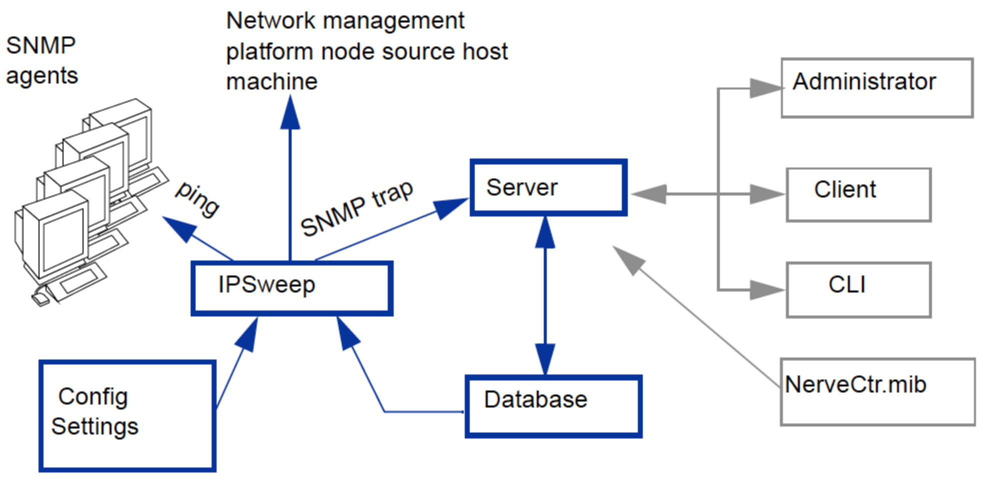


Figure 16: NerveCenter Configured to Discover Nodes

IPSweep extracts subnet IP filter information from the NerveCenter configuration settings, obtains node information from the NerveCenter database, identifies nodes that fall within the subnet range but are not in the database, and sends those nodes a ping (ICMP echo request). If the ping returns a response, IPSweep issues an SNMP trap either to a host specified as the node source or, if no node source is specified, to the local NerveCenter. In a standalone configuration, the trap is sent to NerveCenter. Once NerveCenter receives the trap, the node is added to the NerveCenter database.

IPSweep does not populate the NerveCenter server database; it sends a trap if it gets a ping response from an IP address, at which point the server can add the unknown node to its database.

Note: If you are using NerveCenter in Windows with a Domain Name Server (DNS), the IPSweep behavior model requires that **Enable DNS for Windows Resolution** be selected on the Protocols > TCP/IP > WINS tab of the Network Control Panel.

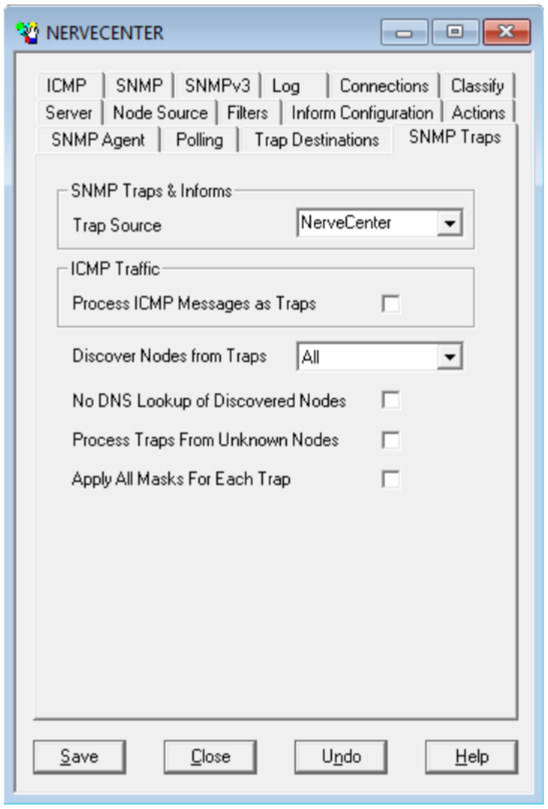
TO POPULATE THE NODE LIST USING THE IPSWEEP BEHAVIOR MODEL

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server.

   See "Connecting Administrator to a NerveCenter Server" on page 25.

2. Select the **SNMP Traps** tab and configure NerveCenter to filter by the appropriate subnet criteria.

   NerveCenter displays the SNMP Traps tab.



SNMP Traps & Informs

Trap Source: NerveCenter

ICMP Traffic

Process ICMP Messages as Traps ☐

Discover Nodes from Traps: All

No DNS Lookup of Discovered Nodes ☐

Process Traps From Unknown Nodes ☐

Apply All Masks For Each Trap ☐

3. Configure NerveCenter to filter by the appropriate subnet criteria.

   See "Filtering Nodes by IP Address" on page 70 for more information.

Caution: If the IP Filter box is empty, the IPSweep behavior model's primary application ipsweep will not run. This precaution is to prevent NerveCenter from trying to discover all the nodes on the Internet.

4. Select the **Server** tab.

   NerveCenter displays the Server tab.

5. Select from the **Discover Nodes from Traps** list **All** or **IP Filter**.

Note: IP Filter only functions if IP Filters are enabled as described in "Enabling and Disabling IP and Hostname Filters" on page 76. If the **Discover Nodes from Traps** field is set to **None**, the IPSweep behavior model will not work.

6. (Optional) Select **No DNS Lookup of Discovered Nodes** to improve performance.

   By default, No DNS Lookup of Discovered Nodes is not selected and NerveCenter performs a DNS lookup for each unknown node. If you select this box, the nodes are added by IP address and NerveCenter does not attempt a DNS lookup.

7. Select **Process Traps From Unknown Nodes**.
8. If you want NerveCenter to turn on the IPSweep behavior model every time the NerveCenter Server is started, select **Enable Discovery at Startup**.
9. From NerveCenter Client, turn on the IPSweep behavior model. See Enabling the IPSweep Alarm in Designing and Managing Behavior Models for details.

NerveCenter periodically runs the script ipsweep. As NerveCenter discovers unknown nodes that fall within its IP filters it will add them to the node list. For more information about NerveCenter's IPSweep behavior model, see Using IPSweep Behavior Model in Designing and Managing Behavior Models.

Populating the Node List Manually

The two most popular methods of initially populating NerveCenter's node list are:

- "Populating Using a Network Management Platform" on page 80
- "Populating Using the IPSweep Behavior Model" on page 83

An alternative is to populate the NerveCenter node list by hand. For each node that NerveCenter will manage, you must use the Node Definition window to specify the node's data, including:

- Name
- Address
- Community string
- Property group

Because defining your own node data demands a considerable amount of attention, this alternative is recommended only for the smallest number of managed nodes.

To populate NerveCenter's node list manually, follow the steps described in "Adding and Deleting Nodes Manually" on page 90.

Maintaining the Node List

To detect and correlate network events, NerveCenter must have basic information about each of the managed nodes it is monitoring.

Once you have initially populated the node list (see "Initially Populating the Node List" on page 80), NerveCenter will need a way to adapt its node list to reflect changes in your network topology. NerveCenter offers several methods for adding and deleting nodes.

These methods include:

- "Synchronization with Your Network Management Platform" below
- "Adding Nodes Discovered from Traps" on the next page
- "Adding and Deleting Nodes Manually" on page 90

Table 11 illustrates how each method affects NerveCenter's node list.

Table 11: Maintaining NerveCenter's Node List

Method	Adds Nodes	Deletes Nodes	Changes Node Data
Resync with the platform	✓	✓	✓
Discover nodes from traps	✓		
IPSweep behavior model	✓		
Manual additions, deletions, and changes	✓	✓	✓

Synchronization with Your Network Management Platform

Over time, a network's topology will change. Eventually your network management platform will add newly discovered devices to its database. It will also delete nodes and change node information. If NerveCenter depends on your network management platform for the data in its node list, it will need to adapt to reflect these changes.

NerveCenter will automatically update its node list to keep in sync with your network management platform's node data. This occurs in the following situations:

- When your network management platform adds a node to its node database. After NerveCenter verifies the node meets the criteria set by its filters, it will add the node to its node list.
- When your network management platform deletes a node from its node database. NerveCenter will delete from its node list any node that is set to Autodelete. Autodelete is the default setting for any new node added to the node list. This setting can be changed in the node's Node Definition Window in the NerveCenter Client. (See Discovering and Defining Nodes in Designing and Managing Behavior Models.)
- When your network management platform changes information about a node in its node database. NerveCenter will make any necessary changes to its node data, including changes in the community string, address, parenting information or the managed/unmanaged state.

Note: If your network management platform unmanages a node in the NerveCenter node list, the unmanaged state will be updated in NerveCenter. However, if your network management platform unmanages a node not found in NerveCenter's node list, the node will not be added to NerveCenter.

Most often, the node list will only be updated a node at a time. Occasionally, NerveCenter will need to perform a complete resynchronization with the platform. A resynchronization gathers from the platform the most current node data for all nodes. This occurs in the following situations:

- The NerveCenter Server is started and successfully connects to the OpenView Platform Adapter (OVPA).
- A connection between the NerveCenter Server and the node source successfully reconnects after being broken.
- The NerveCenter administrator changes the way in which NerveCenter filters by capabilities or system Object Identifiers (OIDs).
- A user manually chooses **Resync** in the **Server** menu of the NerveCenter Client.

The **Machine Name** field on the **Node Source** tab of the NerveCenter Administrator specifies the name of the host running the platform resynchronizing with NerveCenter. (See "Populating Using a Network Management Platform" on page 80 for more details on how to declare a node data source.)

The **Node Source** and **Filters** tabs also specify the parameters NerveCenter uses to filter node data. (See "Filtering Nodes" on page 68.)

Anyone administering NerveCenter should be aware of two important scenarios involving changes to your network management platform's database:

- If the name changes in your network management platform's database, NerveCenter considers it to be a new node.
- If a node is unmanaged in one of your network management platform maps but is managed in another, the node remains in the managed state in NerveCenter's node data.

Caution: Since your network management platform's node is matched to a NerveCenter node using its name, you should use care when changing NerveCenter's node configurations. Resynchronization adds nodes when it cannot find names that match your network management platform's map information. Therefore, if you change a node's name in the Node Definition window, resynchronization will not find a match and will add a node, resulting in two nodes with the same address but different names.

Adding Nodes Discovered from Traps

NerveCenter occasionally receives traps from nodes not included in its node list. In this situation, NerveCenter must make two decisions:

- Whether to process the trap or ignore it (as described in "Processing Traps from Unknown Devices" on page 78).
- Whether to add the unknown node to its node list. This section explains how to configure this setting.

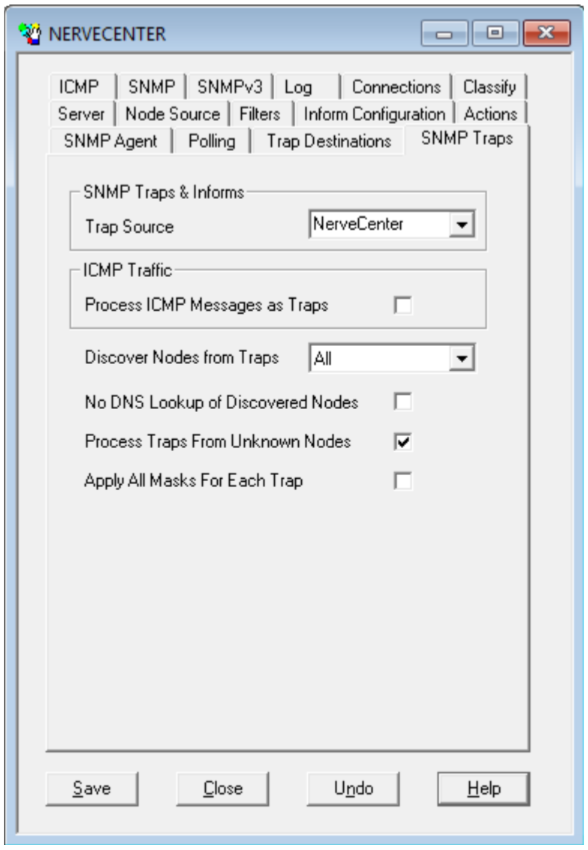
To configure the way unknown nodes affect NerveCenter's node list

1. Open NerveCenter Administrator and connect to the appropriate Server.

   See "Connecting Administrator to a NerveCenter Server" on page 25.

2. Select the **SNMP Traps** tab.

   NerveCenter displays the SNMP Traps tab.



SNMP Traps & Informs
Trap Source: NerveCenter

ICMP Traffic
Process ICMP Messages as Traps: ☐

Discover Nodes from Traps: All
No DNS Lookup of Discovered Nodes: ☐
Process Traps From Unknown Nodes: ✓
Apply All Masks For Each Trap: ☐

3. In the **Discover Nodes from Traps** list, select nodes to place in the NerveCenter node list:

   - **None** - NerveCenter will not add any unknown nodes to the node list.
   - **IP Filter** - NerveCenter will add to its node list an unknown node sending a trap *only* if it meets the criteria established by the node filters.

     IP Filter only functions if IP Filters are enabled. See "Enabling and Disabling IP and Hostname Filters" on page 76 for details.

   - With an **All** setting, NerveCenter adds to its node list any unknown node sending a trap, regardless of the node filters.

4. (Optional) Select **No DNS Lookup of Discovered Nodes** to improve performance.

   By default, No DNS Lookup of Discovered Nodes is not selected and NerveCenter performs a DNS lookup for each unknown node. If you select this box, the nodes are added by IP address and NerveCenter does not attempt a DNS lookup.

5. Select **Save**.

Any time NerveCenter receives a trap from a node not listed in its node list, NerveCenter handles the node according to the criteria you set in the **Discover Nodes from Traps** list box.

Note: NerveCenter sets autodeletes any unknown nodes added from traps. If NerveCenter adds an unknown node discovered from a trap but your network management platform fails to add it, the node will be deleted at the next resynchronization.

Adding and Deleting Nodes Manually

Occasionally, you may want to monitor one or more nodes that do not match the parameters set by the node filters. Or you may want to delete a node or a group of nodes. NerveCenter gives you the ability to alter the node list to your own specifications.

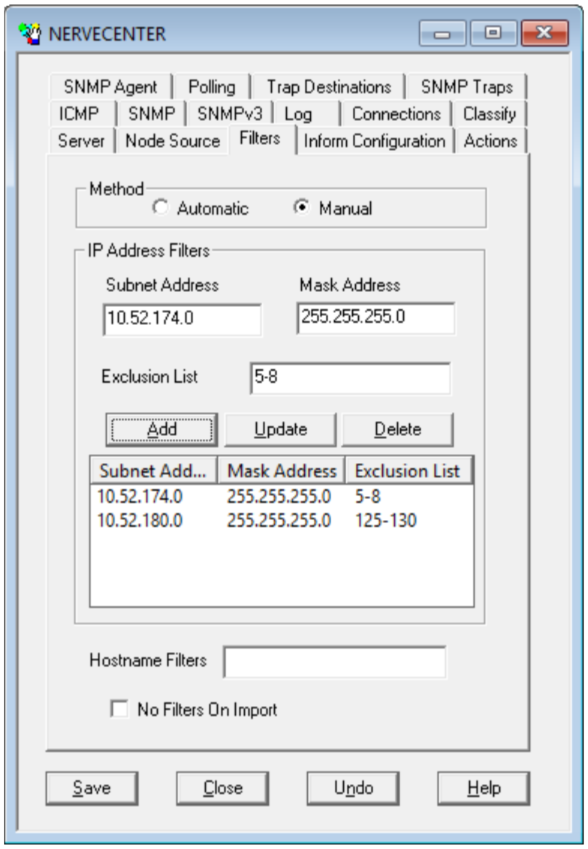
When maintaining the node list manually, you have the following options:

- **Adding a Node Manually** — You add nodes in the NerveCenter Client module, as described in Defining Nodes Manually in Designing and Managing Behavior Models.
- **Deleting a Node Manually** — You delete nodes in the NerveCenter Client module, as described in Deleting Objects in Designing and Managing Behavior Models.
- **Filter Out a Node That was Manually Deleted** — If you do not complete this step, the nodes you delete manually will be added the next time a resynchronization occurs between NerveCenter and the network management platform's database or the next time the IPSweep behavior model runs the ipsweep script. Following are instructions for filtering out a node.

TO FILTER OUT A NODE THAT WAS MANUALLY DELETED

1. Open the NerveCenter Administrator and connect to the appropriate NerveCenter Server.
2. Select the **Filters** tab.

   The Filters tab is displayed.


NERVECENTER
SNMP Agent | Polling | Trap Destinations | SNMP Traps
ICMP | SNMP | SNMPv3 | Log | Connections | Classify
Server | Node Source | Filters | Inform Configuration | Actions
Add
Update
Delete
Save
Close
Undo
Help

Method: Automatic [✓ Manual]

IP Address Filters

Subnet Address: 10.52.174.0
Mask Address: 255.255.255.0
Exclusion List: 5-8

Subnet Add...	Mask Address	Exclusion List
10.52.174.0	255.255.255.0	5-8
10.52.180.0	255.255.255.0	125-130

Hostname Filters:

☐ No Filters On Import

3. In the **IP Address Filters** list, select the subnet address of the node you want to delete.
4. In the **Exclusion List** field, enter the nodes you want deleted from the node list. Separate each node with a comma, but no space; use hyphens to exclude a continuous range of nodes.

Note: If you do not complete this step, deleted nodes will be re-added at the next resynchronization between NerveCenter and the network management platform or the next time the IPSweep model runs the ipsweep script.

5. Select **Update**.
6. Select **Save**.

The nodes are permanently deleted from NerveCenter's database.

Troubleshooting: Managing Node Data

The following list contains some common problems users have when managing node data.

NERVECENTER IS NOT FILTERING A NODE BY A CAPABILITY

Problem: Filtering by capabilities is available only when a network management platform has assigned a specific capability to a node.

Solution: Have your network management platform assign a capability to the node.

See your network management platforms documentation for details.

AFTER SETTING AN IP FILTER, A NODE THAT SHOULD BE MASKED OUT STILL APPEARS IN THE NODE LIST

Problem: IP filters only exclude additional nodes from being added to a node list. It does not actively delete nodes in the node list.

Solution: Either manually delete the node or force a resynchronization with your network management platform.

See "Adding and Deleting Nodes Manually" on page 90 or "Synchronization with Your Network Management Platform" on page 87.

EVEN THOUGH I HAVE ENABLED PROCESS TRAPS FROM UNKNOWN NODES, NERVECENTER DOES NOT UPDATE ITS NODE LIST WHEN IT RECEIVES A TRAP FROM AN UNKNOWN NODE

Problem: Selecting the **Process Traps From Unknown Nodes** box does not affect NerveCenters node list. It only tells NerveCenter to process traps.

Solution: If you want NerveCenter to update its node list based on traps received from unknown nodes, set Discover Nodes from Traps to Filter or All.

See "Adding Nodes Discovered from Traps" on page 88.

NERVECENTER DOES NOT RECOGNIZE MY NETWORK MANAGEMENT PLATFORM AS A VALID SOURCE OF NODE DATA

Problem: Currently NerveCenter is able to retrieve node data.

Solution: Use the IPSweep behavior model to obtain node data.

See "Populating Using the IPSweep Behavior Model" on page 83.

NERVECENTER IS NOT RECEIVING NODE DATA FROM MY NETWORK MANAGEMENT PLATFORM

Problem: They are using different ports.

Solution: Configure the node data source port number to be the same as the platform adapters port.

See "Populating Using a Network Management Platform" on page 80.

THE IPSWEEP BEHAVIOR MODEL WILL NOT WORK

Problem: There are no IP filters. NerveCenter will not discover nodes unless there are IP filters. This precaution is to prevent NerveCenter from trying to discover all the nodes on the Internet.

Solution: Set the appropriate IP filters.

See "Filtering Nodes by IP Address" on page 70.

Problem: The NerveCenter Server is not set to discover nodes from traps.

Solution: Set **Discover Nodes from Traps** to **All** or **Filter**.

See: "Adding Nodes Discovered from Traps" on page 88.

THE NERVECENTER NODE LIST CONTAINS TWO NODES WITH THE SAME ADDRESS BUT DIFFERENT NAMES

Problem: The node name was changed. When a resynchronization occurred between NerveCenter and the network management platform, the platform added the node.

Solution: Change the name of the node in your platform, not in NerveCenters node list.

See "Synchronization with Your Network Management Platform" on page 87.

NERVECENTER DELETES A NODE I ADDED MANUALLY

Problem: At a resynchronization between NerveCenter and a network management platform, NerveCenter deletes any nodes marked autodelete that are not found in the platforms node database.

Solution: When you add the node disable the autodelete feature.

See "Adding and Deleting Nodes Manually" on page 90.

NERVECENTER ADDS A NODE I DELETED MANUALLY

Problem: At a resynchronization between NerveCenter and a network management platform or when NerveCenter runs the Discovery behavior model, NerveCenter adds any new nodes that fall within its filters.

Solution: Exclude the node in the IP filters before deleting it.

See "Filtering Nodes by IP Address" on page 70.

I'M SEEING SEVERAL ERRORS RECORDED IN THE APPLICATION EVENT LOG WINDOW STATING THAT IPSWEEP.EXE IS NOT RUNNING

Problem: Your system is looking for ipsweep.exe in the wrong directory.

Solution: Update the correct path to ipsweep.exe in the alarm action Command in the Discovery behavior model.

Problem: Ipsweep.exe is already currently running.

Solution: Kill the first process and restart the Discovery behavior model.

See Using IPSweep Behavior Model in Designing and Managing Behavior Models for more details.

Managing SNMP Settings

7

This chapter contains information you need to configure NerveCenter for SNMP communication and to support SNMPv3 agents.

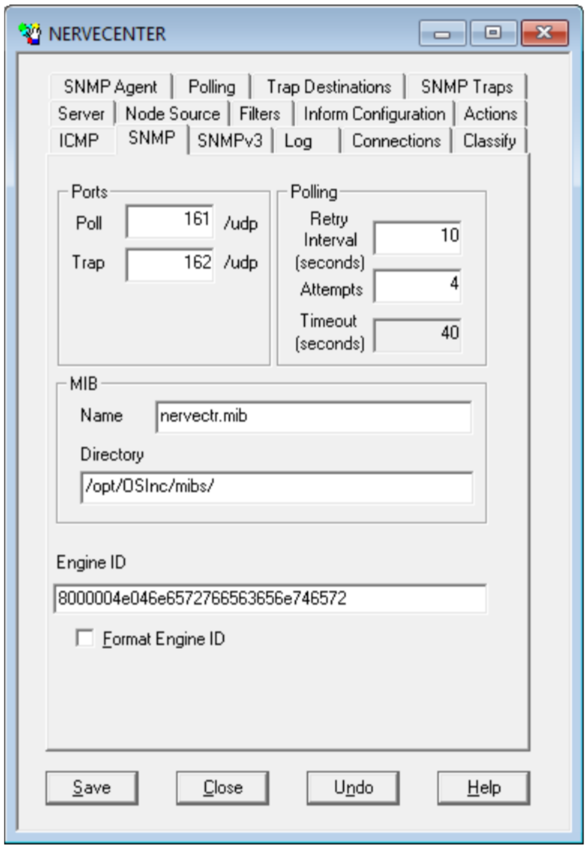
NerveCenter support of SNMP

NerveCenter fully supports SNMP in all its three versions and in the ways its current version (SNMPv3) has been periodically updated. As SNMP is a protocol based on UDP, its configuration takes into account the settings required for a UDP protocol. This includes the settings needed for timeout & retry handling and for port settings.

Timeout & Retry Handling

As with all UDP protocols, it must be defined how to react to the scenario where a protocol request is not answered. NerveCenter permits that a poll attempt should be retried several times before a timeout is declared. This can be set as both a system-wide configuration and, optionally, as a per-node configuration. When a node has been configured for timeout & retry handling, its settings override that system-wide settings.

- To set system-wide SNMP timeout & retry values, open the NerveCenter Administrator and modify the Polling section of the SNMP tab; see "Specifying SNMP Poll Intervals for NerveCenter" on page 109 for instructions.



Ports

Poll 161 /udp

Trap 162 /udp

Polling

Retry Interval (seconds) 10

Attempts 4

Timeout (seconds) 40

MIB

Name nervectr.mib

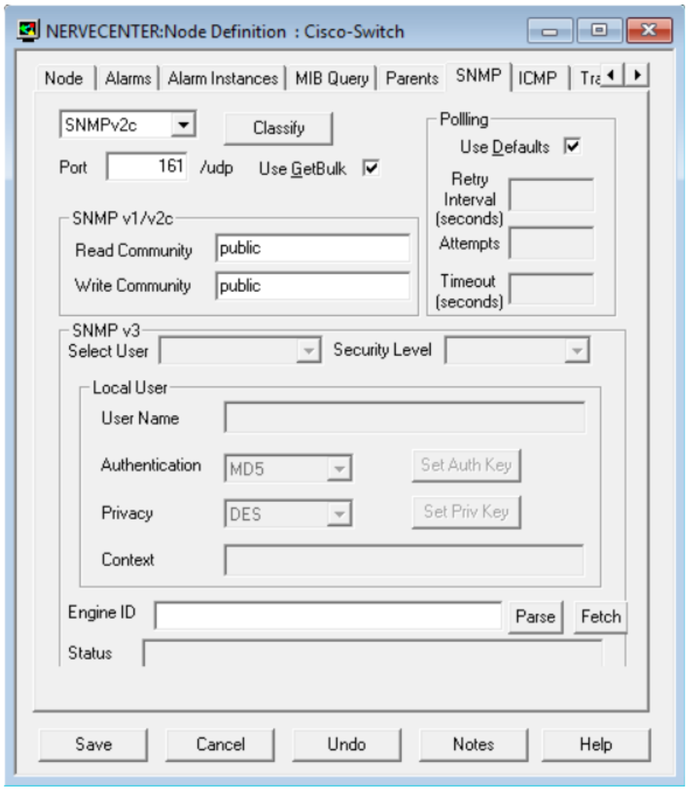
Directory /opt/OSInc/mibs/

Engine ID

8000004e046e6572766563656e746572

☐ Format Engine ID

- To set a per-node override for SNMP timeout & retry values, use the NerveCenter Client application to edit the node configuration. In the Node Definition dialog for the node, go to the SNMP Tab, uncheck the Use Defaults checkbox within the Polling section, and then set the Retry Interval (1-600 sec.) and Attempts (1-11) field as desired.



SNMPv2c

Port 161 /udp Use GetBulk ✓

Use Defaults ✓

Read Community: public

Write Community: public

Authentication: MD5

Privacy: DES

UDP Port Handling

SNMP Traffic to a device needs to be addressed to a known port. In UDP, ports are numbered and the range is from 0 up to 65,535. SNMP uses ports 161/udp and 162/udp for polling and notification destination respectively. These numbers are given by IANA and are in common usage. In NerveCenter, there is both a system-wide setting for these settings and an optional per-node override setting.

- To set the system-wide SNMP port configuration, use the ‘SNMP’ tab of the Administrator application.

  In the Ports section of the SNMP tab, set the Poll and Trap values. The range for each is 0 up to 65,535 inclusive and are set to 161 and 162 by default. These settings will be used for all SNMP Polling and Trap forwarding for all nodes managed by NerveCenter, except for those which have an explicitly prepared per-node setup.

- To set a node-specific SNMP configuration, use the NerveCenter Client to select the node; the polling Port field is on the SNMP tab.

Overview of NerveCenter SNMPv3 Support

NerveCenter supports all versions of SNMP for both polling and notification handling. SNMP v1 and v2c rely on community names for authentication. SNMPv3 replaces the authentication handling of the prior versions and adds privacy (encryption). SNMPv3 also expands on the earlier concept of MIB views to control access to management information by using a View-based Access Control Model (VACM) to determine a user's access level for viewing MIB data.

Following are highlights of NerveCenter support for SNMPv3:

- NerveCenter polls SNMPv3 agents using a configured set of parameters that needs to be set in agreement on both the SNMPv3 agent and within NerveCenter.

  See "Configuring an SNMPv3 Agent for NerveCenter" on page 116.

  See "Configuring SNMPv3 Security Settings" on page 118.

- NerveCenter supports the three security levels defined in SNMPv3 for communicating with SNMPv3 agents. By default, NerveCenter sets the security level to noAuthNoPriv, which means the v3 agent sends and receives messages without authentication or encryption.

  See "NerveCenter Support for SNMPv3 Security" on the facing page for details about security.

  Refer to Changing the Security Level of an SNMPv3 Node in Designing and Managing Behavior Modelsfor details about setting a node's security level.

- NerveCenter supports the HMAC-MD5-96 (MD5) or HMAC-SHA-96 (SHA) protocols for authentication, and DES, 3-DES, AES-128, AES-192, or AES-256 as privacy protocols. Authentication and privacy are defined on a per-node basis. If you change an agent's SNMPv3 configuration, you must likewise configure NerveCenter to use that same configuration to manage the corresponding node in its database.

  Refer to Changing the Authentication Protocol for an SNMPv3 Node in Designing and Managing Behavior Models for details about changing the authentication protocol used by NerveCenter for an agent.

- NerveCenter's SNMPv3 configuration for a node must be correct before NerveCenter can poll or process a trap from that node using SNMPv3. NerveCenter can discover the node version both automatically or manually. If auto-classification is enabled, then a newly added node (e.g., discovered from a trap, added from a management platform, imported from another NerveCenter installation) will be classified at the highest level possible.

Note: NerveCenter auto-classification is disabled by default. You must enable it before NerveCenter can classify nodes added to its database.

See "SNMP Auto and Manual Classification Settings" on page 103.

Refer to Confirming the SNMP Version for a Node in Designing and Managing Behavior Models for details about classifying nodes manually.

- SNMPv3 operations are logged to a file so that you can follow the progress of v3 activities. The log includes information about activities as well as errors that occur while NerveCenter attempts to perform the activities.

  See "Viewing the SNMPv3 Operations Log" on page 122.

  See "SNMP Error Status" on page 126 for information about SNMPv3 errors.

NerveCenter Support for SNMPv3 Security

SNMPv3 enables devices to communicate in a secure fashion using message authentication to validate users and encryption to provide communication privacy. SNMPv3 provides a User-based Security Model (USM) to establish authentication and secrecy.

SNMPv3 nodes can have one of the following security levels:

- **NoAuthNoPriv** — Neither message authentication nor encryption is used while communicating with the agent. No passwords are required.
- **AuthNoPriv** — Message authentication is used without encryption while communicating with the agent. An authentication protocol and password are required to be set up in agreement on the device and within NerveCenter's definition for the respective node. In NerveCenter The authentication protocol and password to be used can be set to either of the global protocols and passwords User #1 or User #2, or can be set on a per-node basis.
- **AuthPriv** — Both authentication and encryption are used when communicating with the agent. Both the authentication and privacy specifics are required to be set up in agreement on the device and within NerveCenter's definition for the respective node. In NerveCenter the protocols and passwords can be set to either of the global User #1 or User #2 definitions, or can be set on a per-node basis.

Communication between any two SNMPv3 entities takes place on behalf of a uniquely identified domain user. The security level used for this communication defines the security services — message authentication and encryption — used while exchanging data. NerveCenter communicates with SNMPv3 nodes on behalf of the NerveCenter poll user in the poll context.

If you do not specify a security level for an SNMPv3 node, NerveCenter uses the NoAuthNoPriv security level by default, which means that message authentication and encryption services are not used for data exchange with the node.

Note: The NerveCenter poll users (User #1 and User #2), contexts, and the authentication and privacy passwords can be changed in NerveCenter Administrator.

The node-specific information (such as version, security level, and authentication protocol) used to poll each SNMPv3 node is configured in NerveCenter Client. A node-specific poll user and associated credentials can be managed from the SNMP tab on the Node screen (see Changing the Security Level of an SNMPv3 Node).

NerveCenter Support for SNMPv3 Digest Keys and Passwords

SNMPv3 allows two devices to communicate in a secure fashion using message authentication and encryption to ensure secrecy. In any SNMPv3 communication, one of the two communicating entities plays a role of authoritative entity for the communication, and communication is performed on behalf of a unique user within the management domain.

The sender of a secure message attaches a code, or digest, for authentication and encrypts the message to ensure privacy. To generate this digest, the sender uses an authentication key at the authoritative entity of the user on whose behalf communication takes place. Similarly, to encrypt a message, the sender uses a privacy key at the authoritative entity of the user on whose behalf communication takes place. These keys are generated from the authentication password and privacy password, respectively.

SNMPv3 specifications have defined a localized key-generation scheme. For every user, the authentication key at every SNMPv3 entity is a function of the snmpEngineID of that entity, the user's authentication password, and the authentication protocol. For every user, the privacy key at every SNMPv3 entity is a function of the snmpEngineID of that entity, the user's privacy password, and the privacy protocol. NerveCenter supports this localized key-generation scheme.

NerveCenter communicates with SNMPv3 nodes on behalf of a NerveCenter user (User #1, User #2, or a local, node-specific user). NerveCenter needs to know the authentication and privacy passwords for this user to generate the keys required for secure communication. Whenever NerveCenter learns the snmpEngineID of a newly discovered SNMPv3 agent with a security level other than NoAuthNoPriv, NerveCenter generates these keys for the NerveCenter poll user on that agent:

- If authentication is required (a security level of AuthNoPriv is specified for the node), the sender uses the authentication key to generate the digest for the message, which is appended to the message.
- If encryption is required (a security level of AuthPriv is specified for the node), the sender uses the privacy key to generate the digest for the message. For this security level, only the privacy digest is required; privacy assumes authentication, and you cannot have encryption without authentication.

On receipt of a secure message, a receiver does the following:

- Separates the message from the digest (authentication or privacy).
- Uses the corresponding key from its local store to generate the message's local digest copy.
- Compares the local digest with the one received in the message. If the two digests match, the recipient authenticates or decrypts the message using the corresponding local key. If they do not match (indicating a lack of authentication), the recipient discards the message.
- The recipient reads and processes the message.

The Need for Node Classification

Each SNMP agent residing on a network device is represented by a corresponding node object in NerveCenter database. Agent attributes such as IP address, port number, and SNMP version are stored as properties of the corresponding node object in NerveCenter database. Whenever NerveCenter needs to communicate with any SNMP agent it monitors, it uses the required attributes from the corresponding node object in its database. When communicating with any SNMP agent, NerveCenter constructs an

SNMP version-specific message based on the node's version information available in the NerveCenter database. Similarly, when receiving a message from a known agent, NerveCenter uses the version information available in the corresponding node object in its database to decide whether to process the message. To communicate with the agent on a managed device, NerveCenter must use the same SNMP version configured at the agent.

Many SNMP agent implementations support multiple versions of SNMP, as more complex enterprise networks are expected to have agents from multiple vendors supporting varied combinations of SNMP versions. A sophisticated network management product like NerveCenter, therefore, must provide a way to specify the SNMP version to be used when communicating with each of these agents. In very large networks, specifying this version information manually for each device quickly becomes an unmanageable task.

NerveCenter can classify nodes automatically based on SNMP version implementation on the corresponding agent, or you can manually classify nodes on a case-by-case basis. Finally, you can specify the SNMP version to use when NerveCenter communicates with the SNMP agent.

The NerveCenter Node Classification Algorithm

The NerveCenter node classification algorithm is depicted in Figure 17:

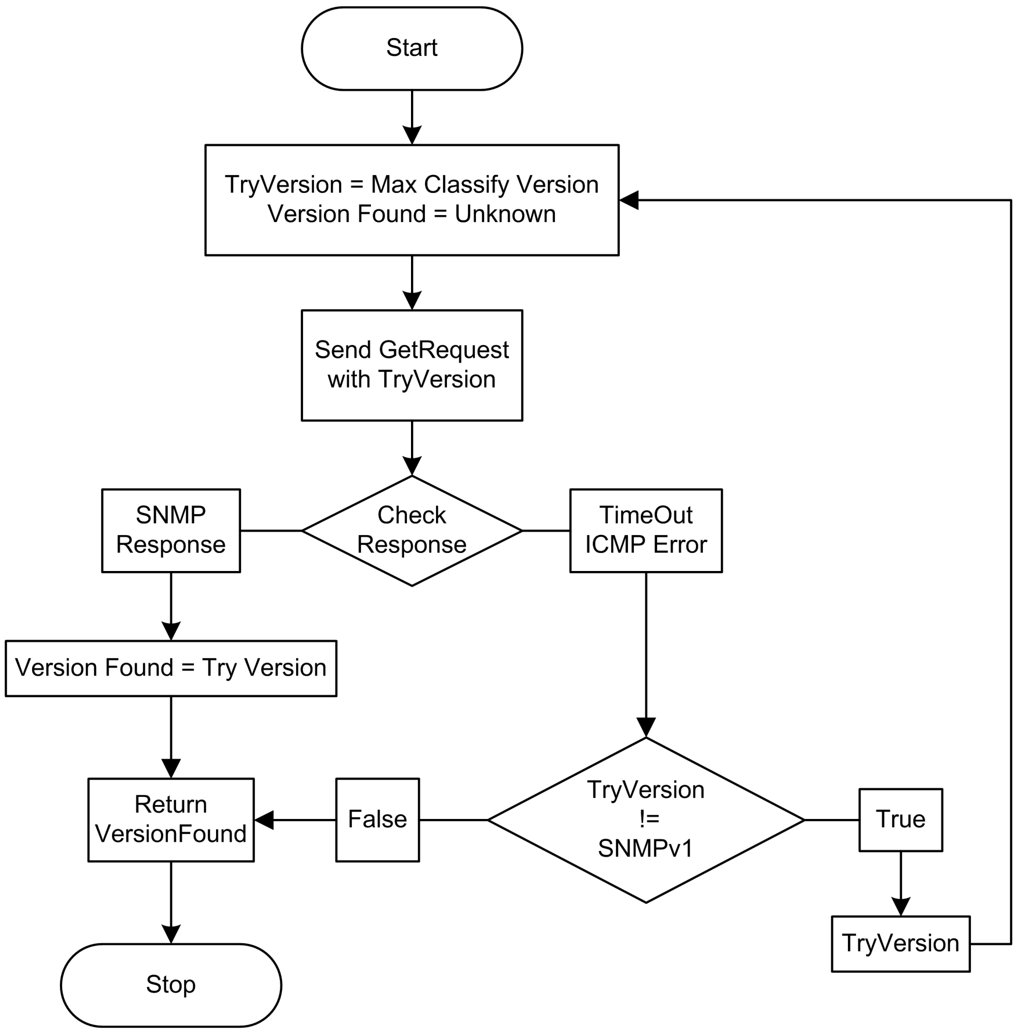


Figure 17: NerveCenter Node Classification Algorithm

How NerveCenter Classifies a Node

NerveCenter tries to classify a node based on the highest version of SNMP implemented on the corresponding SNMP agent. You can, however, limit the highest version NerveCenter attempts to classify in the NerveCenter Administrator. When you specify a maximum classification version, NerveCenter never attempts to classify a node above the version you specified.

When classifying a node, NerveCenter sends an SNMP GetRequest message and listens for the response. Receipt of an SNMP response or SNMP error indicates that the particular version is supported on the corresponding agent. If NerveCenter receives a response other than an SNMP reply, it concludes that the agent does not support that version. To ascertain support for SNMPv1 and SNMPv2c, NerveCenter sends a GetRequest message for SysObjectID with a read community of public.

To ascertain support for SNMPv3, NerveCenter sends a GetRequest message using the SNMPv3 configuration defined for the node. Thus if a node is defined to use SNMPv3 under AuthPriv with MD5 authentication, a digest key of “Kansas”, 3DES privacy, a passkey of “Dorothy”, a user name of “Toto” and an empty context field, all of that information is used in the GetRequest.

As of NerveCenter 5.1.04, there is no longer a default for the Classify operations' SNMPv3 GetRequest. In prior releases a NoAuthNoPriv request was issued under the user name of "initial".

Should the node not have a complete SNMPv3 definition, no SNMPv3 GetRequest is sent and the Classify operation moves to trying SNMPv2. A SNMPv3 configuration is complete if all of the following hold true:

- If the security level is NoAuthNoPriv, then a user name must be specified but the context may be empty.
- If the security level is AuthNoPriv, then the NoAuthNoPriv rules apply, an Authentication protocol must be selected, and the password must be filled in.
- If the security level is AuthPriv, then the AuthNoPriv and NoAuthNoPriv rules apply, the Privacy protocol must be selected, and the password must be filled in.

If NerveCenter does not get SNMP responses or SNMP errors to its GetRequest messages, then NerveCenter designates such nodes as 'unknown'. For details of the algorithm NerveCenter uses to classify nodes, see "The NerveCenter Node Classification Algorithm" on page 101.

You can also test a particular version specified on a node by right-clicking a node in the NerveCenter Client node list view and selecting Test Version. This Test Version is similar to the GetRequest discussed above. If the agent does not support the version specified on the node, a test version failed error message is displayed.

SNMP Auto and Manual Classification Settings

NerveCenter can send requests (manually or automatically) to determine the highest SNMP version supported on a node. When you enable auto-classification, NerveCenter attempts to classify each node automatically when the node is added to the NerveCenter database.

A node must have correct version information, supplied manually or via auto-classification, before NerveCenter can poll or process traps from that node. See the following sections for more information:

- "How NerveCenter Classifies Node SNMP Versions" below
- "When NerveCenter Classifies Node SNMP Versions" on the next page
- "Classifying Nodes Automatically" on page 105
- "Setting a Maximum Classify Value" on page 107

How NerveCenter Classifies Node SNMP Versions

There are two main ways that NerveCenter classifies nodes:

- **Manually** — You can issue a classify command in NerveCenter Client to classify one, several, or all nodes in the database.
- **Automatically** — NerveCenter can be configured to classify nodes when they are added to its database (discovered from a trap, added from a management platform, or imported from another NerveCenter).

Each time NerveCenter attempts to classify a node, it sends a series of classification requests (GetRequest messages) to the node and classifies it based on the responses to those requests. Each request corresponds to an SNMP version — either v1, v2c, or v3.

While classifying a node, NerveCenter attempts to detect the maximum supported SNMP version on the agent up tothe limit configured in NerveCenter Administrator. For example, if you set a maximum classification version of v2c, NerveCenter never attempts to classify nodes any higher than v2c. (However, in the Client module, you can manually specify any version for a node and then test communication with the agent using that version.)

Based on the response to its messages, NerveCenter changes its SNMP version setting for the node.

Caution: Note the following about node classification:
• When NerveCenter attempts to classify a node, NerveCenter's prior SNMP version setting for that node is lost.
• If NerveCenter fails to classify the node, then the node version is set to "Unknown." NerveCenter cannot poll a node with an unknown version.
• A node must have correct version information, either supplied manually or via auto-classification, before NerveCenter can poll or process a trap from the node.

When NerveCenter Classifies Node SNMP Versions

There are two main ways that NerveCenter classifies nodes:

- **On demand** — You can issue a classify command in NerveCenter Client to classify one, several, or all nodes in the database. Procedures for issuing such commands are described in Configuring SNMP Settings for Nodes.
- **Automatically** — You can set up auto-classification in NerveCenter Administrator. When NerveCenter adds nodes to its database (discovered from a trap, added from a management platform, or imported from another NerveCenter), any nodes without version information are classified at the highest possible level. NerveCenter does not attempt auto-classification for nodes that you add manually in Client.

When you enable auto-classification, NerveCenter attempts it in the following instances:

- A node is added through a node file from importutil or from the Client, and the node does not have a version or has the version "Unknown." This would happen, for example, if you were importing the node from a previous version of NerveCenter.
- A node is imported from a NerveCenter Server without a version or with version Unknown.
- A node is added from a trap, and the node's version is not v3. NerveCenter verifies whether these nodes are v1 or v2. If the trap is v3, NerveCenter does not need further verification.
- NerveCenter is co-resident with a network management platform that sends nodes to NerveCenter.

Note: NerveCenter does not attempt to auto-classify nodes that you add manually.

If you choose to disable auto-classification in the NerveCenter Administator, NerveCenter does not poll nodes whose SNMP version is unknown. (You can still classify nodes manually in NerveCenter Client with the available commands.)

Classifying Nodes Automatically

While you can always add and classify nodes manually, NerveCenter can automatically detect agents on network devices it monitors and add the corresponding node objects for those agents. Moreover, NerveCenter can be configured to recognize agents residing on network devices from other network management platforms, another NerveCenter installation, and so on. In such circumstances, if the version information is not available for these automatically detected agents, NerveCenter assigns the version information to corresponding node objects as ‘unknown’. Since NerveCenter does not poll nodes having ‘unknown’ version, its default behavior is to automatically classify these automatically detected agents.

When NerveCenter’s autoclassification feature is enabled, NerveCenter attempts to autoclassify a node in the following situations:

- When nodes are imported via a node file (using ImportUtil or NerveCenter Client), and a node has no SNMP version information or the SNMP version is ‘unknown’. (Such a scenario might occur if you were importing nodes from a previous version of NerveCenter.)
- When nodes are imported from another NerveCenter Server, and a node has no SNMP version or the SNMP version of the node is ‘unknown’.
- When a new node is added to the NerveCenter database because a trap was received from a corresponding agent or source previously unknown to NerveCenter.

NerveCenter auto-classification is enabled for whichever SNMP version you select, but NerveCenter never attempts to auto-classify a node for any version above your selection (see "Setting a Maximum Classify Value" on page 107). For example, if you select v2, NerveCenter can send classification requests for only v1 and v2.

Also, NerveCenter will not attempt to auto-classify a node using SNMPv3 if the node’s v3 definition is incomplete. The server recognizes the cases where a SNMPv3 request simply cannot succeed and in these cases the server skips the v3 testing of the node. Examples of this include a node with no SNMPv3 User Name – a blank user name is not allowed – or where Authentication is indicated but no digest value (passkey) is defined; for these cases the v3 check is skipped by the Classify operation.

Follow the steps below to enable/disable the node auto-classification feature in the NerveCenter Administrator.

TO ENABLE OR DISABLE NODE AUTO-CLASSIFICATION

1. In the NerveCenter Administrator, select the **Classify** tab.
2. To enable auto-classification, check the **Auto Classify** checkbox.

   NerveCenter auto-classification is enabled for whichever version you select, but NerveCenter never attempts to auto-classify a node for any version above your selection. For example, if you select v2, NerveCenter can send classification requests only for v1 and v2.

3. To disable auto-classification, uncheck the **Auto Classify** checkbox.

Note: If you disable auto-classification, bear in mind that NerveCenter does not poll nodes whose SNMP version is unknown.

4. Click **Save**.

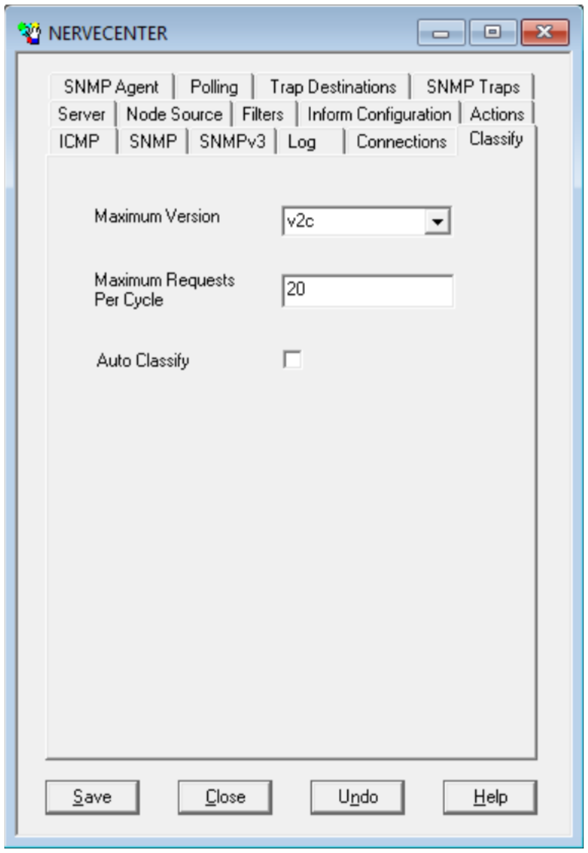
Setting a Maximum Classify Value

Whether using auto-classification or manual node classification (in Client), you specify the highest version you want NerveCenter to detect. When performing classification, NerveCenter will never attempt to classify a node for any version above the version you specify. For example, if you select v2c, NerveCenter can send classification requests only for SNMPv1 and v2c. (However, in the Client module, you can manually specify any version for a node and then test communication with the agent using that version.)

TO SET A MAXIMUM CLASSIFY VALUE

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server. See "Connecting Administrator to a NerveCenter Server" on page 25.
2. Select the **Classify** tab.

   The Classify tab is displayed.



Maximum Version: v2c

Maximum Requests Per Cycle: 20

Auto Classify: ☐

3. Select the highest level you want to classify from the **Maximum Version** list.
4. Click **Save**.

Performance Tuning for Node Classification

SNMPv3-specific operations for large numbers of nodes can cause bursts of traffic on the network. To help you to control this type of sudden SNMPv3-related rise of traffic on the network, you can set the number of requests per cycle and the poll interval.

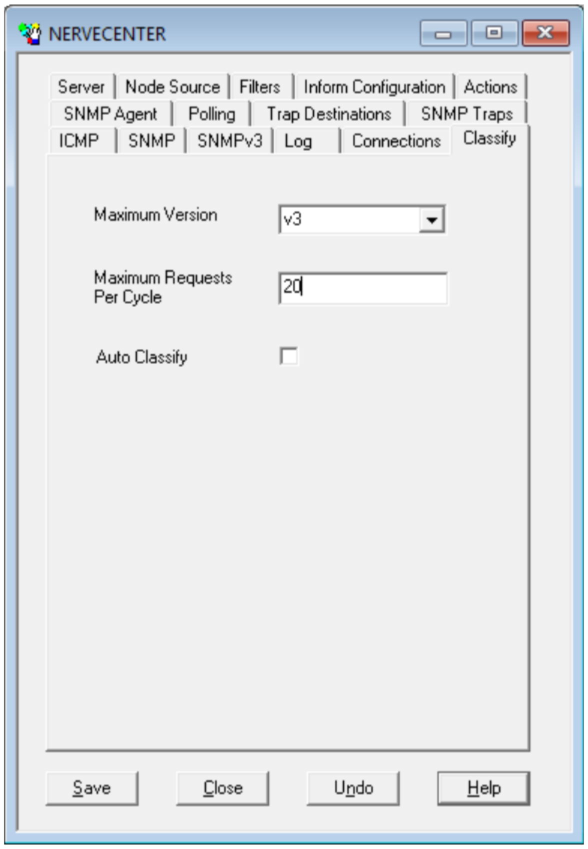
Setting the Maximum SNMPv3 Requests per Cycle

You control SNMPv3 performance by setting a maximum number of requests per processing cycle (approximately one second) for all v3 operations. Consider the number of SNMP messages per second your network devices can handle, and whether messaging uses authentication and privacy keys, which further slows performance.

TO SET THE MAXIMUM NUMBER OF SNMPV3 REQUESTS

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server.
2. Select the **Classify** tab.

   The Classify tab is displayed.


NERVECENTER
Server | Node Source | Filters | Inform Configuration | Actions
SNMP Agent | Polling | Trap Destinations | SNMP Traps
ICMP | SNMP | SNMPv3 | Log | Connections | Classify
Save
Close
Undo
Help

Maximum Version: v3
Maximum Requests Per Cycle: 20
Auto Classify: ☐

3. In the **Maximum Requests Per Cycle** field, enter the maximum number of SNMPv3 requests you want NerveCenter to process. The default value is 20.
4. Click **Save**.

Specifying SNMP Poll Intervals for NerveCenter

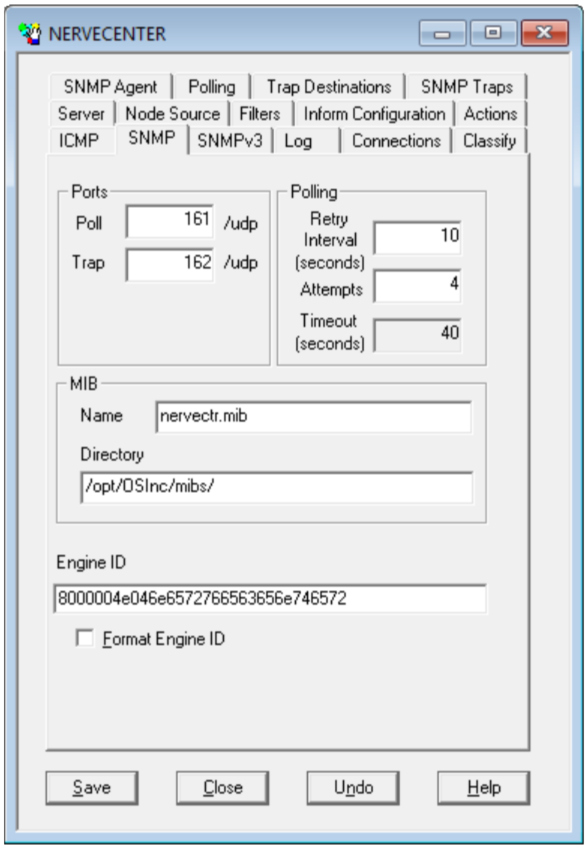
You can change the interval (three attempts by default) at which NerveCenter sends polls to SNMP agents to obtain MIB information. For example, NerveCenter would make four polling attempts to a device that could not respond, the initial attempt and three retries.

The retry interval is significant in that the rate should be high enough to account for the number of retries per interval. For example, if a NerveCenter Server is set to retry three times at a 30 second interval, the polling rate for any of its behavior model's polls should be no lower than 90 seconds.

TO SPECIFY NERVECENTER POLL SETTINGS

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server.
2. Select the **SNMP** tab.

   The SNMP tab is displayed.



Ports
Poll 161 /udp
Trap 162 /udp

Polling
Retry Interval (seconds) 10
Attempts 4
Timeout (seconds) 40

MIB
Name nervectr.mib
Directory /opt/OSInc/mibs/

Engine ID
8000004e046e6572766563656e746572

☐ Format Engine ID

3. In the **Number of retries** field, enter the number of times to reissue unanswered SNMP or ICMP request polls.
4. In the **Retry interval** field, enter the number of seconds NerveCenter should wait for a reply to a poll before issuing another.
5. Select **Save**.

NerveCenter will handle SNMP data according to these settings.

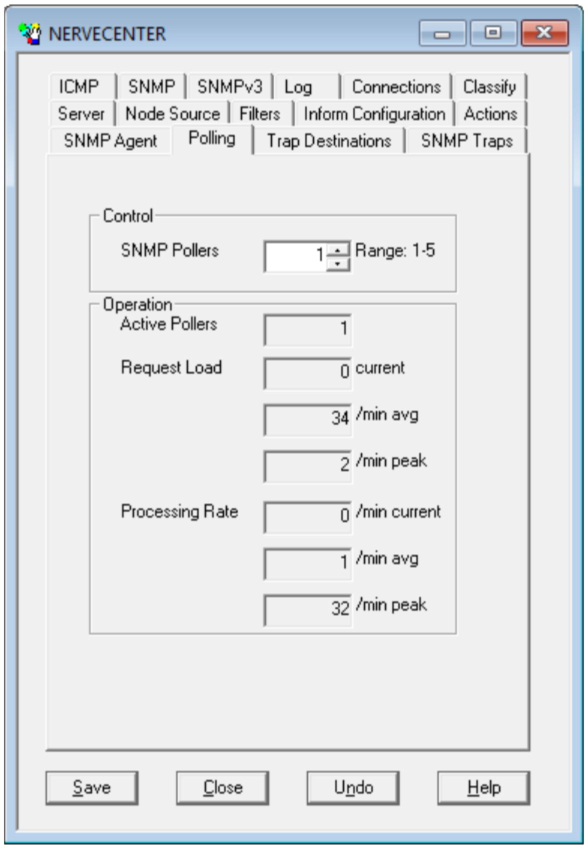
Multi-Threaded Polling

NerveCenter 5.1 introduces support for multi-threaded polling, which can significantly improve performance. Multi-threaded polling is a licensed feature; the License tab of the Server Status window (see 113) displays the number of threads you can use.

TO CONFIGURE MULTI-THREADED POLLING

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server.
2. Select the **Polling** tab.

   The Polling tab is displayed.



Control

SNMP Pollers 1 Range: 1-5

Operation

Active Pollers 1

Request Load 0 current

34 /min avg

2 /min peak

Processing Rate 0 /min current

1 /min avg

32 /min peak

3. Specify the number of polling threads to run in parallel in the **SNMP Pollers** field.
4. Click **Save**.

The Operation section of the Polling tab displays information calculated by the ncserver process and sent to ncadmin. The values are for the entire running of ncserver and are independent of when the ncadmin process connects to the server. (i.e., they are for the life of the ncserver).Table 12 describes the displayed data.

Table 12: Polling Performance Data

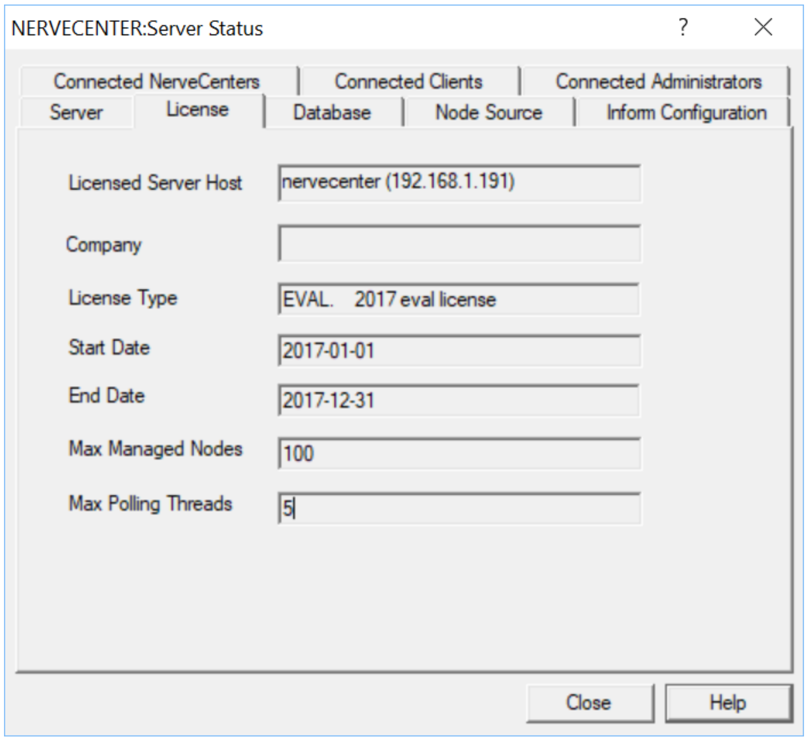
Field		Description
Active Pollers		Displays the number of actively running pollers. This field is useful while system is starting up or adjusting polling rate.
Request Load	current	The number of requests currently being processed by active pollers.
	/min avg	The average number of requests being processed.
	/min peak	The maximum number of requests being processed.
Processing Rate	/min current	The number of requests processed over the last minute.
	/min avg	The average per-minute processing rate.
	/min peak	The maximum per-minute processing rate.

You can view the number of available threads from the License tab of the Server Status window.

To display information about the active server's NerveCenter license

1. Choose **Server Status** from the **Server** menu.
2. Click the **License** tab.

   The License tab is displayed.



Licensed Server Host	nervecenter (192.168.1.191)
Company	
License Type	EVAL. 2017 eval license
Start Date	2017-01-01
End Date	2017-12-31
Max Managed Nodes	100
Max Polling Threads	5

The License tab displays a server's NerveCenter license details; these settings are configured in the hostname.dat license file on the server and cannot be changed here. Table 13 describes the fields on the License tab:

Table 13: Server Status - License Tab Fields

Field	Description
Licensed Server Host	The Server for which the license information is being displayed.
Company	The license owner.
License Type	Indicates if this server is using a standard or evaluation license.
Start Date	Starting date when the license can be used.
End Date	Termination date after which the license is no longer valid.
Max Managed Nodes	The maximum number of nodes that can be managed from this NerveCenter Server.
Max Polling Threads	The number of poll threads that can run in parallel, if the Multi-Threaded Polling feature has been licensed.

Specifying NerveCenter SNMP Ports

NerveCenter has two primary sources of information about network conditions:

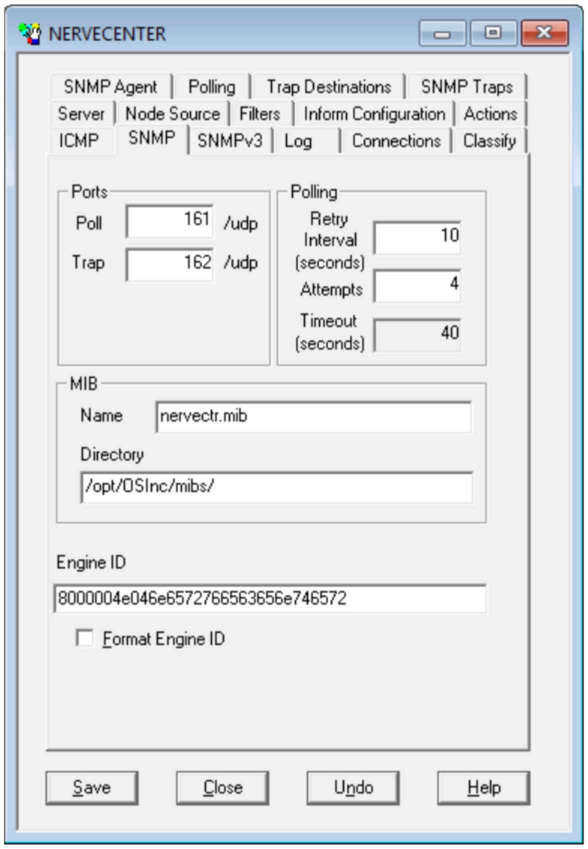
- NerveCenter listens passively for SNMP traps sent by a managed device.
- NerveCenter actively polls the SNMP agents on a managed device.

You can change the ports that NerveCenter uses for receiving traps and sending polls.

To specify SNMP port settings for NerveCenter

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server. See "Connecting Administrator to a NerveCenter Server" on page 25.
2. Select the **SNMP** tab.

   The SNMP tab is displayed.



Ports
Poll 161 /udp
Trap 162 /udp

Polling
Retry Interval (seconds) 10
Attempts 4
Timeout (seconds) 40

MIB
Name nervectr.mib
Directory /opt/OSInc/mibs/

Engine ID
8000004e046e6572766563656e746572

☐ Format Engine ID

3. In the **SNMP Poll** field, enter the number of the port you want NerveCenter to use to communicate with SNMP agents. This port is used to get or set SNMP information.

   The value entered here specifies the port on the node to which NerveCenter sends SNMP polls. You can change the port for any particular node in the node's definition window in NerveCenter Client. It is suggested that you keep the default port number 161.
4. In the **SNMP Trap** field, enter the port number that you want NerveCenter to use for receiving SNMP traps. The default port number is 162.
5. Click **Save**.

Discovery and Initialization of SNMPv3 Agents

For nodes with SNMPv3 agents, NerveCenter requires certain information before the node can be discovered and managed. The process of obtaining this information is call initialization, which occurs in two cases:

- After installation to obtain engine information required for polling.
- When a node's version is changed to SNMPv3 from another version.

Before NerveCenter can initialize a node, its SNMPv3 agent must be configured with the poll user, poll context, and security level.

- See "Configuring SNMPv3 Security Settings" on page 118 for information about the poll user, poll context, and security level.
- See "Configuring an SNMPv3 Agent for NerveCenter" below for configuration details.

Initialization consists of the following:

1. Obtaining the agent's snmpEngineID value for SNMPv3 node discovery. See "Configuring an Initial User for Discovering an SNMPv3 Agent" on the facing page for details.
2. Obtaining additional engine information if required for the chosen level of security.

   NerveCenter requires additional engine information when the NerveCenter user has a security level other than NoAuthNoPriv. In this case, NerveCenter must obtain SNMPv3 agent's snmpEngineBoots and snmpEngineTime values.

Note: SNMPv3 specifications allow secure initial agent configuration wherein an agent need not disclose its snmpEngineID. Network management applications must obtain on their own the snmpEngineID for such agents. NerveCenter cannot initialize or poll such agents.

Configuring an SNMPv3 Agent for NerveCenter

Before NerveCenter can poll SNMPv3 agents, those agents must be configured to support one of the two global NerveCenter users (User #1 or User #2), or configured for a node-specific local user.

User #1 and User #2 definitions are set up through the NerveCenter Administrator. The values supplied for these two definitions are called "global" because nodes that are configured to use either of these two definitions *share* these values — a change to any field within User #1 or User #2 is immediate and effective on *all* nodes that are configured to use that definition.

As an alternative to this shared, global usage mechanism, nodes can be configured with a *local* SNMPv3 definition. The NerveCenter Client is used to set which SNMPv3 definition it will use (User #1, User #2, or Local) and to set the values for a Local definition. Nodes in Local mode have their own private SNMPv3 attributes.

Nodes that reference User #1 or User #2 can be set to access a given security level from the chosen global definition. SNMPv3 polls are formed by NerveCenter using as much of the definition that applies to the selected security level set for each node.

Two passwords (authentication and privacy) and a context can be configured for each NerveCenter user (#1, #2, and Local). You must supply the authentication and privacy passwords for the NerveCenter user configured on your devices. .

Table 14 describes the configuration options for SNMPv3 security.

Table 14: SNMPv3 Agent Configuration

Field	Description
User #1 User User #2 User	String containing the name of the SNMPv3 user account referred to as the global User #1 or User #2 accounts.
The following items can be configured for each user:	
Authentication Protocol	Authentication Protocol (MD5 or SHA-1) for each user.
Authentication Key	Click **Set Auth Key** to enter the password to generate an Authentication Key.
Privacy Protocol	Privacy Protocol (DES, 3-DES, AES-128, AES-192, or AES-256) for each user.
Privacy Key	Click **Set Priv Key** to enter the password to generate a Privacy Key.
Context	This field contains the context string for the account. This information is required only if a context is defined on the SNMPv3 device.

NerveCenter then polls the SNMPv3 agent on behalf of the poll user in this context. In addition to the poll user and context, the parameters described in the following sections must be configured on the SNMPv3 agent.

Note: The following descriptions serve only as guidelines for configuring SNMPv3 agents for NerveCenter. For exact procedural details about configuring SNMPv3 agents, consult the user documentation supplied by your SNMPv3 agent provider.

Configuring an Initial User for Discovering an SNMPv3 Agent

To discover an SNMPv3 node, NerveCenter sends a GetRequest message to that node with the following parameters:

- SecurityLevel: set to (noAuthNoPriv)
- MsgUserName: set to the user name provided by the node's SNMPv3 configuration
- MsgAuthoritativeEngineID: set to zerolength (' ')
- Empty variable bindings (variable-bindings=' ')

Note: For exact procedural details about configuring SNMPv3 agents, consult the user documentation supplied by your SNMPv3 agent provider.

Properly configured SNMPv3 agents respond to GetRequest with a Report PDU, which includes its local snmpEngineID (supplied within the msgAuthoritativeEngineID field). After NerveCenter obtains the snmpEngineID, NerveCenter can discover the node as an SNMPv3 agent.

This process of discovering SNMPv3 agents is recommended in SNMPv3 specifications. Some SNMPv3 agents are preconfigured with an “initial” user and respond to the GetRequest as described without any special type of configuration — see documentation supplied by the SNMPv3 agent provider for detailed information.

When a node is created in NerveCenter (via the NerveCenterAdministrator application), its SNMPv3 “Select User” toggle is set to “User #1” by default. Thus, for most incoming nodes it will be the case that the discovery and classify operations will attempt to use the User Name field from User #1 to contact the SNMP Agent to see whether it can get an snmpEngineID.

If User #1 has an empty User Name field, then the NerveCenter server simply skips any SNMPv3 discovery on the node because SNMPv3 does not allow an empty MsgUserName field to be sent. If the user puts the word “initial” into the User #1 User Name field, then that would be the value attempted for cases where a node is newly received and a discovery or classify is attempted.

NerveCenter can communicate with devices that support any available version of SNMP: v1, v2c, or v3. All SNMP agents and network devices managed by NerveCenter are represented as nodes in the database; the database also stores the SNMP version assigned to each node. Such SNMP version information can be assigned manually or automatically.

Configuring SNMPv3 Security Settings

Before NerveCenter can poll SNMPv3 agents, the agents must be configured to support a NerveCenter user. There are two global accounts, User #1 and User #2 that can be configured within NerveCenter. You can also define accounts on a per-node basis from the Node screen in NerveCenter Client.

After installation, you can change the passwords as appropriate for your network management strategy. Each NerveCenter user can be assigned authentication and privacy protocols, each with its own password-generated key. The security level determines whether passwords are needed:

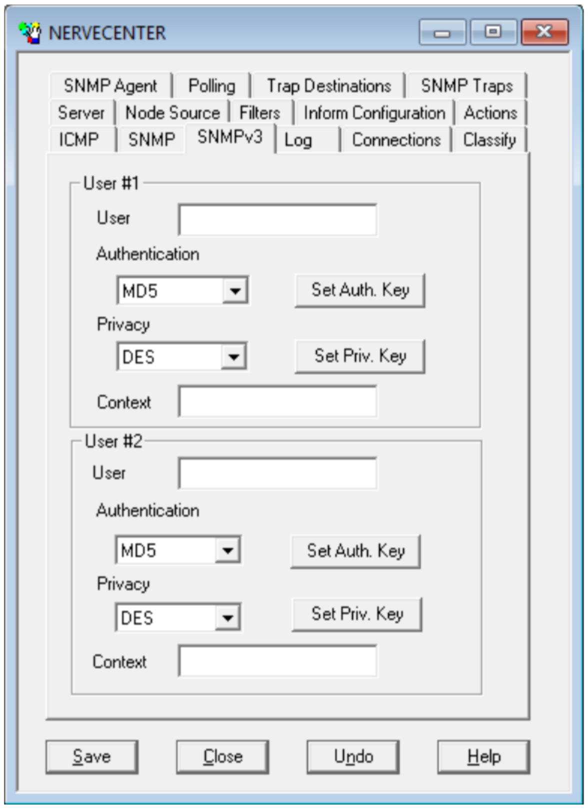
- Authentication password is required for AuthNoPriv and AuthPriv security.
- The Privacy password is required for AuthPriv security.

NerveCenter itself does not require an SNMPv3 context; you may need to enter it to support individual agents that may require it. If you change these values on your agents, you must make the same change in NerveCenter Administrator. If you use contexts, changing context on a regular basis can help ensure privacy; also, if you have multiple NerveCenters managing different device categories, you can associate different user names and contexts with each category.

TO CONFIGURE SYSTEM-WIDE SNMPV3 SECURITY

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server. Select the **SNMPv3** tab.

   The SNMPv3 tab is displayed.

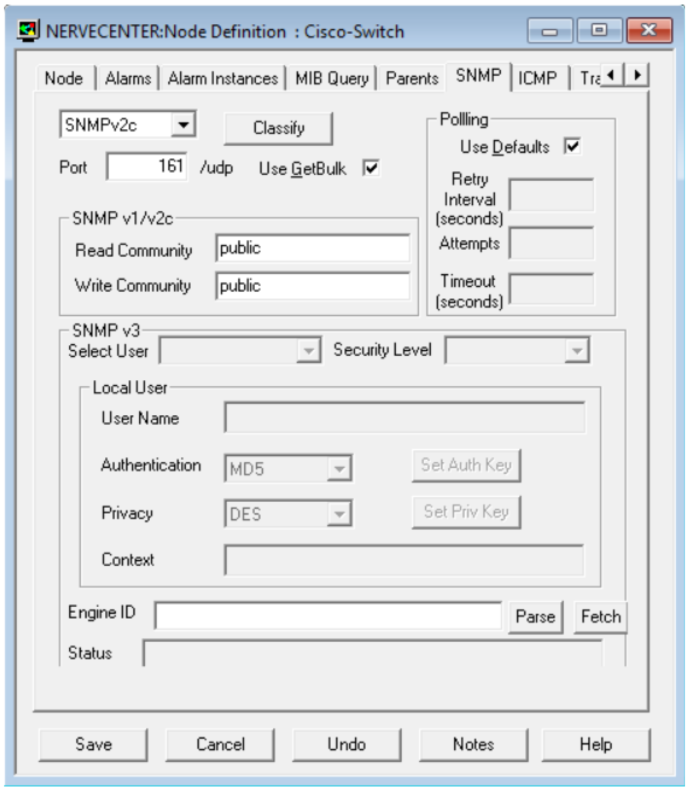


2. Edit the **User #1** or **User#2** account names if necessary.
3. Configure the following for each user as necessary:
   a. Select the **Authentication** protocol (MD5 or SHA-1).
   b. Click **Set Auth.** Key, enter the password in each field, and click **OK**.
   c. Select the **Privacy** protocol (DES, 3DES, AES-128, AES-192, or AES-256).
   d. Click **Set Priv.** Key, enter the password in each field, and click **OK**.
   e. If an agent configuration requires it, enter the **Context** for the account.
4. Click **Save**.

To configure node-specific SNMPv3 settings

1. Select the node in the NerveCenter Client.
2. In the Node Definition dialog, select the **SNMP** tab.

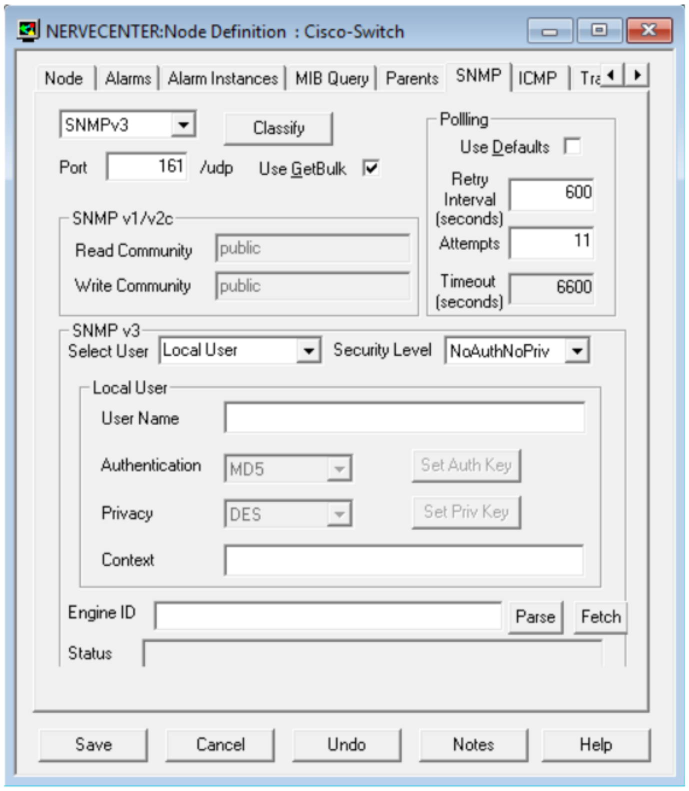
   The SNMP Tab is displayed.



NERVECENTER:Node Definition : Cisco-Switch

SNMPv2c
Port 161 /udp
Use GetBulk ✓
Use Defaults ✓
Retry Interval (seconds)
Attempts
Timeout (seconds)
Read Community public
Write Community public
Select User
Security Level
User Name
Authentication MD5
Privacy DES
Context
Engine ID
Status

3. In the upper-left corner of the tab, select **SNMPv3** from the list.
4. In the **SNMPv3** area, select **Local User** from the Select User list.



SNMPv3

Port 161 /udp Use GetBulk ☑

SNMP v1/v2c
Read Community: public
Write Community: public

Polling
Use Defaults ☐
Retry Interval (seconds): 600
Attempts: 11
Timeout (seconds): 6600

SNMP v3
Select User: Local User
Security Level: NoAuthNoPriv

Local User
User Name:
Authentication: MD5
Privacy: DES
Context:

Engine ID:
Status:

5. Select the node **Security Level** from the list.
6. In the Local User area:
   a. Enter the **User Name** to use for the node.
   b. Optionally, enter a **Context** value for the node. (This is empty by default is and is only needed for special cases within SNMPv3.)
   c. Configure the Local User **Authentication** and **Privacy** as necessary:
      - For **AuthNoPriv** or **AuthPriv** security, select the **Authentication** protocol (MD5 or SHA-1) and click **Set Auth Key** to set the authentication password.
      - For **AuthPriv** security, you must also select the **Privacy** protocol (DES, 3DES, AES128, AES192, or AES256) and click **Set Priv Key** to set the privacy password.
7. Select **Save** to make this configuration permanent in the NerveCenter node management database.

Viewing the SNMPv3 Operations Log

Whenever a NerveCenter Server receives a request for an SNMPv3 operation or an error occurs while attempting to perform an SNMPv3 operation (e.g., v3 initialization fails), the NerveCenter Server logs a message to V3Messages.log, which resides in the NerveCenter installation log directory on the NerveCenter Server host machine. The file contains messages about SNMPv3 operations and errors resulting from requests that originate with any connected NerveCenter Clients, Administrators, and Command Line interfaces.

After logging the error, the NerveCenter Server notifies all connected NerveCenter Clients and Administrators in the following ways:

- If you are logged on to the NerveCenter Client or Administrator that initiated the operation that caused an error condition, NerveCenter displays the error that was logged.
- If you are logged on to a NerveCenter Client or Administrator that did not initiate the error condition, a red icon appears in the status bar; double-click the icon to display the NerveCenter Server with the SNMPv3 error. If your Client or Administrator is connected to more than one Server, the dialog box lists all servers that currently have an error condition.

When your NerveCenter Client or Administrator displays a dialog box with an error condition, you can do either of the following:

- Acknowledge the error condition by “signing the log.” When you sign the log, NerveCenter notes that in the log file and changes the red icon back to green for all connected Clients and Administrators.
- Dismiss the dialog box without acknowledging the error condition, in which case only the icon in your Client or Administrator turns green. The icon remains red for all other connected Clients and Administrators to signal that the NerveCenter Server has an unacknowledged/unsigned error. Moreover, the Server does not indicate acknowledgment in the log file.

You must have administrator rights to initiate an SNMPv3 operation that can result in an error or to acknowledge a logged error condition. If you are logged on with only user rights, you can dismiss the error dialog box but not acknowledge an error condition.

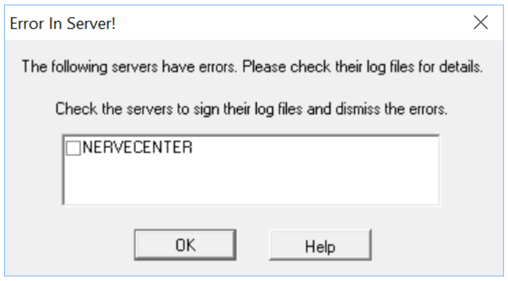
Whether you acknowledge or dismiss the error, all messages remain in the V3Messages.log.

For more information, refer to the following topics:

- "Signing a Log for SNMPv3 Errors Associated with Your Client" on the facing page
- "Signing a Log for SNMPv3 Errors Associated with a Remote Client or Administrator" on page 124
- "Viewing the SNMPv3 Operations Log" on page 125

Signing a Log for SNMPv3 Errors Associated with Your Client

Whenever an SNMPv3 operation is requested or an error occurs while attempting an SNMPv3 operation, the NerveCenter Server logs a message to V3Messages.log. If you are logged in to the NerveCenter Client that initiated the logged request, NerveCenter displays a dialog box with that error.


Error In Server!

The following servers have errors. Please check their log files for details.

Check the servers to sign their log files and dismiss the errors.

☐ NERVECENTER

Figure 18: Operations Log Error in Server Dialog Box for Your Client

Users with administrator rights can acknowledge a logged condition from NerveCenter Client by signing the Operations log. Signing the log causes the icon to turn green in all connected Clients/Administrators.

You can also dismiss the dialog box without acknowledging the error condition. If you are logged on with user rights rather than administrator rights, your only option is to dismiss the dialog box; you cannot sign the Operations log..

To sign the Operations log

1. After viewing the message that NerveCenter displays on your screen, check the Sign the log and dismiss errors checkbox.
2. Click **OK**.

   The icon in the Status Bar turns green for all Clients or Administrators connected to the designated NerveCenter Server. You can later view this message again in the Operations log.

This V3Messages.log file resides in the NerveCenter installation log directory. The file can be viewed in a text editor or word processor.

To dismiss the Error in Server dialog box

- Click **OK** without checking the checkbox.

  In this case, only the icon in your Client turns green. For all other connected Clients and Administrators, the icon remains red and signals to those modules that the NerveCenter Server has some error that remains unacknowledged.

Signing a Log for SNMPv3 Errors Associated with a Remote Client or Administrator

Whenever an error occurs while attempting an SNMPv3 operation, the NerveCenter Server logs a message to V3Messages.log. If you are logged on to a remote NerveCenter Client (one that did not initiate the error condition), the status bar displays a red icon.

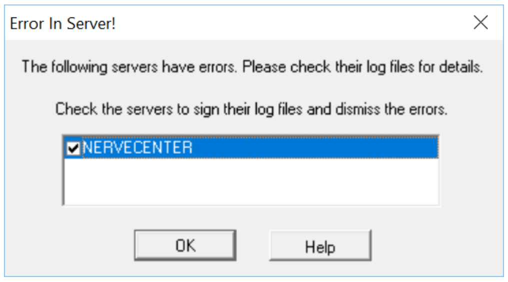
Users with administrator rights can acknowledge a logged condition from NerveCenter Client by signing the Operations log. Signing the log causes the status icon to turn green in all connected Clients/Administrators.

You can also dismiss the dialog box without acknowledging the error condition. If you are logged on with user rights rather than administrator rights, your only option is to dismiss the dialog box; you cannot sign the Operations log..

To sign the Operations log

1. Double-click the red icon in the Status Bar.

   The Error In Server dialog box is displayed.



Error In Server!

The following servers have errors. Please check their log files for details.

Check the servers to sign their log files and dismiss the errors.

☑ NERVECENTER

2. Check the NerveCenter Server or Servers for which you want to sign the log.
3. Click **OK**.

   The icon in the Status Bar turns green for all Clients or Administrators connected to the servers you checked. At a suitable time, you can open the Operations log and view the new message. This file, named V3Messages.log, resides in the NerveCenter installation log directory. The file can be viewed in a text editor or word processor.

TO DISMISS THE ERROR IN SERVER DIALOG BOX

1. Double-click the red icon in the Status Bar.

   The Error In Server dialog box is displayed.

2. Click **OK** without checking any of the checkboxes.

   In this case, only the icon in your Client turns green. For all other connected Clients and Administrators, the icon remains red and signals to those modules that the NerveCenter Server has some error that remains unacknowledged.

Viewing the SNMPv3 Operations Log

Whenever an SNMPv3 operation is requested or an error occurs while attempting the operation, the NerveCenter Server logs a message to the V3Messages.log file, which resides in the NerveCenter installation log directory on the NerveCenter Server host machine.

The file can be viewed in a text editor or word processor. As NerveCenter adds more messages to the file, the file continues to grow until you manually remove old messages.

The log entries resemble the following:

```
 06/20/2017 09:26:29 Tue - Event ID : NC_SERVER; Category ID : NC_
THREAD_V3OP; /
     Error Status : AutoClassifyFail; Error while communicating
using SNMPv1 for 10.52.174.51 /
     because of : NC_PORT_UNREACHABLE;
```

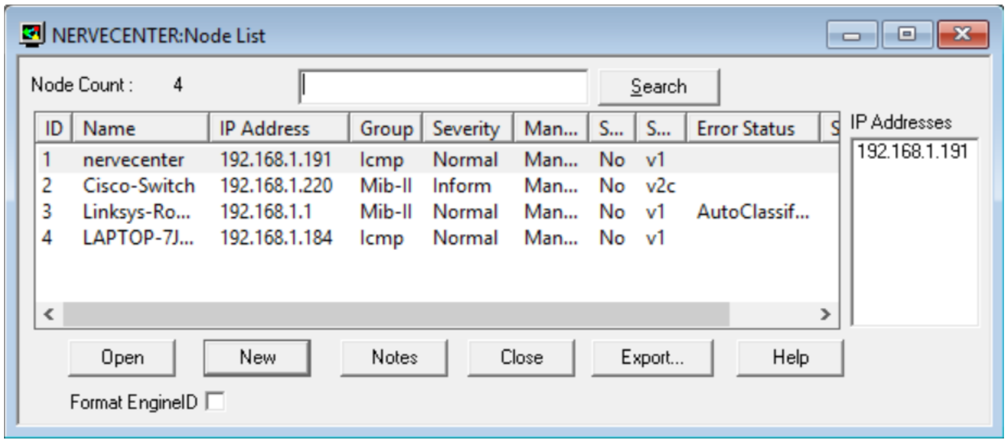
Following are the fields in the log:

Table 15: Fields in the Operations Log

Field	Description
Date/Time	Date and time the record was logged. The format is month/day/year, hour/minute/second, and day (for example, 12/16/2017 11:32:29 Sat).
EventID	This always NC_SERVER.
CategoryID	Name of the thread where the event occurred.
Error Status	One of several error status strings. See *Error Status* for a description of SNMPv3 error status messages and which ones cause polling to stop for a node.
Error Description	Details of the error or operation.

SNMP Error Status

When NerveCenter is unable to complete an SNMP operation on a node, the error status is displayed in the Node List (NerveCenter Client and Web Client) and in the SNMP tab of the node's definition window (NerveCenter Client). Figure 19 shows the Node List window in the Client.


NERVECENTER:Node List

Node Count : 4

ID	Name	IP Address	Group	Severity	Man...	S...	S...	Error Status
1	nervecenter	192.168.1.191	Icmp	Normal	Man...	No	v1	
2	Cisco-Switch	192.168.1.220	Mib-II	Inform	Man...	No	v2c	
3	Linksys-Ro...	192.168.1.1	Mib-II	Normal	Man...	No	v1	AutoClassif...
4	LAPTOP-7J...	192.168.1.184	Icmp	Normal	Man...	No	v1	

IP Addresses
192.168.1.191

Format EngineID

Figure 19: Node List Window

Though most of the error strings correspond to SNMPv3 errors, some are applicable for v1 and v2c errors as well. These are noted in the descriptions below.

Sometimes error conditions can be corrected simply by running the SNMP Test Version poll. Others may require configuration changes to the node's SNMP agent. After changing the configuration of an SNMP agent, always test communication with the node in NerveCenter Client prior to polling the node.

The following list describes each possible SNMP error status.

- **V3InitFail** – An attempt to get the snmpEngine ID of an SNMPv3 agent failed or the SNMPv3 configuration defined for that node is causing a failure at the SNMPv3 communication layer. This can occur either when NerveCenter first attempts to poll the node using the SNMPv3 configuration or at any point when the SNMP agent changes its SNMPv3 configuration. For all of these cases, the V3InitFail is augmented by one of the following values in the SNMPv3 Status field (NerveCenter Client):
  - ConfigurationError – The node's SNMP definition is incomplete with respect to its Security Level. This status is discovered and reported by NerveCenter before issuing an SNMPv3 request to an SNMP Agent.

    Operator intervention is required. The node's SNMP v3 definition must contain a User Name regardless of the Security Level — AuthNoPriv requires an Authentication Protocol and Password; AuthPriv requires Authentication and Privacy Protocols and passwords for each.
  - UnknownUsername – The SNMP Agent reports that the SNMPv3 User Name being sent by NerveCenter is not one of the user names that it has been configured to handle.
  - UnknownContext – The SNMP Agent reports that the SNMPv3 Context being sent by NerveCenter is not appropriate. Many SNMP Agents do not report this value, even if it is the underlying issue. Instead, the SNMP Agent may not issue any response and the operation will time out.

    - UnavailableContext – The SNMP Agent reports that the SNMPv3 Context being sent by NerveCenter is known but inapplicable to the operation (poll, discovery, or classification) being attempted. Many SNMP Agents do not report this value, even if it is the underlying issue. Instead, the SNMP Agent may not issue any response and the operation will time out.
    - UnsupportedSecLevel – The SNMP Agent reports that it cannot handle the Security Level defined in a request issued to it by NerveCenter.
    - UnknownEngineID – Either NerveCenter's SNMP Stack or the SNMP Agent is reporting an issue with the snmpEngineID used for SNMP v3 communication. This can occur if the snmpEngineID is changed on the SNMP Agent during polling.
    - IncorrectAuthPasskey – The SNMP Agent reports that the Authentication passkey (digest) being issued by NerveCenter is not correct. This generally occurs in one of two cases: 1) An incorrect password was entered either on the SNMP Agent or in NerveCenter, or 2) The password was entered correctly at both ends, but the selected Authentication protocol is mismatched between the SNMP Agent and NerveCenter.
- **ClassifyFail** – An attempt to obtain the node's SNMP version failed during a classification attempt. The node's version will be set to "Unknown" and it will not be polled. You can manually change the version or try to classify the node again.
- **AutoClassifyFail** – An auto-classification attempt failed to obtain the node's version. The node's version will be changed to "Unknown" and it will not be polled. You can manually change the version or try to classify the node again.

Note: ClassifyFail and AutoClassifyFail status values are not limited to SNMPv3 agents. If NerveCenter attempts to classify an agent and fails for some reason (e.g., the agent is down), NerveCenter will mark the node with ClassifyFail or AutoClassifyFail regardless of the SNMP version supported on the agent.

- **TestVersionFail** – At attempt to poll the SNMP agent failed. The Test Version poll sends a GetRequest message for a node based on the SNMP version configured for that node.

  If the Test Version poll fails, polling will not happen for this node. In that case, you may need to reconfigure the agent on this node. Then, try running the Test Version poll again (from a node's definition window or the right-click menu in the node list).

Note: TestVersionFail is not limited to SNMPv3 agents. You can test the version of any SNMP agent with this feature.

- **Configuration Mismatch** – Indicates an SNMP trap was received but there is some problem with the configuration on the agent. If NerveCenter is unable to decode a trap due to some unspecified reason (e.g., unsupported authentication or privacy parameters on the agent or an incorrect NerveCenter user name), NerveCenter can receive the trap and add the node to its database if configured to discover nodes via traps. After adding the node to its database, however, NerveCenter assigns an error status of Configuration Mismatch.

Note: Any error that occurs during trap decoding always results in a Configuration Mismatch error.

- **TimeSyncFail** – An attempt to get the node’s snmpEngine boots/timeticks failed. Polling will continue for this node. If any polls successfully reach the node, the node responds with an “Out of time window” report PDU that contains the correct boots/timeticks, and NerveCenter can then update this information for the node. For the initial polls that generate the report PDU, the SNMP_ NOT_IN_TIME_WINDOW trigger will be fired.
  - You can ignore this message, which simply indicates that NerveCenter is getting in sync with that node. You can recover from this error status by right-clicking the node in the Node List and selecting v3TestPoll. If the agent corresponding to the node is up, the test poll should be successful and clear the error message. The SNMPv3 Status field will be set to the following:
- **NotInTimeWindow** – This is the reply sent by the SNMP Agent or declared by NerveCenter’s SNMP stack upon investigating a request or response PDU wherein the SNMPv3 timestamp handling shows a time sync failure.

Managing the SNMP Agent

8

NerveCenter includes a Master SNMP Agent and an SNMP Sub-Agent so that you can monitor NerveCenter with a third-party SNMP application. The data provided by the NerveCenter SNMP Agent is described in the MIB modules located in `InstallPath/mibs/nsmg`.

When you install NerveCenter, you also install the SNMP Master and Sub Agents, though they are disabled by default. You need to first configure the SNMP Master Agent and then enable it before it can process SNMP requests.

The procedure to set up and then start the SNMP Master Agent and the NerveCenter SNMP Sub Agent differs depending on whether you run the NerveCenter Service under the super-user account or a regular user account.

- "Configuring the SNMP Agent and Sub Agent (super-user)" on the next page
- "Configuring the SNMP Agent and Sub Agent (normal user)" on page 132

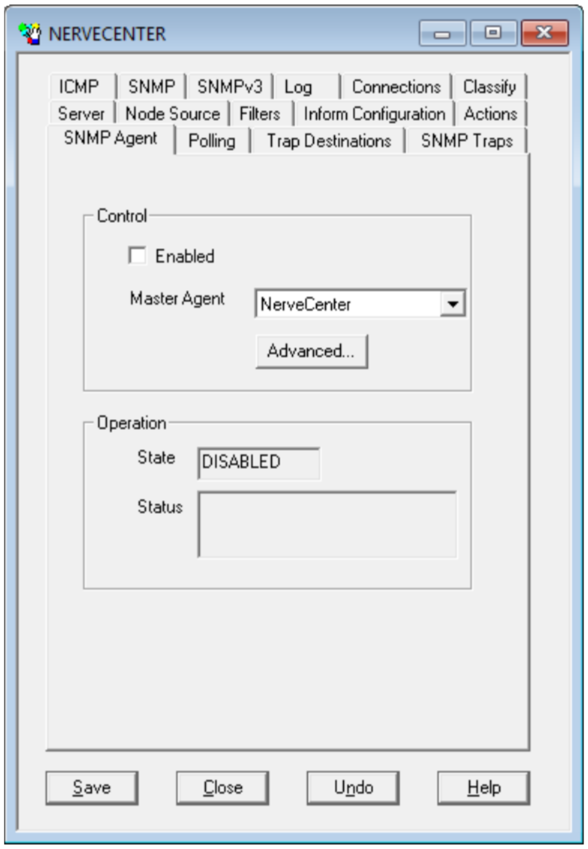
Configuring the SNMP Agent and Sub Agent (super-user)

The following procedures how to enable and disable the SNMP Agent via a super-user account.

To configure the NerveCenterSNMP Agent

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server. See "Connecting Administrator to a NerveCenter Server" on page 25.
2. Select the **SNMP Agent** tab.

   The SNMP Agent tab is displayed.



Enabled [ ]

Master Agent: NerveCenter

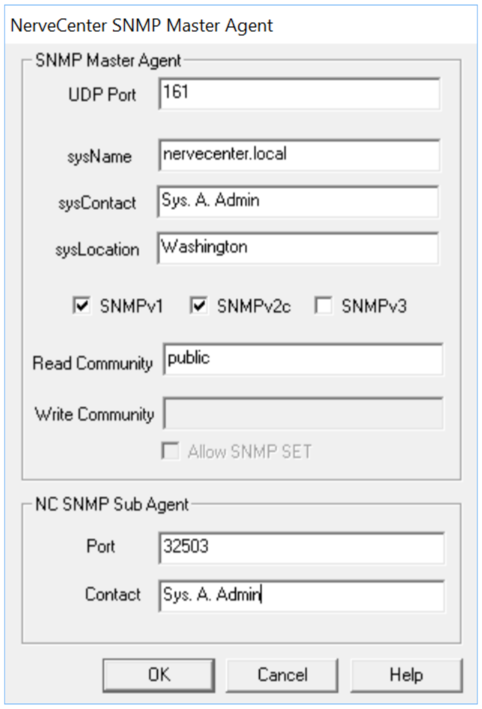
State: DISABLED

Status:

3. Select NerveCenter from the Master Agent list.

4. Click **Advanced**.

   The NerveCenter Master Agent dialog box is displayed.



NerveCenter SNMP Master Agent

SNMP Master Agent

UDP Port: 161

sysName: nervecenter.local

sysContact: Sys. A. Admin

sysLocation: Washington

[✓] SNMPv1 [✓] SNMPv2c [ ] SNMPv3

Read Community: public

Write Community:

[ ] Allow SNMP SET

NC SNMP Sub Agent

Port: 32503

Contact: Sys. A. Admin

5. Enter the **UDP Port** on which you want the SNMP Master Agent to listen for SNMP messages.
6. Enter the **sysName** (system name), **sysContact** (system contact), and **sysLocation** (system location) in the appropriate fields.
7. Select the versions of SNMP you want to support.
8. If you select SNMPv1 or SNMPv2c, enter **Read Community** and **Write Community** strings.
9. Click **OK**.

   You return to the SNMP Agent tab.

10. Select the **Enabled** check box.

Note: To enable the SNMP Agent on UNIX, the NerveCenter server must be running as a process with root privileges.

11. Click **Save**.

If you want to limit network traffic, or have no more need of the SNMP Agent, you can turn it off.

To disable the NerveCenter SNMP Agent (super-user)

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server. See "Connecting Administrator to a NerveCenter Server" on page 25.
2. Select the **SNMP Agent** tab.

   The SNMP Agent tab is displayed.
3. Clear the **Enabled** check box.
4. Click **Save**.

Note: You may also disable the Master Agent permanently using the command

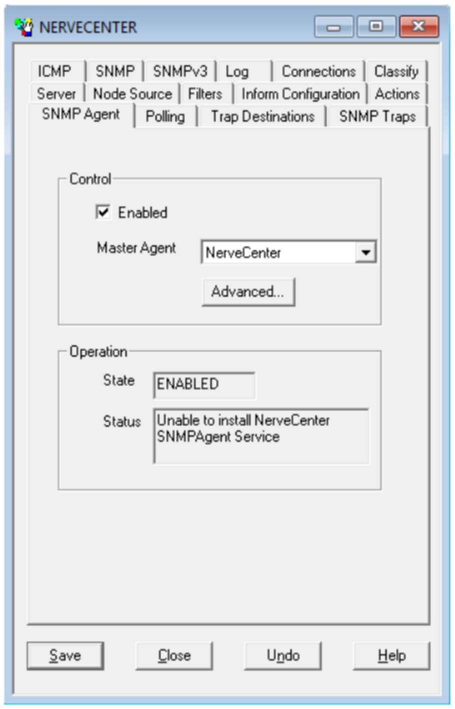
```
# /opt/OSInc/nc/install/ncsnmpdm uninstall
```

Configuring the SNMP Agent and Sub Agent (normal user)

The following procedures describe how enable the SNMP Agent via a normal user account, and how to disable it.

To Configure The NerveCenter SNMP Master Agent and Sub Agent via normal accounts.

1. Open the NerveCenter Administrator and connect to the appropriate NerveCenter Server.
2. Select the **SNMP Agent** tab.


NERVECENTER

Control

☑ Enabled

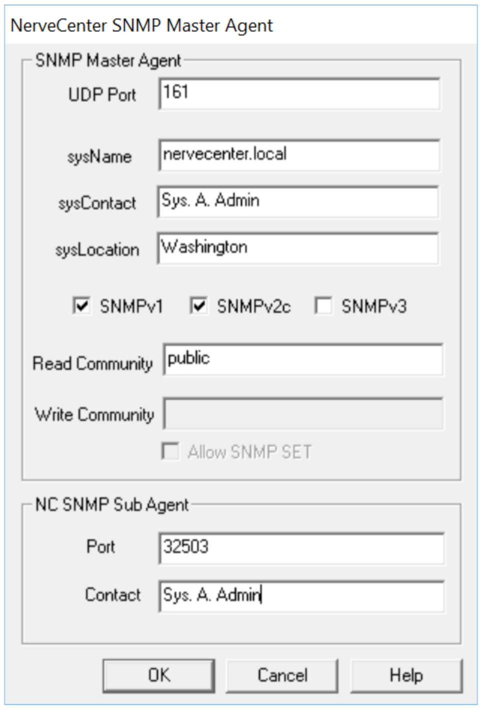
Master Agent: NerveCenter

Operation

State: ENABLED

Status: Unable to install NerveCenter SNMPAgent Service

3. Select **NerveCenter** from the **Master Agent** list.
4. Click **Advanced**.


NerveCenter SNMP Master Agent

SNMP Master Agent

UDP Port: 161

sysName: nervecenter.local

sysContact: Sys. A. Admin

sysLocation: Washington

☑ SNMPv1 ☑ SNMPv2c ☐ SNMPv3

Read Community: public

Write Community:

☐ Allow SNMP SET

NC SNMP Sub Agent

Port: 32503

Contact: Sys. A. Admin

a. Fill out the fields for the **SNMP Master Agent**.

b. Fill out the fields for the **SNMP Sub Agent**.

   Modify the **Port** if necessary (32503 is the default) and enter the **Contact** person(s) responsible for the NerveCenter service on this host.

c. Click **OK** to return to the SNMP Agent tab.

5. Select the **Enabled** checkbox.
6. Select **Save**.

At this point, the service is enabled but cannot start — regular accounts are not permitted to start a service that listens to protected services (UDP port 162 needed by the Master Agent) — so attempts to start it will display an *Unable to install NerveCenter SNMP Agent Service* error. A super-user must perform a one-time procedure on the NerveCenter Server host.

TO START THE SNMP AGENT SERVICE VIA THE SUPER-USER ACCOUNT

1. Open a terminal session on the NerveCenter Service host and access the root account

```
# cd /opt/OSInc/nc/install
# ./ncsnmpdm isstarted
  not started
# ./ncsnmpdm isinstalled
  not install
# ./ncsnmpdm install
# ./ncsnmpdm isinstalled
  installed
# ./ncsnmpdm start
  started (10881)
# ./ncsnmpdm status
  snmpdm (pid 10881) is running…
```

2. Return to the NerveCenter Administrator application.

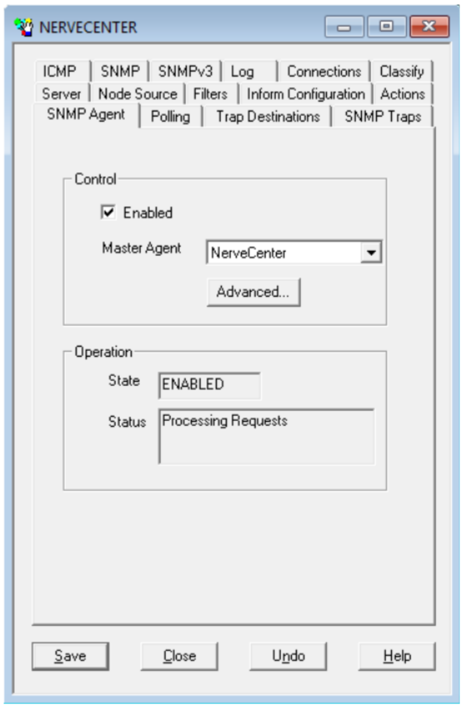
   At this point the Master Agent is running and will automatically restart if the NerveCenter server is rebooted. The NerveCenter Administrator program may now start and stop the Sub Agent.

3. On the SNMP Agent tab, deselect the **Enabled** checkbox and then select **Save**.

   The NerveCenter Server's state is reset, though the Master Agent will still be running.

4. Select the **Enabled** check box again and then select **Save**.

   The NerveCenter Server starts the Sub Agent.



☑ Enabled

Master Agent: NerveCenter

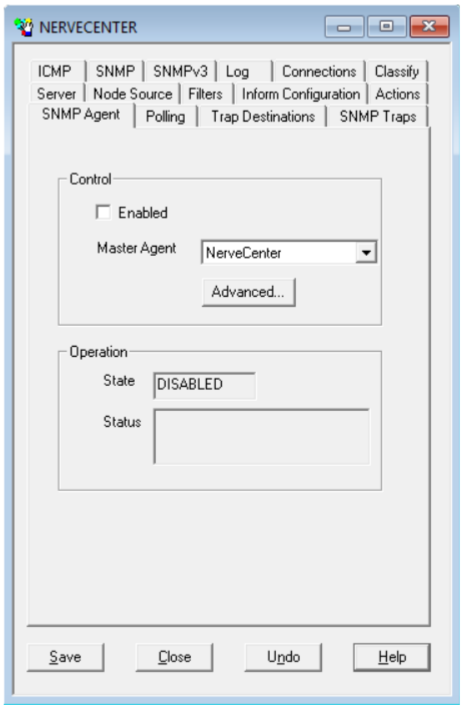
State: ENABLED

Status: Processing Requests

To Disable the NerveCenter SNMP Sub Agent (normal user)

1. Open the NerveCenter Administrator and connect to the appropriate NerveCenter Server.
2. Select the **SNMP Agent** tab.
3. Clear the **Enabled** check box.
4. Select **Save**.

   The NerveCenter Server stops the Sub Agent.


NERVECENTER
ICMP | SNMP | SNMPv3 | Log | Connections | Classify
Server | Node Source | Filters | Inform Configuration | Actions
SNMP Agent | Polling | Trap Destinations | SNMP Traps
Advanced...
Save
Close
Undo
Help

Control

☐ Enabled

Master Agent: NerveCenter

Operation

State: DISABLED

Status:

5. To stop the Master Agent, a super-user must open a terminal session on the NerveCenter Server host and run the following command

```
# /opt/OSInc/nc/install/ncsnmpdm stop
```

Managing ICMP Settings

9

This chapter explains how NerveCenter uses ICMP and the controls available for configuring and monitoring this protocol within NerveCenter.

ICMP Support Overview

ICMP is supported in NerveCenter as a protocol that can be utilized for both monitoring and polling. ICMPv4 and ICMPv6 as defined in RFC792 and RFC1442 respectively, are internal communication protocols embedded within the IP layer of the TCP/IP protocol suite. They are most often employed by internal elements for configuration modification.

The following are the features available for accessing ICMP through NerveCenter

- Nodes can be “pinged” (ICMP Echo / Echo Response) by NerveCenter. The response, or lack thereof, can be used to drive NerveCenter models in the same way done with SNMP polling.
- ICMP messages can be monitored and filtered in the same manner as SNMP Traps. Through the use of NerveCenter Trap Masks, triggers can be created to drive NerveCenter models in the same way as performed with SNMP Notifications.

NerveCenter supports ICMPv4 polling and ICMPv4/v6 monitoring.

ICMP Polling Support

NerveCenter provides a means of accessing the ICMP “ping” operation. The manner of this interface is consistent with the product's SNMP access. A special Property “nl-ping”, defined by one of the MIB Modules provided with NerveCenter, is used to indicate that a poll is to perform an ICMP “ping”. The poll condition then handles examination of the results of the operation, much as NerveCenter handles SNMP polls. This mechanism includes special built-in triggers specific to ICMP polling. See Defining Property Groups and Properties, Using Polls, and Using Other Data Sources , and in Designing and Managing Behavior Models.

ICMP Event Monitoring Support

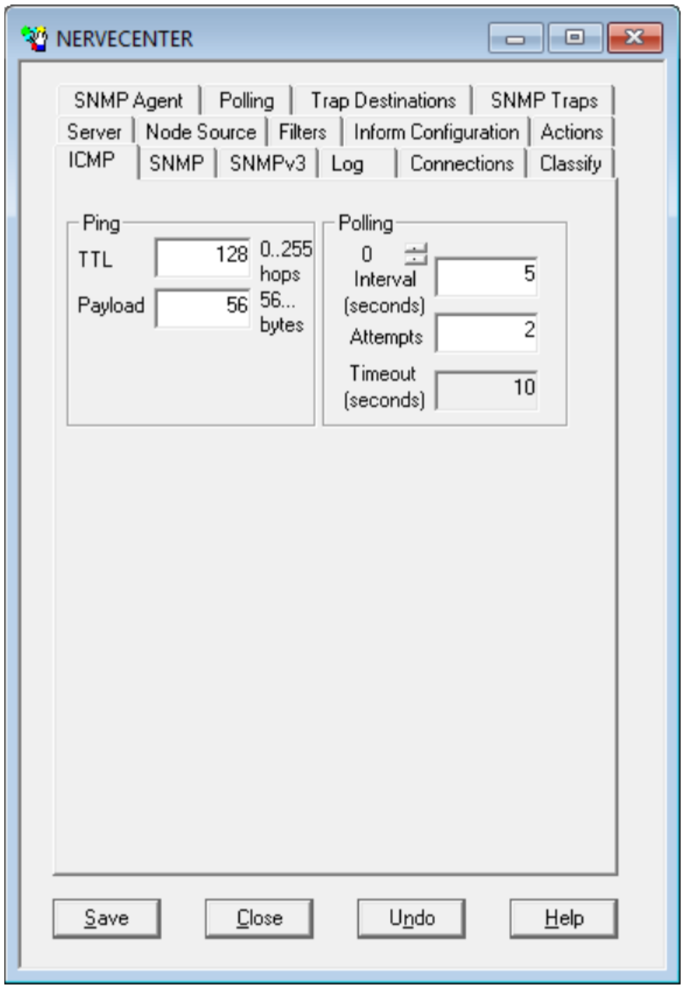
NerveCenter can optionally be configured to treat incoming ICMP messages as events. This feature permits Trap Mask definitions to be created which filter ICMP alerts to the messages of interest. Use of the Trap Mask mechanism for this filtering makes the ability uniform to the event monitoring NerveCenter provides for SNMP. See Using Trap Masks in Designing and Managing Behavior Models.

Configuring ICMP Ping and Polling

ICMP operates much like UDP in that requests posted onto a network might not be responded to. In this case an overall polling operation needs to allow for retries and a per-attempt time interval before declaring the poll to have timed out, and must also be able to define the ping characteristics. As with SNMP polling, NerveCenter supports both system-wide and node-specific ICMP configurations.

TO CONFIGURE SYSTEM-WIDE ICMP SETTINGS

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server.
2. Select the **ICMP** tab.



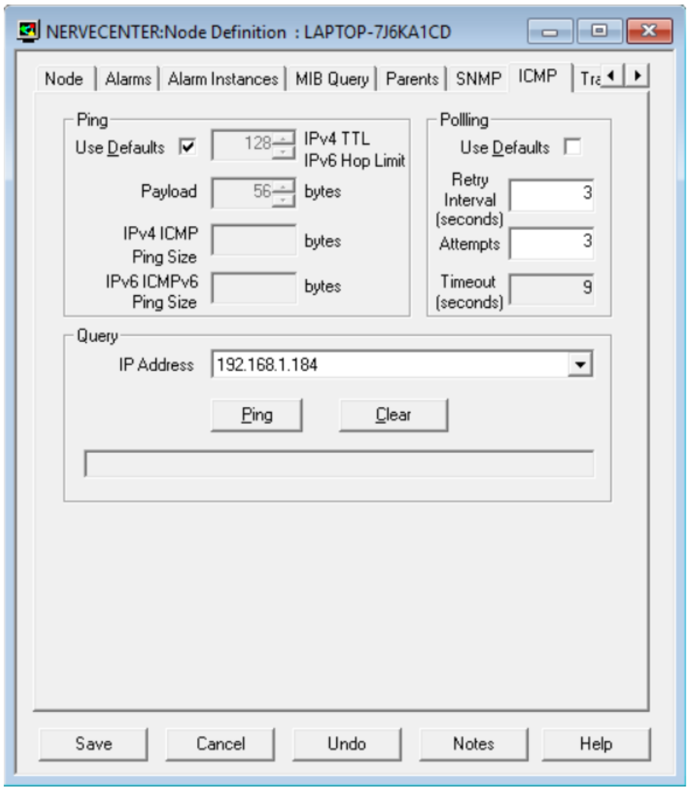
Ping
TTL 128 0..255 hops
Payload 56 56... bytes

Polling
0
Interval (seconds) 5
Attempts 2
Timeout (seconds) 10

3. In the **Ping** section, modify the **TTL** and **Payload** fields as necessary.
4. In the **Polling** section, modify the **Attempts** and **Interval** fields as necessary.
5. Select **Save**.

TO CONFIGURE NODE-SPECIFIC ICMP SETTINGS

1. From the NerveCenter Client, select the node you wish to configure.
2. Open the Node Definition dialog and select the **ICMP** tab.



Ping: Use Defaults ✓; IPv4 TTL / IPv6 Hop Limit 128; Payload 56 bytes

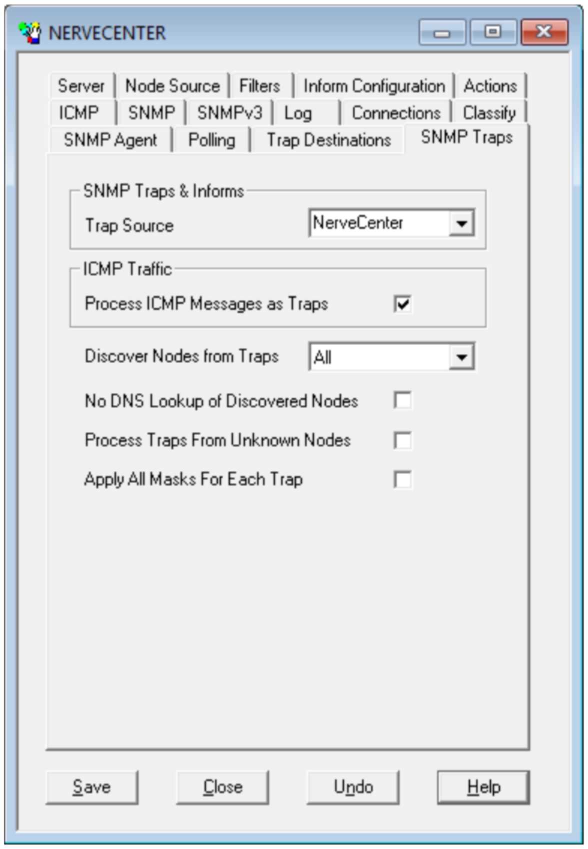
Polling: Use Defaults ☐; Retry Interval (seconds) 3; Attempts 3; Timeout (seconds) 9

Query: IP Address 192.168.1.184

3. To modify **Ping** settings, clear the Use Defaults checkbox and modify the **TTL/Hop Limit** and **Payload** fields as necessary.
4. To modify **Polling** settings, clear the **Use Defaults** checkbox and modify the **Retry Interval** and **Attempts** fields as necessary.
5. Select **Save**.

Configuring NerveCenter for ICMP Event Monitoring

NerveCenter Administrator contains a **Process ICMP Messages as Traps** option on its **SNMP Trap** tab. This control is used to enable/disable ICMP monitoring. ICMP monitoring is enabled only when this setting has been selected and the setting saved to the connected/managed NerveCenter Server. The parameter is persistent over restarts of the NerveCenter Server and does not require a restart of NerveCenter Server for a change to come into effect.


NERVECENTER
Server | Node Source | Filters | Inform Configuration | Actions
ICMP | SNMP | SNMPv3 | Log | Connections | Classify
SNMP Agent | Polling | Trap Destinations | SNMP Traps
Save
Close
Undo
Help

SNMP Traps & Informs
Trap Source: NerveCenter

ICMP Traffic
Process ICMP Messages as Traps [✓]

Discover Nodes from Traps: All
No DNS Lookup of Discovered Nodes [ ]
Process Traps From Unknown Nodes [ ]
Apply All Masks For Each Trap [ ]

When the control is enabled, all received ICMP messages are transformed into SNMPv1 Traps. Any applicable Trap Mask will then be applied to the event. The definition of Trigger Functions can be done in a manner entirely consistent with that used for SNMP event monitoring. The activation of triggers works identically. See Using Trap Masks in Designing and Managing Behavior Models.

When the control is disabled, NerveCenter Server continues to receive ICMP messages but the messages are processed as poll results (etc.) instead of being transformed into traps or forwarded to the user's trap masks.

Managing NerveCenter Security

NerveCenter security allows access for two different groups. A NerveCenter administrator must add users to an appropriate user group before they can access NerveCenter.

The NerveCenter Server authenticates users against the host’s Pluggable Authentication Module (PAM) configuration. Membership in the appropriate groups determines what privileges users have within NerveCenter once they log in successfully.

Note: The User ID under which the NerveCenter Server runs must be a member of the NerveCenter Admins group.

Members of the NerveCenter Admins group (ncadmins) can configure NerveCenter and create or modify behavior models. Members of the NerveCenter Users group can monitor network status and reset alarms. For more details about privileges, see the section "NerveCenter Login Rights" on page 21.

Managing Security on UNIX

User logon attempts to a UNIX- or Linux-based NerveCenter Server are authenticated before access is permitted. Authentication checking occurs when the user attempts to log on with the NerveCenter Administrator or Client applications, the nccmd utility, or the NerveCenter API. Authentication is performed whether the user is running the application on the same host as the NerveCenter Server or is accessing the NerveCenter Server remotely.

NerveCenter integrates with the UNIX or Linux host’s Pluggable Authentication Module (PAM) subsystem to provide authentication. PAM allows NerveCenter logins to be based on the user accounts of the underlying operating system and subject to the authentication rules set up for that host.

To successfully access a NerveCenter Server, a login attempt must satisfy the following:

- The given username must match an existing user account
- The password must be valid for the given username
- The login attempt must be accepted by the host's PAM configuration as defined for the “nervecenter” service
- The account must be a member of either the NerveCenter Administrators (ncadmins) or NerveCenter Users (ncusers) group

NerveCenter does not require special treatment under PAM. The NerveCenter Server accesses PAM using the service name “nervecenter”, under which Administrators create specialized handling for NerveCenter access. If no specialized handling is in place, PAM defaults to matching NerveCenter authentication and account checking against the rules specified for the service name “other” or “OTHER”, depending on the rules by which the host's PAM subsystem operates. PAM is implemented and managed differently by each Linux and UNIX vendor; the network, host, or security administrator(s) for each NerveCenter Server will need to be aware of how PAM operates on their systems and configure it appropriately.

PAM configuration is required only on the Linux or UNIX system where the NerveCenter Server is installed. The NerveCenter Client, NerveCenter Administrator, and other NerveCenter components do not require PAM to be configured, as they use the NerveCenter Server for user account access functions. The user login functionality is handled only by the NerveCenter Server which a user is attempting to access, and the NerveCenter Server's PAM integration determines whether the login is successful.

NerveCenter's PAM integration provides administrators with more control over account access, allowing them to:

- Integrate account control with LDAP or NIS systems
- Select non-default password encryption methods
- Lock out user access with the /etc/nologin file
- Lock out users with expired passwords or blocked accounts

Such features are dependent upon the PAM implementation on the NerveCenter Server host. For example, Blowfish password encryption is supported on Solaris, but not on Red Hat Enterprise Linux.

The NerveCenter installer for Linux and Solaris contains a PAM configuration section that briefly describes the host's PAM configuration with respect to NerveCenter. The installer registers NerveCenter with PAM under the service name “nervecenter”, through which PAM can provide enhanced authentication processing. (Examples of other service names are “passwd”, “login”, “rlogin” and “ssh”, which correspond to the Linux/UNIX applications of the same name; a service name does not need to match the application name.) NerveCenter’s PAM configuration controls access for any NerveCenter component attempting to log on to the NerveCenter Server. PAM authentication methods for the “nervecenter” service name are configured through a “nervecenter” file in the /etc/pam.d directory (Linux and Solaris 11) or with “nervecenter” entries in the /etc/pam.conf file (Solaris 10+). If no “nervecenter” configuration is provided, all systems authenticate using the rules for the default service name “other” or “OTHER”.

The installer places several sample configuration files in the /opt/OSInc/nc/install directory. Additionally, for Linux systems, a “nervecenter” file with a simple default setup is placed in the /etc/pam.d directory if such a directory exists and there is not already a “nervecenter” file present. For Solaris systems, no change is made to the pam.conf file; this is left to your system's administrators.

There is no user access to the PAM configuration from within NerveCenter. The NerveCenter Server accesses PAM to perform authentications and account checking but cannot otherwise alter, view, or reveal the PAM configuration. Users that log in to the NerveCenter Server cannot access their account, its definition, or alter aspects of it (changing an account password is not possible, for example). While PAM can handle different account- and access-related services, NerveCenter accesses the authentication and account management services only. The keywords PAM uses for these services are “auth” and “account” respectively. Thus the “nervecenter” configuration only needs to provide handling for these two services.

Troubleshooting: NerveCenter Connection Issues II

Note: Before following the procedures in this section, you should work through the steps in "Troubleshooting: Connecting to the NerveCenter Server" on page 32.

NCTESTLOGIN RUNS SUCCESSFULLY BUT ADMINISTRATOR/CLIENT ACCESS STILL FAILS

The next step is running the NerveCenter Server with the `–pamdebug` flag, which writes the same `nctestlogin` output to `/var/opt/NerveCenter/log/ncsecurity.log`, but with additional information that may assist in diagnosing the problem.

TO START NERVECENTER WITH THE -PAMDEBUG FLAG

1. Stop NerveCenter:

   ```
   /opt/OSInc/bin/ncstop
   ```

2. Start NerveCenter:

   ```
   /opt/OSInc/bin/ncstart -pamdebug
   ```

3. Attempt the Administrator/Client connection with the same credentials.

On some systems, only accounts with super-user privileges may access PAM API features, in which case you can run the `ncsecurity` application as root via the following commands:

```
# cd /opt/OSInc/nc/bin/
# chown root:ncadmins ncsecurity
# chmod 4755 ncsecurity
or by running the general file ownership and permissions
configuration script ‘ncpermissions’:
# /opt/OSInc/nc/bin/ncpermissions
```

All NerveCenter login attempts are written to `syslog` as a member of the `daemon` facility and messages are sent with the `Warning` priority. The UNIX system administrator can use these values to help configure `syslog` with regards to NerveCenter messaging.

When a user successfully logs in, `syslog` displays a message similar to

```
May 22 16:23:56 bluejay NerveCenter: [ID 651326 daemon.warning]
CATEGORY: 100  “nervectr” login granted with “ncadmins” access.
```

or

```
May 22 16:23:56 bluejay NerveCenter: [ID 651326 daemon.warning]
CATEGORY: 100  “nervectr” login granted with “ncusers” access.
```

When a login attempt fails, `syslog` displays a message similar to

```
May 22 16:23:04 bluejay NerveCenter: [ID 651326 daemon.warning]
CATEGORY: 100  login rejected for “nervectr”
```

When using -`pamdebug`, syslog may contain the following types of additional messages.

- `... login rejected for “nervectr”: unknown user`

  The UNIX host does not recognize the `nervectr` user account.

- `... login rejected for “roger”: PAM authentication check fails`

  In this case the UNIX host recognized the `roger` account, but the PAM Authentication service has rejected the login attempt. This corresponds to PAM auth entries for the `nervecenter` or `other` configuration, and in most cases this reflects an incorrect password.

- `... login rejected for “dharry”: PAM account management check fails`

  The login attempt was rejected by the PAM Account Management service. This corresponds to PAM account entries for the `nervecenter` or `other` configuration. These rejections are usually related to issues such a presence of a `/etc/nologin` file, a blocked account, an expired password, etc..

- `“... login rejected for “tracy”: no group membership`

  The login attempt was denied because the user `tracy` is not a member of the `ncadmins` or `ncusers` groups. However groups are defined on the UNIX host, the current setup does not include the user as required for NerveCenter Server access.

- `“... login rejected for “matthew”: failure accessing groups`

  The login attempt was denied because the `ncadmins` or `ncusers` groups are not defined.

Troubleshooting: Managing NerveCenter Security

The following list contains some common security problems NerveCenter users face. For UNIX-specific tips, see "Troubleshooting: NerveCenter Connection Issues II" on page 143.

USERS CANNOT CONNECT TO THE SERVER

Problem: Users are not members of the appropriate NerveCenter user group on the server machine (ncadmins or ncusers on UNIX or NerveCenter Admins or NerveCenter Users on Windows).

Solution: Check group membership. If users are not members of the appropriate groups, add them. Remember that on Windows, users must log out for changes to take place.

See "Managing Security on UNIX" on page 141.

Problem: Users do not have a valid Windows account.

Solution: Check the user account. See your Windows documentation for more details.

Problem: The server is not running.

Solution: Restart the server.

See "Running the NerveCenter Server" on page 23

USERS IN NCADMINS OR NCUSERS ON UNIX CANNOT CONNECT TO THE SERVER

Problem: ncadmins and ncusers have been created on a NerveCenter host that is also an NIS client.

Solution: When running NerveCenter on an NIS client, the groups ncadmins and ncusers must reside on the NIS Master. Local NerveCenter groups are ignored when the local system is an NIS client. Authentication is done on the NIS Master server. When running in an NIS environment, make sure the NerveCenter groups are defined on the NIS master. The user community members must be members of those groups as well.

See Installing NerveCenter.

USER CAN MONITOR ALARMS, BUT CANT MODIFY BEHAVIOR MODELS

Problem: Users who are members of ncusers (UNIX) or NerveCenter Users (Windows) can only monitor and reset alarms. To be able to modify NerveCenter objects, users must be members of ncadmins (UNIX) or NerveCenter Admins (Windows).

Solution: Add users to the appropriate groups.

See "Managing Security on UNIX" on page 141.

Managing NerveCenter Alarm Actions

10

Alarm actions are important components of NerveCenter behavior models. The user supplies most of the data needed to perform the alarm action when the behavior model is created. However, before some alarm actions can be performed, an administrator must configure particular settings.

Specifying an SMTP Server for Mail Notification

In the event of noteworthy network conditions, NerveCenter can send email alerts.NerveCenter currently offers two options for having a behavior model send an e-mail message:

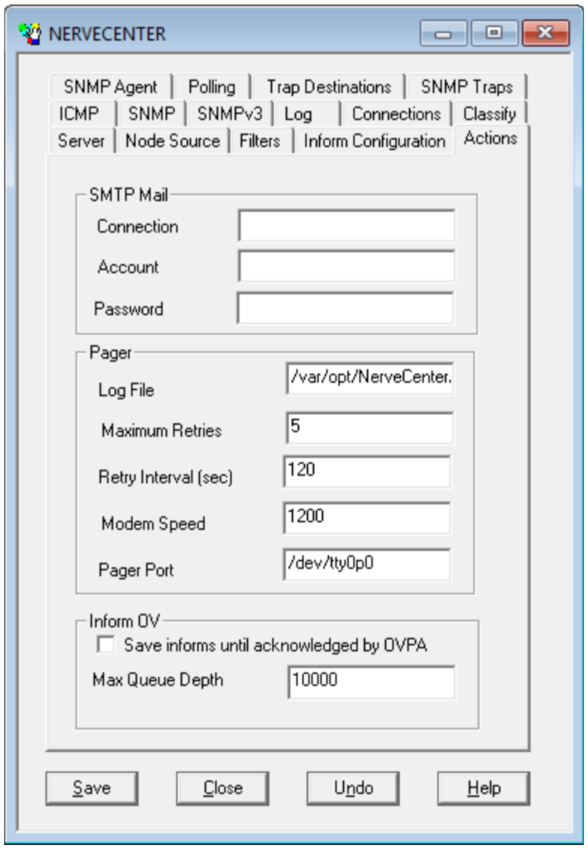
- With an SMTP server.
- With an SMS message.

The recipient or recipients of the SMTP mail message is specified when the behavior model is created. However, before this message can be sent, the administrator must specify which SMTP server NerveCenter uses to send the message.

TO SPECIFY A SMTP SERVER FOR MAIL NOTIFICATION

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server. See "Connecting Administrator to a NerveCenter Server" on page 25.
2. Select the **Actions** tab.

   The Actions tab is displayed.



SMTP Mail

Connection	
Account	
Password	

Pager

Log File	/var/opt/NerveCenter.
Maximum Retries	5
Retry Interval (sec)	120
Modem Speed	1200
Pager Port	/dev/tty0p0

Inform OV

☐ Save informs until acknowledged by OVPA

Max Queue Depth 10000

3. In the **SMTP Mail** section, enter the SMTP host in the **Connection** field, along with the **Account** and **Password** that will be used to send mail/SMS messages.

   NerveCenter will use this account for both sending mail and posting SMS messages. In both cases NerveCenter connects on-demand to the SMTP service, sends the email or message, and then disconnects.

   Connection may be a hostname or IP address of the SMTP service host, optionally with a port number appended; TCP ports 25 and 587 are the most common, though you should verify the port with your SMTP service configuration.

   The values provided in the three SMTP Mail fields are stored on the NerveCenter Server host in `/opt/OSInc/conf/nervecenter.xml`; the password is encrypted.

4. Select **Save**.

   Users can now include a SMTP Mail action in their behavior models.

Specifying Settings for Log Management

NerveCenter allows behavior models to log network data via the Log to File alarm action, which writes information about an alarm transition to an ASCII text file. When a behavior model is created, the user specifies the name of the log as well as the type of data that will be recorded. However, an administrator must specify how NerveCenter will manage these logs.

NerveCenter logs have changed over the course of several releases. Some of the more significant characteristics of NerveCenter logs include:

- Time fields appear in the following format:

  ```
  mm/dd/yy hh.mm.ss day
  ```

  For example, `10/04/98 12.03.44 Wed`

- NerveCenter will delimit fields with a semi-colon
- No spaces will appear around the equal sign (**=**)
- Values will appear in the following format:

  ```
  attribute.instance=value
  ```

  For example, `ifInOctets.3=5`

- NerveCenter will print all fields when a default format is chosen. For example, the same trap in verbose and nonverbose format:

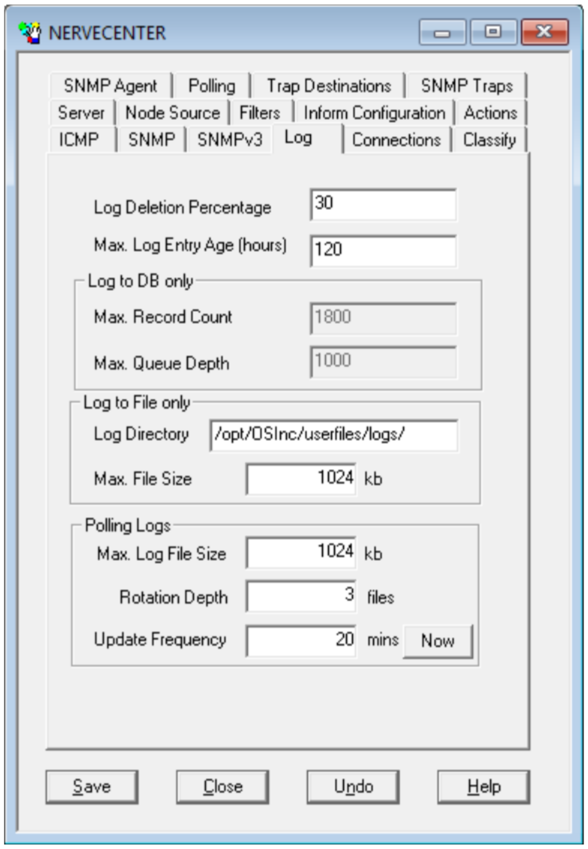
```
verbose: Time=11/11/2002 15:14:43 Mon; LogId=4894;
DestStateSev=Normal; NodePropertyGroup=Mib-II;
NodeName=MyComputer; AlarmName=AllTraps_LogToFile;
OrigState=Ground; TriggerName=allTraps; DestState=Logging;
TrapPduTime=321; TrapPduGenericNumber=4;
TrapPduEnterprise=1.3.6.1.41.78; TrapPduSpecificNumber=0;
TriggerInstance=; TriggerBaseObject=

nonverbose: 11/11/2002 15:14:43 Mon;4875;Normal;Mib-
II;MyComputer;AllTraps_
LogToFile;Logging;allTraps;Logging;321;4;1.3.6.1.41.78;0;;
```

To specify the settings NerveCenter uses to manage logging

1. Open NerveCenter Administrator and connect to the appropriate NerveCenter Server. See "Connecting Administrator to a NerveCenter Server" on page 25.
2. Select the **Log** tab.

   NerveCenter displays the Log tab.



Log Deletion Percentage: 30
Max. Log Entry Age (hours): 120

Log to DB only
Max. Record Count: 1800
Max. Queue Depth: 1000

Log to File only
Log Directory: /opt/OSInc/userfiles/logs/
Max. File Size: 1024 kb

Polling Logs
Max. Log File Size: 1024 kb
Rotation Depth: 3 files
Update Frequency: 20 mins

3. In the **Log Directory** field, type the complete path of the directory where NerveCenter stores the log.

   By default, this field specifies a Log folder created in the NerveCenter directory during installation.

4. In the **Max Log Entry Age** field, type the number of hours you want to keep individual log entries before they are deleted from the rest of the log.

   The way NerveCenter uses this number depends on which logging action is performed:

   - Log to File: NerveCenter checks every file every 20th entry. It deletes any entry it finds that is older than the maximum age specified.
   - Log to Database: NerveCenter waits until the log reaches its size limit. It then deletes any entry it finds that is older than the maximum age specified.

5. In the **Max Log File Size** field, type the size limit, in kilobytes, of the ASCII text file storing the results of the Log to File alarm action.

   This field does not affect the Log to Database alarm action.

6. In the **Max.** Number of Records field, type the highest number of records in the database file storing the results of the Log to Database alarm action.

   This field does not affect the Log to File alarm action.

7. In the **Max.** Queue Depth field, type the highest number of changes that you want queued before saving to the database.

   When the maximum is reached, no more changes are saved until the queue is reduced. You can have more than one change queued for a single database record.

8. In the **Log Deletion Percentage** field, type the percentage of the log to clear when the maximum file size or the maximum number of records is reached.

   NerveCenter checks a log file every 20th entry. If the file log exceeds the amount specified in the Max Log File Size, NerveCenter deletes the percentage specified here, starting with the oldest entries.

9. Select **Save**.

NerveCenter will manage logs according to these settings whenever the Log to File alarm action is used in a behavior model.

Troubleshooting: Managing Alarm Actions

The following list contains some common problems users have when using alarm actions.

NERVECENTER IS NOT SENDING SMTP MAIL NOTIFICATIONS

Problem: A SMTP Server is not specified.

Solution: Specify the connection parameters (host addressing, account, and password) of an SMTP service on the **Actions** tab of the NerveCenter Administrator application.

NERVECENTER SMTP CONFIGURATION HAS BEEN PREPARED BUT NERVECENTER IS NOT SENDING SMTP MAIL

Problem: Despite the SMTP configuration being provided in the Administrator application, NerveCenter is not sending email or SMS messages.

Solution: Verify that the alarm model transitions that indicate an SMTP Mail or SMS Message are being traversed by the running NerveCenter's alarm instances. If the transitions are occurring — such as the firing of a LogToFile action on the same transition — then the server-side handling of the SMTP activity needs to be further explored. When the NerveCenter Server executes an SMTP Mail or SMS Message action, it does so through the `nc-sendmail-action` or `nc-sms-action` commands, located in `/opt/OSInc/nc/bin/`.

These commands are implemented in the Python scripts `nc-sendmail-action.py` and `nc-sms-action.py`. They have been implemented to work with the Python2 environment included wotj RHEL 5/6/7 (and thus CentOS 5/6/7 and OracleLinux 5/6/7) and Solaris 10/11. As your SMTP access depends on the parameters from your SMTP service provider, LogMatrix customer support can help you use these scripts to work through connection issues.

THE LOGS CREATED BY MY BEHAVIOR MODELS ARE USING TOO MUCH DISK SPACE

Problem: The size limits for NerveCenter logging actions are too high.

Solution: Set the maximum sizes to an adequate limit.

See "Specifying Settings for Log Management" on page 149.

Managing the Database

The NerveCenter database does not require much maintenance — it consists of two text files:

- `/opt/OSInc/db/nervecenter.node` contains the node list
- `/opt/OSInc/db/nervecenter.ncdb` contains the rules database

The pre-compiled MIB file `/opt/OSInc/mibs/nervectr.mib` may be considered a third element of this set.

Users should regularly create backups of these files, especially before introducing a substantial or important change to the set of managed nodes or the rules file.

Note: LogMatrix does not provide tools for perform a backup or restore — users should call on the standard UNIX/Linux toolset most appropriate to their environment.

The recommended sequence for backing up or restoring these files is as follows.

1. Use `ncstop` to stop the NerveCenter service.

Caution: Backups and restores should be performed only when the NerveCenter Service is not running on the host.

2. Perform the backup or restore of `nervecenter.node` and `nervecenter.ncdb`; also consider `nervectr.mib`.
3. Use `ncstart` to restart the NerveCenter service

Further, transferring the NerveCenter database between systems is a simple as copying `nervecenter.node`, `nervecenter.ncdb`, and `nervectr.mib` from one host to another (of the same version).

Viewing Information about the Database

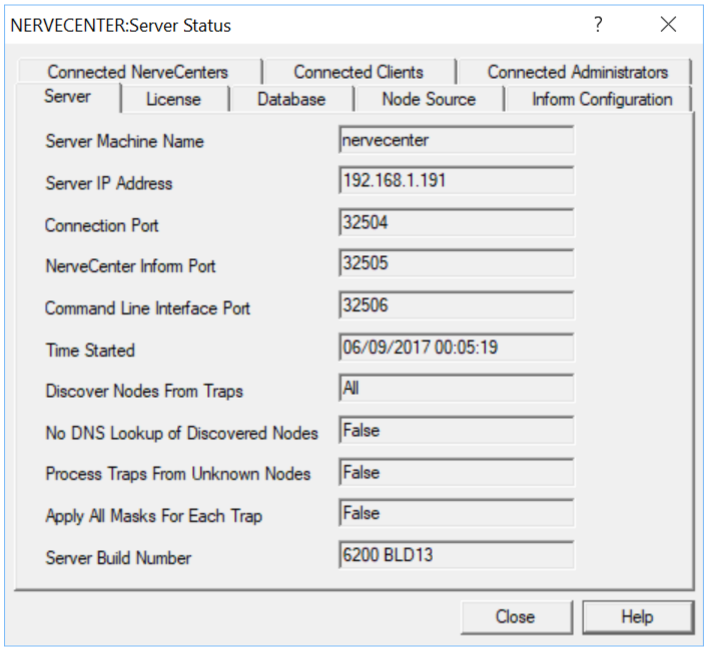
From the NerveCenter Client or Administrator, you can find out the name of the data source, the machine on which the database is kept (if it is different from the machine on which the server runs), the name and location of the database, and whether the database is connected. You can also view the number of alarms, polls, nodes, masks, property groups, and properties stored in the database.

Information about the database is useful when you are troubleshooting or when you just want to verify the database information—for example, to make sure you are going to back up the contents of the correct database.

To View Information About the Database

1. If you are connected to more than one server from either the administrator or client, make sure the server for which you want the database information is the active server.
2. From the **Server** menu, select **Server Status**.

   The Server Status window is displayed.

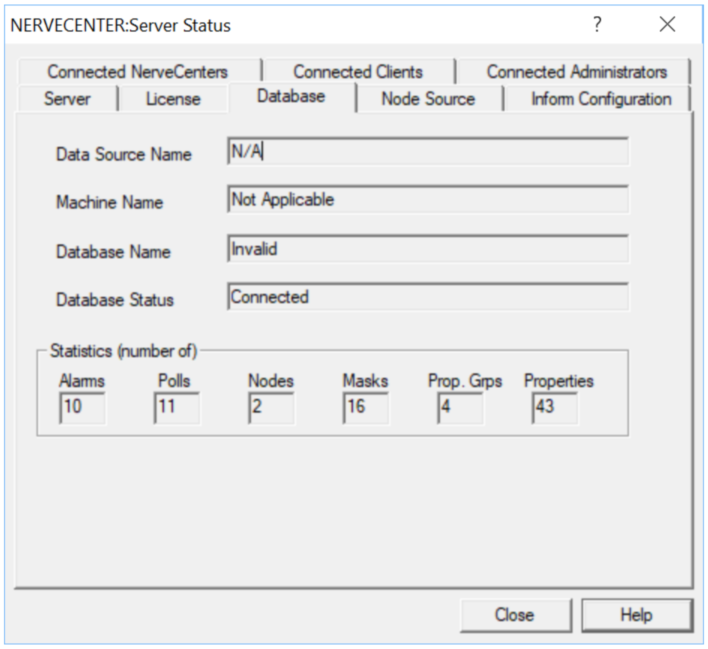


NERVECENTER:Server Status

Server Machine Name	nervecenter
Server IP Address	192.168.1.191
Connection Port	32504
NerveCenter Inform Port	32505
Command Line Interface Port	32506
Time Started	06/09/2017 00:05:19
Discover Nodes From Traps	All
No DNS Lookup of Discovered Nodes	False
Process Traps From Unknown Nodes	False
Apply All Masks For Each Trap	False
Server Build Number	6200 BLD13

3. Select the **Database** tab.

   The Database tab is displayed.



NERVECENTER:Server Status

Data Source Name	N/A
Machine Name	Not Applicable
Database Name	Invalid
Database Status	Connected

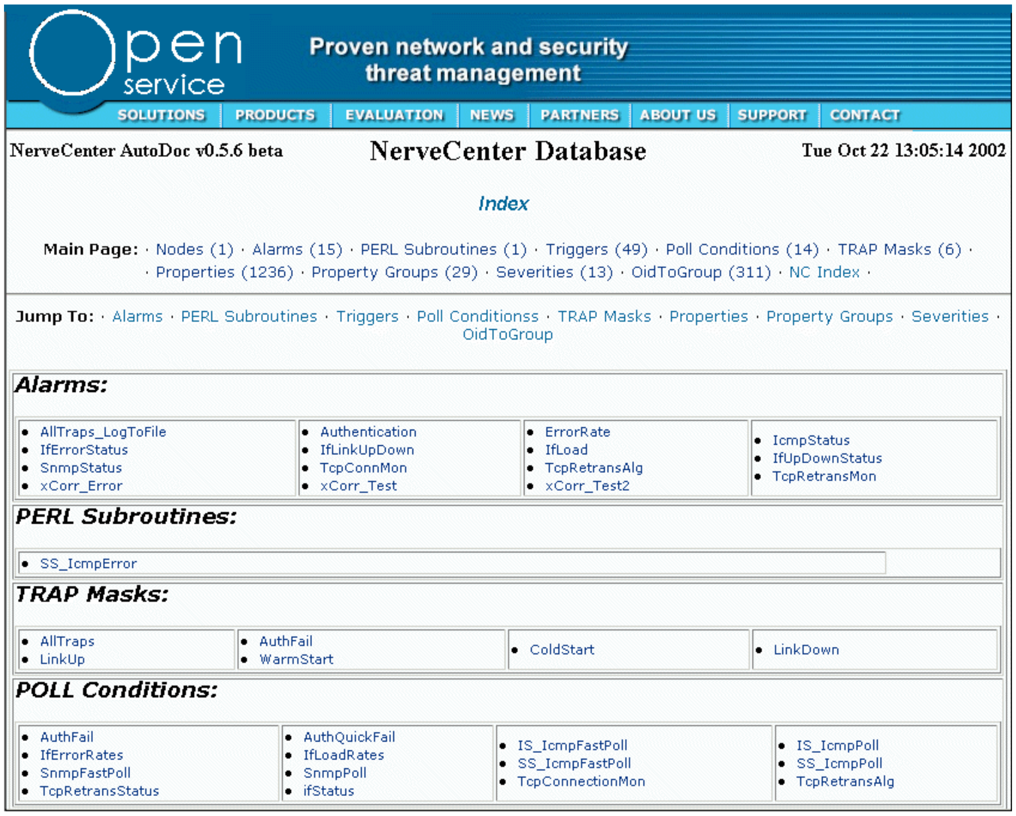
Statistics (number of)

Alarms	Polls	Nodes	Masks	Prop. Grps	Properties
10	11	2	16	4	43

4. When you are done, select **Close** to close the Server Status window.

ncdb2html.pl

NerveCenter includes a utility that converts *.ncdb and *.node files to HTML, so that you can easily know what NerveCenter objects are in your database.



NerveCenter AutoDoc v0.5.6 beta **NerveCenter Database** Tue Oct 22 13:05:14 2002

Index

Main Page: · Nodes (1) · Alarms (15) · PERL Subroutines (1) · Triggers (49) · Poll Conditions (14) · TRAP Masks (6) · · Properties (1236) · Property Groups (29) · Severities (13) · OidToGroup (311) · NC Index ·

Jump To: · Alarms · PERL Subroutines · Triggers · Poll Conditionss · TRAP Masks · Properties · Property Groups · Severities · OidToGroup

Alarms:

• AllTraps_LogToFile • IfErrorStatus • SnmpStatus • xCorr_Error	• Authentication • IfLinkUpDown • TcpConnMon • xCorr_Test	• ErrorRate • IfLoad • TcpRetransAlg • xCorr_Test2	• IcmpStatus • IfUpDownStatus • TcpRetransMon

PERL Subroutines:

• SS_IcmpError

TRAP Masks:

• AllTraps • LinkUp	• AuthFail • WarmStart	• ColdStart	• LinkDown

POLL Conditions:

• AuthFail • IfErrorRates • SnmpFastPoll • TcpRetransStatus	• AuthQuickFail • IfLoadRates • SnmpPoll • ifStatus	• IS_IcmpFastPoll • SS_IcmpFastPoll • TcpConnectionMon	• IS_IcmpPoll • SS_IcmpPoll • TcpRetransAlg

Figure 20: Sample ncdb2html.pl Output

Table 16 details the contents of the HTML output:

Table 16: Output of ncdb2html.pl

File	Contents
ncindex.html	Summarizes the contents of your database. Provides links to all other pages. Contains the following categories: ■ Alarms ■ Perl subroutines ■ Trap masks ■ Poll conditions ■ Triggers ■ Properties ■ Property Groups ■ Severities ■ Oid to Group
nodes.html	Details each alarm in your database, including the following information: ■ Node Name ■ Node Properties ■ Status ■ Property Group ■ Suppressible ■ Auto Delete ■ SNMP Properties ■ Read-Only Community ■ Read-Write Community ■ Port ■ SNMP Version ■ Address List
alarms.html	Details each alarm in your database, including the following information: ■ Status ■ Scope ■ Property ■ Notes ■ State and transition summary

File	Contents
perlsub.html	Lists all Perl subroutines in your database.
triggers.html	Lists all triggers in your database, including the following information: ■ Trigger name ■ Trigger type ■ Where trigger is fired from
polls.html	Details each poll in your database, including the following information: ■ Status ■ Property ■ Base Object ■ Frequency ■ Suppressible ■ Notes ■ Poll Condition
masks.html	Details each trap mask in your database, including the following information: ■ Status ■ Generic trap value ■ Vendor specific trap number ■ Enterprise filter (From/From Only) ■ Simple trigger ■ Notes ■ Mask function
property.html	Lists all properties defined in your database and the property groups they belong to.
propertygrp.html	Lists all property groups in your database and all properties included in each group
severities.html	Details each severity defined in your database, including the following information: ■ Severity Name ■ Color ■ group ■ level ■ platform_level ■ platform_name
oidtogroup.html	Lists all OIDs in your database and the MIB module the OID belongs to.

The *.ncdb and*.node files are the database format on UNIX systems.

TO CONVERT *.NCDB AND *.NODE TO HTML

Note: This script requires Perl 5.6.0 or greater to execute.

1. From a command line or UNIX prompt, navigate to *installation*/**Samples/ncdb2html**.
2. Type the following command:

```
ncdb2html.pl -ncdb=path/filename.ncdb
-node=path/filename.node -h -dir=output_directory
```

The options for this script are detailed in Table 17:

Table 17: ncdb2html.pl Options

Option	Description
-ncdb=*ncdb_file*	Required. Specify the location of ncdb file (**./NerveCenter.ncdb** is default)
-node=*node_file*	Optional. Specify the location of the node file (no default)
-dir=*output_directory*	Specify the directory to place the HTML files (current directory is the default)
-title=*title*	Specify a title for the top of each page (no default)
-h	Hide the community strings on the node page
-sh	Suppress the Open header at the top of each page
-usage	Display usage information and exit

Managing Management Information Bases (MIBs)

By modifying the LogMatrix NerveCenter MIB, you control which devices NerveCenter can effectively manage. The major SNMP MIB definitions and LogMatrix management application MIB definitions are already compiled in the default NerveCenter MIB. You can add other standard MIB definitions and vendors' MIB definitions—many of which are shipped with NerveCenter—as you need them.

Sources of MIB Definitions

Several MIB definitions—the LogMatrix management application definitions and the major Internet standard SNMP definitions—are already compiled into nervectr.mib, the MIB that is installed with NerveCenter.

MIB definitions from other vendors (for Cisco router support, for example) are shipped with NerveCenter. These definitions are not pre-compiled into the NerveCenter MIB because incorporating all the definitions would increase the size of the MIB and could affect performance unnecessarily. You may add the definitions that are appropriate for your environment. You can also add MIB definitions you've gotten from other sources.

Definitions Compiled Into the NerveCenter MIB

The definitions that are already compiled into the NerveCenter MIB are the LogMatrix management application definitions and the major Internet standard SNMP definitions. The definitions are in subdirectories in the mib directory (mibs on UNIX).

Table 18 lists the subdirectories that contain these definitions and includes descriptions of each.

Table 18: Definitions Compiled into the MIB

This directory...	Includes definitions for...
nsmg	NerveCenter management applications
standard-v1	Internet standard RFC SNMPv1 MIB definitions (converted to SNMPv2 compliance)
standard-v2	Internet standard RFC SNMPv2 MIB definitions
standard-v3	Internet standard RFC SNMPv3 MIB definitions

Definitions Not Compiled Into the NerveCenter MIB

Based on your environment, you decide which vendor definitions to compile into the NerveCenter MIB. The definitions are in subdirectories under `mibs/vendors`.

The following vendors provide these definitions and descriptions to help you decide which ones you need:

- 3Com
- Apache
- APC
- ATT
- Bluecoat
- Brocade
- CheckPoint
- Cisco
- Compaq
- CrossBeam
- DEC
- Dell
- EMC
- Epix
- Extreme
- F5
- Fore
- Foundry
- Hewlett-Packard
- Hitachi
- IBM
- Juniper
- Liebert
- Microsoft
- NAT
- NetApps
- ODS
- Oracle
- Retix
- SNMP Research
- Sun
- Symantec
- TippingPoint
- Ungerman Bass
- Veritas
- VMWare
- Xyplex

Definitions From Other Sources

You might want to add other MIB definitions to manage devices that are not supported by the MIBs shipped with NerveCenter. You can add MIB definitions from other sources as long as each one:

- Is a complete ASN.1 module definition that conforms to the standards specifications in RFC1902, *Structure of Management Information for Version 2 of the Simple Network Management Protocol (SNMPv2)*, RFC1903, *Textual Conventions for Version 2 of the Simple Network Management Protocol (SNMPv2)*, and RFC1904, *Conformance Statements for Version 2 of the Simple Network Management Protocol (SNMPv2)*. You can find these RFCs and others on the Web.

Caution: Before NerveCenter can compile a mib it must conform to the ASN.1 module definition standard. See the section "Troubleshooting ASN.1 files" on page 190.

- Does not have the same name as a MIB definition already compiled into your NerveCenter MIB.

Also, remember that your devices must run SNMP agents from your vendor that support the new MIB definitions.

We recommend that you store additional definitions in the mib\user directory (you can create a user directory in the mibs directory on UNIX). Then, when you install a new version of NerveCenter, you can make a copy of this directory to preserve your additional third-party MIBs.

Adding or Removing MIB Definitions

The MIB compiler (mibTool) compiles the MIB definitions referenced in a text file. You can add references to or remove them from this file to control what definitions are compiled into the NerveCenter MIB.

Add MIB definitions if you want to monitor additional devices. Remove MIB definitions if you are no longer monitoring the devices supported by those definitions. (Removing the definitions is not required; however, the NerveCenter MIB will be smaller, which can improve client performance and will make managing the MIB easier.)

If you are using NerveCenter with a network management platform and both applications are monitoring the same agents, make sure both applications are using the correct MIBs for those agents.

See "Sources of MIB Definitions" on page 161 for a list of MIB definitions that are shipped with NerveCenter and what standards MIB definitions are not shipped with NerveCenter must follow.

Note: NerveCenter installs the files nervectr62.mib and mibcomp62.txt. These are copies of the default nervectr.mib and mibcomp.txt in case you need to revert to an unchanged MIB definitions.

The MIB compiler (mibTool) compiles the MIB definitions referenced in a text file. You can add references to this file to control what definitions are compiled into the NerveCenter MIB.

Sample SMIv1 SysUpTime definition	Sample SMIv2 SysUpTime definition
sysUpTime OBJECT-TYPE	sysUpTime OBJECT-TYPE
SYNTAXTimeTicks	SYNTAXTimeTicks
ACCESS read-only	MAX-ACCESS read-only
STATUS mandatory	STATUS current
DESCRIPTION	DESCRIPTION
“The time (in hundredths of a second) since the network management portion of the system was last re-initialized.”	“The time (in hundredths of a second) since the network management portion of the system was last re-initialized.”
::= { system 3 }	::= { system 3 }

Any use of ACCESS within a *.asn1 named in the SMIv2 area of mibcomp.txt causes an error. Any use of MAX-ACCESS within a *.asn1 named in the SMIv1 area of mibcomp.txt causes an error.

TO MODIFY THE MIBCOMP TEXT FILE

1. If you previously modified **mibcomp.txt** or **nervectr.mib**, make back-up copies of them in case you want to recover your changes.

Note: You can modify these files directly; they will not be overwritten during the next installation.

2. In a text editor, open the file.

3. To add definitions, do one of the following:
   - Remove the pound sign (#) in front of definitions that are currently commented out.
   - Add lines for new definitions. Include the directory and the name of the file. For example:

     ```
     users/companyXYZ.asn1
     ```

Note: On Windows, use backslashes. On UNIX, you must use slashes.

4. To remove definitions, do one of the following:
   - Comment out definitions by adding a pound sign (#) in front of the appropriate lines. For example:

     ```
     # xyplex
     #
     # vendors/xyplex/xyplex.asn1
     ```

   - Delete the appropriate lines.
5. Save and close the file.

You have updated the file that includes references to the MIB definitions you need. Now, you must compile the NerveCenter MIB using the file you just created. See "Compiling the NerveCenter MIB" below.

Compiling the NerveCenter MIB

After you modify the mibcomp.txt file, you must compile the NerveCenter MIB. The MIB compiler updates the MIB to support the types of devices you want to monitor. The result of running the mib compiler is the generation of a new MIB file. This file, nervectr.mib, contains a succinct reduction of definitions found in the MIB Modules listed in mibcomp.txt.

TO COMPILE THE MIB

1. At a command prompt, change to the mib directory (mibs on UNIX).
2. Type the following commands and press **Enter**.

   ```
   cd /opt/OSInc/mibs
   mibTool -mibcomp mibcomp.txt
   ```

   As the compiler runs, it displays a series of messages. If a problem needs your attention, the compilation fails. You must resolve the error and re-compile. The error is displayed on the last line on your screen. You will have to do this more than once if you have several errors. Errors are generally syntax problems in third-party MIBs or MIBs that do not comply with the RFC specification. For tips on solving compilation errors, see "Troubleshooting: Managing MIBs" on the facing page.

3. If you did not compile the MIB on the machine that is running the NerveCenter Server, copy the generated nervectr.mib file to that machine.

Caution: The MIB must be located locally on the machine hosting NerveCenter Server. Configure security on the server machine appropriately so no unauthorized users can overwrite or change the MIB. On UNIX, set permissions to read-only using the following command: `chmod 444 nervectr.mib`.

Once you have updated the MIB, file, nervectr.mib, so that it includes the definitions you need, restart the NerveCenter server to load this MIB. If your network management platform uses the NerveCenter MIB, you'll need to update that information in your network management platform. See the documentation for your network management platform for details.

If you added new definitions, you may want to create new property groups that contain properties for the new MIB base objects. See Defining Property Groups and Properties in Designing and Managing Behavior Models.

Troubleshooting: Managing MIBs

Following are some common problems that users encounter when updating NerveCenter MIBs.

As mentioned in "Adding or Removing MIB Definitions" on page 163, MIB modules are dependant on information in other MIB modules. Although a MIB module may compile by itself, there may be problems with dependencies with other MIB modules. This section gives suggestions on how to solve compilation errors.

Note: If MibComp finds a dependency or definition error, it will repeat the error for each MIB module encountered. To find where the problem begins, run MibComp with the **-clean** and **-trace** arguments and note when the problem first occurs.

If a MIB won't compile, consider the following possibilities:

PROBLEM: THE VENDOR ASN1 DEFINITIONS THAT YOU ADDED ARE NOT FORMATTED CORRECTLY.

Solution: Check for spelling and format.

See "Troubleshooting ASN.1 files" on page 190.

PROBLEM: THE MIBCOMP.TXT FILE IS NOT FORMATTED CORRECTLY.

Solution: Check to make sure the appropriate lines are included in the file and not commented out.

See "Adding or Removing MIB Definitions" on page 163.

PROBLEM: THE ASN1 FILE DOES NOT EXIST IN THE DIRECTORY REFERENCED IN THE MIBCOMP.TXT FILE.

Solution: Move the file to the specified directory or correct the path in the mibcomp.txt file.

PROBLEM: YOU MUST RUN THE COMMAND FROM THE MIB DIRECTORY (MIBS ON UNIX).

Solution: Change to the correct directory and run the mibTool command again.

See "Compiling the NerveCenter MIB" on page 164.

PROBLEM: USER OR FILE PERMISSIONS ARE NOT SET CORRECTLY, SO MIBTOOL CANNOT CREATE OR UPDATE THE MIB FILE.

Solution: Check user or file permissions and correct them, if necessary. See your operating system documentation.

PROBLEM: REDEFINITIONS OF MIB ENTITY ENUMERATIONS ACROSS SNMPV1, V2C, AND V3.

Example: WARNING: enumeration conflicts during merge for ifType (continuing)

Solution: These warnings can be safely ignored. Some values, such as ifType or ifOperStatus, have been redefined for different SNMP versions.

PROBLEM: MORE THAN ONE MIB MODULE HAS DEFINED ENTITIES WITH THE SAME NAME.

Example: /opt/OSInc/bin/mgrtool: check_names: Duplicate name with different OIDs: ds1 OID1: 1.3.6.1.2.1.10.18, OID2: 1.3.6.1.3.2 continuing (since -i option was used)

Solution: First, find which MIB modules are defining the duplicated name. When MibComp finds a second definition, it creates an error message similar to the above example. Next, you must research the MIB modules to see if either module has been replaced with a compliant MIB module.

PROBLEM: A MIB MODULE DEPENDS UPON A DEFINITION FORM ANOTHER MIB MODULE.

Example: find_type(): unknown type: SnmpAdminString
mibTool: unable to compile and resolve standard-v3/v3-tgt.my

Solution: Look at the IMPORTS clause of the module that produced the error. In the example, **v3-tgt.my** declares that SnmpAdminString is to be imported from SNMP-FRAMEWORK-MIB. A search of the modules shows that SNMP-FRAMEWORK-MIB is defined in **v3-arch.my**. Therefore, you must name **v3-arch.my** before **v3-tgt.my**.

PROBLEM: MORE THAN ONE MIB MODULE PROVIDES A NAME FOR AN OID.

Example: /opt/OSInc/bin/mgrtool: Warning: Duplicate OIDs with different names.
The first name will appear before the second name.
OID: 1.3.1.4.1.9.7.99999.2, First name: cwRsrcPartCapabilityRpmV2R0160,
Second name: ciscoWanModuleCapabilityV2R00

Solution: Often this is not a problem but should be investigated. In the example, two different Cisco MIB modules state two different names for the same OID. Which device and version of IOS is encountered at run-time will likely resolve which way the OID will be handled. Check which devices you are currently running and whether you need to have the MIB modules with the duplicate names **mibcomp.txt**. A review of your Cisco devices may show that the MIB module that names the first occurrence is not needed.

Error Messages

This appendix explains the error and information messages that you might encounter while using NerveCenter. Possible causes and solutions for the errors are included.

All NerveCenter error messages are written to the host syslog facility. To view messages in the syslog, read the file /var/adm/messages with a text editor or a command such as more.

Each error description is formatted in the following way:

```
Category error_message_number: message: [code_number]
```

Each message is assigned a category, which has a corresponding number. The line listed in the log uses a number to indicate a category, as follows:

Table 19: Error Message Categories

Number	Category
1	NC Server Manager
2	NC Alarm Manager
3	NC Trap Manager
4	NC Poll Manager
5	NC Action Manager
6	NC Protocol Manager
7	NC PA Resync Manager
8	NC Service
9	NC Inform NerveCenter Manager
11	NC LogToFile Manager
12	NC FlatFile Manager
13	NC Alarm Filter Manager
14	NC Deserialize Manager
16	NC DB Manager

The error message number indicates the error type. The error numbers are organized as follows:

Table 20: Error Message Numbers

Number Range	Type of Error
0-999	Users should call customer support.
1000-1999	User can resolve the problem.
2000-2999	User is warned of an event.
3000-3999	User is given an informational message.

The error messages are explained in the following sections:

- "Alarm Filter Manager Error Messages" on the facing page
- "Deserialize Manager Error Messages" on the facing page
- "Flatfile Error Messages" on the facing page
- "Inform NerveCenter Error Messages" on the facing page
- "LogToFile Manager Error Messages" on page 172
- "Poll Manager Error Messages" on page 172
- "Protocol Manager Error Messages" on page 172
- "PA Resync Manager Error Messages" on page 173
- "Server Manager Error Messages" on page 175
- "Trap Manager Error Messages" on page 177
- "NerveCenter installation Error Messages" on page 178

Alarm Filter Manager Error Messages

Table 21: Alarm Filter Manager Error Messages

#	Error	Resolution
1	Lookup failed on line number *value* in File *value*.	
3001	Alarm Filter Manager Initialization successfully finished	

Deserialize Manager Error Messages

Table 22: Deserialize Manager Error Messages

#	Error	Resolution
1	Lookup failed on line number *value* in File *value*.	
3001	Deserialize Thread Manager Initialization successfully finished	

Flatfile Error Messages

Table 23: Flatfile Manager Error Messages

#	Error	Resolution
1	Lookup failed on line number *value* in File *value*.	
3001	Flat File Initialization successfully finished	

Inform NerveCenter Error Messages

Table 24: Inform NerveCenter Manager Error Messages

#	Error	Resolution
1	Lookup failed on line number *value* in File *value*.	
3001	InformNC Manager Initialization successfully finished	

LogToFile Manager Error Messages

Table 25: Log to File Manager Error Messages

#	Error	Resolution
1	Lookup failed on line number *value* in File *value*.	
3001	LogToFile Manager Initialization successfully finished	

Poll Manager Error Messages

Table 26: Poll Manager Error Messages

#	Error
3001	Poll Manager Initialization successfully finished
3002	CPollManagerWnd:OnPollOnOff, PreCompild of PollEvent with Poll Id %ld failed

Protocol Manager Error Messages

Table 27: Protocol Manager Error Messages

#	Error	Resolution
1	Building copy of node list failed.	N/A
2	Building copy of poll property list failed.	N/A
3	I.nitialization of protocol methods failed	N/A
4	Initialization of ping socket failed.	N/A
5	Creation of SNMP socket failed, socket error code: %d	N/A
6	Error in ping socket: %s	N/A
7	Error in ping socket: create socket failed.	N/A
8	Error in ping socket: async select failed.	N/A
1000	Looking for the %s key in the configuration settings.	Use the Administrator to enter the SNMP values in the configuration settings.
1001	Ncuser user ID is not found.	Add ncuser user ID to your system.
3000	Initialization successfully finished.	N/A
3001	Invalid value in configuration settings for SNMP retry interval, using default of 10 seconds.	Use the Administrator to enter an SNMP retry interval.

#	Error	Resolution
3002	Invalid value in configuration settings for number of SNMP retries, using default of 3 retries.	Use the Administrator to enter a number of SNMP retries.
3003	Invalid value in configuration settings for default SNMP port, using default of 161.	Use the Administrator to enter the default SNMP port number.

PA Resync Manager Error Messages

Table 28: PA Resync Manager Error Messages

#	Error	Resolution
1	Error getting local host name for encoding resync request, socket error code: %d	N/A
2	Encoding resync request failed	N/A
3	Sending resync request failed with zero bytes sent	N/A
4	Sending resync request failed: %s	N/A
5	Memory allocation error, trying to notify of connection status	N/A
6	Memory allocation error, creating node list	N/A
7	Memory allocation error, creating a resync node	N/A
8	Parent status not sent during resync	
10	Parents not computed during resync with map host. Check OVPA. OVPA database must have nc host node.	
500	Socket Error: (%d)	
1000	Error looking for the %s key in the NerveCenter configuration settings	Use the Administrator to enter configuration settings.
1001	Attempt to connect to %s on port %d failed: %s	Make sure the platform host is up and running and that the name exists in the hosts file.

#	Error	Resolution
1002	Resync connection attempt failed: %d	Make sure the platform host is up and the platform adapter is running.
1500	The connection to % was closed	
1501	Send failed with zero bytes sent	
1505	%s. The address is already in use	Make sure you are not running two instances of the same application on the same machine.
1506	%s. The connection was aborted due to timeout or other failure	Make sure the physical network connections are present.
1507	%s. The attempt to connect was refused	Make sure the server is running on the remote host.
1508	%s. The connection was reset by the remote side	Make sure the remote peer is up and running.
1509	%s. A destination address is required	A destination address or host name is required.
1510	%s. The remote host cannot be reached	Make sure the routers are working properly.
1511	%s. Too many open files	Close any open files.
1512	%s. The network subsystem is down	Reboot the machine.
1513	%s. The network dropped the connection	Make sure the peer is running and the network connections are working.
1514	%s. No buffer space is available	This might be because you are running several applications, or an application is not releasing resources.
1515	%s. The network cannot be reached from this host at this time	Make sure the routers are functioning properly.
1516	%s. Attempt to connect timed out without establishing a connection	Make sure the machine is running and on the network.
1517	%s. The host cannot be found	Make sure you can ping the host, check you hosts file or DNS server.
1518	The network subsystem is unavailable	Make sure the network services are started on machine.

#	Error	Resolution
1519	%s. Invalid host name specified for destination	The host name cannot be resolved to an IP address. Enter the name to the hosts file or DNS server.
1520	The specified address in not available	Make sure the host name is not zero. Try pinging the host.
3000	initialization successfully finished	N/A
3001	Node resync from map host was not requested because either host name or port number is missing	If you are trying to disable a connection to the platform adapter, then this message is OK. If you want to be connected to the platform adapter, then use the Administrator to check the map host settings.
3500	Connection to %s was successful	N/A

Server Manager Error Messages

Table 29: Server Manager Error Messages

#	Error	Resolution
2	Perl create failed.	N/A
3	Initialization of *value* manager thread failed.	N/A
4	Failed to restore MibDirectory in configuration settings.	N/A
5	Failed to open configuration settings while trying to restore mib information.	N/A
6	Discrepancy in data. File: SERVER_CS.CPP, Line: *value.*	N/A
10	Conflict in data. File: SERVER_CS.CPP, Line: *value.*	N/A
11	Internal Error. File: SERVER_CS.CPP, Line: *value.*	N/A
20	Cannot read configuration settings value: Bind.	N/A
21	Cannot connect to Tcpip configuration settings information.	N/A
22	Cannot read configuration settings value: IPAddress.	N/A

#	Error	Resolution
23	Couldn't find *value* in map.	N/A
24	Error while reading database. Poll/Mask:*value* uses a simple trigger that doesn't exist in database.	N/A
25	Please report error number *value* to technical support.	N/A
26	User validation failed: Unable to communicate with ncsecurity process :*value.*	~
1002	Initialization failed, cannot find ncperl.pl.	Check NCPerl.pl location.
1003	Failed to open MIB: *value.*	Check MIB location.
1004	Failed to parse MIB.	Invalid MIB. Check configuration to see if the correct MIB is specified.
1010	Failed to validate poll: *value*. The poll will be turned off.	Check the poll condition using the Client Application.
1100	*value* (database error).	Try to resolve using the message. If not, call support.
1103	Version table validation failed. NC_Version table doesn't exist in database.	Upgrade the NerveCenter database.
1200	Failed to open configuration settings while trying to restore mib information.	Use the NerveCenter Administrator to check the configuration settings. Invalid key is likely.
1202	Cannot connect to configuration settings.	Use the NerveCenter Administrator to check the configuration settings. Invalid key is likely.
1203	Cannot open key *value.*	Use the NerveCenter Administrator to check the configuration settings.
1204	Cannot add value *value*.	Use the NerveCenter Administrator to check the configuration settings. Invalid key is likely.
1205	Cannot read configuration settings value in MapSubNets key.	Use the NerveCenter Administrator to check the configuration settings. Invalid key is likely.
1206	Invalid configuration settings Entry for the value Method in the Platform key.	Only Manual and Auto are allowed. Check for case.
1207	Cannot read configuration settings value: *value*	Use the NerveCenter Administrator to check the configuration settings. Invalid key is likely.

#	Error	Resolution
1208	Cannot write configuration settings Value: *value*	Use the NerveCenter Administrator to check the configuration settings. Invalid key is likely.
1300	*value* (Import behavior/database error).	Try to resolve using the message. If not, call - support.
1313	Server alarm instance maximum exceeded. Please restart Server.	Restart server.
3001	Request to delete the node *value* failed because the node doesn't exist.	N/A
3002	Failed to find socket in server's map. Line: *value*.	
3003	Exiting due to a SIGTERM signal.	
3004	Primary thread initialization successful.	

Trap Manager Error Messages

Table 30: Trap Manager Error Messages

#	Error	Resolution
1	Error in TrapManagerWnd::Initialize - failed to create GetHostByAddr thread.	
2	Error in TrapManagerWnd::LaunchTrapper - failed to create trapper process.	
3	Error in TrapManagerWnd::CreateCheckTrapperThread - failed to create new thread.	
8	Error in TrapManagerWnd::Initialize - Failed to create trap stream socket.	
9	Error in TrapManagerWnd::Initialize - Failed to listen on trap stream socket.	
10	Error in TrapManagerWnd::OnTraceTraps - Failed to create trace file for traps.	
1001	CTrapManagerWnd::OnTrapExist - gethostbyname from trap data with snmptrap failed for *value.*	
1003	CTrapManagerWnd::OnInvalidSignature - Error in receiving data on NC socket.	Check for consistent version numbers of trapper and NerveCenter executables.

#	Error	Resolution
2001	MS Trap service threw exception in GetTrap.	Make sure you aren't making SNMP get requests to port 162.
2002	Error processing trap data.	Make sure you aren't making SNMP get requests to port 162.
3001	Trap Manager Initialization successfully finished.	
3002	Check Trapper—Trapper process died. restarting Trapper.	

NerveCenter installation Error Messages

Table 31: NerveCenter Installation Error Messages (UNIX)

Error	Resolution
Space under *dirname* is INSUFFICIENT to install LogMatrix NerveCenter	Free up space in the file system by removing files, or choose another place for installation.
The directory *dirname* must reside on a local disk	The directory you specified for NerveCenter installation is on a disk that is not on the local file system. Pick a new directory or re-mount the disk.
Write permission is required by root for *dirname* directory	The directory you specified for NerveCenter installation does not have write permission for root. Choose another directory or change the permissions.
Please create the desired destination directory for NerveCenter and re-run the installation script	The directory you specified for NerveCenter installation does not exist. Choose another directory or create the original.
Invalid mount point	The installation script could not find the CD-ROM drive and prompted you for its location. The path you specified was not valid. Verify that the drive exists, is mounted, and is configured correctly.
ProcessName is running on the system. Please exit from (or kill) *processName* process.	The installation script found that the nervectr or ovw process was running. Exit from or kill the process and re-run the installation script.
These processes must be stopped before NerveCenter can be installed. Please kill these processes and re-run the installation script.	The installation script found processes that need to be killed before installation, you were prompted to stop them, and you said no. You must manually exit from or kill the processes and re-run the installation script.

Error	Resolution
hostname is not a valid host name	The host that you provided to the script is not a valid host. Check the name of the host (capitalization, spelling, and so on) and try again.
I don't know how to install on this architecture	Installation is supported for Solaris. The script issues this message if attempting to install on an architecture that is not in this set.
Can't cd to *installation_ path*/userfiles	Make sure the directory exists and has appropriate permissions.
Can't open *hostname*.conf	The script couldn't create the file or couldn't open an existing configuration file. Check *installation_path*/userfiles to make sure that root has permission to write in this directory, that *hostname*.conf has read permission set, if it exists, and that localhost.conf exists and has read permission set.
Can't create *hostname*.ncdb Can't create *hostname*.node	The script was attempting to create the indicated file by copying data from another file. Check *installation_path*/userfiles to make sure that root has permission to write in this directory, and that localhost.*ext* exists and has read permission set.
Can't open /etc/rc Couldn't re-create /etc/rc Couldn't modify /etc/rc	The script couldn't modify /etc/rc to call the NerveCenter rc script. Edit the file and add a line that executes *installation_ path*/bin/rc.openservice. There's no need to rerun the installation script after this correction.
Can't append to /etc/rc.local	The script couldn't modify /etc/rc.local to call the NerveCenter rc script. Edit the file and add a line that executes *installation_ path*/bin/rc.openservice. There's no need to rerun the installation script after this correction.
Can't create /etc/rc2.d/K94ncservice on Solaris	The script couldn't create the NerveCenter rc script /etc/rc2.d/K94ncservice on Solaris. Copy *installation_path*/bin/rc.openservice to /etc/rc2.d//K94ncservice. There's no need to rerun the installation script after this correction.

Error	Resolution
An error occurred in trying to contact the Server “*hostname*”. As a result, the information that you have specified cannot be used to complete this NIS update. Unable to modify *filename*. It doesn’t exist! Unable to modify *filename*. File size is 0!	The script was attempting to update system services and failed. Correct the specific error (perhaps the host name or file name was entered incorrectly) and rerun the script. If the error isn’t easily corrected, you can edit /etc/services yourself. Make sure that the following lines are included in the file: SNMP 161/udp SNMP-trap 162/udp If you’re running NIS, be sure to make these changes on the NIS server, change to the NIS directory, and run make services.

Troubleshooting NerveCenter

B

NerveCenter’s complexity means that users will inevitably run into difficulties at some point. Before calling NerveCenter Technical Support, there are several first steps you can take. Besides checking the obvious—the NerveCenter Server is running, the license has not expired, etc.—you may want to look at information found in this appendix.

Common problems

The following list includes some of the more common problems users face when administering NerveCenter.

THE NERVECENTER SERVER

- "Troubleshooting: Running the NerveCenter Server" on page 23.
- "An alarm causes the NerveCenter Server to crash every time I start it" on page 25.
- "UNIX will not start the NerveCenter Administrator" on page 36.
- "Troubleshooting: Connecting to the NerveCenter Server" on page 32.
- "I misspelled a server name while trying to connect and now the misspelled name appears in the NerveCenter Administrator Server Name list" on page 36..
- "I need to make the same changes to several NerveCenter Servers" on page 45.
- "Importing imputil.ini caused unwanted changes to a NerveCenter Server" on page 45.
- "I cannot log in to a UNIX NerveCenter Server." on page 46

THE NERVECENTER NODE LIST

- "Troubleshooting: Managing Node Data" on page 92.
- "Troubleshooting: Managing Node Data" on page 92.
- "Troubleshooting: Managing Node Data" on page 92.
- "Troubleshooting: Managing Node Data" on page 92.
- "Troubleshooting: Managing Node Data" on page 92.
- "Troubleshooting: Managing Node Data" on page 92.
- "Troubleshooting: Managing Node Data" on page 92.
- "Troubleshooting: Managing Node Data" on page 92.
- "Troubleshooting: Managing Node Data" on page 92.
- "Troubleshooting: Managing Node Data" on page 92.

ALARM ACTIONS

- "Troubleshooting: Managing Alarm Actions" on page 152.

MANAGEMENT INFORMATION BASES (MIBS)

- "If a MIB won’t compile, consider the following possibilities:" on page 165.
- "Troubleshooting: Managing MIBs" on page 165.

Using Trace Counters to Troubleshoot NerveCenter

A useful tool for troubleshooting NerveCenter is the trace counters feature. Trace counters enable NerveCenter to keep track of its own activities, such as the number of SNMP traps that have been received, polls requested, Informs sent, and quite a few other operations. The cumulative count for an operation reflects total activity for either of the following durations:

- For as long as the NerveCenter Server has been running
- If you have reset a counter, since the reset was performed.

Trace counters can help you troubleshoot communication or behavior model problems.For example, if Informs are not being received by your network management platform for a particular network condition, you might trace a backward path starting with the Inform Action counters to find out where the problem arises. If informs are being sent, the problem is not with your alarm, polls, or masks. If no Informs are requested or sent, you might examine whether triggers are being issued or whether SNMP traps are being captured in trap masks by examining the counters in the Trigger and Trap pages.

If you are having difficulties with NerveCenter, a LogMatrix Technical Support representative may provide further instructions for using the Trace counters, including enabling trace logging.

Note: NerveCenter’s trace logging feature should only be used as directed by Technical Support. Trace logging quickly consumes large blocks of memory and could impact Server performance.

Even though the data presented by the trace counters are read-only, you can reset a counter. See "Using Trace Counters" on the facing page for details about viewing, resetting, and refreshing counters.

The following sections describe the meaning of each counter:

- "Alarm Action Counters" on page 185
- "Database Counters" on page 185
- "Inform Counters" on page 186
- "Inform NerveCenter Counters" on page 186
- "Log to File Counters" on page 186
- "Server Counters" on page 187
- "Node Source Counters" on page 187
- "Poll Counters" on page 187
- "Trap Counters" on page 189
- "Trigger Counters" on page 189

Using Trace Counters

This section describes how to access and use the trace counters.

TO VIEW NERVECENTER TRACE COUNTERS

1. From the Server menu, choose **Trace Counts**.

   The Trace Counters window is displayed.

2. Select one of the tabs.

   The corresponding set of counters is displayed.

3. Select the **Refresh** button to update all counters.

Once you have the counters displayed, you can reset one or more counters as described in "Resetting Counters" below. You may also periodically need to update all counters as described in "Refreshing the Trace Counter Display" on the next page.

Resetting Counters

When using NerveCenter Trace Counters to troubleshoot a problem, you may want to reset one or more counters. For example, if Informs are not being received by your network management platform for a particular network condition, you can isolate the problem by turning off alarms not related to the condition, resetting the related counters, and then tracing the activity for the alarm you want to monitor.

TO RESET NERVECENTER TRACE COUNTERS

1. In the Trace Counters window, select the page containing the counters you want to reset.
2. Do either of the following:
   - Select the **Reset** checkbox next to the counter or counters you want to reset, and then select the **Reset** button. The counters you checked are reset.
   - To reset all counters in a page, select the **Reset All** button.

Refreshing the Trace Counter Display

When you open the Trace Counter window, you see the current values for all counters. Once opened, however, the counter information is updated only upon command. To update counter values, you must refresh the view manually.

To update NerveCenter trace counters once the window is opened

- Select the **Refresh** button.

  All counters are updated.

Table 32 provides a reference for all procedures related to the trace counters. Each of these procedures are performed within a NerveCenter Administrator after it is connected to a NerveCenter Server

Table 32: Procedures Related to Trace Counters

Task	Procedure
To view the trace counters...	From the **Server** menu, choose **Trace Counts**.
To view a specific counter...	Within the **Trace Counters** window, select the appropriate tab.
To reset one or more counters...	After selecting each of the relevant **Reset** checkboxes, select **Reset**.
To reset all the counters in a page...	Select **Reset All**.
To update all the counters...	Select **Refresh**.

Alarm Action Counters

The Action tab of the Trace Counters window provides read-only information about the current NerveCenter Server's automated actions. See "Using Trace Counters to Troubleshoot NerveCenter" on page 182 for more information about using the counters.

Table 33: Alarm Action Counters

Group	Field	Description
Specific action	Action Type	Select the type of action you want to monitor.
	Requested	The number of specified actions that have been requested.
	Pending	The number of specified actions that are pending.
	Completed	The number of specified actions that are completed.
Totals for All Actions	Requested	The total number of all actions that NerveCenter has requested (as opposed to the number of actions for the selected action type).
	Pending	The total number of all actions that are currently pending.
	Completed	The total number of all actions that have been completed.

Database Counters

The Database tab of the Trace Counters window provides read-only information about the current NerveCenter Server's automated actions. See "Using Trace Counters to Troubleshoot NerveCenter" on page 182 for more information about using the counters.

Table 34: Database Counters

Group	Field	Description
DB Operations	Successful	The number of NerveCenter operations that have been written to the database.
	Pending	The number of database entries waiting to be written.
DB Connections	Lost	The number of lost connections with the database.
	Restored	The number of lost connections with the database that were later restored.

Inform Counters

The Inform tab of the Trace Counters window provides read-only information about the current NerveCenter Server's automated actions. See "Using Trace Counters to Troubleshoot NerveCenter" on page 182 for more information about using the counters.

Table 35: Inform Counters

Group	Field	Description
Universal PA	Requested	The number of informs requested to be sent to NerveCenter's Universal Platform Adapter.
	Sent	The number of informs that have been sent to NerveCenter's Universal Platform Adapter.

Inform NerveCenter Counters

The Inform NC tab of the Trace Counters window provides read-only information about the current NerveCenter Server's automated actions. See "Using Trace Counters to Troubleshoot NerveCenter" on page 182 for more information about using the counters.

Table 36: Inform NerveCenter Counters

Field	Description
Received	The number of informs the current NerveCenter Server has received from other NerveCenter Servers.
Processed	The number of informs the current NerveCenter Server has received and processed from other NerveCenter Servers.

Log to File Counters

The Log File tab of the Trace Counters window provides read-only information about the current NerveCenter Server's automated actions. See "Using Trace Counters to Troubleshoot NerveCenter" on page 182 for more information about using the counters.

Table 37: Log to File Counters

Field	Description
Requested	The number of Log to File actions requested.
Pending	The number of Log to File actions that are still pending.
Lost	The number of Log to File actions that did not successfully log and are considered to be lost.
Deleted	The number of Log to File actions that have been removed from the log file.

Server Counters

The Server tab of the Trace Counters window provides read-only information about the current NerveCenter Server's automated actions. See "Using Trace Counters to Troubleshoot NerveCenter" on page 182 for more information about using the counters.

Table 38: Server Counters

Field	Description
Alarms transitioned	The number of alarm instances that the current NerveCenter Server has transitioned.
Messages to client	The number of messages the current NerveCenter Server has sent to a connected NerveCenter Client.
Client connections	The number of NerveCenter Clients that have connected to the current NerveCenter Server.

Node Source Counters

The Node Source tab of the Trace Counters window provides read-only information about the current NerveCenter Server's automated actions. See "Using Trace Counters to Troubleshoot NerveCenter" on page 182 for more information about using the counters.

Table 39: Node Source Counters

Field	Description
Nodes received	The total number of nodes that NerveCenter has received from a node source and stored in its database received during the last resync
Nodes updated	The total number of updates that NerveCenter has received from a node source during the last resync.
New nodes created	The number of nodes added to the NerveCenter node list when nodes were added to the network management platform.
Nodes auto deleted	The number of nodes removed from the NerveCenter node list.

Poll Counters

The Poll tab of the Trace Counters window provides read-only information about the current NerveCenter Server's automated actions. See "Using Trace Counters to Troubleshoot NerveCenter" on page 182 for more information about using the counters.

Table 40: Poll Counters

Group	Field	Description
SNMP	Requests	The number of SNMP polls requested.
	Pending	The number of polls waiting for an SNMP reply poll request.
	Responses Received	The number of SNMP responses that have been received from SNMP requests.
	Error Received	The number of SNMP and ICMP errors resulting from SNMP requests.
	Polls Retried	The number of SNMP polls that were reissued.
	Polls Timed Out	The number of SNMP polls that did not receive a response and therefore timed out.
ICMP	Requests	The number of ICMP polls requested.
	Pending	The number of polls waiting for an ICMP reply to the poll request.
	Responses	The number of ICMP responses that have been received from ICMP requests.
	Error Received	The number of ICMP errors resulting from ICMP requests: The following types of messages indicate errors: ICMP_UNREACH, ICMP_SOURCEQUENCH, ICMP_REDIRECT, ICMP_TIMXCEED, ICMP_PARAMPROB.
	Polls Retried	The number of ICMP polls that were reissued.
	Polls Timed Out	The number of ICMP polls that did not receive a response and therefore timed out.

Trap Counters

The Trap tab of the Trace Counters window provides read-only information about the current NerveCenter Server’s automated actions. See "Using Trace Counters to Troubleshoot NerveCenter" on page 182 for more information about using the counters.

Table 41: Trap Counters

Field	Description
Received	The number of SNMP traps that the current NerveCenter Server has received and processed.
Failed	The number of SNMP traps that the current NerveCenter Server has received but was unable to process.
Ignored	The number of SNMP traps that the current NerveCenter Server has received but ignored. NerveCenter ignores traps if NerveCenter is configured not to process traps from unknown nodes and the node sending the trap is outside NerveCenter’s specified IP subnet filter range.

Trigger Counters

The Trigger tab of the Trace Counters window provides read-only information about the current NerveCenter Server’s automated actions. See "Using Trace Counters to Troubleshoot NerveCenter" on page 182 for more information about using the counters.

Table 42: Trigger Counters

Field	Description
Masks	The number of triggers that NerveCenter has fired from trap masks.
Actions	The number of triggers that NerveCenter has fired from alarm actions.
Polls	The number of triggers that NerveCenter has fired from polls.
Total	The total number of triggers that NerveCenter has fired.

Troubleshooting ASN.1 files

Many mibcomp error messages are caused by problems with ASN.1 MIB files. This section describes how to make sure your ASN.1 files can be processed correctly by the mibcomp utility and how to correct common ASN.1 errors. If you receive mibcomp error messages, check your ASN.1 files for the following possible problems:

- Improper characters
- Improper use of period, underscore, and hyphen
- Improper capitalization
- Improper table construction
- Duplicate type definitions
- Object identifier format
- Ignored **IMPORT** definitions

Use of Characters

Blank spaces, carriage returns (^M), and line feeds (^J) are considered white space, used only as separators.

The following are the only characters you can use in ASN.1:

```
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ
0123456789
:=,{}<>.()[]-’”;|
```

Except in a comment, you cannot use the following characters:

```
!@#$%^&*_+\~‘?/
```

Note the following exceptions:

- You can use any character in a comment.
- An object name can have only alphabetic and numeric characters and hyphens (but never two hyphens in a row), and cannot end with a hyphen.

Period, Underscore, and Hyphen

The only nonalphanumeric character you can use in a type or value is the hyphen (-). Do not use a period (.) or an underscore (_).

Two consecutive hyphens (--) begin comments. Two consecutive hyphens or the end of a line can end comments. When you end a comment with a second pair of hyphens, mibTool tries to convert the material between the second pair of hyphens and the end of the line. For example, given the following line:

```
-- This is a comment -- This is not a comment.
```

The mibTool utility tries to convert “This is not a comment.” Don’t use two consecutive hyphens in an object name.

Capitalization

Key words and predefined types must contain only uppercase letters and must not be reserved character sequences. Reserved character sequences are listed below:

Table 43: Reserved character sequences

`ABSENT`	`DEFINED`	`INTEGER`	`REAL`
`ANY`	`DEFINITIONS`	`MAX`	`SEQUENCE`
`APPLICATION`	`END`	`MIN`	`SET`
`BEGIN`	`ENUMERATED`	`MINUS-INFINITY`	`SIZE`
`BIT`	`EXPLICIT`	`NULL`	`STRING`
`BOOLEAN`	`EXPORTS`	`OBJECT`	`TAGS`
`BY`	`EXTERNAL`	`OCTET`	`TRUE`
`CHOICE`	`FALSE`	`OF`	`UNIVERSAL`
`COMPONENT`	`FROM`	`OPTIONAL`	`WITH`
`COMPONENTS`	`IDENTIFIER`	`PLUS-INFINITY`	
`correct`	`IMPLICIT`	`PRESENT`	
`DEFAULT`	`INCLUDES`	`PRIVATE`	

Type references and type declarations must begin with an uppercase letter. Enumeration names, value names, and object identifiers must begin with a lowercase letter.

Table Construction

If a table isn't being displayed properly, it might be missing an **INDEX** clause or a **SEQUENCE** definition. An **INDEX** clause looks like this:

```
INDEX(instance)
        ::= (parent)
```

A **SEQUENCE** definition for each table in the ASN.1 file is necessary for compliance with RFC1212. For example, a MIB might have:

```
commonB                OBJECT IDENTIFIER ::= { rr2board 1 }
boardIndex                              OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-only
STATUS mandatory
DESCRIPTION
                       "The slot number of this board."
::= { commonB 1 }

.
.
.
```

etc.

This prevents the mibcomp utility from producing usable output because it cannot detect that boardIndex is an attribute of a tabular object. One way to fix this is to add extra material between commonB and the first attribute, as follows:

```
commonB OBJECT IDENTIFIER ::= { rr2board 1 }
-- -- Modified for 1212 compliance --
commonB OBJECT-TYPE
SYNTAX CommonB
ACCESS read-write
STATUS mandatory
INDEX { boardIndex }
:= { rr2board 1 }
CommonB ::= SEQUENCE {
boardIndex
INTEGER,
boardName
OCTET STRING,
boardType
INTEGER,
boardTotalPorts
INTEGER,
boardStatus
INTEGER,
boardPortsOn
INTEGER         }
boardIndex OBJECT-TYPE
SYNTAX INTEGER
ACCESS read-only
STATUS mandatory
DESCRIPTION
"The slot number of this board."
::= { commonB 1 }
```

```
                .
                .
          etc.
```

Do not nest tables. Nesting can happen indirectly—for example, when a type that contains a table is referenced in a table without the connection being obvious. To avoid nesting, you can use an index from one table to index another.

Duplicate Type Definitions

If the mibcomp utility encounters duplicate type definitions, it uses the first definition it encounters and then displays the following message:

```
Warning(file.asn1:about line n):

Duplicate type defined: TypeName
```

It then continues processing without pausing for you to respond. Make sure the type definitions are all the same, or rename the types and run mibTool again.

Object Identifier Format

The only object identifier format mibTool accepts is:

```
{objname integer}
```

Any other format results in the error message:

```
Illegal Object Identifier Value
```

If your MIB contains an object identifier like:

```
open OBJECT IDENTIFIER ::= { osi 1 2}
```

you could change it to an acceptable object identifier by splitting the definition as follows:

```
open OBJECT IDENTIFIER ::= { osi 1}
open OBJECT IDENTIFIER ::= { osi 2}
```

Ignored IMPORT Definitions

The mibcomp utility ignores the **IMPORT** definition.

For example,

```
MIB-A: x = 1
MIB-B: x = 2
MIB-C: import x from MIB-B
```

In MIB-C, the utility takes x as 2 rather than as 1 if MIB-A is parsed first. Before parsing, check for **IMPORT** clauses and copy the **IMPORT** definitions directly into the MIBs containing them.

Controlling NerveCenter from the Command Line

C

NerveCenter provides commands that you can run from a UNIX shell or a DOS Command Prompt window. NerveCenter also enables you to run a command line interface (CLI) from which you can issue single commands, open a CLI interactive session, or run a CLI script.

NerveCenter Command Line Interface

You can use a command line interface (a UNIX shell or a DOS cmd window) to add, list, delete, and control nodes, alarms, polls, and masks. You can also connect to, display the status of, and disconnect from NerveCenter servers — either manually or from a script.

The following sections describe how to control NerveCenter from the command line, and the "Command Line Interface Reference" on page 198 describes each specific command. The procedures in this appendix assume that you are in the NerveCenter bin directory or that your search path includes that directory.

Entering a Single Command in a UNIX Shell or a DOS Prompt

TO ISSUE A SINGLE COMMAND IN A UNIX SHELL OR AT A DOS PROMPT

1. Open a UNIX shell or a DOS Command Prompt window.
2. Type

   ```
   nccmd -s ServerName -u UserName -p Password -n Port Command
   ```

   where *ServerName* is the name of the server you want to contact, *UserName* and *Password* are valid for that NerveCenter server, *Port* is the command-line-interface port (32506 by default), and *Command* is the command you want to run. Then press Enter.

Note: User IDs and passwords are encrypted before being sent over the network.

If you are already connected to a server via a previous connect or select server command, the command you issue is directed to that server, and the -s switch is unnecessary.

The command is executed and informs you if an error was encountered. The command syntax is specified in "Command Line Interface Reference" on page 198.

In the following example, a user named John lists all the polls that are suppressed. For this example, the server is named sales, John's user ID is johndoe, and his password is anytown.

```
nccmd -s sales -u johndoe -p anytown -n 32506 list poll -S yes
```

Entering Multiple Commands in an Interactive Session

TO OPEN AN INTERACTIVE SESSION AND ENTER MULTIPLE COMMANDS

1. Open a UNIX shell or a DOS Command Prompt window.
2. Do one of the following:
   - To open a session without connecting directly to a server, type **nccmd** and press Enter.

     The nccmd command prompt (`nc>`) appears.
   - To open an interactive session and connect directly to a server, type

     ```
     nccmd -sServerName-u UserName-pPassword -n Port
     ```

     where *ServerName* is the name of the NerveCenter server you want to contact, *UserName* and *Password* are a valid for that NerveCenter server, and *Port* is the server's command-line-interface port (32506 by default). Then press Enter.

     The nccmd command prompt (**>**) appears and you are connected to the server.

     All subsequent commands are sent to that server until you connect to an additional server or issue a select server command.
3. Type the commands you want to run, pressing Enter after each one.

   The command syntax is specified in "Command Line Interface Reference" on page 198.

   Each time you press Enter, you are returned to the nccmd command prompt. If any command returns an error, nccmd quits, returns you to the shell prompt or the Command Prompt, and displays the error.
4. To exit the interactive session, type **quit** at the prompt and press Enter.

   You are returned to a shell or command Prompt.

In the following example, a user named Kim opens an interactive session, connects directly to the server, deletes a node, disconnects from the server, and quits the session. For this example, the server is named sales, Kim's user ID is kimz, her password is NoMuZak, and the name of the node she is deleting is N0128.

```
C:\>nccmd -s sales -u kimz -p NoMuZak -n 32506
nc>delete node -n N0128
nc>disconnect
nc>quit
```

Running a Script Containing One or More Commands

TO RUN A SCRIPT THAT CONTAINS ONE OR MORE COMMANDS

1. Create a text file that contains the commands in the order you would enter them as if you were in an interactive session.

   The command syntax is specified in "Command Line Interface Reference" on the next page.

2. Open a DOS Command Prompt window or a UNIX shell.
3. Do one of the following:

review - j - check original parsing

- To connect to a NerveCenter server and have the server execute the commands in your file, enter a command of the following form:

  ```
  nccmd -s ServerName -u UserName -p Password -n Port -f Filename
  ```

  ServerName is the name of the NerveCenter server you want to contact, *UserName* and *Password* are a valid user name and password for that NerveCenter server, *Port* is the command-line-interface port for the server (32506 by default), and *Filename* is the name of the file containing the commands you want to run.

Note: If *Filename* is a file name or path name that contains spaces, the name must be enclosed in quotation marks, for example, “C:\Program Files\LogMatrix\NerveCenter\Bin\Cmd.nc”.

- To prevent having to connect to the server explicitly each time you want to run your script, you can edit your script so that its first line is a connection command. This command must have the form:

  ```
  connect -s ServerName -u UserName -p Password -n Port
  ```

  Once you’ve performed this step, you can type **nccmd -f** *Filename* from the DOS or a shell prompt to connect to the server and execute the commands in your script.

  If any command returns an error, nccmd quits, displays the error, and returns you to the Command Prompt or UNIX prompt.

In the following example, a user named Juan writes a script that connects to a server, turns on an alarm named Monitor, and disconnects from the server. For this example, the server is named sales, Juan’s user ID is juanr, and his password is CaFFieNe. The script is named SetAlarm and it looks like this:

```
connect -s sales -u juanr -p CaFFieNe -n 32506
set alarm -e on -w -a Monitor
disconnect -s sales
```

Juan runs the script by entering the following command at the command prompt:

```
>nccmd -f SetAlarm
```

Command Line Interface Reference

You can use NerveCenter commands to control NerveCenter servers, nodes, alarms, polls, and masks from the command line interface from a scriptfile, single command, or interactive session.

The commands are generally made up of an action and an object on which you perform the action. For example, the command **list mask** is made up of the list action, which is performed on the mask object. Command line switches are case-insensitive.

This reference consists of the following sections:

- "Server Commands" below
- "Object Commands" on page 200
- "General Commands" on page 207
- "Notes for CLI Commands" on page 208
- "CLI Command Examples" on page 208

Server Commands

Use these commands to list, select, disconnect from, or display information about connected NerveCenter Servers in your CLI session. These commands are listed in the order that they appear in the syntax diagram.

```
serverCommand [-serverSwitch [parameter]] [-serverSwitch
[parameter]]...|-?
```

serverCommand

One of the several server-related keywords and their switch/parameter pairs described below. *serverCommand* also includes the connect command and its switch/parameter pairs described in "OPTIONS" on page 216 of "nccmd" on page 216.

NerveCenter directs server commands to the active server if no server is specified. The active server is the last server connected to or selected using the select server command.

Table 44: Server Commands

serverCommand	Description	Options

`clear trigger -t` *`triggerName`* `[-i` *`alarmInstanceID`*`]`	Cancels any pending *triggername* triggers that were queued by its own alarm instance. If no alarm instance is supplied, then *triggername* is canceled for all alarms in which it is pending.	`-t` *`triggerName`*
		Name of the trigger to be cleared.
		`-i` *`alarmInstanceID`*
		(Optional) *alarmInstanceID* is a numeric value indicating the alarm instance for which NerveCenter is clearing the specified trigger.

serverCommand	Description	Options
`disconnect`	Disconnects from the active server.	

`fire trigger`
`-t` *`triggerName`*
`-n` *`nodeName`*`|$ANY`
`-o` *`subobjectName`*`|$ANY`
`[-d` *`delay`*`]`
	Fires the specified trigger. If no delay is supplied, then NerveCenter fires the trigger immediately.	`-t` *`triggerName`*
		Name of the trigger to be cleared.
		`-n` *`nodeName`*`|$ANY`
		Name of the node on which to fire the trigger or any node ($ANY).
		`-o` *`subobjectName`*`|$ANY`
		The name of the subobject on which the trigger is fired or any subobject ($ANY).
		`-d` *`delay`*
		(Optional) *delay* is the number of seconds before NerveCenter fires the specified trigger.

`list`	Lists the connected servers.	
`select -s` *`serverName`*	Makes a connected *serverName* the active server.	

`set server`
`{-t` *`timeoutInterval`*`}`
`|{-w` *`width`*`}|-?`
	Allows you to configure how the server commands respond.	`-t` *`timeoutInterval`*
		timeoutInterval is the number of seconds that the CLI waits for the NerveCenter server to acknowledge a CLI request. The default is five seconds (UNIX) and one second (Windows). -t *timeoutInterval* is commonly used in situations where a server is busy or the connection is slow. Should a timeout occur, the connection is still maintained, but the CLI displays the message: **Error. No data was received within allowed time**.
		`-w` *`width`*
		width is the column width (1-50 characters; 12 is the default) for tabular output from the **list** command.
		`-?`
		Displays help for the **set server** commands.

-switch parameter

One of several switch/parameter pairs described below. Parameter switches are case-insensitive and may be preceded by either a hyphen (-) or a backslash (\).

```
show server -serverSwitch
```

Displays information about the active server.

Table 45: Server Switches

-*serverSwitch*	**Description:**
-a	Display information about the NerveCenter administrators connected to the active server.
-c	Display information about the NerveCenter clients connected to the active server.
-d	Display information about the NerveCenter database.
-h	Display information about the Inform host.
-i	Display information about the active server.
-l	Display information about the NerveCenter license key.
-m	Display information about the map host.
-s	Make *serverName* the active server. Used only with the *serverCommand* **select**.
-?	Display help information about the specified *serverCommand*.

Object Commands

Use these commands to delete, list, or set alarms, trap masks, nodes, and polls. These commands are listed in the order that they appear in the syntax diagram.

```
{add | delete | list object {-switch parameter} | -? } |
{set object {-switch parameter -w targetObject} | -? }
```

add node

Adds the specified node. A unique node name (`-n`) and at least one IP Address (`-i`) must be specified. All other parameters are optional and will be set to a default value if omitted.

Usage

```
add node -n (NodeName) -i (IpAddress) [-i IpAddressN] [-p
PortNumber]
        [-m (yes|no)] [-s (yes|no)] [-d (yes|no)] [-g PropertyGroup]
        [-v (1|2|3|u)] [-c ReadCommunity] [-t WriteCommunity]
        [-3 (User1|User2|Local)] [-u v3username] [-z v3context]
        [-l (NoAuthNoPriv|AuthNoPriv|AuthPriv)]
        [-a (SHA|MD5) ] [-h (AuthPassword)]
        [-o (DES|3DES|AES128|AES192|AES256)] [-y (PrivacyPassword)]
```

```
add -?
```

Returns possible options for the 'add' action command.

```
add node -?
```

Returns the syntax description of the add node command.

Required Switches

```
-n NodeName
```

A unique node name for the server. Node names are not case-sensitive.

```
-i IpAddress
```

The IP Address of the node, specified in dotted format. To specify multiple addresses, use multiple *-i* flags with a single address following each. At least one address must be specified. The validity or uniqueness of the IP address is not verified.

Optional Switches

```
[-p SnmpPortNumber]
```

Set the port number for SNMP operations of the new node to the specified value. If this switch is omitted the node will be set to port 161.

```
[-m (yes|no)]
```

Set the new node to be managed (yes), or not managed (no). The default is yes.

```
[-s (yes|no)]
```

Set the new node to be suppressed (yes), or not suppressed (no). The default is no.

```
[-d (yes|no)]
```

Set the new node to be eligible for automatic delete (yes), or not eligible (no). The default is yes.

`[-g PropertyGroupName]`

Set the property group of the new node to the specified value. The PropertyGroup must already exist for the server. If this switch is omitted the PropertyGroup of the node will be set to “NCDefaultGroup”. PropertyGroup names are case-sensitive.

`[-v (1|2|3|u)]`

Set the SNMP version of the new node to the specified value. The default is version 1.

Optional Switches, SNMPv1 or v2 (`[-v 1]` or `[-v 2]`)

`[-c ReadCommunity]`

Set the read community string of the new node to the specified value. This switch will be ignored unless the SNMP version (-v flag) is specified as '1', '2', or 'u'. If this switch is omitted and the node is not version 3 the node will be set to read community 'public'.

`[-t WriteCommunity]`

Set the write community string of the new node to the specified value. This switch will be ignored unless the SNMP version (-v flag) is specified as '1', '2', or 'u'. If this switch is omitted and the node is not version 3 the node will be set to write community 'public'.

Optional Switches, SNMPv3 (`[-v 3]`)

`[-3 (User1|User2|Local)]`

Sets the SNMPv3 type. The default is Local.

`[-l (NoAuthNoPriv|AuthNoPriv|AuthPriv)]`

Set the security level of the new node to the specified value. This switch will be ignored unless the SNMP version (-v flag) is specified as '3'. If this switch is omitted for version 3 nodes, the security level defaults to 'NoAuthNoPriv'.

Optional Switches, SNMPv3 Local definition (`[-v 3 -3 Local]`)

`[-u v3username]`

Sets an SNMPv3 user name.

`[-z v3context]`

Sets the SNMPv3 context. The default is `-z ""` (empty).

`[-a (SHA|MD5)]`

Sets the authentication protocol of the new node to the specified value. This switch will be ignored unless the SNMP version (-v flag) is specified as '3' and the security level has been specified as 'AuthPriv' or 'AuthNoPriv'. If this switch is omitted for version 3 nodes, the authentication protocol defaults to SHA.

`[-h (AuthPassword)]`

Sets an SNMPv3 passkey.

`[-o (DES|3DES|AES128|AES192|AES256)]`

Sets the SNMPv3 privacy mode. The default is DES.

`[-y (PrivacyPassword)]`

Sets the SNMPv3 privacy passkey.

The add node command indicates whether or not the node was added. If successful, the text "'NodeName' was added" is returned. If a failure occurred, the text "Error" is returned, followed by the reason for the failure.

Rules

1. The node name must be provided. `[ -n (NodeName) ]`
2. The node name must not already be in the server's node list.
3. At least one IP Address `[ -i IpAddress ]` must be given.
4. The Property Group, if specified, must be known to the server

Examples

The following five examples demonstrate the basic command structure.

- The simplest form of "add node":

```
NC> add node -n gateway -i 192.168.1.1
```

- The same command, with argument aliases (see "`add node -? aliases`"):

```
NC> add node -nodename gateway -ipaddr 192.168.1.1 -ipaddr
        194.22.19.251
```

- Adding a SNMPv2c router where the Property Group is Mib-II-switch:

```
NC> add node -n switch -I 192.168.1.100 -g Mib-II-switch -v 2
        -c public -t dontTouchThis
```

- Adding a SNMPv3 device, providing the v3 communication configuration:

```
NC> add node -n archiver -i 192.168.1.23 -g Mib-II -v 3 -c public
        -3 Local -u Admin -l authNoPriv -a SHA -h lArg3rThAnL1f3
```

- Adding a node that has two IP addresses but is not to be monitored:

```
NC> add node -n ncserver -i 127.0.0.1 -i 192.168.1.30 -g Mib-II
        -m no -d no
```

The next four examples demonstrate how to create different SNMP node types, with the command output included after each step:

- Adding an SNMPv1 node:

```
NC> add node -n Zeus -i 10.150.1.2 -m yes -s no -d no -g Mib-II
         -v 1 -c public -t private

Zeus was added

NC> list node -n Zeus

Name         Group        Severity     Managed      Suppressed  SNMPVer.
Zeus         Mib-II       Normal       Yes          No          V1
No. of Object(s) = 1
```

- Adding an SNMPv2 node:

```
NC> add node -n Athena -i 10.150.1.3 -m yes -s no -d yes -g Mib-II
        -v 2 -c public -t owl

Athena was added

NC> list node -n Athena

Name             Group    Severity     Managed      Suppressed  SNMPVer.
Athena           Mib-II   Normal       Yes          No          V2
No. of Object(s) = 1
```

- Adding a node where the SNMP version is specified as Unknown:

```
NC> add node -n Aries -i 10.150.1.4 -m no -d yes -s yes -g Mib-II -v u

Aries was added

NC> list node -n Aries

Name             Group            Severity     Managed      Suppressed  SNMPVer.
Aries            Mib-II           Normal       No           Yes         Unk.
No. of Object(s) = 1

NC> list -node -g Mib-II

Name         Group        Severity     Managed      Suppressed  SNMPVer.
Aries        Mib-II       Normal       No           Yes         Unk.
Athena       Mib-II       Normal       Yes          No          V2
Zeus         Mib-II       Normal       Yes          No          V1
No. of Object(s) = 3
```

- Adding an SNMPv3 node with "MotherEarth" as the user name, with the context "EarthMother", and the NoAuth/NoPriv Security Level:

```
NC> add node -n Hera -i 10.150.1.5 -m yes -v 3 -3 Local
         -u MotherEarch -z EarthMother -l NoAuthNoPriv

Hera was added

NC> list node -g Mib-II

Name            Group       Severity    Managed     Suppressed  SNMPVer.
Aries           Mib-II      Normal      No          Yes         Unk.
Athena          Mib-II      Normal      Yes         No          V2
Zeus            Mib-II      Normal      Yes         No          V1
No. of Object(s) = 3

NC> list node -n *

Name        Group       Severity    Managed     Suppressed  SNMPVer.
Hera        NCDefault.. Normal      Yes         No          V3
Aries       Mib-II      Normal      No          Yes         Unk.
Athena      Mib-II      Normal      Yes         No          V2
Zeus        Mib-II      Normal      Yes         No          V1
No. of Object(s) = 5
```

`delete`

Deletes the specified *object*. One switch/parameter pair is required.

`list`

Lists the specified alarm, mask, node, or poll when specified with a switch/parameter pair. When used without a switch/parameter pair, lists the connected and active servers (if any).

`object`

Specifies the type of NerveCenter object that you are deleting, listing, or setting attributes for. *object* can be: alarm, mask, node, or poll.

`-`*`switch parameter`*

One of several switch/parameter pairs described below. Parameter switches can be preceded by either a hyphen (-) or a backslash (\).

`-a` *`name`* `| *`

Identifies a specific NerveCenter alarm or poll (*name*) or all (*) alarms or polls. *name* is case-sensitive. -a does not apply to trap masks. See **-m**, **-n**, and **-p**.

`-a` *`authenticationProtocol`*

Identifies NerveCenter nodes by their authentication protocols. Valid *authenticationProtocol* values are: MD5 (Message Digest algorithm) and SHA (Secure Hash Algorithm). The default protocol is MD5. For privacy, the CBC-DES protocol will be used, since this is the default provided by Envoy.

`-c` *`communitystring`*

Identifies the SNMPv1/v2c Read Community String for nodes. Used by the 'set node' and 'add node' commands."

`-e on | off`

Identifies NerveCenter objects by their state: enabled (on) or disabled (off).

`-g` *`propertyGroup`*

Identifies NerveCenter nodes by their property group.

`-l` *`securityLevel`*

Identifies NerveCenter nodes by their SNMPv3 security level. Valid *securityLevel* values are: NoAuthNoPriv (no authentication protocol, no privacy password), AuthNoPriv (authentication protocol, no privacy password), and AuthPriv (authentication protocol, privacy password).

`-m` *`name`* `| *`

Identifies a specific NerveCenter trap mask (*name*) or all (*) trap masks. *name* is case-sensitive. -m only applies to trap masks. See **-a**, **-n**, and **-p**.

`-m yes | no`

Specifies NerveCenter managed nodes (yes) or unmanaged nodes (no).

`-n` *`name`* `| *`

Identifies a specific NerveCenter node (*name*) or all (*) nodes. *name* is case-sensitive. To display extended SNMPv3 attributes (security level, authentication protocol, and error status), use **-x v3**. See **-a**, **-m**, and **-p**.

`-p` *`name`* `| *`

Identifies a specific NerveCenter poll (*name*) or all (*) polls. *name* is case-sensitive. See **-a**, **-m**, and **-n**.

`-r` *`propertyName`*

Identifies NerveCenter alarms or polls by their property. *propertyName* is case-sensitive.

`-r` *`errorStatus`*

Identifies NerveCenter nodes by their error statuses. Valid *errorStatus* values are: AuthKeyFail, AuthPrivKeyFail, V3InitFail, ClassifyFail, TimeSyncFail

`-s enterprise | node | subobject | instance`

Identifies NerveCenter alarms by their scope.

`-s yes | no`

Identifies NerveCenter nodes their state: suppressed (yes) or unsuppressed (no).

`-t communitystring`

Identifies the SNMPv1/v2c Write Community String for nodes. Used by the 'set node' and 'add node' commands."

`-v` *`SNMPVersion`*

Identifies NerveCenter masks, nodes, or polls by the version of the SNMP agent. Valid *SNMPVersion* values are: 1, 2, or 3.

`-w` *`targetObject`*

Flag indicating that target object follows for the set command.

The arguments preceding the -w switch determine which attributes of an object(s) are set and what they are set to. Those that follow **-w** determine which object(s) are modified. For example, the sequence **-r** *property1* **-w -r** *property2* would change the property of all objects with property2 to property1.

`targetObject`

Switch/parameter pair that identifies NerveCenter objects whose attributes are being set.

`-x v3`

Displays all SNMPv3 nodes' extended attributes (security level, authentication protocol, and error status), which are not displayed by the **list node -n** command.

`-?`

Display help information about the delete/list/set *object* command.

`set`

Sets attributes for a specified object. One switch/parameter pair and **-w** *targetObject* is required.

General Commands

Use these commands to get command line help, quit the NerveCenter CLI, or to resync with the network management platform. These commands are listed in the order that they appear in the syntax diagram.

```
-? | quit | reset alarm_instance -i instanceNumber [-s stateName] | resync
```

`-?`

Displays help or, if used without an action keyword or *object*, displays general information about help. If used with an action alone, gives general command information such as syntax and the objects to which it can be applied. If used with an action and *object*, gives detailed information about the command's syntax.

`quit`

Ends the CLI session and automatically disconnects from each NerveCenter Server.

`reset alarm_instance`

Transitions the specified alarm to the ground state.

`-i` *`instanceNumber`* `[-s` *`stateName`*`]`

Identifies the alarm to be rest to ground by a valid alarm instance number (required) and the state of the alarm (optional).

`resync`

Updates the NerveCenter node list with information from the network management platform. NerveCenter automatically executes a resync at startup and when you reconnect to the platform after the connection was broken.

Notes for CLI Commands

If you leave out any mandatory parameters, nccmd returns an error. In general, nccmd also returns an error if it encounters any syntax errors (for example, unknown or incorrectly specified parameters) and does not issue the command to the connected server. Specific error messages are returned by nccmd if the server encounters errors.

CLI Command Examples

The following command starts a CLI interactive session, connecting to the Blueridge server, with the user ID GRakauskas, and using port 32506:

```
nccmd connect -s blueridge -u GRakauskas -n 32506
```

This command sets the property to mynodes for all alarms that are enabled:

```
set alarm -r mynodes -w -e on
```

The command that follows assigns all nodes belonging to the troublemaker property group to the criticaldevices property group:

```
set node -g criticaldevices -w -g troublemaker
```

In this example, all alarms set to enterprise scope are displayed:

```
list alarm -s enterprise
```

NerveCenter UNIX Shell Commands

D

There are certain NerveCenter commands that you run from a UNIX shell. This section lists each of these commands in alphabetical order.

Note: This appendix assumes that you are in the NerveCenter `/bin` directory or that your search path includes that directory. NerveCenter installation includes this directory in your search path by default.

Caution: This appendix documents the commands that are intended for NerveCenter users to run. Other commands (e.g., ncsnmppoller) are for LogMatrix Customer Support, and if used incorrectly could corrupt your NerveCenter installation.

importutil

NAME

importutil

SYNOPSIS

```
importutil imputil.ini
```

DESCRIPTION

Enables you to reconfigure a setting on more than one NerveCenter Server at a time by changing one file and importing it to all the relevant servers.

OPTIONS

`imputil.ini`

File that contains NerveCenter Server settings that you import to all NerveCenter Servers. imputil.ini resides in the /userfiles directory .

USAGE

imputil.ini is made of a number of sections that include a section header and keys. Before making any changes, create a backup copy of imputil.ini.

Caution: You will not be able to restore the original imputil.ini after making changes to the file, unless you first make a backup copy.

Delete all but the relevant sections to be changed.

All sections in the file are optional. If you remove a section, including the section header and keys, ImportUtil does not change or delete values in the NerveCenter configuration settings for that key.

For example, if you are changing a value found only in the section [CONFIG_SERVER], you delete all sections except the section header and the values in the [CONFIG_SERVER] section. ImportUtil will only change the values pertaining to that section.

Any new values left in imputil.ini will overwrite the old values. To avoid having placeholders overwrite legitimate values, delete any unnecessary keys before running ImportUtil.

For example, if within the [CONFIG_SERVER] section you only want to change the value of the key InformNCListenPort, delete all but the following:

```
[CONFIG_SERVER]

InformNCListenPort = port
```

Caution: If you are configuring either [CONFIG_PLATFORM_NETNODENOTIFY] or [CONFIG_ PLATFORM_MAPSUBNETS], you need to include all values, including old values. ImportUtil deletes values from the NerveCenter configuration settings that are not included in these sections. Please read the comments before each section in this file for more information.

Change the values by replacing the placeholders after the equal sign (=) with valid values. Unless otherwise noted, you may not leave the value after a key blank.

Note: You must either be in the same directory as the imputil.ini file or include the full pathname of the imputil.ini file.

NerveCenter notifies you upon successful completion of the reconfiguration. For more information about each section in imputil.ini, refer to Table 46.

Table 46: inputil.ini Value Descriptions

imputil.ini value pair	Description
[CONFIG_PLATFORM_MAPSUBNETS] This section configures the IP filtering for node lists imported from a network management platform. If the method in [CONFIG_PLATFORM] is AUTO, the entries in this section are ignored. **Caution:** You must include all values from this section, including old values. ImportUtil deletes values from the NerveCenter configuration settings that are not included in these sections. **Note:** If you want to apply filters to a list of nodes you are importing, first run importutil to change the filters. Then run importutil again to import the nodes.	

MapSubNet1 = Address;Mask;Exclusion list
MapSubNet2 = Address;Mask;Exclusion list
	You can filter nodes by subnet. For more details about this value, see "Filtering Nodes by IP Address" on page 70.

[CONFIG_PLATFORM_NETNODENOTIFY] This section configures what information NerveCenter sends to the Network Management Platform. **Caution:** You must include all values from this section, including old values. ImportUtil deletes values from the NerveCenter configuration settings that are not included in these sections.	
host/port/type = EVENT_ ONLY | EVENT_AND_ SYMBOL | SYMBOL_ONLY [Min Severity] [Properties]	The type of information sent to the network management platform. For more details about these values, see the NerveCenter Platform Integration Guide.
[CONFIG_SERVER] This section configures the NerveCenter Server.	
ServerConnPort = port	The NerveCenter Server communicates with other applications, such as NerveCenter Client and NerveCenter Administrator, on a special connection port. For more details about this value, see "Configuring the NerveCenter Server Connection Port" on page 48. If you change a NerveCenter Server's communication port number, be sure all applications, such as NerveCenter Administrator, have a matching port number. See "Changing the NerveCenter Administrator Server Port" on page 49 for more details.

imputil.ini value pair	Description
CLIConnPort = port	The port number you want the NerveCenter Server to use for client/server communication. For more details about this value, see "Changing the NerveCenter Client Server Port" on page 50.
InformNCListenPort = port	Before a NerveCenter Server can receive an Inform, it must be configured to have a listening port number. For more details about this value, see "Configuring NerveCenter to Receive Inform Actions" on page 51.
NCMibName = MibFileName	The filename of the mib file you want to import into NerveCenter. You enter the path in [CONFIG_SERVER_PATH].
EnableDiscovery = TRUE | FALSE	If true, NerveCenter adds nodes discovered by traps to the database. For more details about this value, see "Adding Nodes Discovered from Traps" on page 88.
DiscoverNodesFromTraps = All | Filter | None	For more details about this value, see "Populating Using the IPSweep Behavior Model" on page 83.
ApplyAllMasksForEachTrap = TRUE | FALSE	Set to TRUE if you want NerveCenter to process every incoming SNMP trap against all defined trap masks that are currently enabled. Selecting this option forces masks to process traps regardless of the state of their associated alarms. A mask processes traps even when its associated alarm is turned off or is not in a state that can be transitioned by the mask's trigger. There are times, however, when masks do not process traps though this option may be checked. A mask processes traps only when the following conditions are met: ■ The mask is turned on. ■ The incoming trap matches the generic, specific, or enterprise OID values defined in the mask.
[CONFIG_SERVER_PATH] Path information.	
MibDirectory = Location of Mib	The path to the mib file you want to import into NerveCenter. You enter the filename in [CONFIG_SERVER].
LogDirectory = Location of logs	The location of any log created by a Behavior Model or other component of NerveCenter. For details about this value, see "Specifying Settings for Log Management" on page 149.
[CONFIG_SERVER_SNMP] This section configures SNMP settings.	
SnmpRetry = No. of retries	The number of times to reissue unanswered SNMP request polls. For more details about this value, see "Specifying SNMP Poll Intervals for NerveCenter" on page 109.

imputil.ini value pair	Description
SnmpInterval = No. of Intervals	The number of seconds NerveCenter should wait for a reply to a poll before issuing another. For more details about this value, see "Specifying SNMP Poll Intervals for NerveCenter" on page 109.
DefaultPort = port no.	The port number NerveCenter uses for SNMP traps. For more details about this value, see "Specifying SNMP Ports for NerveCenter" on page 55.
[CONFIG_SERVER_SNMPV3] This section configures the SNMPv3 settings.	
PollUser = poll user	The SNMPv3 user name. For more details about this value, see "Configuring SNMPv3 Security Settings" on page 118.
PollContext = poll context	The SNMPv3 context. For more details about this value, see "Configuring SNMPv3 Security Settings" on page 118.
AutoClassify = on | off	Whether NerveCenter attempts to classify the SNMP format of discovered nodes. For more details about this value, see "Classifying Nodes Automatically" on page 105.
AuthPwd = unencrypted (plain text) authentication password	The Authentication password for an SNMPv3 node. For more details about this value, see "Configuring SNMPv3 Security Settings" on page 118.
PrivPwd = unencrypted (plain text) privacy password	The Privacy password for an SNMPv3 node. For more details about this value, see "Configuring SNMPv3 Security Settings" on page 118.
MaxClassifyVer = v1 | v2c | v3	If autoclassification is enabled, you can limit the level to which NerveCenter attempts to classify nodes. For more details about this value, see "Setting a Maximum Classify Value" on page 107.
MaxReqPerCycle = Max no. of requests going out of v3opmanager per second	You control SNMPv3 performance by setting a maximum number of requests per processing cycle (approximately one second) for all v3 operations including classification, SNMP test version poll, version change, authentication protocol change, security level change, and initialization requests. For more details about this value, see "Setting the Maximum SNMPv3 Requests per Cycle" on page 108.
[CONFIG_SERVER_LOGS] This section configures the NerveCenter Logs.	
MaxLogEntryAge =	The number of hours you want to keep individual log entries before they are deleted from the rest of the log. For more details about this value, see "Specifying Settings for Log Management" on page 149.

imputil.ini value pair	Description
MaxLogFileSize =	The size limit, in kilobytes, of the ASCII text file storing the results of the Log to File alarm action. For more details about this value, see "Specifying Settings for Log Management" on page 149.
MaxNumDBRecords =	The highest number of records in the database file storing the results of the Log to Database alarm action. For more details about this value, see "Specifying Settings for Log Management" on page 149.
LogDelPercentage =	The percentage of the log to clear when the maximum file size or the maximum number of records is reached. For more details about this value, see "Specifying Settings for Log Management" on page 149.
MaxQueueDepth =	The highest number of changes that you want queued before saving to the database. For more details about this value, see "Specifying Settings for Log Management" on page 149.
[CONFIG_SERVER_SMTPMAIL] This section configures SMTP mail.	
SMTPHost =	The name of the host running your SMTP mail server. For more details about this value, see "Specifying an SMTP Server for Mail Notification" on page 147.
[IMPORT_MODEL] Use this section to import a NerveCenter Behavior Model.	
File = model_path_and_filename	For more details about this value, see Importing Node, Object, and Behavior Model Files in Designing and Managing Behavior Models.
[IMPORT_NODE] Use this section to import a NerveCenter Node List.	
File = node_path_and_filename	For more details about this value, see Importing Node, Object, and Behavior Model Files in Designing and Managing Behavior Models.

mibTool

NAME

mibTool

SYNOPSIS

```
../bin/mibTool [-o filename] [-loc path ] [-inc mib_name] -mibcomp
                                filename [-namemap] [-strict] [-e filename]
```

DESCRIPTION

mibTool is the NerveCenter MIB compiler that is installed with the Server component. mibTool produces a compiled NerveCenter MIB identical to the one produced by mibcomp in prior releases. mibTool, however, provides a marked performance improvement and has a far greater allowance for SNMP syntax variances as compared to mibcomp.

mibTool is a Java application and requires that Java 1.8 or higher is installed on the system from which it is run. NerveCenter 5.1 supports the prior MIB compiler, mibcomp, though support will be withdrawn in the next release.

OPTIONS

`-o` *`filename`*

Specifies the output filename for compiled MIB information (nervectr.mib, by default).

`-loc` *`path`*

The directory being used as a MIB repository; all MIBs in this directory are checked if an entity lookup fails.

`-inc` *`mib_name`*

Includes the specified MIB in the output file.

`-mibcomp` *`filename`*

Specifies the file that contains the MIB file paths to compile (mibcomp.txt by default).

`-namemap`

Produces the mibnamemap.txt file for use with older versions of NerveCenter.

`-strict`

Compiles in strict mode, which succeeds only if all imports are completely accurate.

`-e` *`filename`*

Writes compilation errors to the specified file.

ERROR MESSAGES

Duplicate entity detected

Multiple mib objects were detected with the same name but different OIDs. All objects under the enterprises OID will be renamed to *MIB-NAME_entityName* to allow the objects to be accessible from within NerveCenter.

Not renaming "mib object"

A duplicate entity was found but its OID was not under enterprises therefore the entity was not renamed as described in the "Duplicate entity detected" message.

Duplicate OID found

Multiple mib objects with different names were found using the same OID. The compiler will use the first name it encounters and discard the remaining duplicates.

Requested entity "mib object" imported from MIB was found in ATL-MIB instead

The compiler could not find the mib object in the file specified in the imports clause, but it was found in another mib file. This behavior only occurs when compiling without the `-strict` flag.

No such file or directory

A file specified on the command line or within the mibcomp.txt file could not be found.

nccmd

NAME

nccmd

SYNOPSIS

```
nccmd [connect -s server -u userID [-p password] -n port [-f scriptFile |
CLIcommand]] | [-?]
```

DESCRIPTION

Runs the NerveCenter command line interface (CLI). You can issue single commands, open a CLI interactive session, or run a CLI script.

OPTIONS

`connect`

Connects to the specified server. Once you are connected to a server, all subsequent commands are directed to that server.

`-s` *`server`*

Name of the server to which you want to connect. Required.

`-u` *`userID`*

User ID under which you want to connect to the server. The default is the user ID specified during the previous connection or by the nccmd command line parameter, if any. Required.

`-p` *`password`*

Password for the user ID you specified. The default is the password specified during the previous connection or by the nccmd command line parameter. **-p** is required for sites that use NerveCenter security.

`-n port`

Number of the port you want to use for client/server communication. The default command-line-interface port is 32506. Required.

`-f scriptfile`

Filename of the script or batch file containing CLI commands.

Note: If *scriptfile* is a file or path name that contains spaces, it must be enclosed in quotes.

To avoid connecting to the server each time you run your script, you can edit your script so that its first line is a connection command. Once done, you can type **nccmd -f** *scriptfile* from a Windows cmd prompt or UNIX shell to connect to the server and execute the script. If any command returns an error, nccmd quits, displays the error, and returns you to the Windows command prompt or UNIX shell prompt.

`CLIcommand`

Any one of several command line interface commands listed in the "Command Line Interface Reference" on page 198.

`-?`

Provides command-line help for the nccmd command.

nccheckdirs

Name

nccheckdirs: Script to check and correct NerveCenter directory ownership and permission settiongs.

Synopsis

```
$ /opt/OSInc/nc/bin/nccheckdirs
```

Description

Checks the ownership and permission settings of core directories used by the installed NerveCenter service. If run with the ‘root’ account, automatically corrects any encountered issues; running nccheckdirs with a non-root account only presents a list of issues.

ncget

Name

ncget: Utility to perform SNMP queries.

Synopsis

```
$ /opt/OSInc/bin/ncget -help
$ /opt/OSInc/bin/ncget -v1 [OPTIONS] target community varbind(s)
$ /opt/OSInc/bin/ncget -v2c [OPTIONS] target community varbind(s)
$ /opt/OSInc/bin/ncget -v3 [OPTIONS] target username varbind(s)
```

Description

Can only be used when NerveCenter service is running (see ncstatus). Performs live SNMP operation(s), as specified with OPTIONS and to retrieve the listed varbind(s) from the named destination. The overall functionality is very similar to `snmpget` and `snmpwalk` as found in the Linux `net-snmp-utils` package. See `ncget -help` for examples and descriptions of OPTIONS and how varbinds are given. Can only be run by accounts which are members of group `ncadmins`.

nclistener

Name

nclistener: Utility to listen for incoming SNMP notifications.

Synopsis

```
$ /opt/OSInc/bin/nclistener [n]
```

Options

n

The number of seconds to run before exiting. If omitted, nclistener runs continuously.

Description

Prints to the terminal window a decomposition of received SNMP notification traffic (Trap and Inform PDUs). Can only be run by accounts which are members of group ‘ncadmins’.

ncpermissions

Name

ncpermissions: Script to setup NerveCenter file and directory ownerships and permissions.

Synopsis

```
# /opt/OSInc/nc/bin/ncpermissions
```

Description

This script is used as part of the post-installation steps; however it can be run later to help return a NerveCenter installation back to its original file and directory ownership and permission settings. It may only be run from the root account.

ncping

Name

ncping: Utility to invoke a ICMP ‘ping’ operation.

Synopsis

```
$ /opt/OSInc/nc/bin/ncping  target(s) [OPTIONS]
```

Options:

`-t`

timeout value, in seconds, to wait for a response. default is 3 seconds. Timout can be expressed as x.yy where x is the number of seconds and yy is the hundredths of a second (centiseconds).

`-r`

retries value, the number of retries to make when timeout occurs

`-ttl`

the IPv4 TTL or IPv6 Hop Count

`-s`

size value, used to set the size of the `ping` request packet.

Description

NerveCenter implementation of `/bin/bin` from the Linux `iputils` package. Can only be run when the NerveCenter Service is running (see ncstatus), and only by accounts which are members of group ‘ncadmins’.

Example:

```
$ /opt/OSInc/bin/ncping 192.168.1.1 192.168.1.242 192.168.1.3
listening for responses (next 3.00 seconds)
Sending to 192.168.1.1
192.168.1.1 is alive  [6ms.]
Sending to 192.168.1.242
192.168.1.242 is alive  [6ms.]
Sending to 192.168.1.3
timeout!
```

ncrelease

Name

ncrelease: Displays the version information for the installed release of NerveCenter

Synopsis

```
$ /opt/OSInc/nc/bin/ncrelease
```

ncstart

Name

ncstart

Synopsis

```
$ /opt/OSInc/bin/ncstart [OPTIONS]
```

Options

`-counters`

Writes data once a minute to `/opt/OSInc/userfiles/logs/TraceCounters.log`

`-queues`

Writes data once a minute to `/opt/OSInc/userfiles/logs/TraceQueues.log`

`-nogetbulk`

Disables any use of SNMP GetBulk for this run of NerveCenter.

Description

Starts the NerveCenter Service, if not already running. Can be run by the 'root' account or by an account that has been setup to run NerveCenter Service.

SEE ALSO

"ncstop" on the next page

ncstatus

Name

ncstatus: Provides summary information about a running NerveCenter Server.

Synopsis

```
/opt/OSInc/bin/ncstatus [ OPTIONS ]
```

Options

`-p`

Show the process tree of the running NerveCenter Service

`-s`

Show summary of running NerveCenter Service

Description

ncstatus is not present on NerveCenter 5.x or 6.x systems; it can be downloaded from http://docs.logmatrix.com/NerveCenter/6.2.00/index.html#tools. Runs on RHEL4/5/6 for NerveCenter v5.x and v6.x.

Examples

```
$ /opt/OSInc/bin/ncstatus
  nervecenter service running (pid 2467)
$ /opt/OSInc/bin/ncstatus  -p
  Processes:
  ncserver (pid 2467)  0.6 %cpu, 291648 KiB
  ncsnmppoller (pid 2768)  0.6 %cpu,  16476 KiB
  brassd (pid 2796)  0.0 %cpu,   4256 KiB
  brassd (pid 5862)  0.0 %cpu,   4200 KiB
  ncsnmpagt (pid 17819)  0.0 %cpu,  12840 KiB
```

ncstop

NAME

ncstop

SYNOPSIS

```
$ /opt/OSInc/bin/ncstop
```

DESCRIPTION

Stops the running NerveCenter Service. Can be run by the ‘root’ account or, if ncstart was run by a non-root account, then by that same non-root account.

SEE ALSO

"ncstart" on page 220

ncversion

Name

ncversion: Displays version information about installed NerveCenter components

Synopsis

```
$ /opt/OSInc/bin/ncversion [-a] [COMPONENT]
```

Options

`-a`

Show installed product version information, including build id.

`COMPONENT`

Can be one or more NerveCenter shell commands, separated by spaces.

Examples

```
$ /opt/OSInc/bin/ncversion
  LogMatrix NerveCenter 6.2.00
$ /opt/OSInc/bin/ncversion -a
  LogMatrix NerveCenter 6.2.00 6200 BLD13
$ /opt/OSInc/bin/ncversion ncserver ncget
  LogMatrix NerveCenter 6.2.00
  ncserver Version 6.2.00 (6200 BLD13 HF20170510A), Linux(el3)/x86
(32-bit) Copyright (C) 1989-2015 LogMatrix Inc.
  ncget Version 6.2.00 (6200 BLD13), Linux(el6)/x86-64 (64-bit)
Copyright (C) 1989-2015 LogMatrix Inc.
```

paserver

NAME

paserver

SYNOPSIS

```
paserver [-d] [-g] [-h|-?] [-n ON|OFF] [-nhost] [-nport] [-o] [-p]
                        [-r] [-scm {a|m|r|s}] [-t] [-tcfg] [-thost] [-toneway]
                        [-tport] [-ttype] [-u ON|OFF] [-v]
```

DESCRIPTION

Runs the NerveCenter Universal Platform Adapter for integrating NerveCenter with IBM Tivoli Netcool/OMNIbus.

OPTIONS

`-d`

Runs the Universal Platform Adapter from the command line in debug mode and outputs debug messages to the console. The next time the host machine boots, the Universal Platform Adapter will run as a service or daemon again.

`-g`

(Windows only) Registers the Universal Platform Adapter as an Event Source.

`-h | -?`

Displays command line help for the Universal Platform Adapter switches.

`-n`

Enables or disables NerveCenter integration with IBM Tivoli Netcool/OMNIbus.

On Windows, when starting paserver from the command line, you must specify either **-d** or **-scm** *option* in combination with either **-n ON** or **-u ON**.

`-nhost`

Defines the machine on which the Netcool probe is located. The default is localhost.

`-nport`

Defines the port the NerveCenter platform adapter uses to communicate with IBM Tivoli Netcool/OMNIbus. The default is 32510.

`-o`

(Windows only) Records values into Registry. Any options (other than -scm) become a part of the standard configuration.

To use this switch, you should first stop the Universal Platform Adapter. You must restart the Universal Platform Adapter.

`-p`

Defines the platform adapter's listening port. The default is 32509.

Note: This number must match the one in NerveCenter Administrator.

`-r`

(Windows only) Shortened version of -scm r. Removes the Universal Platform Adapter as a service. It also removes Registry entries created at install time.

`-scm a|m|r|s`

(Windows only) Changes settings in the service control manager.

`a`

(Windows only) Installs the Universal Platform Adapter as a service, marking it as autostart. The service will start following this command.

`m`

(Windows only) Installs the Universal Platform Adapter as a service or daemon, marking it as start on demand. The service will not start following this command.

`r`

(Windows only) Removes the Universal Platform Adapter as a service. If it the service is running, it will be stopped.

`s`

(Windows only) Starts the Universal Platform Adapter as a service. This may be combined with **a** or **m**.

`-tcfg`

The full qualified path/filename for the Event Adapter configuration file. The default is /opt/OSInc/userfiles/nctec.cfg

`-thost`

The machine on which the Tivoli Event Server is located. The default is localhost.

`-tport`

The port the NerveCenter platform adapter uses to communicate with Tivoli Event Server. The default is 32510.

`-ttype`

The type of connection to be established with the Tivoli Event Server:

- 0 Connectionless delivery to agent.
- 1 Connection-oriented (secure) delivery to agent.
- 2 Default type of connection.

The default is 0.

`-v`

Views current Universal Platform Adapter settings.

NOTES

If you want these settings to take effect every time the Universal Platform Adapter is started, edit the file pastart to include these switches. pastart resides in NerveCenter \bin directory.

On Windows, when starting paserver from the command line, you must specify either **-d** or **-scm** *option* in combination with either **-n ON** or **-u ON**.

trapgen

Name

trapgen: Utility to create and send a SNMP notification

Synopsis

```
$ /opt/OSInc/bin/trapgen [OPTIONS] destination [OPTIONS] [VARBINDS]
```

Description

Much like the Linux `net-snmp-utils` utility's `snmptrap` command, creates an SNMP v1, v2c, or v3 Trap or Inform message using the parameters given and issues it to the given destination. This command may only be used while the NerveCenter Service is running (see ncstatus), and only by accounts listed as members of group ‘ncadmins’.

See ‘trapgen -help’ for examples and descriptions of options and varbind syntax.

Traprcv

NAME

traprcv

DESCRIPTION

The traprcv command displays the SNMP Trap messages received by the NerveCenter Trap service. This utility can be useful when debugging behavior models. When you run the traprcv command from a command line or shell prompt, any trap you receive is shown on the screen.

Traprcv can receive the following traps and informs:

- SNMPv1 Traps
- SNMPv2c Traps
- SNMPv2c Informs
- SNMPv3 Traps
- SNMPv3 Informs

Glossary

ACTION ROUTER

Performs standard NerveCenter alarm actions in response to detected network conditions and based on additional conditions that you specify. In NerveCenter Client, you first define a set of conditions along with the actions to be completed. Then, you can assign the Action Router action to an alarm transition. When the transition occurs, NerveCenter evaluates all existing conditions to determine whether any have been met and, therefore, whether one or more conditional actions should be performed. All alarms that are sent to the Action Router are evaluated against all the current rules.

See also "alarm action" below.

ACTION ROUTER RULE

An Action Router rule is composed of a conditional statement and one or more actions. The conditional statement can contain one or more conditions logically joined together.

ADMINISTRATOR RIGHTS

Users with administrator logon rights can customize NerveCenter. In the NerveCenter Client, administrator rights are required to create or modify the objects used in behavior models. Those with administrator rights belong to the NerveCenter Admins (ncadmins) group on UNIX.

See also "user rights" on page 240.

ALARM

A NerveCenter object that detects a trigger generated by a poll, trap mask, Perl subroutine, or other alarm. An alarm is a finite state machine that transitions from one state to the next and performs any actions assigned to the transition. Each transition is triggered by its own set of network data, which is defined in the associated trigger generator. When an alarm detects its first trigger, the alarm transitions to the next state, where it remains until another trigger is received—either from the same or another trigger generator. The sequence of transitions enables NerveCenter to monitor persistent, simultaneous, or sequential events that, taken together, could indicate a critical or important condition.

ALARM ACTION

A NerveCenter automated response that helps you manage network activity and stay informed about network conditions. You can assign one or more actions to any transition in an alarm state diagram.

Actions fall into four main categories: notification, logging, triggering other alarms, and correcting network conditions. Examples include sending e-mail, issuing a page, logging data, sending an SNMP trap, executing a command, and sending an Inform message to a network management platform. In addition, NerveCenter actions can be performed conditionally based on criteria that you specify using the Action Router.

ALARM CIRCULAR TRANSITION

A circular transition is one in which the From State and To State are the same for a transition in an alarm state diagram. When a trigger containing the polled data arrives at this transition, the transition occurs and returns the alarm to the same state. The effect is that the alarm continues looping with each instance of the trigger. This type of transition usually includes an alarm counter action that tracks the number of occurrences and fires a trigger when the number reaches a certain value.

ALARM INSTANCE

A single instance of a detected network event. Each instance is one active occurrence of an alarm definition that tracks a current network or system condition through its own copy of the alarm's state diagram. For example, one alarm might have five distinct alarm instances, each tracking the same condition on a different node.

See also "alarm scope" below.

ALARM INSTANCE ID

The unique identifier for an alarm instance managed by a NerveCenter Server. The instance ID is listed in the Alarm History window for an alarm instance.

NerveCenter includes a variable $AlarmInstanceID that you can use in Perl subroutines, command actions, and log actions to specify the instance ID associated with an alarm.

ALARM SCOPE

A setting that determines whether an alarm instance monitors a subobject (for example, a port), several MIB objects on a subobject, a node, or an entire enterprise. Scope is assigned in an alarm's definition window.

If an alarm is using node scope, each alarm instance tracks the alarm's states for a single, distinct node. If an alarm is using subobject scope, each instance tracks the alarm's states for a MIB base object on a node, for example, an interface on a router. Instance scope alarms track instances for every interface or port that fits the polled condition regardless of the base object. Enterprise scope alarms track an event for the enterprise as a whole.

ALARM SEVERITY

An indication of the urgency of a detected event. When creating an alarm, you assign each state a severity level, which is defined by a name and unique color. NerveCenter ships with severity options ranging from normal to critical for a fault condition, and from very low to saturated for a traffic condition. If your network management platform has event severities, you can map NerveCenter's severities to match those on your platform.

ALARM STATE

Corresponds to a network condition that an alarm is monitoring. To monitor certain network conditions, or states, you must specify these states in the alarm's state diagram. Each state listens for certain triggers. Once the correct trigger is fired, the alarm transitions to the corresponding state.

ALARM STATE DIAGRAM

Specifies which detected conditions are correlated, in what order, and which responses (if any) are assigned to each stage. For each alarm that you create, you design a state diagram to specify the states you want to monitor. The state diagram can detect persistent, simultaneous, or sequential events that, taken together, indicate a critical or important network condition. Incoming triggers transition the alarm from one state to another.

ALARM SUMMARY VIEWS

NerveCenter Client provides two views for monitoring network activity. Both windows can be opened from the Admin menu or the toolbar:

Alarm Summary - In the Alarm Summary window, you monitor alarm instances for the connected server. If you are connected to more than one server, you can change the active server and view the Alarm Summary window for the new server. To do this, select the server you want to view from the Active Server drop-down listbox on the NerveCenter toolbar.

Aggregate Alarm Summary - In the Aggregate Alarm Summary window, you can monitor alarm instances collectively for all servers to which you are connected.

ALARM TRANSITION

A change from one alarm state to another, prompted by a "trigger" on page 239. When an alarm transitions from its ground state, the transition creates an alarm instance. Further triggers affect the current alarm instance, each generating a transition in the alarm. During a transition, NerveCenter carries out any alarm actions that are assigned to the transition.

ASN.1 FILE

MIB base objects are defined using Abstract Syntax Notation One (ASN.1), a language that's understood by network management protocols. The ASN.1 language is described in the ISO documents ISO.8824 and ISO.8825.

See also "Management information base (MIB)" on page 233.

ATTRIBUTE

NerveCenter terminology includes two meanings for the word attribute:

A MIB object that contains an actual value. For example, sysContact, ifInOctets, and ipInDiscards are all attributes in the industry-standard MIB-II because they contain strings, numbers, or other values. The value associated with an attribute can be static (for example, the speed of the interface) or dynamic (for example, an entry in a routing table).

A value that you assign to a NerveCenter object. For example, if you assign a property to a poll, that property is an attribute of the poll. NerveCenter's Set Attribute alarm action allows you to change certain node, poll, trap mask, or alarm attributes when a specified network condition is detected.

AUTHENTICATION

In SNMPv3 communication, the process of confirming the parties (i.e. entities) that communicate with each other as well as the timeliness of the messages received at an SNMPv3 entity.

AUTHENTICATION KEY

The seed used to generate a message digest, as specified in the authentication protocols, to authenticate SNMPv3 messages. Both sender and receiver generate the digest, and the receiver matches the generated digest against the digest accompanying the message received to authenticate the message.

AUTHENTICATION PROTOCOL

Protocol used with SNMPv3 communication that allows you to verify the sender and timestamp of a message. Two authentication protocols are currently defined: HMAC-MD5-96, which is based on MD5, and HMAC-SHA-96, which is based on SHA-1 cryptographic algorithms. NerveCenter supports both protocols.

AUTHNOPRIV

Security level that requires message authentication services to be used while communicating with an SNMPv3 entity.

AUTHPRIV

Security level that requires both the message authentication and message encryption services to be used while communicating with an SNMPv3 entity.

BASE OBJECT

See "MIB base object " on page 233.

BEHAVIOR MODEL

The group of all alarm, trigger generator, and property group definitions required to detect and handle a particular network or system behavior. A typical behavior model consists of an alarm with all its supporting trigger generators, though behavior models can have multiple alarms. Any NerveCenter object can be associated with one or more behavior models.

You can customize the behavior models that ship with NerveCenter as well as create new behavior models.

BOOTS

See "engine boots" on the facing page.

CBC-DES

A privacy protocol as specified in SNMPv3 specifications to be used for message encryption in communication between two SNMPv3 entities with AuthPriv security level. This protocol is supported by NerveCenter.

CONTEXT

Every communication between two SNMPv3 entities takes place on behalf of a *user* (a uniquely identified entity in the SNMPv3 management domain), and potentially in some *context* (a uniquely identified entity for access control generally configured on conventional SNMPv3 agents/nodes) with any one of three security levels. These three parameters (user, context, and security level) are used together by VACM (View based Access Control Mechanism defined in SNMPv3 specifications) to grant access to any MIB data at the agents/nodes. Context can be thought of as the MIB information available to a particular user who seeks information from an agent using a certain security level. NerveCenter requires a context only if defined by the agent.

CORRELATION EXPRESSION

A Boolean expression you can use to create alarms.

DES

A privacy protocol as specified in SNMPv3 to be used for message encryption in communication between two SNMPv3 entities with AuthPriv security level. This protocol is supported by NerveCenter. See also CBC-DES.

DIGEST

The hashing code generated for message authentication using authentication key. This digest is appended to the message being sent out by a sending SNMPv3 entity. The receiving SNMPv3 entity separate out this digest from the message, generate the digest again from the message using the locally available authentication key and then compares the two digests for message authentication.

DISCOVERY

The process of discovering nodes and adding them to the NerveCenter database. When you enable discovery, if the NerveCenter database does not already contain a node matching the source of an SNMP trap or NerveCenter Inform, it adds that node to the database. You can also customize NerveCenter's Discovery behavior model, which uses a TCP/IP sweep program (ipsweep.exe) to populate your node list. Alternatively, the NerveCenter node list can be populated from your network management platform.

DOWNSTREAM ALARM SUPPRESSION

A NerveCenter behavior model that uses parent-child topology information to determine the status of nodes and suppress polling on nodes that are down.

ENGINE BOOTS

Number of times an SNMPv3 engine has been started or re-initialized (i.e., booted) since its engine ID was last configured.

ENGINE ID

An SNMPEngineID value with a length of up to 32 octets that uniquely identifies an SNMPv3 engine within the SNMPv3 management domain. There is one SNMP engine ID for every instance of a NerveCenter Server.

ENGINE INFORMATION

An SNMPv3 agent's engine ID, boots, and time ticks.

ENGINE TIME TICKS

As defined in SNMPv3 specifications, number of seconds elapsed since the last engine boot. When this counter reaches its maximum limit, it is reset to 0 and the engine boots value is incremented by 1.

ENTERPRISE SCOPE

A setting for an alarm definition that determines that there will be at most one alarm instance monitoring the entire enterprise.

See also "alarm scope" on page 228.

ESCAPE CHARACTER

If you enter an expression that contains a period, NerveCenter interprets the string as if it were a BaseObject.Attribute. In order to use a period in any other context, you must use an escape character (\) with the period. This applies to filenames, IP addresses, or any other string containing a literal "." character. For example, if your script refers to a fully qualified host name, you might enter something like the following:

```
if ($nodeName ne "myhost\.mydomain\.com" {
FireTrigger (mytrigger);
}
```

EVENT

A detected network or system condition. An event can be forwarded to one or more NerveCenters and network management platforms.

GENERIC TRAP NUMBER

Indicates the nature of an SNMP trap detected by a NerveCenter trap mask. There are seven different types of traps, numbered 0 through 6. The first six (0-5) are industry-standard traps and are described in any standard SNMP text. NerveCenter also provides an option for selecting all six traps. Trap number 6 is reserved for enterprise-specific traps and NerveCenter Inform actions.

See also "specific trap number " on page 238.

HMAC-MD5-96

See "MD5" on the facing page.

HMAC-SHA-96

See "SHA" on page 237.

INFORM MESSAGE

A message indicating a specified network condition. You can configure alarms to send a NerveCenter Inform when a particular network event is detected. NerveCenter can send the Inform data to one or more NerveCenters and network management platforms.

In NerveCenter Administrator, you specify the Inform recipients in the Inform tab and determine which types of events are forwarded (for example, those above a specified severity level). This allows you to manage which platforms handle which events. You can use separate criteria for each platform that is to receive events and coordinate alarms from multiple NerveCenters so that one central NerveCenter determines which events should be forwarded to the platform.

INSTANCE

The identity of a particular MIB base object when there are multiples of that base object on a node. For example, if a managed node has four ports, it has four instances of the ifEntry base object, numbered one through four.

In NerveCenter, a base object and its instance is called a subobject.

INSTANCE SCOPE

A setting for an alarm definition that lets you monitor one or more base objects in an alarm instance. Instance scope is similar to Subobject scope, though Instance scope lets you monitor any instance for different base objects.

See also "alarm scope" on page 228.

IP ADDRESS

The 32-bit host address defined by the Internet Protocol in STD 5, RFC 791. It is usually represented in dotted decimal notation, for example, 192.164.10.0.

LOGON RIGHTS

NerveCenter Client users are grouped into two categories, depending on their logon rights:

- Those with administrator rights can customize and create NerveCenter behavior models and save all changes to the NerveCenter database. These users belong to the NerveCenter Admins group (ncadmins).
- Those with user rights belong to the NerveCenter Users group (ncusers). They perform such tasks as monitoring and resetting alarms. They cannot modify the NerveCenter database, for example, by making changes to nodes, property groups, polls, masks, or alarms.

MANAGED NODE

A node that can be targeted by NerveCenter polls. Only managed nodes are polled. SNMP traps, however, are received from nodes regardless of their managed state.

See also "node" on page 235.

MANAGEMENT INFORMATION BASE (MIB)

A defined collection of device information that's governed by SNMP. An agent's MIB contains the configuration and status values for the particular type of device. A specific type or class of management information is called a MIB base object.

MIB-II defines the common set of objects for network management of TCP/IP-based intranets. Other enterprise-specific MIBs can be defined to support specific pieces of hardware.

See also "Simple Network Management Protocol (SNMP) " on page 237.

MAP

The graphic representation of your network and its devices that a network management platform provides.

MASK

See "trap mask" on page 239.

MD5

Message authentication/hashing algorithm as defined in HMAC-MD5-96 specifications, intended for secure message exchanges between any two entities communicating with each other. The algorithm takes a message of arbitrary length and produces a 128-bit message digest. This is one of the two algorithms suggested in SNMPv3 specifications for message authentication. This protocol is supported by NerveCenter.

MIB

See "Management information base (MIB)" above.

MIB BASE OBJECT

An object that is managed by a network manager. Base objects contain attributes that have values. In MIB-II, for example, system, ifEntry, icmp, and ipAddrEntry are all base objects. System contains attributes such as sysDescr and sysUpTime that have associated values.

Some base objects, such as system, are simply the MIB group name and contain a single set of attributes. These are zero-instance base objects because they have no separate instances. Other base objects represent one instance out of two or more. IfEntry, for example, is one interface on a device that may contain several interfaces. Connecting the base object name and instance number with a period, as in ifEntry.2, fully qualifies the base object instance that is being referenced.

IBM TIVOLI NETCOOL/OMNIBUS

A network management platform that can be configured to receive NerveCenter Inform messages. The Netcool platform host requires NerveCenter's universal platform adapter to communicate with NerveCenter.

MULTITIER BEHAVIOR MODEL

A behavior model that contains two or more alarm definitions. A transition in one alarm instance fires triggers to feed other transitions in other alarm instances. This is useful when you want to correlate conditions that first need to be correlated themselves, or when you want to correlate conditions for a node, but need information from another node to determine the final outcome.

This type of behavior model can exploit NerveCenter's alarm scope capability. For example, if you want to be notified when a device has high port traffic, you can design a subobject scope alarm to detect high traffic for a single port. This alarm sends a trigger to a second alarm each time heavy traffic is detected on the port. The second alarm uses node scope and processes at most one trigger for the port. The second alarm notifies you once when the node's port is busy.

NERVECENTER ADMINISTRATOR

The NerveCenter software module that enables you to customize and manage NerveCenter operations. Run the NerveCenter Administrator program, ncadmin.exe, from the Start menu on your Windows 7, 8.1, or 10 desktop.

NERVECENTER ADMINISTRATOR

A system or network administrator who is responsible for configuring and maintaining NerveCenter operations and, optionally, installing NerveCenter.

NERVECENTER CLIENT

The NerveCenter software module that enables you to create and manage behavior models as well as monitor and report on network activity. Run the NerveCenter Client program, client.exe, from the Start menu on your Windows 7, 8.1, or 10 desktop.

NERVECENTER ENTERPRISE TRAP NUMBER

The enterprise-specific MIB object identifier (OID) for NerveCenter is 1.3.6.1.4.1.78, used to create trap masks that detect Inform messages.

NERVECENTER OBJECT

A NerveCenter entity that can be part of a behavior model used to manage your network. The following are NerveCenter objects: nodes, property groups and their properties, polls, trap masks, and alarms.

NERVECENTER SERVER

The background engine that handles event correlation and manages the NerveCenter database. Each time you use the NerveCenter Client or Administrator, you must connect to a server. The server can be installed on the same machine as the NerveCenter Client or Administrator applications.

NETWORK MANAGEMENT PLATFORM

NerveCenter supports several network management platforms, and communicates with each using a platform adapter. The platform adapter must be installed and running on the platform host before a connection can be established with NerveCenter.

IBM Tivoli Netcool/OMNIbus can only receive NerveCenter messages and display those messages in their event or message browser. It uses the NerveCenter universal platform adapter:

NOAUTHNOPRIV

Security level that does not require message authentication or encryption services for communication between any two SNMPv3 entities. That means communication between the two SNMPv3 entities communicating with this (i.e. NoAuthNoPriv) security level is not secure.

Communication between two SNMPv3 entities takes place on behalf of a *user* (a uniquely identified entity in the SNMPv3 management domain), and possibly in some *context* (a uniquely identified entity for access control generally configured on conventional SNMPv3 agents/nodes).

NODE

Any device on the network that can be managed. Examples of nodes are servers, workstations, printers, hubs, routers, bridges, and gateways.

When NerveCenter is integrated with a network management platform, you can configure NerveCenter's node list to be populated by that platform. In addition, NerveCenter can be configured to add unknown nodes that send it a trap. You can also populate the list manually or by using the Discovery behavior model.

See also "managed node" on page 232.

NODE SCOPE

Alarm definition setting that determines if each alarm instance monitors a node.

See also "alarm scope" on page 228.

OBJECT IDENTIFIER, OBJECT ID, OR OID

A unique SNMP name given to each MIB base object, identified by the value of the sysObjectID object in the system group of MIB-II. The name is written as a sequence of integers separated by periods. For example, the sequence 1.3.6.1.2.1.1.1.0 specifies a system description, where 1.3.6.1.2.1.1 specifies the base object system, 1.3.6.1.2.1.1.1 specifies the attribute sysDescr (which has a value), and 1.3.6.1.2.1.1.1.0 specifies an instance (0)

Objects associated with a vendor or equipment type also have object identifiers. You can obtain the object identifier for a network node by opening its Node Definition window. Select the Query Node tab and then select the **Get** button.

OID

See "object identifier, object ID, or OID " above.

PASSWORDS (SNMPV3)

Before NerveCenter can poll SNMPv3 nodes, you must supply the proper passwords for the NerveCenter user communicating with that node. These passwords must be configured on your SNMPv3 agents as well as in NerveCenter Administrator. You need to provide the following passwords:

- **Authentication -** Required for providing authentication services when NerveCenter communicates with the agent using either an AuthNoPriv or AuthPriv security level.
- **Privacy -** Required for providing encryption services when NerveCenter communicates with the agent using an AuthPriv security level.

Depending on the security level you provide in NerveCenter Client for the node corresponding to an SNMPv3 agent, NerveCenter will use an appropriate combination of passwords to provide message authentication and message encryption services. No passwords are required if using NoAuthNoPrive security.

PING

An ICMP echo request sent to a network device that returns an echo reply from the device. ICMP is an error-control protocol.

PLATFORM ADAPTER

A NerveCenter process that facilitates communication between NerveCenter and a network management platform. The universal platform adapter enables NerveCenter to send messages to a platform.

See also "Universal Platform Adapter (paserver)" on page 239.

POLL

A NerveCenter object that monitors the network for conditions of interest by polling SNMP agents on managed nodes for specific MIB data. These polls are compare to a user-defined poll condition and trigger expression. If the condition is satisfied the poll generates a trigger that signals one or more alarms, at least one of which must be enabled and include a transition that the poll can trigger.

POLL CONDITION

Defines the condition to be detected on each node that is tracked. When a poll condition is satisfied, the poll generates a trigger that signals one or more alarms.

You can create conditions, such as threshold crossings or rates of change for tracked values, using arithmetic and relational operators. You can string together combinations of conditions using logical (Boolean) operators.

POLL RATE

The interval (in seconds, minutes, or hours) that a poll waits between requests for node MIB data and evaluation of the associated poll condition.

PRINCIPAL

The person who has access to an SNMPv3 agent.

PRIVACY KEY

The seed used by the privacy protocol to encrypt messages while communicating with an SNMPv3 entity using AuthPriv security level. The privacy key is described in SNMPv3 specifications.

PROPERTY

A text string that is a member of one or more property groups. Properties fall into two categories: MIB base objects and user-defined strings. Poll and alarm definitions use properties when determining whether a device should be monitored.

The property group associated with a managed node must contain the properties for every MIB base object used in a poll for that node. The property groups shipped with NerveCenter contain the base objects relevant for the group. For example, the Mib-II property group contains properties for each base object in MIB-II. User-defined strings are used to restrict the targeted subset of nodes for a poll or alarm.

PROPERTY GROUP

A collection of one or more text strings called properties. Property groups allow you to categorize nodes into logical or managerial units. The groups can be based on device type, location, priority, supported MIBs, business function, or any other useful characteristic.

Each managed node belongs to a property group and, therefore, is associated with all of the group’s properties. These properties are used to restrict which polls and alarms monitor the node. Assigning a node to a property group that contains multiple properties allows the node to be targeted by multiple behavior models.

RESYNCHRONIZATION (RESYNCH)

When NerveCenter is integrated with a network management platform, NerveCenter issues a Resync command upon start-up. This command updates NerveCenter's entire node list with information from the platform. The update contains all nodes within NerveCenter's subnet filters, if applicable, and includes details about each node's parents.

You can manually refresh the NerveCenter node list by selecting the **Resync** command from the client's Server menu. If this command is not available, the connection to the node source is down.

See also "synchronization" on the next page.

SCOPE

See "alarm scope" on page 228.

SECURITY LEVEL

The use of message authentication and message encryption services for communication between any two SNMPv3 entities. NerveCenter supports the following security levels while communicating with SNMPv3 agents:

- **NoAuthNoPriv** - Neither message authentication nor message encryption services are used for communication between two SNMPv3 entities. As such, the communication is not secure.
- **AuthNoPriv** - Message authentication services are used without message encryption for communication between two SNMPv3 entities. This ensures only authenticity of the messages being exchanged between two SNMPv3 entities. The messages are not encrypted.
- **AuthPriv** - Both message authentication and message encryption services are used for communication between two SNMPv3 entities. This is the most secure way of communication defined in SNMPv3 specifications at the present time.

SEVERITY

See "alarm severity " on page 228.

SHA

A secure hash algorithm specified in the Secure Hash Standard (SHS, FIPS PUB 180). This is one of the two algorithms suggested in SNMPv3 specifications for message authentication. This protocol is supported by NerveCenter.

SIMPLE NETWORK MANAGEMENT PROTOCOL (SNMP)

An industry-standard protocol that defines how network management systems exchange information with their managed nodes. Software processes called SNMP agents reside on each managed device and track defined sets of data. A database of network information, called a management information base (MIB), is associated with both the manager and agent. A manager and agent may exist on the same system.

SNMP messages allow a management application to set and retrieve agents' managed data. SNMP traps allow an agent to send events to a management application.

SMART POLLING

A NerveCenter feature that reduces the network overhead associated with SNMP management. A poll doesn't solicit data from nodes unless an alarm definition uses the poll's trigger. Even then, the poll is issued to a specific node only when the alarm exists in a state that the poll can trigger. For example, if a behavior model correlates high traffic followed by high error rates, a node is not polled for error rates unless it fulfills the high traffic condition first.

SNMP

See "Simple Network Management Protocol (SNMP) " on the previous page.

SNMP TRAP

An asynchronous notification that an SNMP agent can send to a management application. Traps are identified by a generic number (0–6), an enterprise-specific number, and an enterprise name. Related information data is typically bundled with a trap in structured formats called variable bindings. NerveCenter detects and filters SNMP traps with trap masks.

SPECIFIC TRAP NUMBER

An SNMP trap number associated with an enterprise MIB. When defining trap masks, generic trap numbers (0-6) indicate the nature of the SNMP trap being reported. In addition, each enterprise, or vendor, can have any number of subtrap numbers, called specific trap numbers, within the category 6 generic trap. Each vendor defines its own number of enterprise traps, their associated specific numbers, their functions, and their variable bindings.

See also "generic trap number" on page 231.

STATE DIAGRAM

See "alarm state diagram " on page 228.

SUBOBJECT

A base object and an instance connected by a period. Examples from MIB-II include ifEntry.3, system.0, and ipForwardEntry.2.

SUBOBJECT SCOPE

A setting for an alarm definition that determines that each alarm instance monitors a subobject. Subobject scope indicates that conditions for each subobject are tracked separately with separate alarm instances. For example, one alarm instance would track conditions on ifEntry.1, another would track conditions on ifEntry.2, and so on.

See also "alarm scope" on page 228.

SUPPRESSION

NerveCenter provides the following types of suppression as a way to manage unresponsive nodes.

- **Node suppression** - When a node fails to respond, it can be suppressed manually or through an alarm action. Normal polling does not occur for the node as long as the node remains suppressed.
- **Poll suppression** - By default, NerveCenter polls are suppressible. This means they do not poll nodes that are suppressed. However, there may be specific polls that you want to occur on a node even when it is suppressed. In such a case, you can set the poll to insuppressible.

Suppression affects only polling; SNMP traps are still received from a managed node even though it is suppressed.

SYNCHRONIZATION

When NerveCenter is integrated with a network management platform, the platform uses a process called synchronization to update NerveCenter whenever the platform adds, updates, or deletes nodes. This type of update does not include parent information.

See also "resynchronization (resynch)" on page 237.

TIME TICKS

See "engine time ticks" on page 231.

TRANSITION

See "alarm transition" on page 229.

TRAP

See "SNMP trap" on the previous page.

TRAP MASK

A NerveCenter object that captures SNMP traps received from managed nodes and fires a trigger that transitions an alarm instance. A mask can detect a standard SNMP trap identified by one of six generic categories, an enterprise-specific trap, and, with the advanced Trigger Function feature, a trap whose contents match user-specified criteria. At least one alarm must be enabled and include a transition that can be triggered by the mask’s trigger.

TRIGGER

A flag sent to one or more alarms to indicate that an event has been detected or a condition satisfied. The trigger transitions an alarm from one state to another. The alarm must be enabled and include a transition that can be triggered by this particular trigger.

Triggers are generated by polls, trap masks, and the Fire Trigger or Perl Subroutine alarm actions. You specify a trigger using the FireTrigger() function in a poll condition, Perl subroutine, or trap mask trigger function. NerveCenter generates its own built-in triggers, designed to detect unresponsive nodes.

TRIGGER FUNCTION

A Perl script that you define for a NerveCenter trap mask. The script is called whenever a detected SNMP trap matches the mask’s data fields.

The Perl script issues triggers based upon the contents of the trap’s variable bindings. When a suitable trap is detected, NerveCenter executes the trigger function, supplying data from the trap’s variable bindings. Your script evaluates the variable-binding contents and conditionally fires triggers or assigns property groups.

TRIGGER GENERATOR

A NerveCenter object or action that fires a trigger when a certain condition is detected. The trigger is sent to one or more alarms, where it transitions one or more alarm states.

The following are NerveCenter trigger generators: polls, trap masks, Perl subroutines that include the FireTrigger() function, and alarm actions (Send Trap, Fire Trigger). NerveCenter generates its own built-in triggers when it detects unresponsive nodes.

UNIVERSAL PLATFORM ADAPTER (PASERVER)

A NerveCenter platform adapter process that resides on IBM Tivoli Netcool/OMNIbus. The universal platform adapter enables NerveCenter to send Inform messages.

USER RIGHTS

Those with user rights belong to the NerveCenter Users (ncusers) group. They perform such tasks as monitoring and resetting NerveCenter alarms. They cannot modify the NerveCenter database, for example, by making changes to nodes, property groups, polls, masks, or alarms.

VARIABLE BINDINGS

An array of related information that accompanies an SNMP message. The variable bindings are typically defined in standard or vendor-specific MIB definitions (.ASN1 files). The format of each variable binding is defined by SNMP. NerveCenter polls and trap masks obtain variable bindings from the nodes they monitor.

ZERO-INSTANCE BASE OBJECT

A MIB base object that has only one instance and, therefore, contains a single set of attributes. For example, system is a zero-instance base object because it contains attributes (like sysLocation) that have only one associated value per node. In contrast, ifEntry is a base object with multiple instances. If a node had four ports, the node would has four instances of the ifEntry base object, numbered one through four, and each instance has associated attributes.

Index