



LogMatrix®  
NerveCenter™

## Monitoring NerveCenter

Version 6.2

## **COPYRIGHT**

Portions ©1989-2017 LogMatrix, Inc. All rights reserved.

## **DISCLAIMERS**

LogMatrix, Inc. ("LogMatrix") makes no representations or warranties, either expressed or implied, by or with respect to anything in this manual, and shall not be liable for any implied warranties of merchantability or fitness for a particular purpose or for any indirect, special or consequential damages.

These applications are available through separate, individual licenses. Not every feature or application described herein is licensed to every customer. Please contact LogMatrix if you have licensing questions.

No part of this publication may be reproduced, stored in a retrieval system or transmitted, in any form or by any means, photocopying, recording or otherwise, without prior written consent of LogMatrix. While every precaution has been taken in the preparation of this book, LogMatrix assumes no responsibility for errors or omissions. This publication and the features described herein are subject to change without notice.

The program and information contained herein are licensed only pursuant to a license agreement that contains use, reverse engineering, disclosure and other restrictions.

## **TRADEMARKS**

LogMatrix is registered in the U.S. Patent and Trademark Office. NerveCenter and the LogMatrix Logo are trademarks of LogMatrix, Inc.

All other products or services mentioned in this manual may be covered by the trademarks, service marks, or product names as designated by the companies who market those products.

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>

<b>Introduction</b>	<b>1</b>
NerveCenter Documentation .....	1
LogMatrix Technical Support .....	4
<b>Understanding NerveCenter</b>	<b>5</b>
What is NerveCenter? .....	5
How NerveCenter Manages Nodes .....	5
Defining a Set of Nodes .....	6
Detecting Conditions .....	6
Correlating Conditions .....	7
Responding to Conditions .....	11
Main NerveCenter Components .....	13
The NerveCenter Server .....	14
The NerveCenter Database .....	14
NerveCenter Interfaces .....	17
Role in Network Management Strategy .....	20
Standalone Operation .....	21
Using Multiple NerveCenter Servers .....	22
Integration with Network Management Platforms .....	22
<b>Getting Started with the NerveCenter Client</b>	<b>25</b>
Starting the NerveCenter Client .....	25
Connecting to a Server .....	26
Connecting to a Server Manually .....	27
Connecting to a Server Automatically .....	29
Sharing MIB Information from Multiple Servers .....	31
Selecting the Active Server .....	32
Deleting a Server from the Server List .....	32
Changing the Server Port on the Client .....	33

Setting Up Alarm-Instance Filters .....	34
Filtering Alarms by IP Range .....	35
Filtering Alarms by Severity .....	42
Filtering Alarms by Property Groups .....	46
Associating a Filter with a Server .....	49
Rules for Associating Filters with Alarms .....	51
Specifying Heartbeat Messaging .....	52
Modifying the Heartbeat Message Interval .....	53
Deactivating Heartbeat Messaging .....	54
Disconnecting from a Server .....	55
<b>Monitoring Alarms</b> .....	<b>57</b>
Viewing Alarm Information .....	57
Using the NerveCenter Client .....	58
Interpreting Alarm-Instance Information .....	62
Getting Information about an Alarm .....	63
Getting Information about a Trigger .....	64
Built-In Triggers .....	69
Viewing Alarm Instance History .....	72
Using the NerveCenter Client .....	72
Reading Logged Data .....	76
Determining Where Data is Being Logged .....	76
How to Interpret Logged Data .....	78
<b>Resetting Alarms</b> .....	<b>81</b>
Resetting an Alarm Instance to Ground .....	81
Resetting an Alarm Instance to a Non-Ground State .....	82
Resetting Node Alarm Instances .....	83
Resetting All Instances of an Alarm .....	85
<b>Monitoring SNMP Status and Operations</b> .....	<b>87</b>
Viewing the SNMPv3 Operations Log .....	87
Signing a Log for SNMPv3 Errors Associated with Your Client .....	88
Signing a Log for SNMPv3 Errors Associated with a Remote Client or Administrator .....	89
Viewing the SNMPv3 Operations Log .....	90

SNMP Error Status .....	91
<b>Monitoring Nodes</b>	<b>95</b>
Viewing Related Alarms .....	95
Querying Nodes .....	98
Viewing Parent Node Status .....	103
<b>Checking Server Status</b>	<b>107</b>
Server Tab .....	108
License Tab .....	109
Database Tab .....	110
Node Source Tab .....	111
Inform Configuration Tab .....	112
Connected NerveCenters Tab .....	113
Connected Clients and Connected Administrators Tabs .....	113
<b>Communications and Data</b>	<b>115</b>
<b>Error Messages</b>	<b>119</b>
Alarm Filter Manager Error Messages .....	121
Deserialize Manager Error Messages .....	121
Flatfile Error Messages .....	121
Inform NerveCenter Error Messages .....	121
LogToFile Manager Error Messages .....	122
Poll Manager Error Messages .....	122
Protocol Manager Error Messages .....	122
PA Resync Manager Error Messages .....	123
Server Manager Error Messages .....	125
Trap Manager Error Messages .....	127
NerveCenter installation Error Messages .....	128
<b>Index</b>	<b>131</b>



*Monitoring Your Network* 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](http://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 <i>Release Notes</i> 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	<b>Bold</b>	Press <b>Tab</b> Enter <b>ovpa -pc</b>
<ul style="list-style-type: none"> <li>■ A variable you substitute with a specific entry</li> <li>■ Emphasis</li> <li>■ Heading or Publication Title</li> </ul>	<i>Italic</i>	Enter <i>./installdb -f IDBfile</i>
Code samples, code to enter, or application output	Code	<code>iifInOctets &gt; 0</code>
Messages in application dialog boxes	<i>Message</i>	<i>Are you sure you want to delete?</i>
An arrow ( > ) indicates a menu selection	>	Choose <b>Start &gt; Programs &gt; NerveCenter</b>

**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](mailto: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](mailto:support@logmatrix.com).

## 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 Manages Nodes

To perform its job of event automation, NerveCenter relies on the definition of *behavior models*. These models are constructed from NerveCenter objects (which we'll discuss in detail later) and define:

- Which nodes the 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

## Defining a Set of Nodes

NerveCenter can get the list of devices to monitor from a network management platform, discover them on the network, or import this information from another NerveCenter database.

NerveCenter assigns to each managed node a set of *properties*, and these properties determine which behavior models apply to a node. 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.

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

As is discussed in the section [Role in Network Management Strategy on page 20](#), NerveCenter can collect network and system data from a variety of sources. However, most frequently NerveCenter 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 SNMP v2c and v3 features allow SNMP to be utilized both reliably and securely.

## Correlating Conditions

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

- How these conditions, or some subset of them, are related
- The underlying cause of a set of conditions, or the problem to which they 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 below](#)
- [Finding a Set of Conditions on the next page](#)
- [Looking for a Sequence of Conditions on page 9](#)

### 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 below.

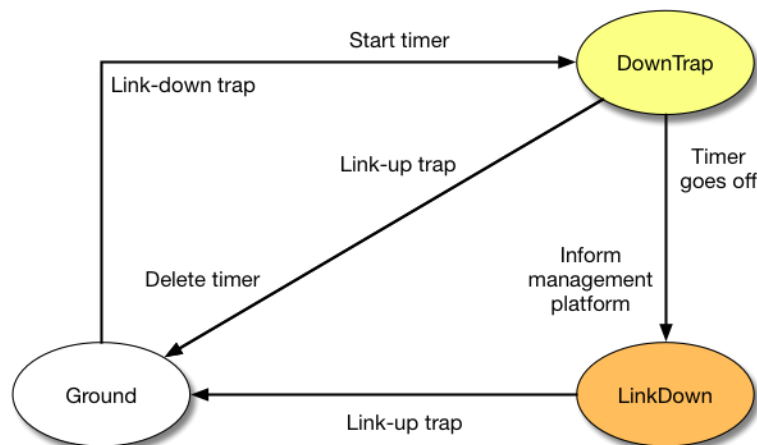


Figure 1: State Diagram for Detecting a Link-Down Condition

Let's say that 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 the identification of a set of conditions. For example, let's say that you're monitoring the interfaces on a router. To be notified when a low-speed interface goes down or when a high-speed interface goes down, you might use the following state diagram.

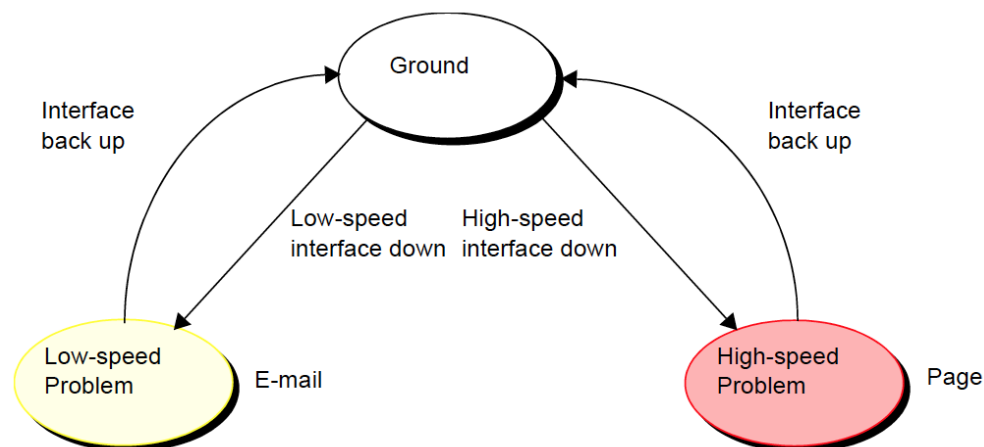


Figure 2: State Diagram for Detecting a Router Interface Problem

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 this set of conditions is met, a problem exists on an interface that is probably used for a dial-up connection.

The second set of conditions is the same as the first, except that the last condition is that the interface's current bandwidth is greater than or equal to 56K. If this 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 you to correlate conditions by looking for sequences of conditions. This type of correlation is possible because, in NerveCenter, 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 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.

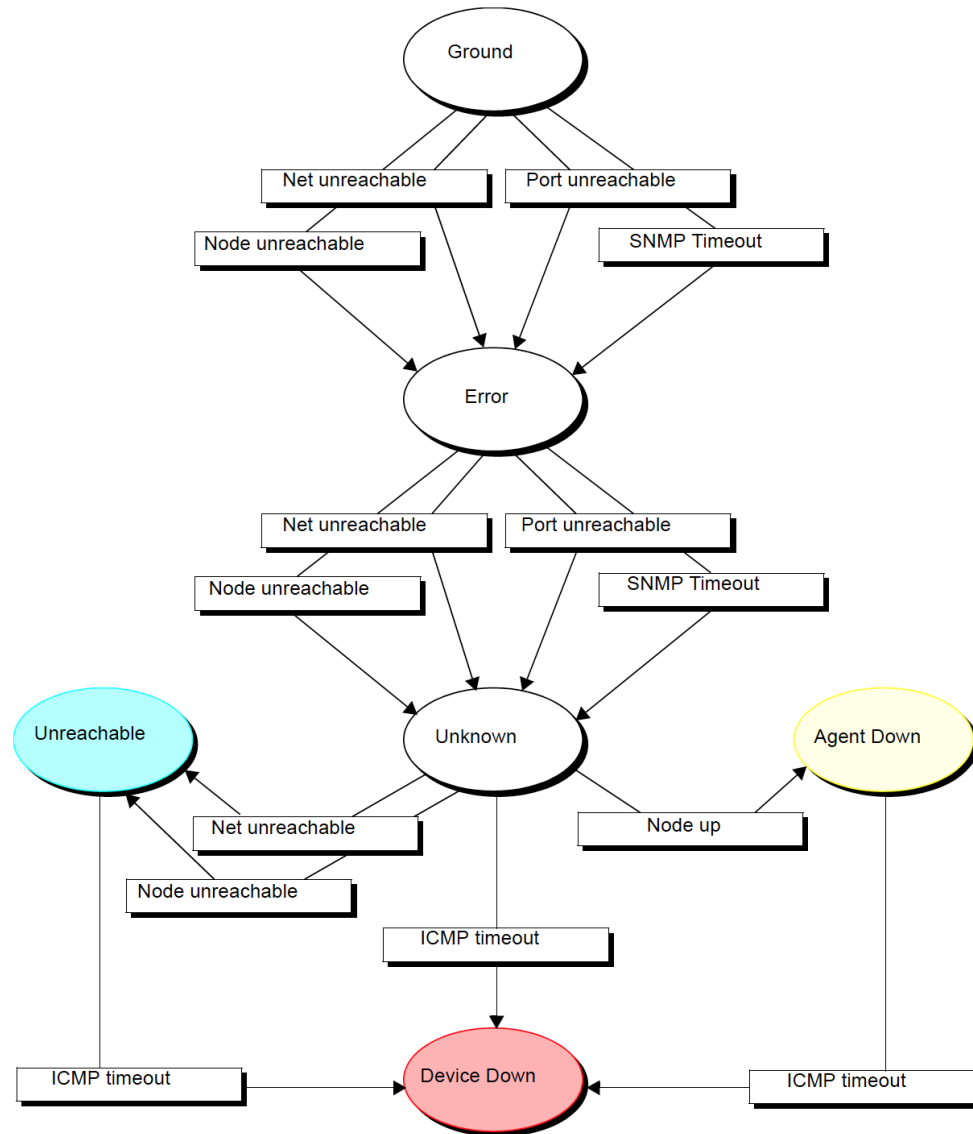


Figure 3: State Diagram for Determining Node Status

**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 13](#)

### 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 a notification to an administrator using either a SMS message or SMTP mail.
- You can page an administrator.
- 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's 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 20](#).

## 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
- system log (syslog)

Log to File writes information about an event to a file. The EventLog action writes information about an event to the system log.

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 that processes 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 any 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

## Main NerveCenter Components

NerveCenter is a distributed client/server application and includes the following components:

- [The NerveCenter Server \(page 14\)](#)
- [The NerveCenter Database \(page 14\)](#)
- [NerveCenter Interfaces \(page 17\)](#)

## The 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.

## The NerveCenter Database

The NerveCenter database is primarily a repository for the NerveCenter objects that make up a set of behavior models. The principal objects used in these models are:

- Nodes
- Property groups and properties
- Polls
- Trap masks
- Alarms

For brief explanations of what these objects are and how they are used, see [Objects in the Database below](#).

A set of objects that define many useful behavior models ships with NerveCenter and is available as soon as you've installed the product. For a list of these predefined behavior models, see the section [Predefined Behavior Models on page 16](#).

On UNIX systems, the NerveCenter database is implemented as a flat file.

### Objects in the Database

This section contains brief definitions of the basic objects used in the construction of behavior models.

- **Nodes** - A node represents either a workstation or a network device, such as a router. Each node has an attribute called its property group that controls which behavior models NerveCenter will employ in managing the node.

**Note:** Strictly speaking, a node is not part of a behavior model; rather, it is the entity managed by a behavior model.

- **Property groups and properties** - As mentioned above, each node has a property group. This property group is simply a container for a set of properties, which are strings that typically either describe the type of node or name an object in the MIB used to manage the node. It is actually a node's properties, rather than its property group, that determine whether a particular behavior model will be used to manage that node.

- **Polls** - A poll defines what MIB variables NerveCenter should request the values of, how those values should be evaluated, and what action the poll should take. If the poll takes an action, it will be to fire a *trigger*, which may cause a state transition in one of NerveCenter's finite state machines.
- **Trap masks** - A trap mask describes an SNMP trap and contains the name of a trigger. If NerveCenter receives an SNMP trap that matches the description given in the trap mask, NerveCenter fires a trigger with the name defined in the trap mask. If NerveCenter receives a trap that does not match a trap mask, it discards that trap.
- **Alarms** - NerveCenter's finite state machines are called *alarms*. Each alarm defines a set of operational states (such as Normal and Down) and transitions between the states. Transitions are effected by the receipt of the proper trigger and can have actions associated with them. If actions are associated with a transition, the server performs these actions each time the transition takes place.

### Behavior Models

Once a set of managed nodes has been defined, NerveCenter's monitoring activities are controlled by a set of *behavior models*. A behavior model is the group of NerveCenter objects required to detect and take action upon a single network condition, such as high traffic on an interface.

The central object in each behavior model is a deterministic finite state machine called an *alarm*. For instance, the alarm shown in Figure 4 tracks the level of traffic on an interface.

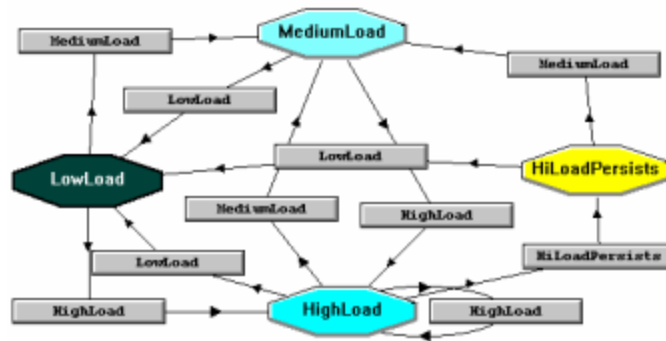


Figure 4: Alarm State Diagram

The possible states in this alarm are low, medium, and high. And these states have the *severities* Normal, Medium, and High, respectively. (The color of each state denotes its severity.) The gray rectangles in the alarm represent *state transitions*.

What about the inputs and outputs of the state machine? The inputs are called *triggers* and can come from several sources. For example, one predefined NerveCenter poll queries the SNMP agent on a device for the level of traffic on, and the capacity of, each interface on the device. If the level of use exceeds a certain percentage of the capacity for an interface, the poll fires the trigger *mediumLoad*, which can cause a state transition in an alarm.

The outputs of an alarm are called *alarm actions*. These actions are associated with the transition from one state to another by the designer of a behavior model, and NerveCenter performs these actions each time the transition occurs. There are many possible actions, including the following:

- Sending an audible alert to the workstation on which the NerveCenter Client is being run
- Executing a program or script
- Deleting a node from the NerveCenter database
- Informing a network management platform of a condition
- Logging information to a disk file
- Sending mail to an administrator
- Paging an administrator
- Sending an SNMP trap
- Setting a MIB attribute

### Predefined Behavior Models

When you install NerveCenter and create a new database, that database contains the objects that make up a number of predefined behavior models. These include behavior models for:

- Detecting authentication failures
- Monitoring the error rate on network interfaces
- Monitoring link-up and link-down traps
- Monitoring the amount of traffic on network interfaces
- Indicating the status of network interfaces: up, down, and so on
- Detecting errors that inhibit accurate SNMP device management
- Determining whether a device is down, unreachable, or up with/without an agent
- Giving early warning concerning TCP connection saturation
- Verifying that the current TCP retransmission algorithm is the most efficient
- Categorizing devices based on TCP retransmission activity
- Logging information about SNMP traps

NerveCenter also includes predefined behavior models that you can import to monitor specific vendors' devices and additional models for troubleshooting, interface status, data collection, and downstream alarm suppression. For more information about behavior models, see [Behavior Models and Their Components](#) in *Designing and Managing Behavior Models*.

## NerveCenter Interfaces

This section introduces NerveCenter's principal user interfaces:

- "The NerveCenter Administrator" below
- The NerveCenter Client Console
- "The NerveCenter Client" on page 19
- "The Command Line Interface" on page 20

### 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.

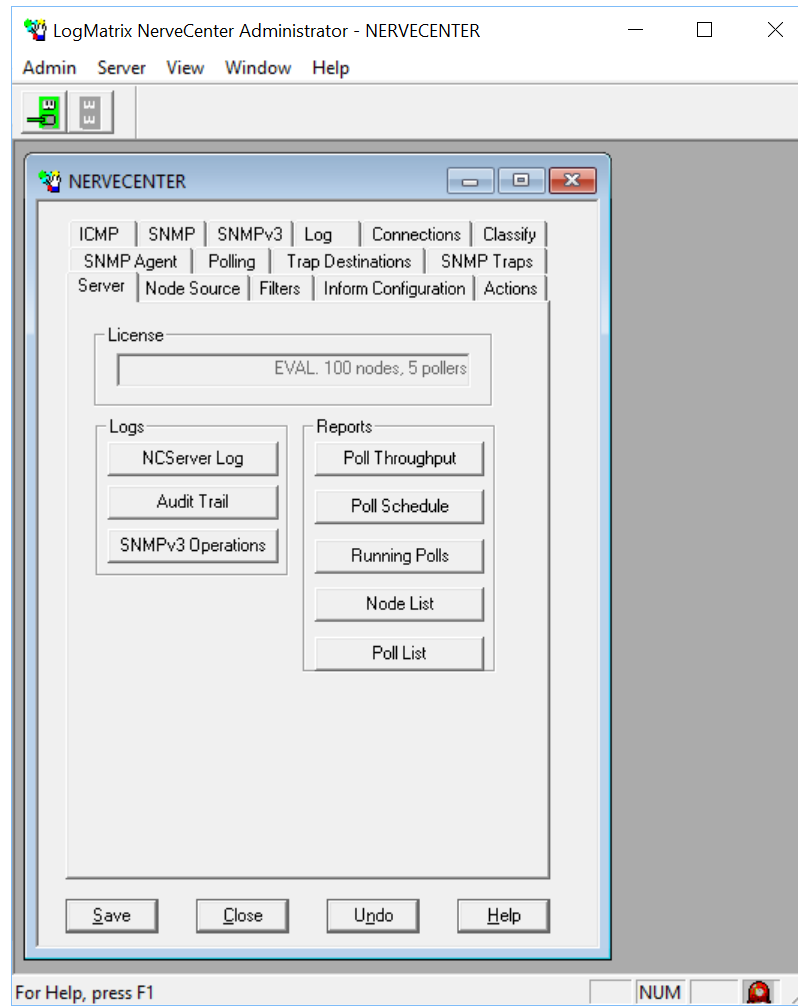


Figure 5: NerveCenter Administrator



## The NerveCenter Client

The figure below shows the GUI for the NerveCenter Client.

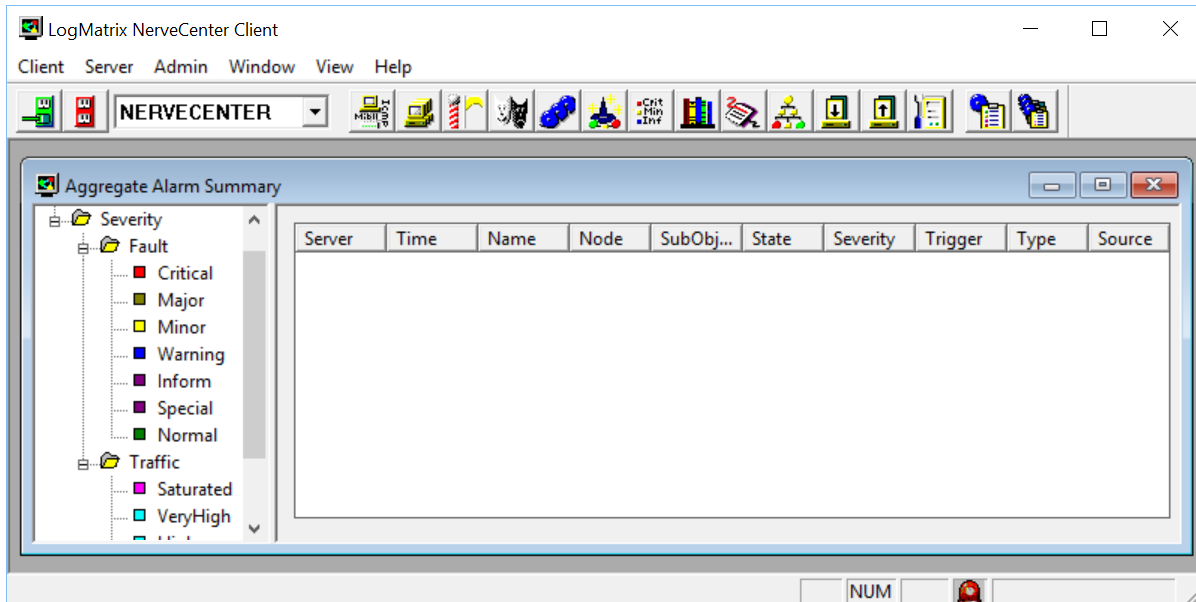


Figure 6: NerveCenter Client

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](#) 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 facing 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 22 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 22 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 7](#).

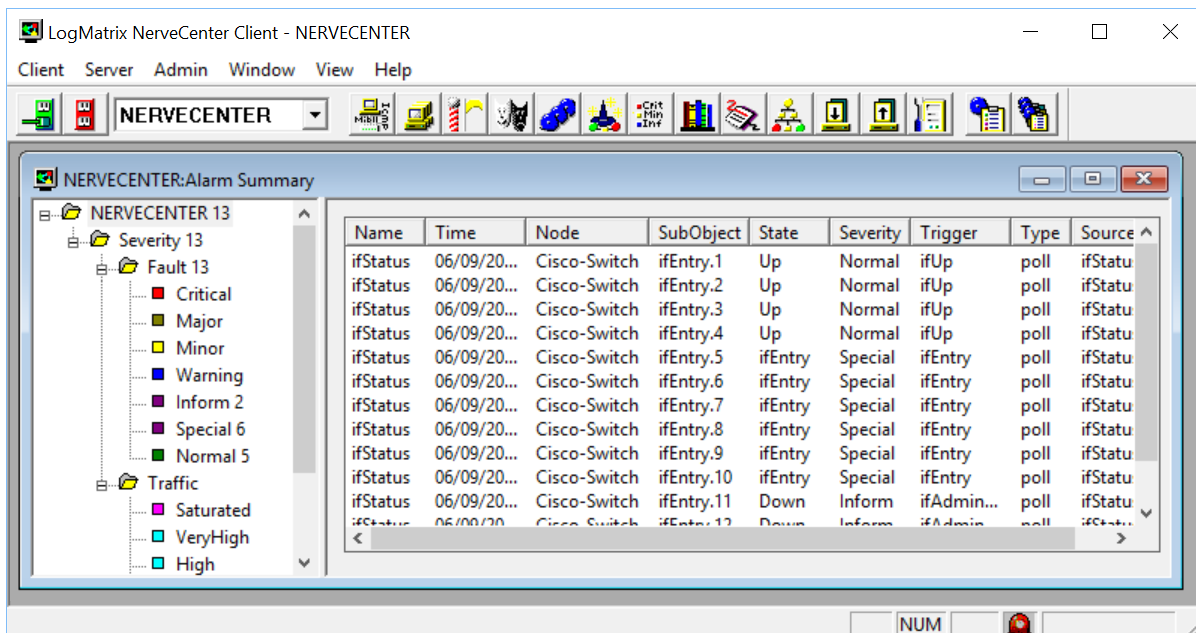


Figure 7: NerveCenter's Alarm Console

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 8).

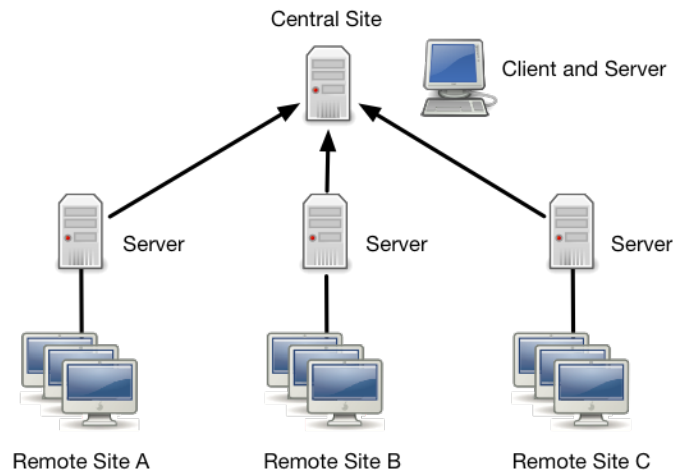


Figure 8: Distributed NerveCenter Servers

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.



Before you can begin monitoring your network using the NerveCenter Client, you must start the client and then establish a connection between the client and one or more NerveCenter servers. You may also want to set up alarm filters to control which alarm instances the NerveCenter Client will display information about.

## Starting the NerveCenter Client

### TO START THE NERVECENTER CLIENT

- Select the **Start** menu. Select **Programs > LogMatrix NerveCenter > Client**.  
NerveCenter displays the NerveCenter Client window.

**Note:** These steps depend on a typical NerveCenter installation. The directory path may be different.

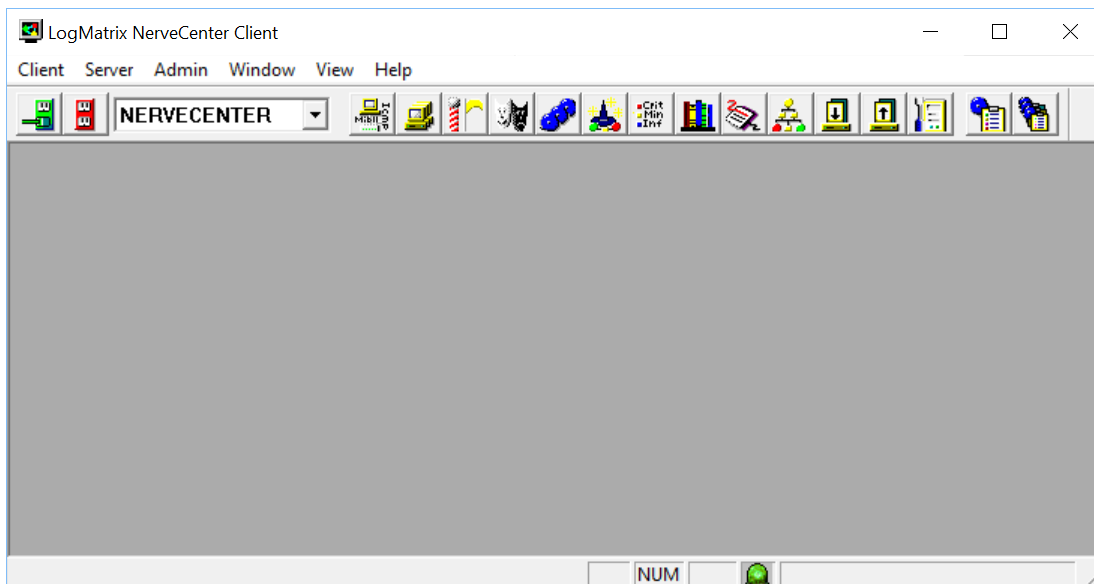


Figure 9: The NerveCenter Client Window

Most buttons and menu options are not enabled until you connect the client to a NerveCenter server.

## Connecting to a Server

Before you can use the client, you must connect the client to a NerveCenter server. This server collects data from managed devices, creates alarm instances, and performs the actions defined in alarms. The server also gives the client access to information about alarm instances and the status of nodes.

You can connect your client to more than one server at one time and view information about all the active alarm instances being managed by those servers. However, only one server can be the *active* server. The active server determines which NerveCenter database is used when you ask for a list of polls or the definition of an alarm.

For information on connecting to a NerveCenter server, see the following subsections:

- [Connecting to a Server Manually on the facing page](#)
- [Connecting to a Server Automatically on page 29](#)
- [Sharing MIB Information from Multiple Servers on page 31](#)

You may also be interested in the following topics, which relate to connecting to a server:

- [Selecting the Active Server on page 32](#)
- [Deleting a Server from the Server List on page 32](#)
- [Changing the Server Port on the Client on page 33](#)



## Connecting to a Server Manually

If you haven't configured the client to connect to one or more servers at startup, or if you want to establish a connection with a server that you don't typically use, you must establish your connection with the server manually.

### TO CONNECT TO A NERVECENTER SERVER MANUALLY

1. From the **Server** menu, select **Connect**.

The Connect to Server window displays.

2. In the **Server Name** field, type the hostname or IP address of the machine where the NerveCenter server is running or select a hostname or IP address from the **Server Name** drop-down list.

The first time you connect to a server, the drop-down 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 won't display the names of machines to which you're already connected.) For information on removing an entry from the drop-down list box, see the section [Deleting a Server from the Server List on page 32](#).

3. 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 Users (ncusers) or NerveCenter Admins (ncadmins) group on the NerveCenter Server host.

4. Select the **Connect** button.

If the machine to which you try to connect is not running the NerveCenter server, you see the message **The server did not respond**.

When the client successfully connects to the server, all of the buttons in the button bar become enabled, and the Aggregate Alarm Summary window appears.

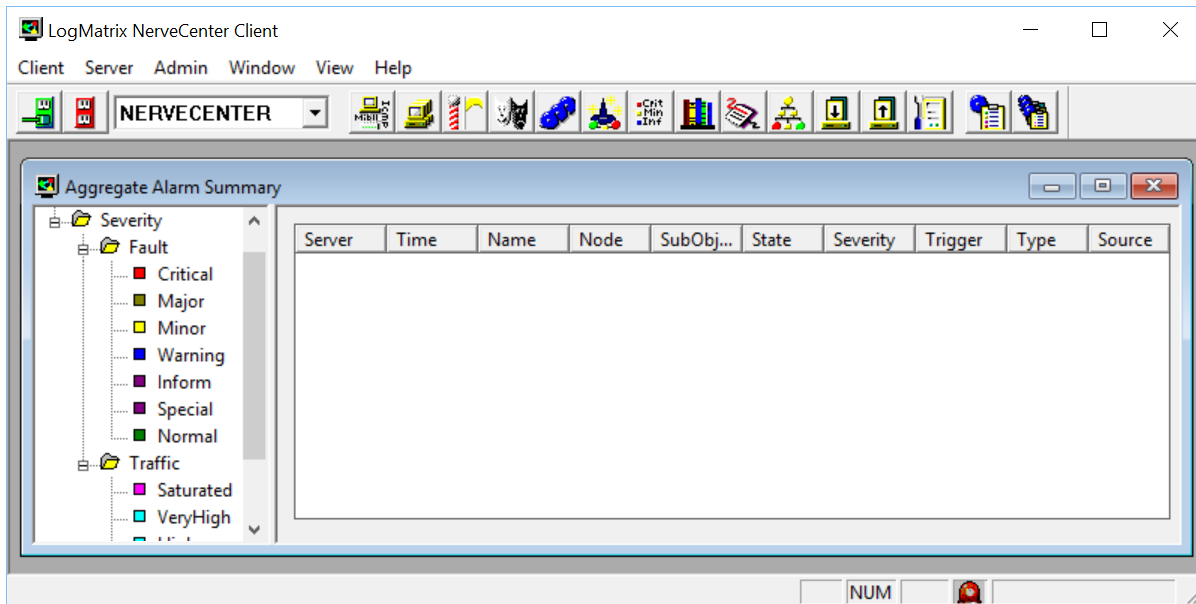


















Figure 10: Client Connected to a Server

Table 1 lists the client windows you can reach by using the buttons in the client's toolbar.

Table 1: Windows Accessible from Toolbar

Button	Window
	Opens the Connect to Server window, from which you can connect the client to a NerveCenter server.
	Opens a Client message window containing the prompt <b>Disconnecting from Hostname</b> . Use this window to confirm that you want to disconnect the client from a NerveCenter server.
	Opens the Property Group List window. From this window, you can view the currently defined property groups and the properties that each property group contains.
	Opens the Node List window. From this window, you can view a list of the nodes defined in the NerveCenter database and a brief definition of each node.
	Opens the Poll List window. From this window, you can view a list of the polls defined in the NerveCenter database and a brief definition of each poll.
	Opens the Mask List window. From this window, you can view a list of the trap masks defined in the NerveCenter database and a brief definition of each trap mask.
	Opens the Alarm Definition List window. From this window, you can view a list of the alarms defined in the NerveCenter database and open a definition window for each alarm.
	Displays a list of currently defined correlation expressions. Correlation expressions enable you to create alarms from Boolean expressions.

Button	Window
	Opens the Severity List window, from which you can view a list of the severities defined in the NerveCenter database. (A severity has a name, a severity level, and a color associated with it.)
	Opens the Perl Subroutine List window. From this window, you can view a list of the currently defined Perl subroutines.
	Opens the Action Router Rule List window. From this window, you can view a list of the current set of rules that you have defined for the Action Router.
	Opens the Import Objects and Nodes dialog. From this dialog, you can import behavior models from one NerveCenter to another.
	Opens the Export Objects and Nodes dialog. From this dialog, you can export specific objects or groups of objects from one database to another.
	Opens the Server Status dialog. This dialog provides you with a comprehensive view of all your NerveCenter server settings.
	Opens the Alarm Summary window. This window presents information about the current alarm instances for the active server.
	Opens the Aggregate Summary window. This window presents information about the current alarm instances for all the servers to which you're connected.

## Connecting to a Server Automatically

If you want to establish a connection with the same set of servers each time you run the client, you can use NerveCenter's Autoconnect feature.

**Note:** Before you activate the Autoconnect feature, you might want to manually connect to the NerveCenter Server, to verify that you can indeed access the server.

## TO SET UP A LIST OF SERVERS TO WHICH YOU'LL CONNECT AT STARTUP

1. From the client's **Client** menu, choose **Configuration**.

The Client Configuration dialog displays.

2. Enter the hostname or IP address of the server to which you want to connect in the **Server Name** field.
3. Generally, you'll leave the default value in the **Server Port** field.  
However, if the administrator who configured the server you want to connect to has changed the server port to be used for client/server communication, you must enter the new port number here. The NerveCenter Client uses this same port number for every NerveCenter Server to which it attempts to connect.
4. Check the **Autoconnect** checkbox.
5. 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 Users (ncusers) or NerveCenter Admins group (ncadmins) on the NerveCenter Server host.
6. Select the **Add** button.  
The server's name and automatic-connection status are displayed in the list near the bottom of the window.
7. Repeat the process for each server you want to connect to automatically.

8. Select the **OK** button.

When you restart and log on to the client, you will be connected to the servers that have an Autoconnect status of On. Alternatively, you can connect, or reconnect, to these servers by selecting **Autoconnect** from the client's **Server** menu.

## Sharing MIB Information from Multiple Servers

The NerveCenter Client needs a copy of the same MIB file that a NerveCenter Server uses to provide MIB base objects and attributes. If you intend to connect to multiple servers that use the same MIB file, you can direct NerveCenter to share MIB information. When you use this option, the NerveCenter Client saves only the MIB information sent to it by the first connected server.

For more information about MIBs, refer to [Managing Management Information Bases \(MIBs\)](#) in *Managing NerveCenter*.

### TO SHARE MIB INFORMATION

1. Disconnect from any connected servers.
2. From the client's **Client** menu, choose **Configuration**.

The Client Configuration dialog is displayed.

The screenshot shows the 'Client Configuration' dialog box with the following details:

- Server Connections** tab is selected.
- Connection Information** section:
  - Server Name: NERVECENTER
  - User ID: ncadmin
  - Server Port: 32504
  - Password: [REDACTED]
  - Autoconnect
- Buttons: Add, Update, Delete
- Server List** table:
 

Name	Autoconnect
NERVECENTER	Off
NC6200-CENTOS6-G	Off
- Server Port**: 32504
- Heartbeat Configuration**:
  - Heartbeat
  - Retry Interval (sec): 30
- Share MIB (client uses MIB from first connected server)
- Buttons: OK, Cancel, Help

3. Select the **Share MIB** checkbox.

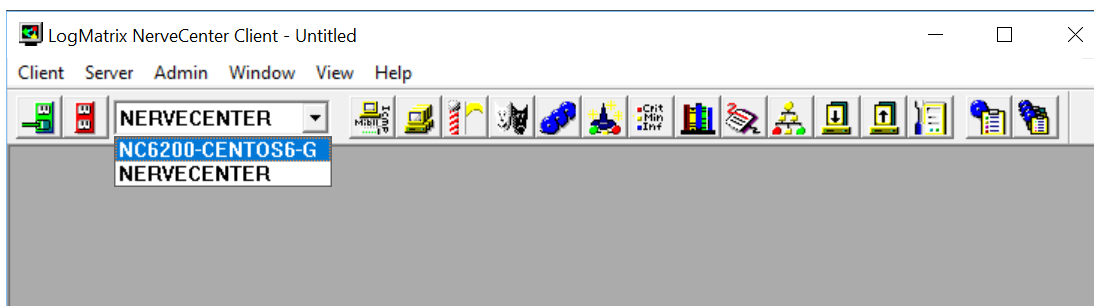
4. Select the **OK** button.

## Selecting the Active Server

The active server is the one whose database you can read data from. That is, you have access to this server's alarm definitions, poll definitions, and so on. You can view alarm instances for any number of servers at the same time.

### TO MAKE A PARTICULAR SERVER THE ACTIVE SERVER

1. Display the server drop-down list on the client's button bar.



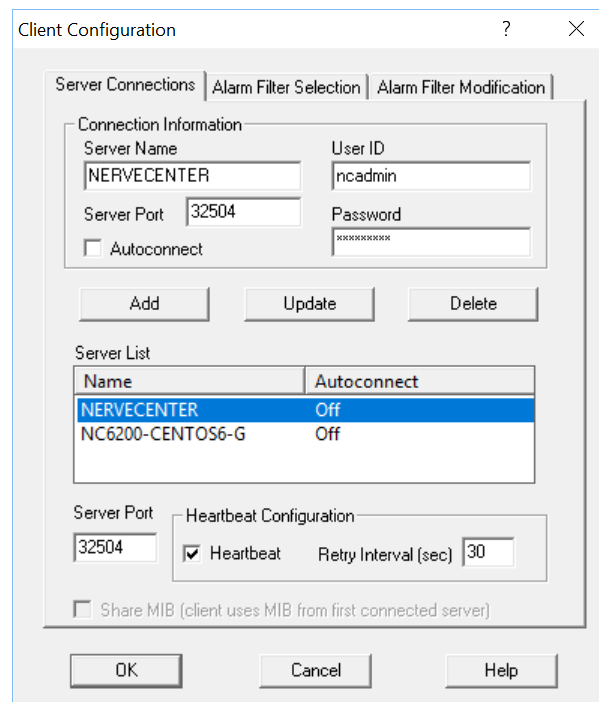
2. Select from the list the name of the server you want to make the active server.  
The name of the active server appears in the drop-down list box.

## Deleting a Server from the Server List

NerveCenter maintains a list of servers that a client has connected to, or attempted to connect to, in the past. This list is used in the Connect to Server window, which you use to establish a connection to a server manually, and it also appears in the Client Configuration window. This list may contain the names of servers that you will never connect to again, or, even worse, the misspelled names of servers you were unable to connect to because of a misspelling.

**TO DELETE THE NAME OF A SERVER FROM THE SERVER LIST**

1. From the client's **Client** menu, select **Configuration**.  
NerveCenter's Client Configuration window is displayed.



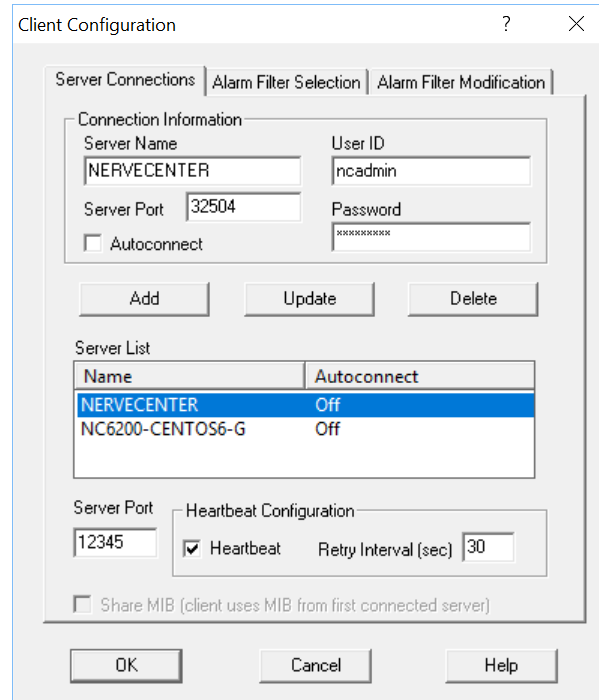
2. In the **Server List** near the bottom of the window, select the server name you want to remove from the server list.
3. Select the **Delete** button.
4. Select the **OK** button.

## Changing the Server Port on the Client

Each NerveCenter server uses a special port on its host for client/server communication. By default, servers use port 32504; however, the person who configures the NerveCenter server can change the number of this communication port if port 32504 is being used by another application. If this number is changed on the server side, you must make a corresponding change on the client side before you will be able to connect to the server.

## TO CHANGE THE CLIENT'S SERVER PORT

1. From the client's **Client** menu, choose **Configuration**.  
The Client Configuration window is displayed.



2. In the **Server List** near the bottom of the window, select the name of the server that uses the non-default port number.  
Connection information for that server is displayed.
3. Type the new port number in the **Server Port** text field.
4. Select the **OK** button.

## Setting Up Alarm-Instance Filters

Before or after you've connected to the servers from which you want to retrieve alarm instances, you can set up one or more alarm-instance filters, per server. These filters control which alarm instances are displayed in the NerveCenter Client. You can filter alarm instances by:

- The IP address of the instance's node
- The severity of the instance's state
- The property group associated with the instance's node

If you filter alarm instances by a severity, only instances whose states have this severity will be displayed in the client. Filters based on property groups and IP address ranges work similarly.



A single filter can contain any combination of:

- A list of subnets
- A list of severities
- A list of property groups

These filters offer two advantages. First, they limit the number of alarm instances that will show up in the client, enabling you to focus your attention on the alarm instances that are specifically of interest to you. Using filters also improves the performance of the client, since NerveCenter only transfers to the client those alarm instances that match the filter criteria.

For information on how to build an alarm-instance filter and on how to associate a filter with a server, see the sections listed below:

- [Filtering Alarms by IP Range below](#)
- [Filtering Alarms by Severity on page 42](#)
- [Filtering Alarms by Property Groups on page 46](#)
- [Associating a Filter with a Server on page 49](#)
- [Rules for Associating Filters with Alarms on page 51](#)

## Filtering Alarms by IP Range

When you filter alarms by IP range, you are specifying that you only want to display alarm instances in the NerveCenter Client from particular nodes identified by their IP addresses. See [IP Subnet Filter Exclusion Rules on page 38](#), for more about filtering alarms by IP ranges. Although you can create a filter simply based on an IP range, a single filter can contain any combination of:

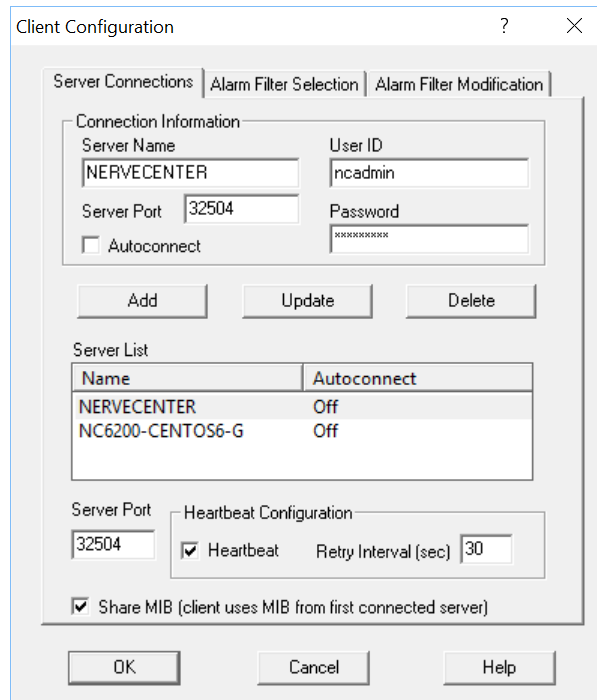
- A list of subnets
- A list of severities
- A list of property groups

For information on how to build an alarm-instance filter based on severities and property groups, see the respective section listed below:

- [Filtering Alarms by Severity on page 42](#)
- [Filtering Alarms by Property Groups on page 46](#)

## TO CREATE AN ALARM FILTER BASED ON AN IP RANGE

1. Choose **Configuration** from the **Client** menu.  
The Client Configuration dialog is displayed.



The screenshot shows the 'Client Configuration' dialog box with the 'Alarm Filter Modification' tab selected. The dialog is divided into several sections:

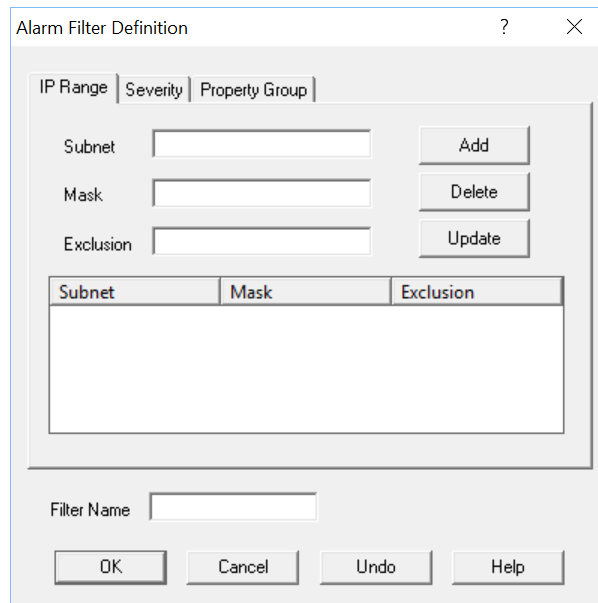
- Connection Information:** Fields for Server Name (NERVECENTER), User ID (ncadmin), Server Port (32504), and Password (masked with asterisks). There is an unchecked 'Autoconnect' checkbox.
- Buttons:** 'Add', 'Update', and 'Delete' buttons.
- Server List:** A table with columns 'Name' and 'Autoconnect'.

Name	Autoconnect
NERVECENTER	Off
NC6200-CENTOS6-G	Off
- Heartbeat Configuration:** A 'Server Port' field (32504) and a 'Heartbeat Configuration' section with a checked 'Heartbeat' checkbox and a 'Retry Interval (sec)' field (30).
- Share MIB:** A checked checkbox labeled 'Share MIB (client uses MIB from first connected server)'.
- Bottom Buttons:** 'OK', 'Cancel', and 'Help' buttons.

2. Select the **Alarm Filter Modification** tab.  
The Alarm Filter Modification page is displayed.



3. Select the **New** button.  
The Alarm Filter Definition dialog is displayed.



4. If you want to filter alarm instances based on the IP addresses of the alarm instances' nodes, perform the steps below for each subnet you want to be part of the filter. That is, you want to see information about instances whose nodes have IP addresses on these subnets.
    - a. Enter an IP address in the **Subnet** text field.  
The IP address must consist of four octets separated by periods.
    - b. Enter a subnet mask in the **Mask** text field.  
The subnet mask must consist of four octets separated by periods. Taken together with the subnet address, this mask defines the subnet whose nodes you're monitoring.
    - c. In the **Exclusion** text field, enter the last octet of the IP address of any node on the subnet that you're not monitoring.  
You can enter multiple exclusions separated by commas. You can also enter a range of excluded nodes using a hyphen. For example, if you enter 24, 76-78 in the Exclusion field, the nodes whose addresses end in 24, 76, 77, and 78 will be excluded by the filter.
    - d. Select the **Add** button.
    - e. Repeat [Step a](#) to [Step d](#) to add other subnets to the alarm filter.
  5. Enter a name for your filter in the **Filter Name** field.
  6. Select the **OK** button.  
The Alarm Filter Definition dialog is closed and you return to the Client Configuration dialog box.
- 

You've now defined an alarm filter based on an IP range. Before the client will use the filter, however, you must associate the filter with a server. For instructions on how to create this association, see the section [Associating a Filter with a Server on page 49](#).

### IP Subnet Filter Exclusion Rules

When you filter by subnet, you specify which subsets of nodes are managed by NerveCenter. Filtering does not apply to nodes that have been imported from a file or from another NerveCenter. For an example, see [IP Subnet Filter Examples on page 40](#).

You can exclude specific nodes that belong to the filter by entering an exclusion. To exclude one or more nodes, you must specify the full subnet and mask, and then enter the individual nodes you want excluded. Enter the part of the IP address that is not affected by the subnet's mask.

NerveCenter filters Class B and C networks.

- **In a Class C network**, the first three octets of the address specify the network and the last octet specifies the host. For example, in network 194.123.45.0, the 194.123.45 value pertains to the network. The remaining octet is used to identify nodes (up to 254) on the network, and you can exclude nodes by specifying ID values in this octet.
- **In a Class B network**, only the first two octets of the address specify the network. For example, in network 132.45.0.0, the 132.45 value pertains to the network. The remaining two octets are used to identify nodes, and you can exclude nodes by specifying ID values in these two octets.

### Example

In the following example, the node whose IP address is 134.204.179.40 is excluded from the filter (the node is filtered out and, therefore, is not managed by NerveCenter).

134.204.179.0

255.255.255.0

40

### Rules for Exclusions

- You can enter several nodes separated by a comma. NerveCenter accepts comma-separated values with or without spaces following the commas. You can enter the node values in any order.

The following three examples (each on a separate line) illustrate valid exclusions:

7, 8, 9, 15

7, 8, 9, 15

8, 7, 9, 15

- You can enter a range of values using a hyphen.

For example, you can enter as an exclusion range: **40-60**

You can also enter the range in inverse order: **60-40**

- You can include multiple entries for the same subnet if you have values or ranges that are not incremental.

- For example, you can enter as a filter:

134.204.179.0

255.255.255.0

7, 8, 9

134.204.179.0

255.255.255.0

40-60

134.204.179.0

255.255.255.0

70-90

- You can combine ranges, for example:

134.204.179.0

255.255.255.0

40-60, 70-90

- You can also combine formats, for example:

134.204.179.0

255.255.255.0

7-9, 31, 33, 40-60

### IP Subnet Filter Examples

The following examples can help you understand how to filter nodes for Class B and C networks.

#### Class C Network

The following subnet filters are for two sample nodes:

- Sample node #1 with IP address: 197.204.179.25
- Sample node #2 with two IP addresses:
  - 134.204.179.40
  - 197.204.179.7

The filter values in [Table 2](#) have the following effects on the sample nodes:

Table 2: Class C Network Examples

Subnet Mask Exclusion	Results of Filter
134.204.179.0 255.255.255.0	This filter does not contain any exclusions. Node #1 is not on this subnet and is not included in the filter or managed by NerveCenter. Node #2 is included in the filter because it's on the subnet.
134.204.179.0 255.255.255.0 25,40	Node #1 is not on this subnet and is not included in the filter. Node #2 is listed as an exclusion and is not included in the filter.
197.204.179.0 255.255.255.0 7-20	Node #1 is included. Node #2 is not included because it's listed in the exclusion range.
197.204.179.0 255.255.255.0 7-20 134.204.179.0 255.255.255.0 40	Node #1 is included in the first subnet. Node #2 is not included because it's listed as an exclusion on both subnets.
197.204.179.0 255.255.255.0 25,40	Node #1 is not included because it's listed as an exclusion. Node #2 is included.

### Class B Filters

The following subnet filters are for two sample nodes:

- Sample node #1 with IP address: 132.45.160.10
- Sample node #2 with IP address: 132.45.174.10

The mask you use for this filter is 255.255.0.0.

Table 3: Class B Filter Examples (Set One)

Subnet Mask Exclusion	Results of Filter
132.45.0.0 255.255.0.0	Both nodes are included in the filter and managed by NerveCenter.
132.45.0.0 255.255.0.0 174.10	Node #1 is included in the filter. Node #2 is excluded from the filter. The filter includes all nodes except 132.45.174.10.
132.45.0.0 255.255.0.0 160.10- 174.5	Node #1 is listed in the exclusion range and is excluded from the filter. Node #2 is included in the filter.
132.45.0.0 255.255.0.0 10	Both nodes are excluded from the filter and, therefore, neither node is managed by NerveCenter. The filter includes all nodes except 132.45.xxx.10, where xxx can be any value greater than 1 and less than 255.

If you use a subnet mask of 255.255.240.0, you would get different results.

- Sample node #1 with IP address: 132.45.160.10
- Sample node #2 with IP address: 132.45.174.10

You must first apply the filter before determining the node's ID. The filter values in the table below have the following effects:

Table 4: Class B Filter Examples (Set Two)

Subnet Mask Exclusion	Results of Filter
132.45.160.0 255.255.240.0 174.10	The node is not included in the filter. The filter includes all nodes except 132.45.174.10.
132.45.160.0 255.255.240.0 10	Neither node is included in the filter. The filter includes all nodes except those ending in .10. The third octet of an excluded node can be 174 or any value between 160 and 174.

## Filtering Alarms by Severity

When you filter alarms by severity, you are specifying that you only want to display alarm instances in the NerveCenter Client from particular nodes identified by the severity of the alarm instance's state.

Although you can create a filter simply based on severity, a single filter can contain any combination of:

- A list of subnets
- A list of severities
- A list of property groups

For information on how to build an alarm-instance filter based on IP range and property groups, see the respective section listed below:

- [Filtering Alarms by IP Range on page 35](#)
- [Filtering Alarms by Property Groups on page 46](#)



## TO CREATE AN ALARM FILTER BASED ON SEVERITY

1. Choose **Configuration** from the **Client** menu.  
The Client Configuration dialog is displayed.

Client Configuration

Server Connections | Alarm Filter Selection | Alarm Filter Modification

Connection Information

Server Name: NERVECENTER      User ID:

Server Port:       Password:

Autoconnect

Add      Update      Delete

Server List

Name	Autoconnect
NERVECENTER	Off

Server Port: 32504      Heartbeat Configuration

Heartbeat      Retry Interval (sec): 30

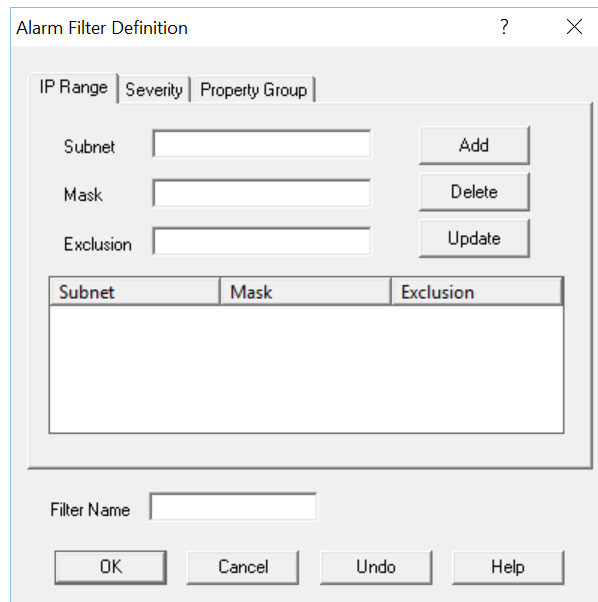
Share MIB (client uses MIB from first connected server)

OK      Cancel      Help

2. Select the **Alarm Filter Modification** tab.  
The Alarm Filter Modification page is displayed.

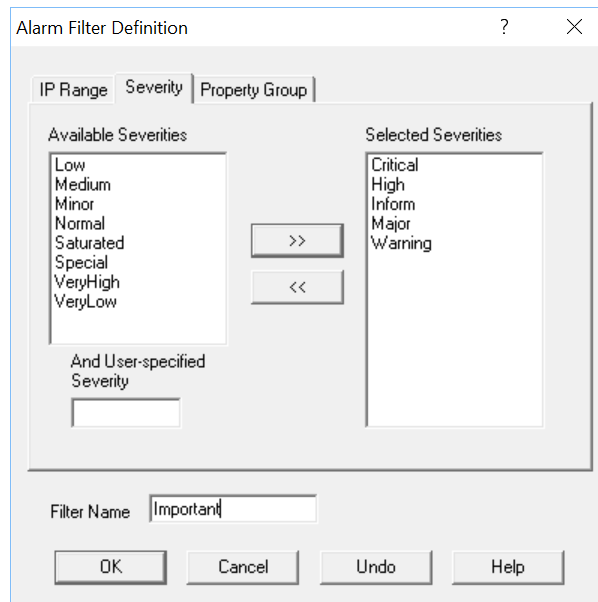


3. Select the **New** button.  
The Alarm Filter Definition dialog is displayed.



This is the dialog you use to define your filter.

4. Select the **Severity** tab.  
The Severity tab is displayed.



5. In the **Available Severities** list, for each severity you want to use in your filter, select the severity and then select the >> button. That is, you want to see information about alarm instances whose states have these severities.

The severities in this list box are the union of the severities defined by all of the servers to which you're connected. You can also add a user-defined severity to the list of severities to filter by entering it in the **And User-specified Severity** text box, and then clicking >>.

The name of the severity is moved to the **Selected Severities** list. Information about alarm instances with this severity will be displayed in the alarm summary views.

To remove a severity from the **Selected Severities** list, select the severity and then click <<.

6. Enter a name for your filter in the **Filter Name** field.
7. Select the **OK** button.  
You return to the Client Configuration dialog box.

---

You've now defined an alarm filter based on severity. Before the client will use the filter, however, you must associate the filter with a server. For instructions on how to create this association, see the section [Associating a Filter with a Server on page 49](#).

## Filtering Alarms by Property Groups

When you filter alarms by property groups, you are displaying alarm instances in the NerveCenter Client from particular nodes belonging to one or more property groups. While you can create a filter based on membership within a property group, a single filter can contain any combination of subnets, severities, or property groups.

For more on building an alarm-instance filter based on an IP range and severities, see the respective section listed below:

- [Filtering Alarms by IP Range on page 35](#)
- [Filtering Alarms by Severity on page 42](#)

### TO CREATE AN ALARM FILTER BASED ON PROPERTY GROUPS

1. Choose **Configuration** from the **Client** menu.  
The Client Configuration dialog is displayed.

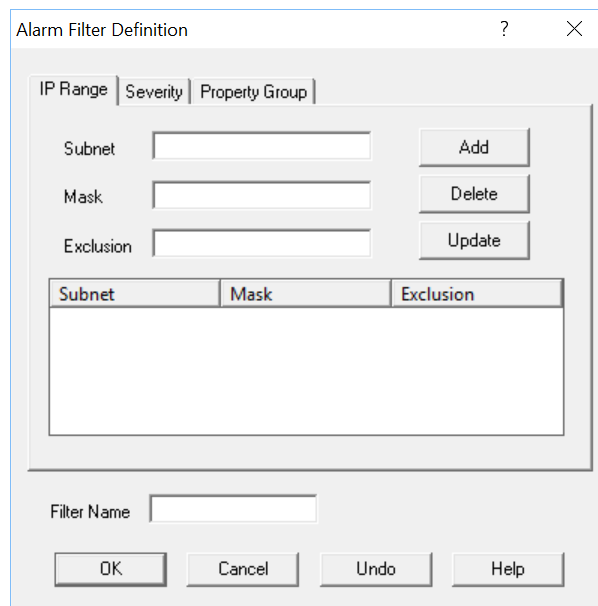
The screenshot shows the 'Client Configuration' dialog box with the 'Alarm Filter Selection' tab selected. The dialog is divided into several sections:

- Connection Information:** Contains fields for 'Server Name' (filled with 'NERVECENTER'), 'User ID', 'Server Port', and 'Password'. There is also an 'Autoconnect' checkbox which is currently unchecked.
- Buttons:** 'Add', 'Update', and 'Delete' buttons are located below the connection information fields.
- Server List:** A table with two columns: 'Name' and 'Autoconnect'. It contains one entry: 'NERVECENTER' with 'Off' in the 'Autoconnect' column.
- Server Port:** A field containing '32504'.
- Heartbeat Configuration:** Includes a checked 'Heartbeat' checkbox and a 'Retry Interval (sec)' field set to '30'.
- Share MIB:** An unchecked checkbox with the text '(client uses MIB from first connected server)'.
- Bottom Buttons:** 'OK', 'Cancel', and 'Help' buttons.

2. Select the **Alarm Filter Modification** tab.  
The Alarm Filter Modification tab is displayed.



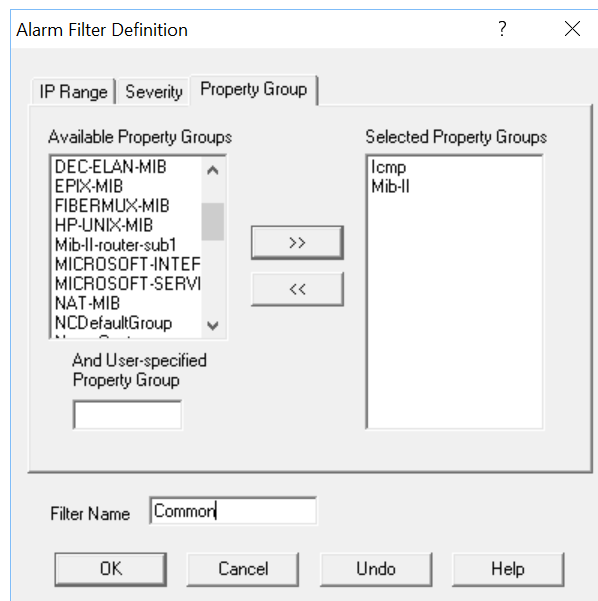
3. Select the **New** button.  
The Alarm Filter Definition dialog is displayed.



The screenshot shows the "Alarm Filter Definition" dialog box with the "IP Range" tab selected. The dialog has three tabs: "IP Range", "Severity", and "Property Group". Under the "IP Range" tab, there are three input fields: "Subnet", "Mask", and "Exclusion". To the right of each field is a button: "Add" for Subnet, "Delete" for Mask, and "Update" for Exclusion. Below these fields is a table with three columns: "Subnet", "Mask", and "Exclusion". The table is currently empty. At the bottom of the dialog, there is a "Filter Name" input field and four buttons: "OK", "Cancel", "Undo", and "Help".

This is the dialog you use to define your filter.

4. Select the **Property Group** tab.  
The Property Group tab is displayed.



The screenshot shows the "Alarm Filter Definition" dialog box with the "Property Group" tab selected. The dialog has three tabs: "IP Range", "Severity", and "Property Group". Under the "Property Group" tab, there are two lists: "Available Property Groups" and "Selected Property Groups". The "Available Property Groups" list contains: DEC-ELAN-MIB, EPIX-MIB, FIBERMUX-MIB, HP-UNIX-MIB, Mib-II-router-sub1, MICROSOFT-INTEF, MICROSOFT-SERVI, NAT-MIB, and NCDDefaultGroup. Below this list is a text input field labeled "And User-specified Property Group". Between the two lists are two buttons: ">>" and "<<". The "Selected Property Groups" list contains: Icmp and Mib-II. At the bottom of the dialog, there is a "Filter Name" input field containing the text "Common" and four buttons: "OK", "Cancel", "Undo", and "Help".

5. In the **Available Property Groups** list, for each property group of each alarm instance's node, perform the steps below for each property group you want to be part of the filter. That is, you want to see information about instances whose nodes belong to these property groups.

The property groups in this list box are the union of the property groups defined by all of the servers to which you're connected.

The property group is moved to the **Selected Property Groups** list. Information about alarm instances with this property will be displayed in the alarm summary views. Optionally, you can also add a user-defined property group to the list of properties to filter by entering a property group in the **And User-specified Property Group** text box, and then click **>>**.

To remove a property group from the **Selected Properties** list, select it and then click **<<**.

6. Enter a name for your filter in the **Filter Name** field.
7. Select the **OK** button.

You return to the Client Configuration dialog box.

---

You've now defined an alarm filter based on property groups. Before the client will use the filter, however, you must associate the filter with a server. For instructions on how to create this association, see the section [Associating a Filter with a Server](#) below.

## Associating a Filter with a Server

When you define an alarm filter, that filter is not used to filter alarm instances from all connected servers. It is only used to filter alarm instances from a server with which you have explicitly associated it.

## TO ASSOCIATE AN ALARM FILTER WITH A NERVECENTER SERVER

1. Choose **Configuration** from the **Client** menu.

The Client Configuration dialog is displayed.

Client Configuration

Server Connections | Alarm Filter Selection | Alarm Filter Modification

Connection Information

Server Name: NERVECENTER    User ID:

Server Port:     Password:

Autoconnect

Add    Update    Delete

Server List

Name	Autoconnect
NERVECENTER	Off

Server Port: 32504    Heartbeat Configuration

Heartbeat    Retry Interval (sec): 30

Share MIB (client uses MIB from first connected server)

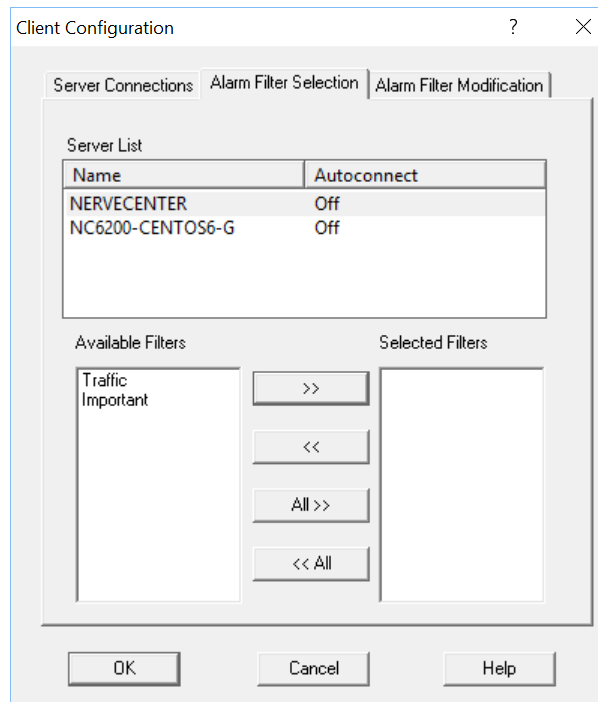
OK    Cancel    Help

2. Select a server from the list of servers at the bottom of the dialog.

The name of the server appears in the **Server Name** text field in the Connection Information group box. This is the server with which you will associate your alarm filter.



3. Select the **Alarm Filter Selection** tab.  
The Alarm Filter Selection page is displayed.



4. Select a filter from the **Available Filters** list.  
This is the filter you want to associate with the server you selected in [Step 2](#).
5. Select the >> button to move the filter from the **Available Filters** list to the **Selected Filters** list.  
To remove a filter from the **Selected Filters** list, select the filter and then select the << button.
6. Select the **OK** button at the bottom of the dialog.

## Rules for Associating Filters with Alarms

When deciding whether to apply multiple filters to your alarms, you should keep in mind the following general rules:

- Multiple filters are ORed together
- Multiple conditions in a single filter are ANDed together

### Multiple Filters are ORed Together

When you select more than one filter for a server, each filter is independent of the other filters. Their behavior is equivalent to a logical OR operation.

For example, say you associate two filters with a NerveCenter Server. The two filters are defined as follows:

- Filter #1 is configured to display only those alarms that have a severity level of Critical.
- Filter #2 is configured to display only those alarms coming from the network 132.168.196.0.

When both filters are applied to a server, you see the following alarms:

- Alarms with a Critical severity level from all existing networks defined for the server.
- From the network 132.168.196.0, you see all alarms regardless of severity.

### Multiple Conditions in a Single Filter are ANDed Together

If, instead of the above view, you want to limit your alarms to Critical instances coming from the network 132.168.196.0, you need to create one filter with both of those conditions. You would create one filter that:

- Specifies a severity level of Critical, and
- Specifies an IP range of 132.168.196.0.

With this filter applied to the server, you see only those alarms that have a Critical severity level *and* that come from network 132.168.196.0. One filter with multiple conditions is equivalent to a logical AND operation; each condition is ANDed with the other conditions for optimum filtering.

## Specifying Heartbeat Messaging

The NerveCenter Client sends a message called a *heartbeat* to each connected NerveCenter Server on a standard interval. This messaging ensures the reliability of communications between the server and client. If a server fails to respond after three consecutive heartbeat messages from the client, a message box is displayed on the client console to alert the operator of the server's heartbeat failure. (In such cases, you should check with your network administrator to obtain the status of that particular NerveCenter Server.)

You can set the interval at which the NerveCenter Client sends a heartbeat to the NerveCenter Server (30 seconds by default). You can also choose to deactivate heartbeat messaging.

See the following sections for more information:

- [Modifying the Heartbeat Message Interval on the facing page](#)
- [Deactivating Heartbeat Messaging on page 54](#)

## Modifying the Heartbeat Message Interval

You can change the interval NerveCenter Client uses to send heartbeat messages to verify its connection with your NerveCenter Servers.

### TO MODIFY THE HEARTBEAT MESSAGE INTERVAL

1. Choose **Configuration** from the **Client** menu.  
The Client Configuration dialog is displayed.

The screenshot shows the 'Client Configuration' dialog box with the 'Heartbeat Configuration' panel selected. The 'Heartbeat' checkbox is checked, and the 'Retry Interval (sec)' is set to 30. The 'Server List' table shows one entry: 'NERVECENTER' with 'Autoconnect' set to 'Off'.

Name	Autoconnect
NERVECENTER	Off

2. In the **Heartbeat Configuration** panel, make sure the **Heartbeat** checkbox is checked. If it's not checked, heartbeat messaging is turned off.
3. In the **Retry Interval** field, enter the number of seconds you want NerveCenter Client to wait between heartbeat messages. The default is 30 seconds. (The number of retries is three.)

**Note:** When you modify heartbeat messaging, it applies to all NerveCenter Servers to which this client connects.

4. Select the **OK** button.

## Deactivating Heartbeat Messaging

The NerveCenter Client sends heartbeat messages on an interval that you specify (or by default, every 30 seconds) to verify its connection with your NerveCenter Servers. If you choose, you can deactivate (or activate) heartbeat messages going to and from *all* your connected servers.

### TO DEACTIVATE HEARTBEAT MESSAGES

1. Choose **Configuration** from the **Client** menu.  
The Client Configuration dialog is displayed.

Client Configuration

Server Connections | Alarm Filter Selection | Alarm Filter Modification

Connection Information

Server Name: NERVECENTER    User ID: ncadmin

Server Port: 32504    Password: \*\*\*\*\*

Autoconnect

Add    Update    Delete

Server List

Name	Autoconnect
NERVECENTER	Off
NC6200-CENTOS6-G	Off

Server Port: 32504    Heartbeat Configuration

Heartbeat    Retry Interval (sec): 30

Share MIB (client uses MIB from first connected server)

OK    Cancel    Help

2. In the Heartbeat Configuration panel, uncheck the **Heartbeat** checkbox.

**Note:** If there is no check mark in this checkbox, heartbeat messaging has already been deactivated for NerveCenter Client. When you activate or deactivate heartbeat messaging, it applies to all NerveCenter Servers to which this client connects.

3. Select the **OK** button.

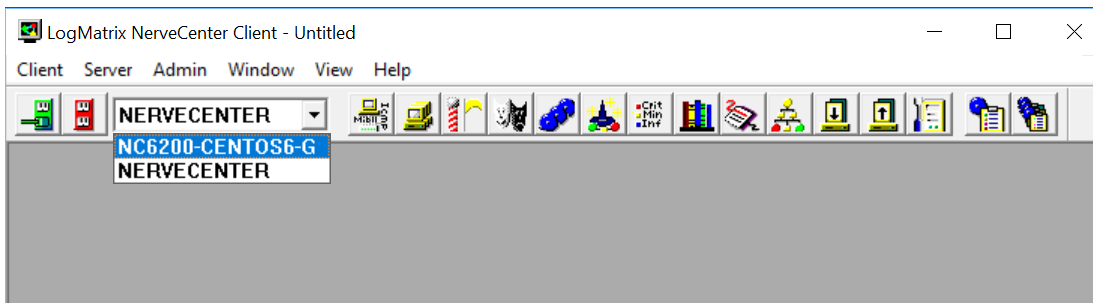
Heartbeat deactivation takes effect the next time you connect NerveCenter Client to one or more of your NerveCenter Servers.

## Disconnecting from a Server

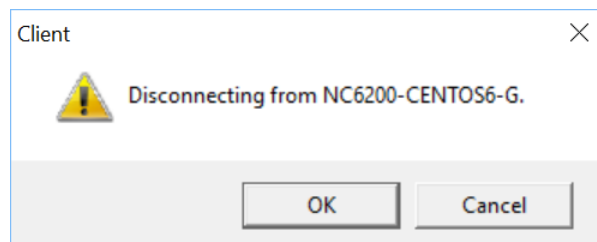
When you exit the client, all connections to NerveCenter servers are broken. However, you may also want to disconnect the client from a server without stopping the client.

### TO DISCONNECT THE CLIENT FROM A SERVER

1. From the server drop-down list on the client's button bar, select the server with which you want to break the connection.



2. From the client's **Server** menu, choose **Disconnect**.  
You see a pop-up window that asks you to confirm that you want to disconnect from the selected server.



3. Select the **OK** button.



---

# Monitoring Alarms

When NerveCenter detects a condition it is looking for, it creates an alarm instance that tracks that condition. For instance, if your site uses the behavior model that includes the alarm `ifLoad`, NerveCenter monitors network traffic on a set of interfaces. If it detects a traffic level above a certain threshold, it creates an alarm instance to track that condition.

The NerveCenter Client features interfaces that enable you to monitor these alarm instances. This chapter discusses:

- The interfaces that these clients provide for monitoring alarm instances
- How to interpret the information these interfaces present
- How to examine an alarm instance's history

## Viewing Alarm Information

When an alarm instance is instantiated or undergoes a transition, you need to know certain things about the alarm transition that just took place:

- Which alarm was instantiated or underwent a transition? If the name of the alarm was `ifLinkUpDown`, you know that you received a link-down or a link-up trap.
- What node was the alarm instance monitoring? Was it monitoring a particular interface on a device?
- What state is the alarm instance now in? And what is the severity of that state? If the alarm involved is `ifLinkUpDown` and the current state is `LinkDown` (Major severity), you know that a communication link is down.
- What NerveCenter object caused the alarm instance to be instantiated or undergo a transition? If you know that the poll `MediumLoad` caused the instantiation or transition, you can look at the definition of that poll to determine exactly what condition is being reported.

The NerveCenter Client provides interfaces that present you with summary information about the current alarm instances in which you're interested. For information about bringing up these interfaces and about the information they present, see [Using the NerveCenter Client on the next page](#)

## Using the NerveCenter Client

The NerveCenter Client provides two windows that you can use to view information about current alarm instances:

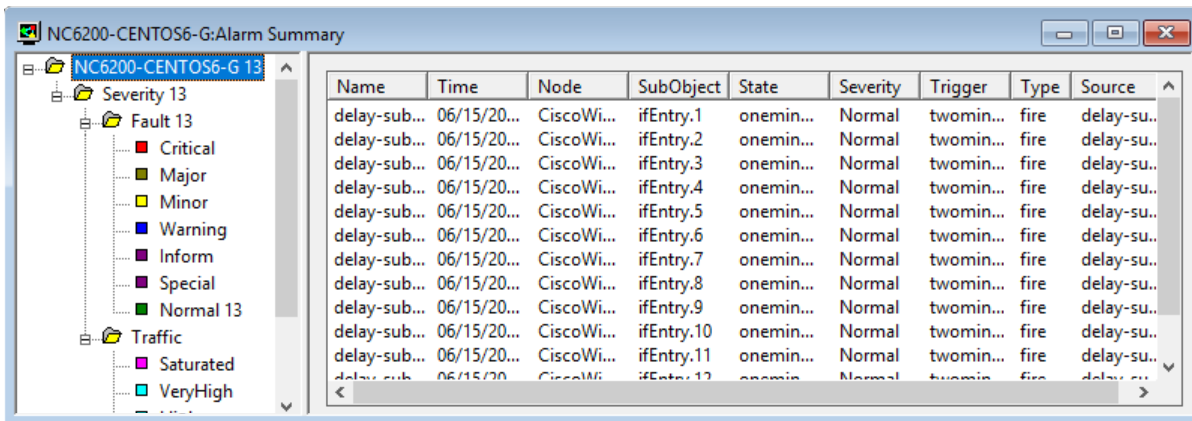
- The Alarm-Summary window
- The Aggregate Alarm Summary window

If you're only connected to one server or are only interested in viewing alarm instances from one server at a time, you should use the Alarm Summary window. On the other hand, if you are connected to multiple servers and want to be able to view alarm instances from all servers at once, you should use the Aggregate Alarm Summary window.

### TO OPEN THE ALARM SUMMARY WINDOW

- From the client's **Admin** menu, choose **Alarm Summary**.

The Alarm Summary window is displayed.



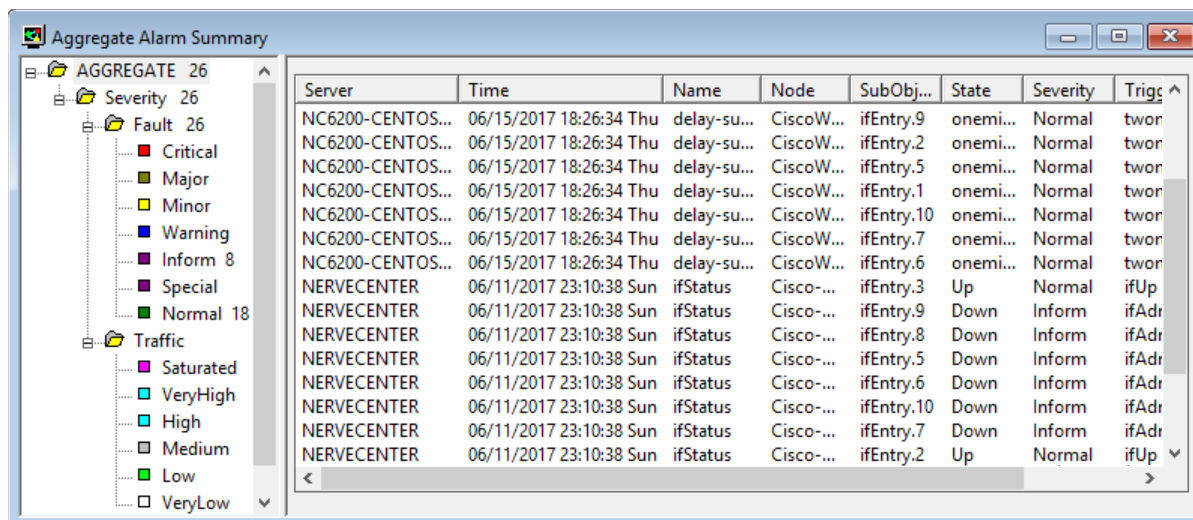


---

**TO OPEN THE AGGREGATE ALARM SUMMARY WINDOW**

- From the client's **Admin** menu, choose **Aggregate Alarm Summary**.

The Aggregate Alarm Summary window is displayed.



These windows are very similar. Both contain a tree view of the current alarm instances in the left pane, and details about the current alarm instances in the right pane. For information about how to interpret the information in these two panes, see the following sections:

- [The Tree View on the next page](#)
- [The Alarm-Detail View on page 61](#)

### The Tree View

The left pane in both the Alarm Summary window and the Aggregate Alarm Summary window contains a tree of severities.

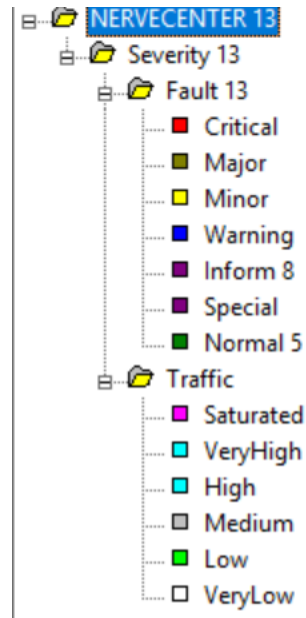


Figure 11: Severity Tree

The only difference between the two trees is that the top folder in the Alarm Summary window represents the active server, while the top folder in the Aggregate Alarm Summary window represents all of the servers to which you are connected.

This tree view has two purposes:

- It enables you to see at a glance 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. For instance, the tree above indicates that there are five alarm instances of Major severity.
- It enables you to control which alarm instances appear in the alarm detail pane. If you select the Major icon, only alarm instances of Major severity will appear in the alarm detail pane.

## The Alarm-Detail View

The right pane in both the Alarm Summary and Aggregate Alarm Summary windows contains an alarm detail view that presents quite a bit of information about each current alarm instance, as shown in [Figure 12](#).

Server	Time	Name	Node	SubObj...	State	Severity	Trigger	Type
NC6200-CENTOS6-G	06/15/2...	delay-s...	CiscoW...	ifEntry.3	onemi...	Normal	twomi...	fire
NC6200-CENTOS6-G	06/15/2...	delay-s...	CiscoW...	ifEntry.9	onemi...	Normal	twomi...	fire
NC6200-CENTOS6-G	06/15/2...	delay-s...	CiscoW...	ifEntry.6	onemi...	Normal	twomi...	fire
NC6200-CENTOS6-G	06/15/2...	delay-s...	CiscoW...	ifEntry.10	onemi...	Normal	twomi...	fire
NC6200-CENTOS6-G	06/15/2...	delay-s...	CiscoW...	ifEntry.4	onemi...	Normal	twomi...	fire
NC6200-CENTOS6-G	06/15/2...	delay-s...	CiscoW...	ifEntry.11	onemi...	Normal	twomi...	fire
NC6200-CENTOS6-G	06/15/2...	delay-s...	CiscoW...	ifEntry.2	onemi...	Normal	twomi...	fire
NC6200-CENTOS6-G	06/15/2...	delay-s...	CiscoW...	ifEntry.12	onemi...	Normal	twomi...	fire
NC6200-RHEL72-G	06/17/2...	ICMP-E...	virthost...		Reacha...	Normal	Ping_H...	poll
NERVECENTER	06/11/2...	ifStatus	Cisco-...	ifEntry.1	Up	Normal	ifUp	poll
NERVECENTER	06/11/2...	ifStatus	Cisco-...	ifEntry.3	Up	Normal	ifUp	poll
NERVECENTER	06/11/2...	ifStatus	Cisco-...	ifEntry.2	Up	Normal	ifUp	poll
NERVECENTER	06/11/2...	ifStatus	Cisco-...	ifEntry.13	Up	Normal	ifUp	poll
NERVECENTER	06/11/2...	ifStatus	Cisco-...	ifEntry.4	Up	Normal	ifUp	poll

Figure 12: Alarm Detail Pane

This is the pane you'll use for most of your monitoring.

[Table 5](#) explains what information is available for each alarm instance.

Table 5: Fields in Alarm Detail Pane

Column	Description
Server	The name of the server that is managing the alarm instance. This column is present only in the Aggregate Alarm Summary window.
Time	The date and time at which the alarm instance's most recent transition occurred.
Name	The name of the alarm from which the alarm instance was created. If you have any question about what condition a particular alarm is monitoring, you can use the NerveCenter Client to view a definition of the alarm. For information about viewing such a definition, see the section <a href="#">Getting Information about an Alarm</a> on page 63.
Node	The hostname of IP address of the node the alarm instance is monitoring.
SubObject	The subobject associated with the alarm instance. This subobject consists of a MIB base object plus an instance number, for example, ifEntry.1. The instance usually tells you which interface on a device is being monitored.
State	The current state of the alarm instance. The name of the state should indicate the condition NerveCenter is reporting. For example, if an instance of the alarm IfUpDownStatus is monitoring an interface and the current state of the alarm instance is "down," the operational status of the interface is down.
Severity	The severity of the alarm instance's current state.

Column	Description
Trigger	The name of the trigger that caused the most recent alarm transition.
Type	The type of trigger that caused the most recent alarm transition. The possible types are poll, mask, fire (alarm), and built-in.
Source	The name of the poll, mask, or alarm that generated the trigger. Or, in the case of a built-in trigger, the name of the trigger. Given the name of the trigger that caused the transition and the name of the object that generated the trigger, you can pinpoint the exact cause of a transition. See the section <a href="#">Getting Information about a Trigger on page 64</a> for details on this subject.

The alarm detail pane is designed primarily for reading. However, there are a couple of actions you can take from this pane.

- You can select any of the column headings to sort the alarm-instance entries alphabetically by the values in that column. Selecting the column heading a second time reverses the order of the entries.  
This feature is useful for tasks such as ordering alarm instances by node.
- You can double-click the entry for an alarm instance to open an alarm-history window. For more information about alarm history, see the section [Viewing Alarm Instance History on page 72](#).

## Interpreting Alarm-Instance Information

The alarm-instance information that you can view using the NerveCenter Web Client and the NerveCenter Client is meant to stand on its own, that is, to provide you with all the information you need concerning a network condition. However, until you become familiar with all of the behavior models being used at your site, you might need some supplementary information. For example, suppose you see the summary information shown in [Figure 13](#):

Server	Time	Name	Node	SubObj...	State	Severity	Trigger	Type	Source	^
NERVECENTER	06/11/2017 23:10:38 Sun	ifStatus	Cisco-WirelessVPNRouter	ifEntry.5	Down	Inform	ifAdmi...	poll	ifStatus	
NERVECENTER	06/11/2017 23:10:38 Sun	ifStatus	Cisco-WirelessVPNRouter	ifEntry.6	Down	Inform	ifAdmi...	poll	ifStatus	
NERVECENTER	06/11/2017 23:10:38 Sun	ifStatus	Cisco-WirelessVPNRouter	ifEntry.7	Down	Inform	ifAdmi...	poll	ifStatus	

Figure 13: Summary Alarm Information

It's clear what node is being reported on. However, if you're not familiar with the Authentication alarm, it may not be clear what it means for an instance of this alarm to be in the state Alert3. (As it turns out, this state indicates that a node has received three authentication-failure traps within a ten-minute period.) To find out what this state means, you can use the NerveCenter Client to look at the documentation for, and definition of, this alarm. For information on this subject, see the section [Getting Information about an Alarm on the facing page](#).

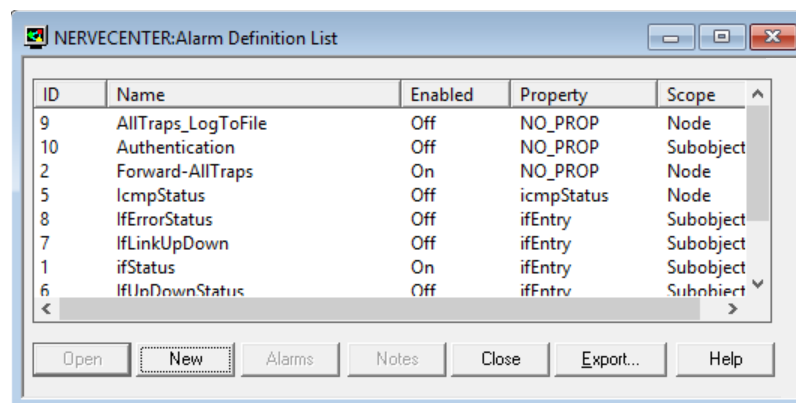
Also, it may not always be clear what condition caused the Source to generate the trigger that led to the most recent alarm transition. In the figure above, a mask called AuthFail generated the trigger authFail. You can probably guess that a trap mask responded to an authentication-failure trap. But what if you were monitoring an instance of the alarm ifErrorStatus (which monitors the percentage of error packets on an interface) and the poll MediumErrorRate fired the trigger mediumErrorRate. You could infer that NerveCenter had seen a moderate number of error packets on an interface, but what constitutes a medium error rate? You can find out by using the NerveCenter Client to read the documentation for, or definition of, the MediumErrorRate poll. For further information on interpreting the meaning of a trigger, see the section [Getting Information about a Trigger on the next page](#).

## Getting Information about an Alarm

If you're monitoring alarm instances and have a question about a particular alarm or alarm state, you can easily obtain information about the purpose of the alarm and what states are defined.

### TO GET THIS INFORMATION

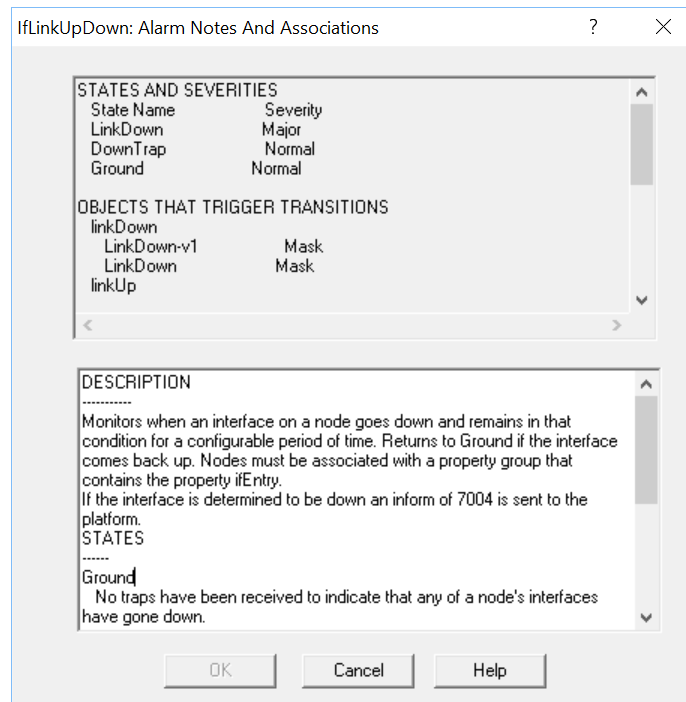
1. From the NerveCenter Client's **Admin** menu, choose **Alarm Definition List**.  
The Alarm Definition List window is displayed.



2. Select the alarm you're interested in.  
The **Notes** button is enabled.

3. Select the **Notes** button.

The Alarm Notes and Associations dialog displays.



These notes should include the following information:

- A list of states and their severities
- A list of transitions and the objects that can trigger those transitions
- A list of the actions associated with each transition
- A brief description of the purpose of the alarm
- A description of each state in the alarm
- Information about other alarms that are part of the same behavior model

**Note:** If you need further information about an alarm, you can look at its definition. To view this definition, select the alarm in the Alarm Definition List window and then select the **Open** button.

## Getting Information about a Trigger

If you're monitoring alarm instances and want further information about the cause of an alarm transition, you can obtain that information using the NerveCenter Client. The specific procedure you should follow depends on the type of the trigger that caused the transition. [Table 6](#) directs you to the appropriate subsection.

Table 6: Finding Information about the Source of a Trigger

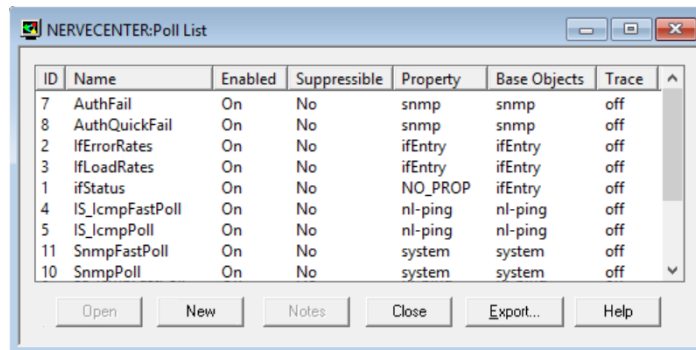
Type	See this Section
poll	<a href="#">A Trigger Generated by a Poll below</a>
mask	<a href="#">A Trigger Generated by a Mask on the next page</a>
fire (alarm)	<a href="#">A Trigger Generated by an Alarm on page 68</a>
built-in	<a href="#">Built-In Triggers on page 69</a>

### A Trigger Generated by a Poll

If you're monitoring alarm instances and see that an alarm transition took place when the poll CsCpuBusy fired the csCpuBusy trigger, how can you determine what condition caused the poll to fire this trigger? To get this information, follow the procedure below.

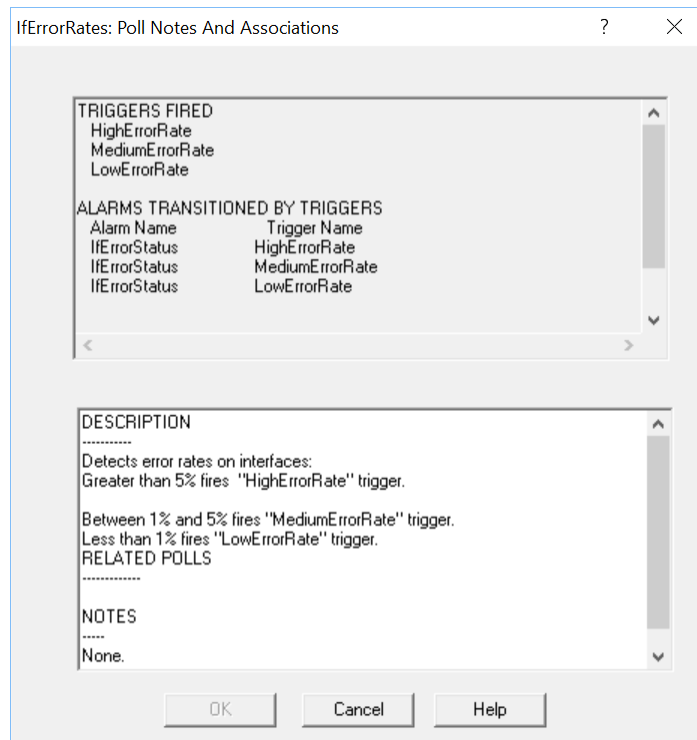
#### TO DETERMINE WHY A POLL FIRED A TRIGGER

1. From the NerveCenter Client's **Admin** menu, choose **Poll List**.  
The Poll List window is displayed.



2. Select the poll you're interested in.  
The **Notes** button is enabled.

3. Select the **Notes** button.  
The Poll Notes dialog displays.



For each poll, the note should include the following information:

- A list of the triggers the poll can fire
- A list of the alarms in which the poll can cause a transition
- A list of other objects that can fire any one of the triggers fired by this poll
- A brief description of the poll and its poll condition

**Note:** If you need additional information about a poll, you can look at its definition. To view this definition, select the poll in the Poll List window and then select the **Open** button.

### A Trigger Generated by a Mask

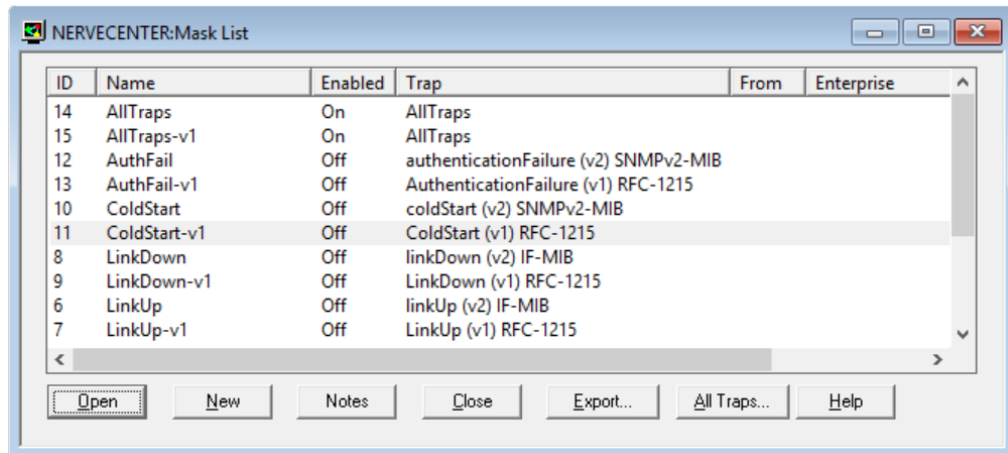
If you're monitoring alarm instances and see that an alarm transition took place when the SynBoardPowerFail trap mask fired the synBoardPsTrap trigger, how can you determine what condition caused the mask to fire this trigger? To get this information, follow the procedure below.



## TO DETERMINE WHY A MASK FIRED A TRIGGER

1. From the NerveCenter Client's **Admin** menu, choose **Mask List**.

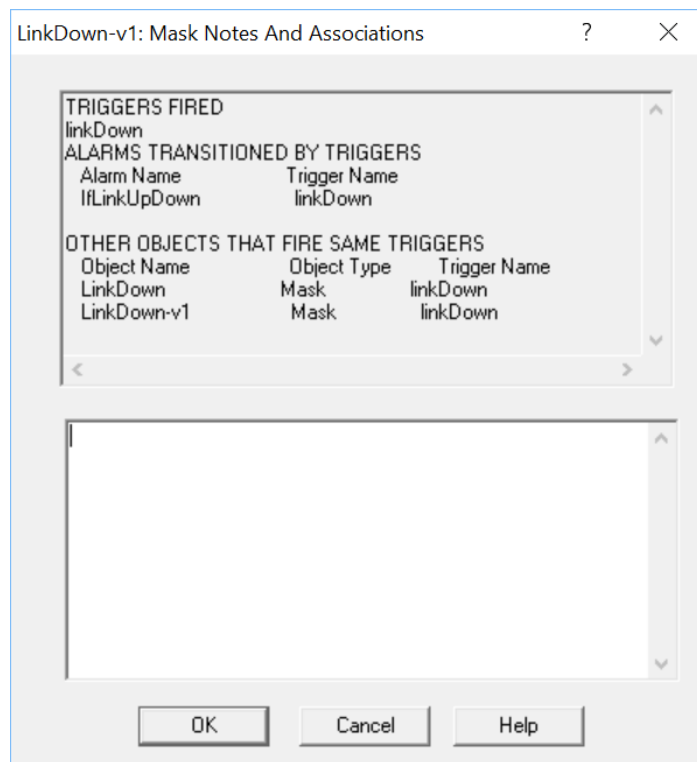
The Mask List window is displayed.



2. Select the mask you're interested in.

The **Notes** button is enabled.

3. Select the **Notes** button.  
The Mask Notes dialog displays.



For each mask, the note should include the following information:

- A list of the triggers that this mask can fire
- A list of the alarms in which this mask can cause a transition
- A list of other objects that fire any one of the triggers fired by this mask
- A brief description of the trap detected by this mask

**Note:** If you need additional information about a mask, select it in the Mask List window and click **Open** to see its definition

### A Trigger Generated by an Alarm

If you're monitoring alarm instances and see that a transition in one alarm took place when another alarm fired a trigger, you can determine what condition caused the Source alarm to fire this trigger by looking at the notes (documentation) for the Source alarm. Follow the procedure mapped out in the section [Getting Information about an Alarm](#) on page 63.

## Built-In Triggers

If you're monitoring alarm instances and an instance changes states because of a built-in trigger, you can consult the table below to determine why NerveCenter generated the built-in trigger. [Table 7](#) lists all the built-in triggers that NerveCenter can fire.

**Note:** NerveCenter uses all uppercase letters to designate built-in trigger names.

Table 7: Built-In Triggers

Trigger Name	Meaning
CANNOT_SEND	A local error occurred while NerveCenter was trying to send an SNMP message.
ERROR	An SNMP or ICMP request did not result in a valid response. After firing the ERROR trigger, NerveCenter fires a second trigger that indicates the specific nature of the error.
ICMP_ERROR	Indicates an ICMP error. The ICMP_ERROR trigger contains the ICMP/IP fields from the error message.
ICMP_TIMEOUT	NerveCenter sent an ICMP ping to a node and did not receive a response. This trigger generally indicates that the node in question is down.  NerveCenter uses the number of retries and retry interval specified on the SNMP tab in the Administrator. Refer to <a href="#">Specifying SNMP Poll Intervals for NerveCenter</a> in <i>Managing NerveCenter</i> for details.
ICMP_UNKNOWN_ERROR	NerveCenter sent an ICMP ping to a node and received an invalid response. This trigger is no longer used except for the purpose of backward compatibility with version 3.5. We recommend you use it sparingly in the current version.
INFORM_CONNECTION_DOWN	A NerveCenter Inform host connection with OVPA or paserver is down.
INFORM_CONNECTION_UP	A NerveCenter Inform host connection with OVPA or paserver was down but is now back up.
INFORMS_LOST	The number of NerveCenter Informs that were unacknowledged and lost, usually while the inform host connection with OVPA was down.
NET_UNREACHABLE	Indicates that the IP routing layer could not find a route to the network containing the polled node, usually because at least one router was down. This trigger indicates nothing about the status of the node.  This trigger can be issued only if you have a router between the workstation running NerveCenter and the polled node.

Trigger Name	Meaning
NODE_UNREACHABLE	Indicates that the IP routing layer could not find a route to the destination node. This trigger indicates nothing about the status of the node.  This trigger can be issued only if you have a router between the workstation running NerveCenter and the polled node.
PORT_UNREACHABLE	NerveCenter sent a message to a node, and there was no response from the port to which the message was sent.
RESPONSE	NerveCenter sent an SNMP message and received a valid response from the agent on the destination node.
SNMP_AUTHORIZATIONERR	An SNMPv3 authorization error caused because there is a mismatch between one or all of the rows of <b>vacmAccessTable</b> and the packet. Reasons include: context name mismatch ( <b>vacmAccessContextPrefix</b> ); security model is not used ( <b>vacmAccessSecurityModel</b> ); incorrect security level ( <b>vacmAccessSecurityLevel</b> ); unauthorized to read the MIB view for the SNMP context ( <b>vacmAccessReadViewName</b> ); unauthorized to write to the MIB view for the SNMP context ( <b>vacmAccessWriteViewName</b> ); unauthorized to notify the MIB view for the SNMP context ( <b>vacmAccessNotifyViewName</b> )
SNMP_BADVALUE	NerveCenter tried to set the value of an attribute in a MIB, but the value it supplied was inappropriate for the attribute. The value may have been of the wrong type, of the wrong length, or invalid for some other reason.
SNMP_DECRYPTION_ERROR	The SNMPv3 engine dropped packets because they could not be decrypted. The 32-bit counter, <b>usmStatsDecryptionErrors</b> , is greater than zero.
SNMP_ENDOFTABLE	NerveCenter fires <b>SNMP_ENDOFTABLE</b> when it finds no more rows while performing an SNMP walk of a MIB table. For example, you could walk <b>IfTable</b> to determine the number of DSO interfaces a node contains.
SNMP_GENERR	A GetRequest, GetNextRequest, or SetRequest failed for some unknown reason (general error).
SNMP_NOSUCHNAME	NerveCenter sent to an SNMP agent a GetRequest, a GetNextRequest, or a SetRequest, and the agent that was contacted was unable to perform the requested operation because: <ul style="list-style-type: none"> <li>■ The name of the attribute to be read did not match exactly the name of an attribute available for get operations in the relevant MIB view</li> <li>■ The name of the attribute to be read did not lexicographically precede the name of an attribute available for get operations in the relevant MIB view</li> <li>■ The attribute to be set was not available for set operations in the relevant MIB view</li> </ul>

Trigger Name	Meaning
SNMP_NOT_IN_TIME_WINDOW	The SNMPv3 engine dropped packets because the boots and timeticks sent in the PDU appeared outside of the authoritative SNMP agent's time window. The 32-bit counter, <b>usmStatsNotInTimeWindows</b> , is greater than zero.
SNMP_READONLY	The error readOnly is not defined in RFC 1157. However, some vendors' agents do use this error-status code. As the name implies, the error usually indicates that an agent has received a SetRequest (from NerveCenter, in this case) for an attribute whose access type is read only.
SNMP_TIMEOUT	NerveCenter sent an SNMP message to an agent and did not receive a response. This trigger indicates either that a node's SNMP agent is down or that the node itself is down.  NerveCenter uses the number of retries and retry interval specified on the SNMP tab in the Administrator. Refer to <a href="#">Specifying SNMP Poll Intervals for NerveCenter</a> in Managing NerveCenter for details.
SNMP_TOOBIG	An SNMP agent did not respond normally to a GetRequest, GetNextRequest, or SetRequest from NerveCenter because the size of the required GetResponse would have exceeded a local limitation.
SNMP_UNAVAILABLE_CONTEXT	The SNMPv3 engine dropped packets because the context contained in the message was unavailable. The 32-bit counter, <b>snmpUnavailableContexts</b> , is greater than zero.
SNMP_UNKNOWN_CONTEXT	The SNMPv3 engine dropped packets because the context contained in the message was unknown. The 32-bit counter, <b>snmpUnknownContexts</b> , is greater than zero.
SNMP_UNKNOWN_ENGINEID	The SNMPv3 engine dropped packets because they referenced an <b>snmpEngineID</b> that was not known to the SNMPv3 engine. The 32-bit counter, <b>usmStatsUnknownEngineIDs</b> , is greater than zero.
SNMP_UNKNOWN_USERNAME	The SNMPv3 engine dropped packets because they referenced a user that was not known to the SNMPv3 engine. The 32-bit counter, <b>usmStatsUnknownUserNames</b> , is greater than zero.
SNMP_UNSUPPORTED_SEC_LEVEL	The SNMPv3 engine dropped packets because the requested security level is unknown or unavailable. The 32-bit counter, <b>usmStatsUnsupportedSecLevels</b> , is greater than zero.
SNMP_WRONG_DIGEST	The SNMPv3 engine dropped packets because they didn't contain the expected digest value. The 32-bit counter, <b>usmStatsWrongDigests</b> , is greater than zero.
UNKNOWN_ERROR	Some other error occurred.

One additional trigger, `USER_RESET`, is not available from the list of built-in triggers in NerveCenter. NerveCenter fires `USER_RESET` to trigger another state for an existing alarm instance when you reset the alarm instance using the right-click pop-up menu in the Alarm Summary or Aggregate Alarm Summary windows.

## Viewing Alarm Instance History

Using the alarm-instance viewers provided by the NerveCenter Client, you can view all the current alarm instances for the servers to which you're connected. Sometimes, however, you also need historical information about an alarm instance. For example, let's say that a current alarm instance tells you that an interface on a router has been experiencing high traffic for the last ten minutes. You might also want to see whether this is a new problem or whether it has happened before. To get this information, you can ask to see the alarm instance's history. This history includes information about the alarm instance's twenty most recent transitions.

**Caution:** When an alarm instance returns to Ground state, it is deleted and no history for the instance is retained. To track a particular condition on a device or an interface across alarm instances, a behavior model must record data about alarm transitions in a log file. For information on reading logs, see the section [Reading Logged Data on page 76](#). For information on creating logs, see the manual [Alarm Actions](#) in *Designing and Managing Behavior Models*.

## Using the NerveCenter Client

This section explains how to view the history of an alarm instance using the NerveCenter Client. To view this information, you bring up the Client's Alarm History window.

## TO OPEN THE ALARM HISTORY WINDOW

- From the Alarm or Aggregate Alarm Summary window, double-click an alarm instance.  
The Alarm History window is displayed.

The screenshot shows the 'Alarm History' window for a specific alarm instance. The top pane displays a state diagram with nodes: 'Ground', 'up-wait', 'oneminu', 'one-min', 'ifEntry', 'again', 'twomin', and 'onemint'. The bottom pane shows a 'History List' table with columns: Time, Node, From ..., To Sta..., Severity, Trigger, Type, Source, and #.

Time	Node	From ...	To Sta...	Severity	Trigger	Type	Source	#
06/15/2017 18:...	Cisco...	Ground	up-w...	Normal	ifEntr...	poll	IfEntr...	1
06/15/2017 18:...	Cisco...	up-w...	one-...	Special	onem...	fire	delay-...	2
06/15/2017 18:...	Cisco...	one-...	again	Normal	ifEntr...	poll	IfEntr...	3
06/15/2017 18:...	Cisco...	again	onem...	Normal	twom...	fire	delay-...	4

The top pane in the Alarm History window displays the state diagram for the relevant alarm, and the list at the bottom of the window displays the transitions that have led to the alarm instance's current state. The data displayed for each transition is similar to that displayed for an alarm instance in the Alarm or Aggregate Alarm Summary window. The only new columns are From State, To State, and #. These columns hold the state of the alarm instance before the transition, the state of the instance after the transition, and the number of the transition (first, second, and so forth).

The Alarm History window displays read-only information that cannot be modified. From this window, you can view the following:

Table 8: Alarm Instance History

Historical Item	Information Available
Alarm configuration	The alarm's property and scope appear in their respective fields. The unique identifier for the current alarm instance. Each alarm instance is assigned a unique instance ID by its associated NerveCenter Server.
Detail for each alarm instance	The list near the bottom of the window displays a line for each transition. Each line includes the following: <ul style="list-style-type: none"> <li>■ Name of the trigger that caused the transition.</li> <li>■ Origin and destination states.</li> <li>■ Poll, mask or alarm that triggered the transition.</li> <li>■ Node whose agent caused the transition.</li> <li>■ Severity of the final state.</li> <li>■ Sequence of transitions for the instance. The 20 most recent transitions are listed in order of occurrence.</li> </ul>
Transition activity	By selecting each entry in the list in order and watching the transitions change to red, you can follow along as the transitions lead to the current alarm state.
Transition configuration	Double-click the transition whose configuration you want to view. You can view the origin and destination states, the trigger that caused the transition, and any associated actions.
State severity	Double-click the state whose severity level you want to view.
Associated NerveCenter objects	When you select an entry in the list, the <b>Node</b> , <b>Poll</b> , <b>Mask</b> , or <b>Alarm</b> buttons are enabled, for each instance associated with one or more of these objects. Click a button to view the related node, poll, mask, or alarm definition.
Notes	Select the <b>Notes</b> button to view notes about the alarm.
Counters	Select the <b>Counters</b> button to see the names of any counters used in the alarm, along with the set value of each.

The preceding figure shows the history of an instance of the alarm `IfErrorStatus`, which monitors the percentage of error packets on an interface. As you can see, the instance first transitioned from low to high, and since then has bounced back and forth between the high and medium states. The times associated with the transitions (you can't see all of them) indicate that the instance has been in the high state most of the time.



There are several actions that you can take from the Alarm History window:

- If you click the first transition in the alarm-history list, the corresponding transition in the state diagram is highlighted, as shown in Figure 14.

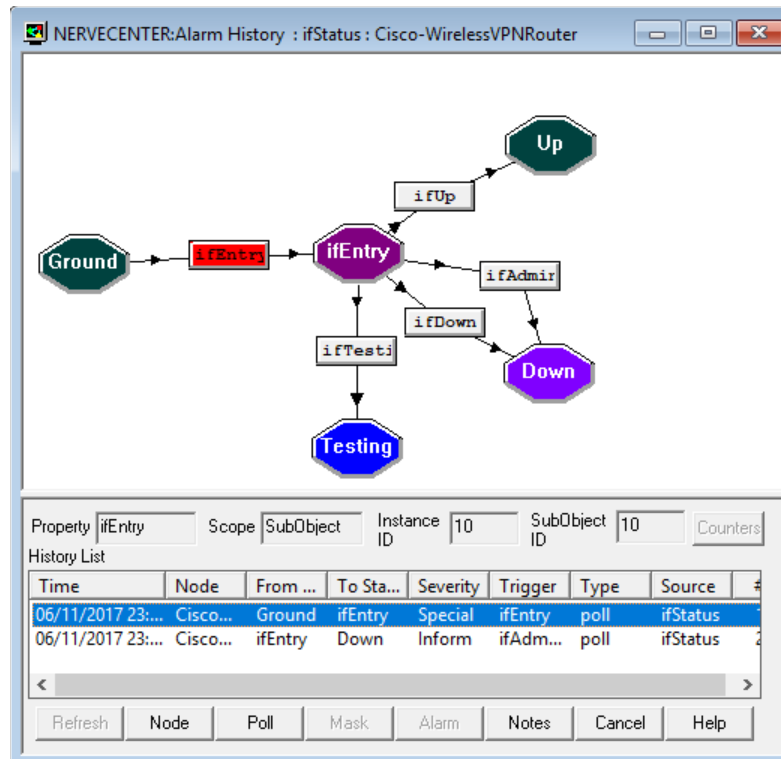


Figure 14: Alarm History Window

By selecting the transitions in order—from first to last—you can watch the history of the alarm instance in the state-diagram pane.

- When you select a transition from the transition list, the **Node** button and typically either the **Poll**, **Mask**, or **Alarm** button are enabled. Selecting an enabled button opens a definition window that presents a definition of one of the following objects:
  - The node that the alarm instance is monitoring.
  - The poll that generated the trigger that caused the transition.
  - The trap mask that generated the trigger that caused the transition.
  - The alarm that generated the trigger that caused the transition.
- If the **Refresh** button becomes enabled while you're viewing an alarm instance's history, the alarm instance has undergone a transition while you've had the Alarm History window open. Select the **Refresh** button, and NerveCenter will update the information about this latest transition to the transition list.

## Reading Logged Data

As was mentioned in the section [Viewing Alarm Instance History on page 72](#), NerveCenter does not maintain a lot of historical information about network conditions. It remembers the last twenty transitions for each current alarm instance; however, when that alarm instance is deleted (when it returns to Ground), even that history is lost.

To preserve historical information about a network problem, a behavior model must log data about alarm transitions to a file or to the system log (UNIX syslog). To take advantage of this logged data, all you need to know is where the data is being logged and how to interpret it.

You can also manage the size of logs as well as the length of time they are stored by setting parameters in NerveCenter Administrator. For more information, see [Specifying Settings for Log Management in Managing NerveCenter](#).

For more information about where NerveCenter writes log data and how you should interpret this data, see the following subsections:

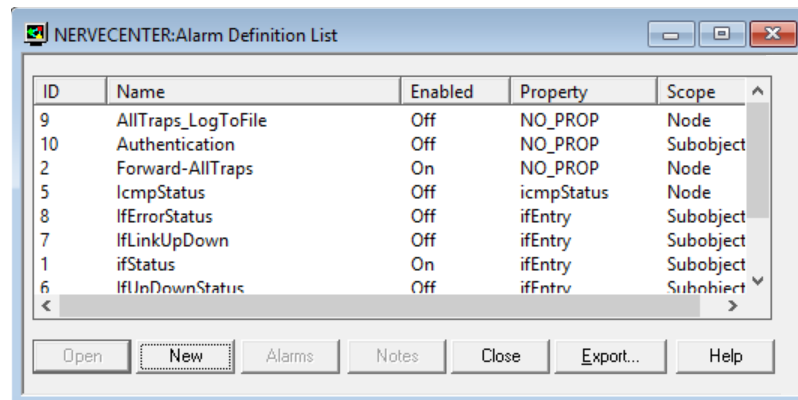
- [Determining Where Data is Being Logged below](#)
- [How to Interpret Logged Data on page 78](#)

## Determining Where Data is Being Logged

To determine whether an alarm logs data about any of its transitions and, if so, where it logs that data, you should look at the alarm's notes using the NerveCenter Client.

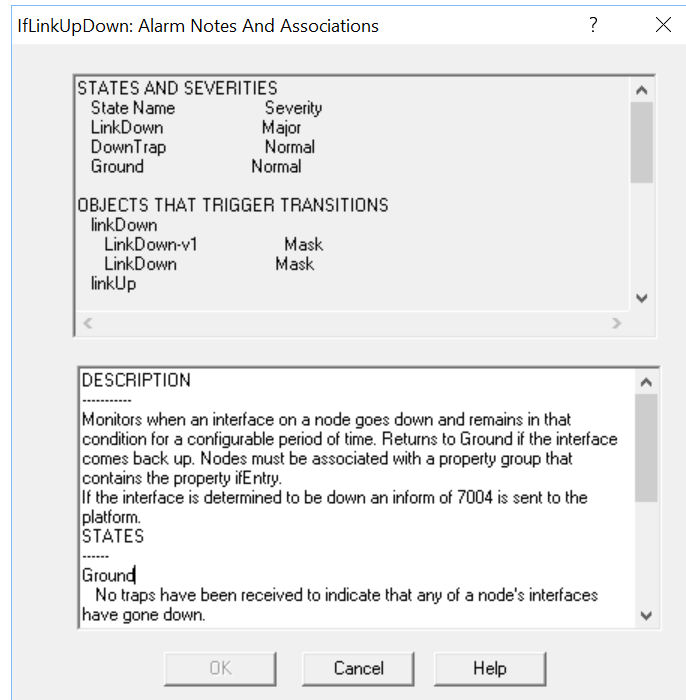
### TO VIEW AN ALARM'S NOTES

1. From the NerveCenter Client's **Admin** menu, choose **Alarm Definition List**.  
The Alarm Definition List window is displayed.



2. Select the alarm you're interested in from the list of alarms.  
The **Notes** button is enabled.

3. Select the **Notes** button.  
The Alarm Notes and Associations dialog displays.



This dialog contains documentation for the alarm you selected and describes, among other things, any logging actions. For a Log to File action, the notes specify the log file to which data is written. An EventLog action causes NerveCenter to log data to one of the following locations:

- /var/adm/messages (Solaris)
- /var/log/messages (Linux)

A Log to Database action causes NerveCenter to log to the NerveCenter database.

## How to Interpret Logged Data

Once you've determined where the data you're interested in is being logged, you need to know how to interpret that data. [Figure 15](#) shows a sample entry from a log file.

```
Time=06/17/2017 16:00:48 Sat; LogId=1; DestStateSev=Normal;
NodePropertyGroup=Mib-II;
NodeName=cisco-2600-switch-14; AlarmName=AllTraps_LogToFile;
OrigState=Ground;
TriggerName=allTraps; DestState=Logging; TrapPduTime=1194180;
TrapPduEnterprise=
1.3.6.1.6.3.1.1.5.3; TrapOid=1.3.6.1.6.3.1.1.5.3
TriggerInstance=21942; TriggerBaseObject=
ifEntry; Attribute ifIndex.22=22; Attribute ifAdminStatus.22=1;
Attribute ifOperStatus.22=2
```

Figure 15: Log File Entry

[Table 9](#) explains what information the fields in this report contain.

Table 9: Fields in a Log Entry

Field	Contains
Time	Date and time the record was logged. The format of the time is <i>mm/dd/yyyy hh:mm:ss day</i> (for example, 8/31/2017 14:32:22 Thur).
LogID	Identification number of the log entry. NerveCenter assigns a sequential number to each log entry.
DestStateSev	Severity of the transition's destination state.
NodeProperty	Property group of the node that caused the alarm to change states.
NodeName	Name of the node that caused the alarm to change states.
AlarmName	Name of the alarm.
OrigState	Name of the state from which the alarm moves when the logged transition occurs.
TriggerName	Name of the trigger that causes the alarm to move from the Ostate to the Nstate.
DestState	State of the alarm after the logged transition occurs.
TrapPduTime	The contents of a trap's timestamp field. Used only when the transition is caused by a trap-mask trigger.
TrapPduGeneric	The contents of a trap's generic-trap field. Used only when the transition is caused by a trap-mask trigger.
TrapPduEnterprise	The contents of a trap's enterprise field. Used only when the transition is caused by a trap-mask trigger.
TrapPduSpecific	The contents of a trap's specific-trap field. Used only when the transition is caused by a trap-mask trigger.
TriggerInstance	The specific base object instance for which the transition occurred.

Field	Contains
TriggerBaseObject	The base object associated with the transition.
Attribute ...	The variable bindings of the trigger that caused the transition. Each variable binding is printed in the format <i>Attribute attribute.instance=value</i> .

**Note:** If a log file was created in non-verbose mode, its entries will not contain the labels shown in the figure above, but only a series of semi-colon-separated values.

When you're processing a log file created by a particular alarm, keep in mind the following rules:

- If the alarm has Enterprise scope, all the entries in the log constitute a single data set.
- If the alarm has Node scope, all entries that refer to the same node make up a data set.
- If the alarm has Subobject scope, the entries that share a node and subobject constitute a data set.
- If the alarm has Instance scope, the entries that share an instance of an alarm, regardless of the MIB objects listed, constitute a data set.
- A log that is the result of a poll transition contains only data pertinent to a poll. For example, TriggerBaseObject will contain a value, but TrapPduTime will not.
- A log that is the result of a trap mask transition contains only data pertinent to a trap. For example, TrapPduTime will contain a value, but TriggerBaseObject will not.



Some alarms are designed to return to the Ground state when the condition they detect goes away. For example, the predefined alarm `ifLoad` tracks the level of traffic on an interface. As traffic increases, instances of this alarm may move from the Ground state to the medium state to the high state. Then, as traffic subsides, they may transition from the high state to the medium state to the Ground state. When these instances return to Ground state, they are automatically deleted from the Alarm Summary window.

Other alarms, however, are designed to remain in a terminal state until they are manually reset. For example, an instance of the predefined alarm `Authentication` transitions to the Intrusion state if a node receives four or more authentication-failure traps in a ten-minute period. The instance then remains in this state until it is manually reset.

The NerveCenter Client provides you with several ways to reset alarm instances. Most commonly, you'll select an alarm instance in the Alarm Summary or Aggregate Alarm Summary window and reset that instance to Ground. However, you can also select an instance in either of these windows and reset the state of the instance to any state allowed by the alarm. In addition, the NerveCenter Client enables you to reset either all alarm instances associated with a node or all alarm instances derived from the same alarm definition in a single operation.

If you reset an alarm to ground, any pending triggers fired by that alarm are cleared if the **Clear Triggers for Reset To Ground or Off** checkbox is checked in the Client's alarms definition window for the alarm.

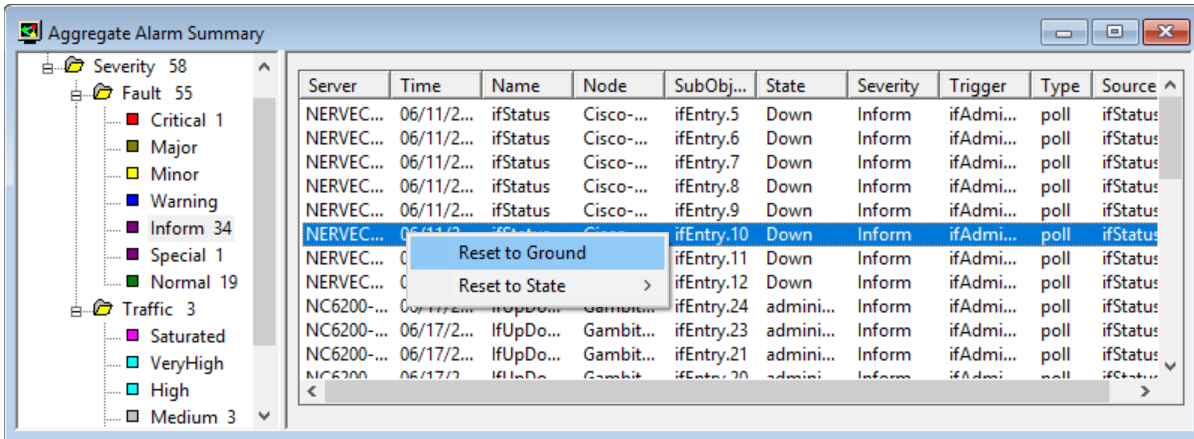
## Resetting an Alarm Instance to Ground

To reset one or more alarm instances to Ground from the Alarm or Aggregate Alarm Summary window, follow the directions below.

### TO RESET AN ALARM INSTANCE

1. Select an alarm instance in the Alarm Summary or Aggregate Alarm Summary window.  
You can also select a number of alarm instances and reset all of them to Ground at once.

2. With your cursor positioned over the selected alarm instance, right-click to bring up the reset pop-up menu.



3. Select **Reset to Ground** from the pop-up menu.

The alarm instance reset to Ground and removed from the Alarm Summary list; however, if the network condition that caused that instance to be created in the first place still exists, a new alarm instance will be created to track that condition.

**Note:** In the Alarm Summary windows, if you select the instances for a single severity and change those instances to a state with a different, non-ground severity, they disappear from the current instance list. You can view them in the alarm list for their new severity.

If the **Clear Triggers for Reset To Ground or Off** checkbox is checked in the alarm's definition window, any pending triggers fired by that alarm are cleared when you reset the alarm to ground.

## Resetting an Alarm Instance to a Non-Ground State

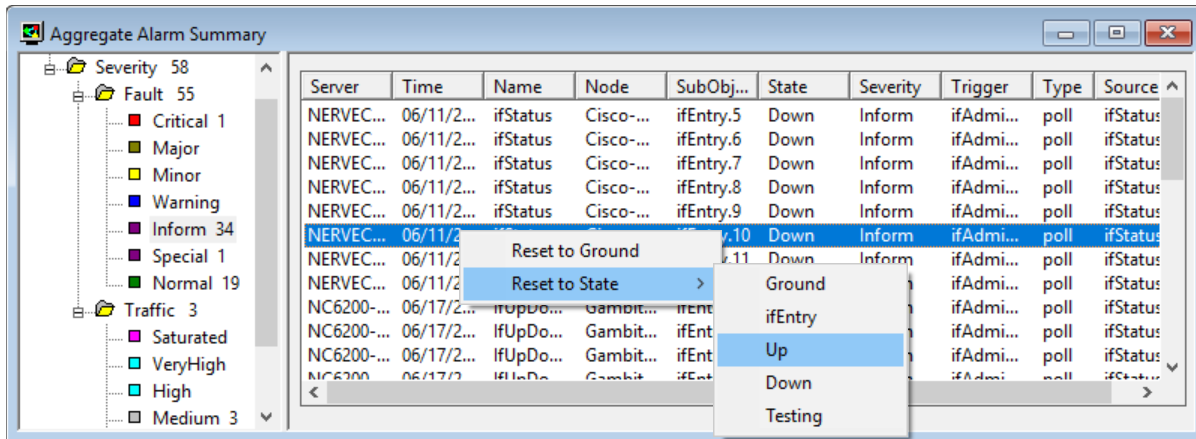
Sometimes you want to set an alarm instance to a state other than Ground. To reset one or more alarm instances to a state other than Ground from the Alarm or Aggregate Alarm Summary window, follow the directions below.

### TO RESET AN ALARM INSTANCE

1. Select an alarm instance in the Alarm Summary or Aggregate Alarm Summary window.  
You can also select a number of alarm instances and reset all of them to a particular state at once. All of the instances must be derived from the same alarm definition.
2. With your cursor positioned over the selected alarm instance, right-click to bring up the reset pop-up menu.



3. Move your cursor over the **Reset to State** entry to bring up the state pop-up menu.



4. Select the state to which you want to reset the instance from the pull-right menu.  
The entry for the alarm instance will show that it is now in the state you selected. The trigger that caused the transition to that state is the built-in trigger `USER_RESET`.

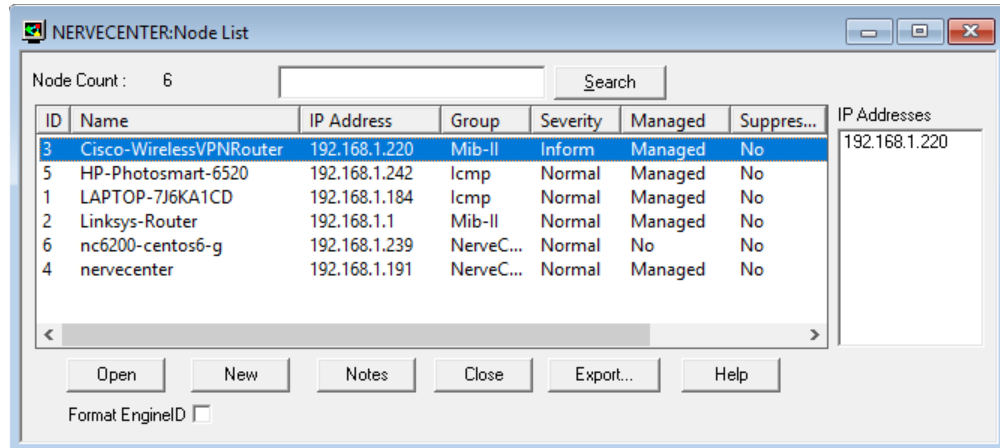
## Resetting Node Alarm Instances

Once you've identified a node that is experiencing a problem and have addressed the problem, you may want to reset all the alarm instances monitoring that node to Ground. You can reset all of these alarm instances using the procedure below:

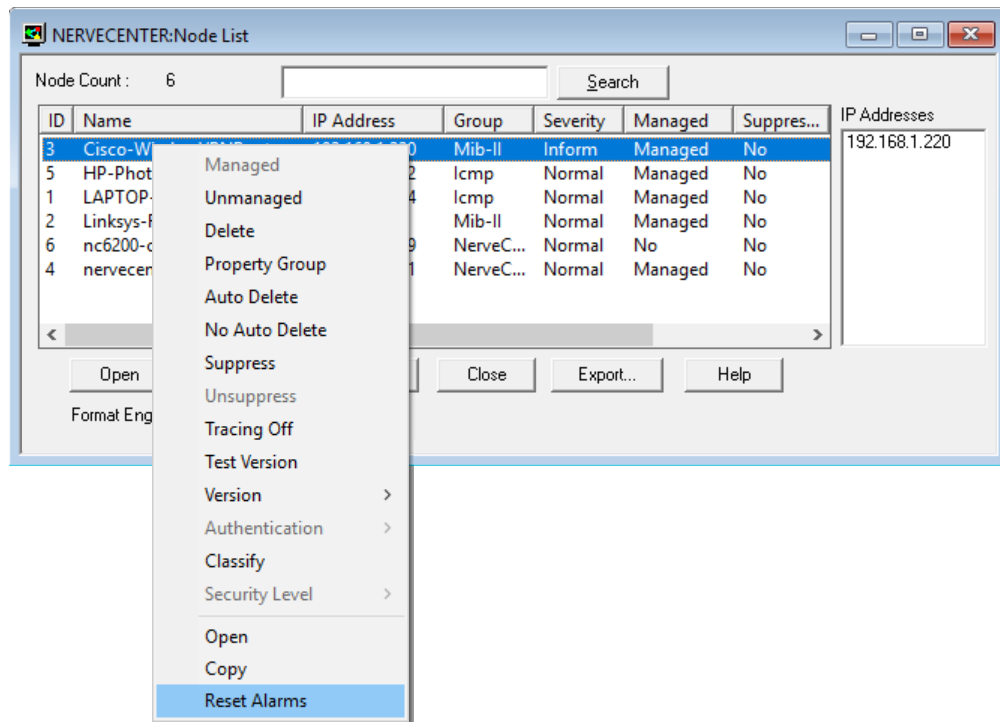
## TO RESET THE ALARM INSTANCES

1. From the client's **Admin** menu, choose **Node List**.

The Node List window is displayed.



2. Select the node whose alarm instances you want to reset.
3. Right-click over the selected instance to bring up the node pop-up menu.



4. Select **Reset Alarms** from the pop-up menu.

**Note:** You can also reset all node alarm instances by opening the Node Definition window, clicking the **Alarms** tab, and clicking **Reset All**.

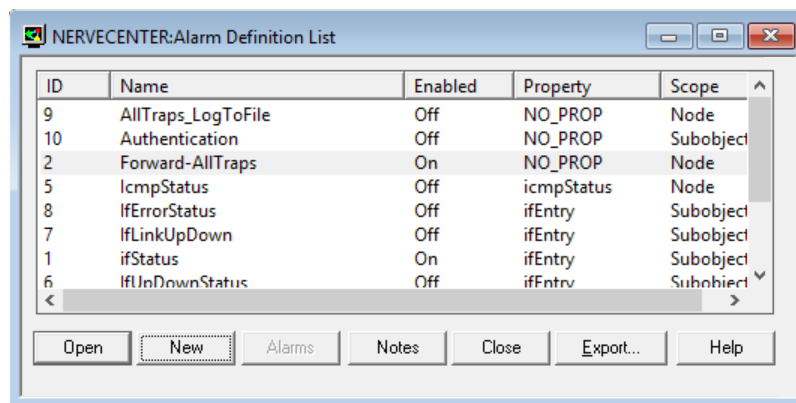
## Resetting All Instances of an Alarm

When you select an alarm from the Alarm Definition List and perform a reset operation on it, you reset to Ground all the current instances of that alarm. That is, if you are using the Authentication behavior model and instances of the Authentication alarm have been instantiated for three nodes, all three instances will be deleted when you reset the Authentication alarm.

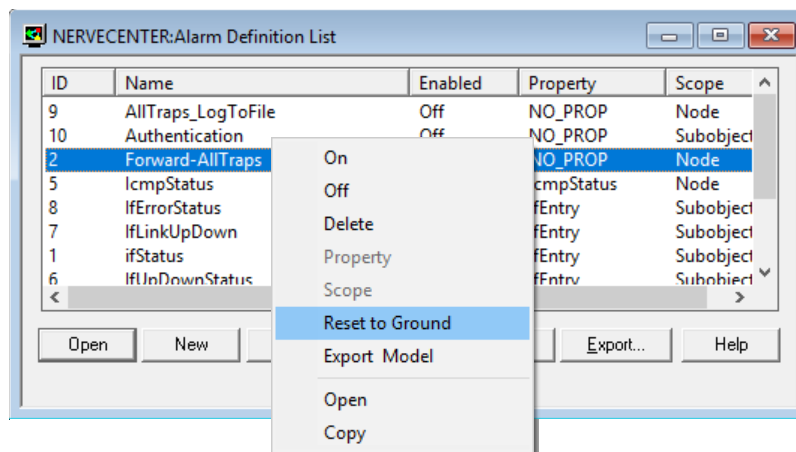
### TO RESET AN ALARM

1. From the client's **Admin** menu, choose **Alarm Definition List**.

The Alarm Definition List window is displayed.



2. Select the alarm whose instances you want to reset from the list.
3. With your cursor over the selected alarm, right-click to bring up the alarm pop-up menu and select **Reset to Ground**.





# Monitoring SNMP Status and Operations

SNMP version 3 is an update of SNMP that addresses security and administration. The following topics describe how NerveCenter provides support for SNMPv3. NerveCenter logs all SNMP operations to a file that you can use to track events and errors. In addition, the NerveCenter Client and Web Client provide error messages in their respective node lists for nodes with SNMP-related problems.

## 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" below
- "Signing a Log for SNMPv3 Errors Associated with a Remote Client or Administrator" on the facing page
- "Viewing the SNMPv3 Operations Log" on page 90

## 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.

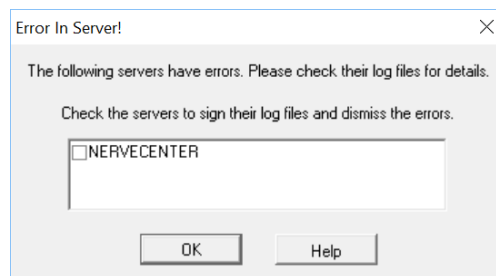


Figure 16: Operations Log Error in Server Dialog Box for Your Client

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

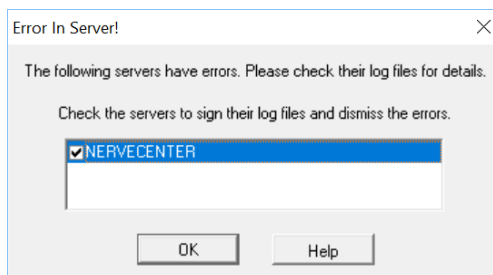
Whenever an error occurs while attempting an SNMPv3 operation, the NerveCenter Server logs a message to V3Messages.log. If you are logged on to a remote NerveCenter Client (one that did not initiate the error condition), the status bar displays a red icon.

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.



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 10: 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 <i>Error Status</i> 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 17](#) shows the Node List window in the Client.

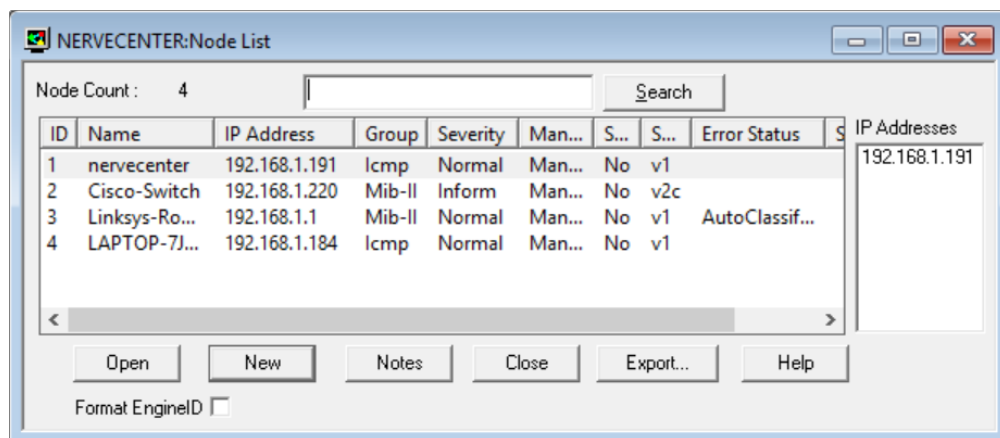


Figure 17: Node List Window

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.



The NerveCenter Client enables you to perform two types of node-related tasks. First, it enables you to view a list of alarms that can be instantiated for a node and a list of the alarms that are currently instantiated for a node. Second, the NerveCenter Client enables you to quickly query a node to determine its status.

## Viewing Related Alarms

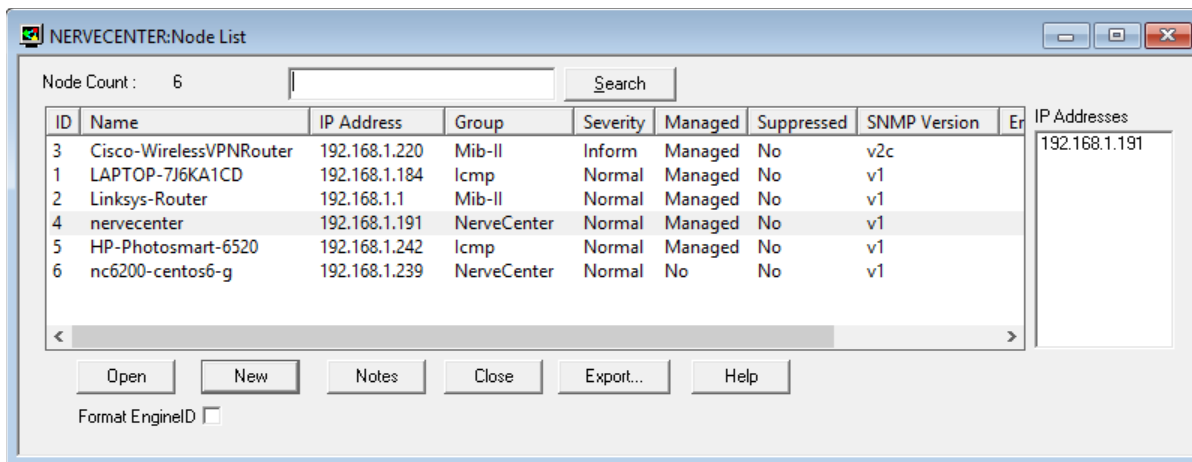
The NerveCenter Client can provide you with lists of:

- The alarms that can be instantiated for a node.
- The alarms that *have been* instantiated for a node. For each alarm instance, NerveCenter lists an alarm name, the enabled status of the alarm, the alarm's state, the subobject the alarm is monitoring, and the time at which the alarm instance was created. This information enables you to monitor the status of a node from the Node Definition window instead of the Alarm or Aggregate Alarm Summary window.

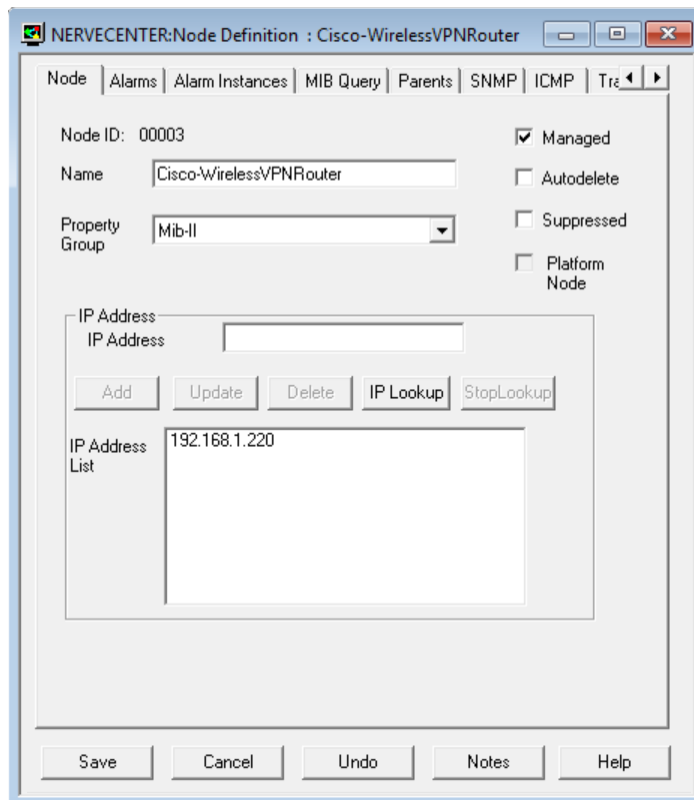
### TO VIEW THE ALARMS RELATED TO A NODE

1. From the Client's **Admin** menu, choose **Node List**.

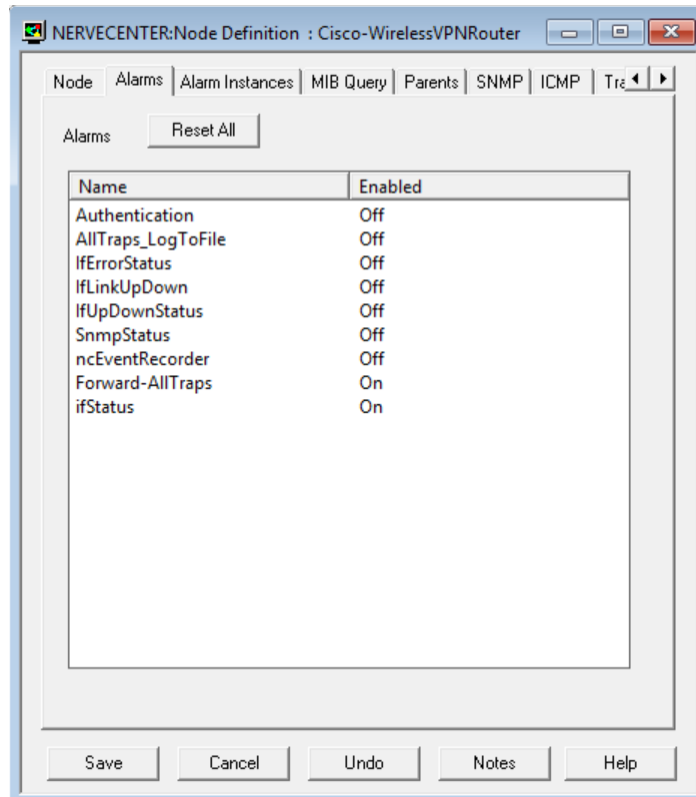
The Node List window is displayed.



2. Double-click the name or IP address of the node in which you're interested.  
The Node Definition window is displayed with the definition of the node you selected.



3. Select the **Alarms** tab.  
The Alarms tab is displayed.



In this example, the only alarm that has been instantiated is IfLoad. This alarm instance is monitoring interface 2 and is in the state medium, indicating that there is a moderate level of traffic on this interface.

The other alarms in the list are alarms that will be instantiated if:

- All the necessary NerveCenter objects are enabled
- The conditions that the alarms are designed to monitor actually occur

To see the documentation for an alarm, double-click the entry for an alarm to bring up the Alarm History window; then, click the **Notes** button.

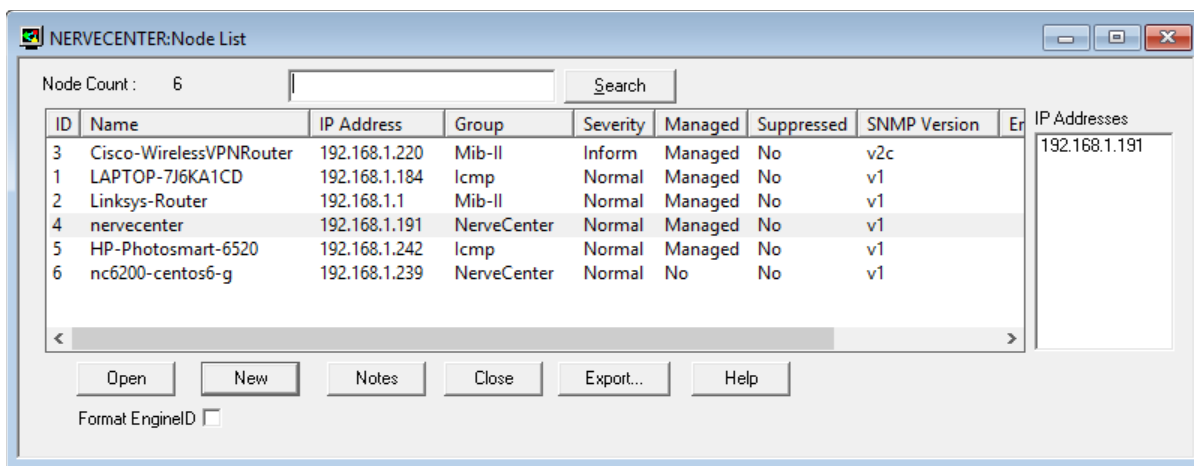
## Querying Nodes

In addition to listing a node's current alarm instances, the NerveCenter Client can query a node to determine whether the node is up and whether its SNMP agent is up—without using a behavior model. To determine whether a node is up or down, you send an ICMP ping to the node, and to determine the status of a node's SNMP agent, you send an SNMP GetRequest to the node asking for information about the system object.

### TO QUERY A NODE

1. From the Client's **Admin** menu, choose **Node List**.

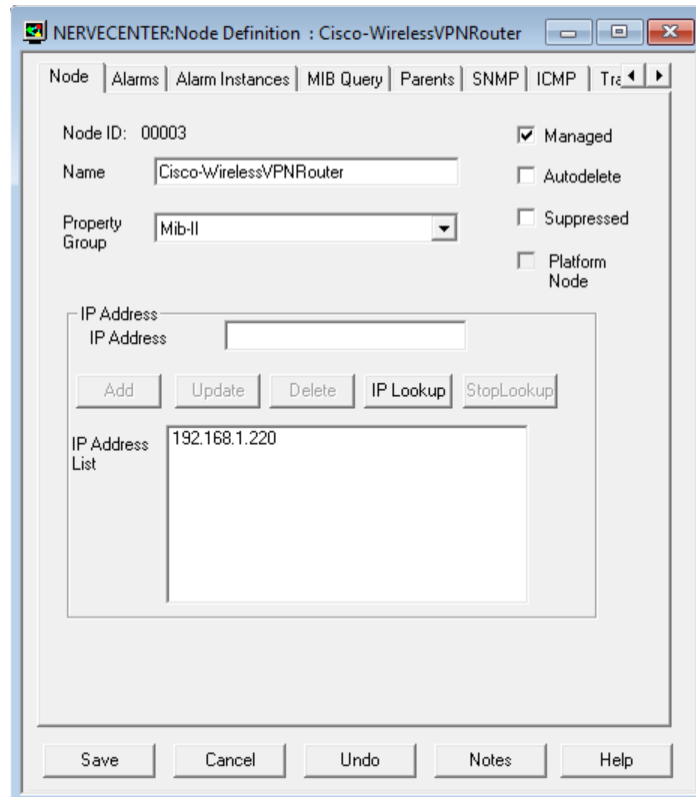
The Node List window is displayed.



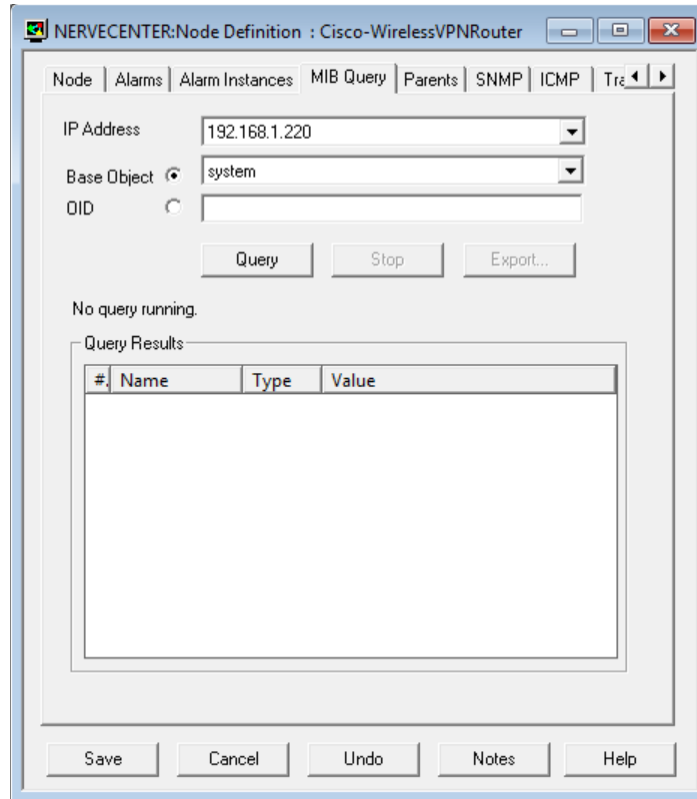
2. Double-click the name or IP address of the node in which you're interested.

The Node Definition window is displayed with the definition of the node you selected.

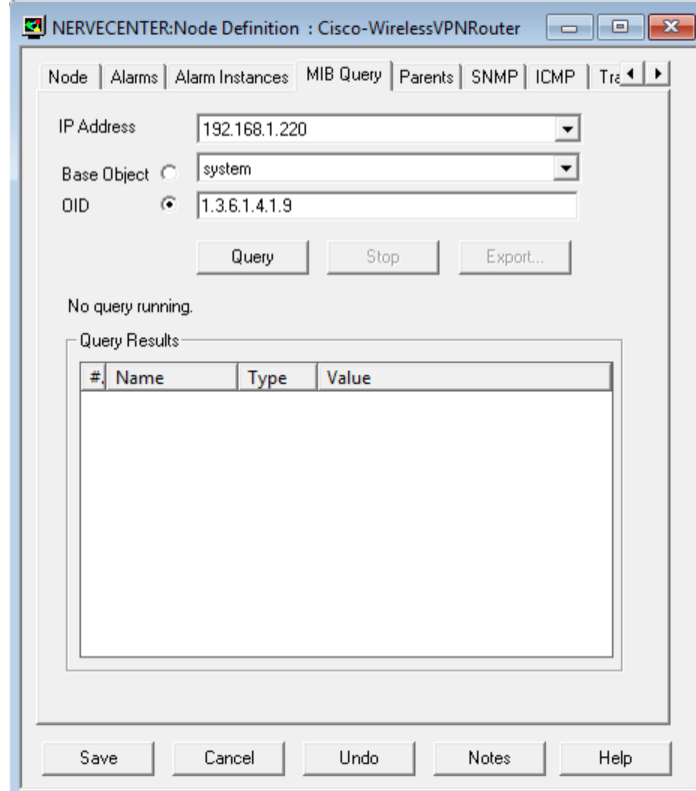
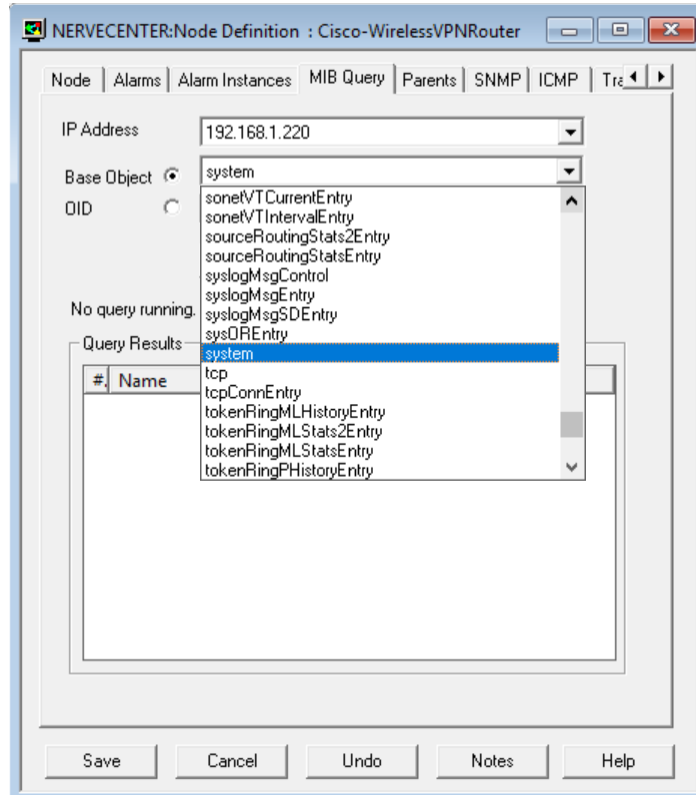




3. Select the **MIB Query** tab.  
The MIB Query tab is displayed.

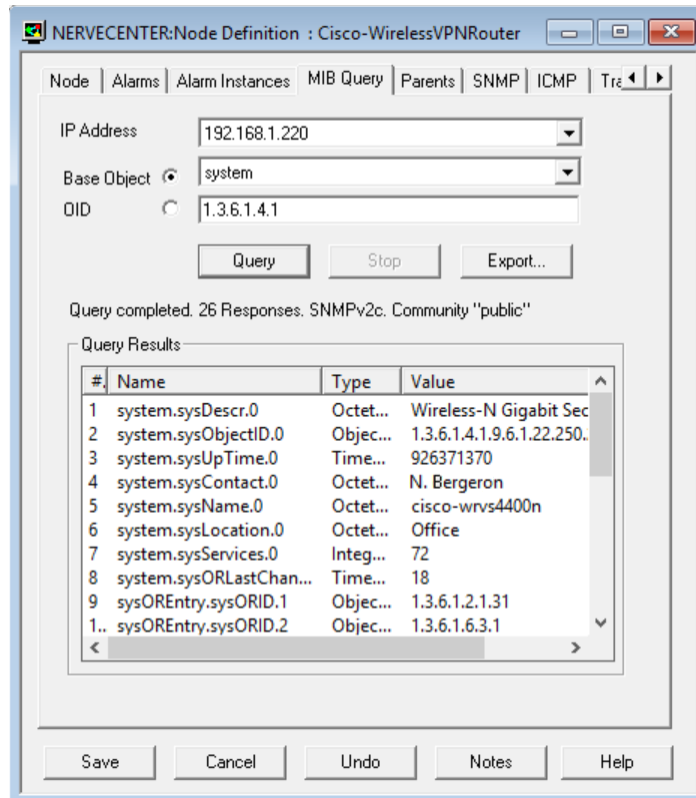


4. If the node you're querying is multihomed, select the IP address you want to use for the query from the **IP Address** drop-down list box.
5. Select the radio button to specify whether you want to poll a **Base Object** (ex: system, interfaces, ifEntry, ifXEntry, tcp) or poll using an object identifier (**OID**).
6. If you are polling a base object, select one from the list; if polling with an OID, enter the OID into the text field.



7. Select the **Query** button.

As returns are made to NerveCenter by the agent for the indicated Base Object or OID, they are shown in the table. If nothing is returned, the table area is use to report the timeout.



- Use the **Stop** button to halt a query that has been started.
- Use the **Export** button to upload the data from a query into the spreadsheet program on your Windows desktop.

#	Name	Type	Value
1	system.sy	OctetStrin	Wireless-N Gigabit Security Router with VPN
2	system.sy	Object ID	1.3.6.1.4.1.9.6.1.22.250.2
3	system.sy	TimeTicks	9.26E+08
4	system.sy	OctetStrin	N. Bergeron
5	system.sy	OctetStrin	cisco-wrvs4400n
6	system.sy	OctetStrin	Office
7	system.sy	Integer32	72
8	system.sy	TimeTicks	18
9	sysOREntr	Object ID	1.3.6.1.2.1.31
10	sysOREntr	Object ID	1.3.6.1.6.3.1
11	sysOREntr	Object ID	1.3.6.1.2.1.49
12	sysOREntr	Object ID	1.3.6.1.2.1.4
13	sysOREntr	Object ID	1.3.6.1.2.1.50
14	sysOREntr	Object ID	1.3.6.1.6.3.16.2.2.1
15	sysOREntr	OctetStrin	The MIB module to describe generic objects for network interface sub-layers
16	sysOREntr	OctetStrin	The MIB module for SNMPv2 entities
17	sysOREntr	OctetStrin	The MIB module for managing TCP implementations
18	sysOREntr	OctetStrin	The MIB module for managing IP and ICMP implementations
19	sysOREntr	OctetStrin	The MIB module for managing UDP implementations
20	sysOREntr	OctetStrin	View-based Access Control Model for SNMP.
21	sysOREntr	TimeTicks	5

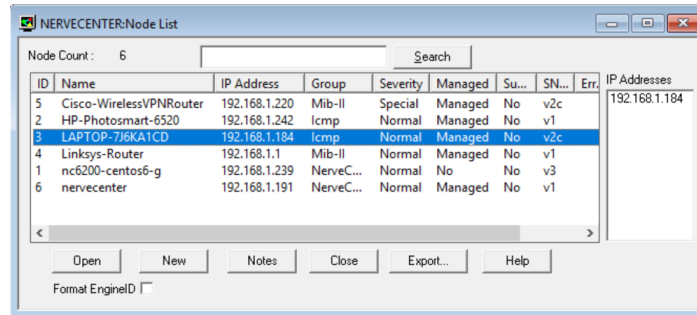
## Viewing Parent Node Status

NerveCenter monitors parent-child relationships and uses this information for *downstream alarm suppression*, suppressing alarms from any nodes that are downstream from a down router. The **Parents** tab of the Node Definition window displays the status that is obtained from NerveCenter's SetNodeStatus Perl subroutines.

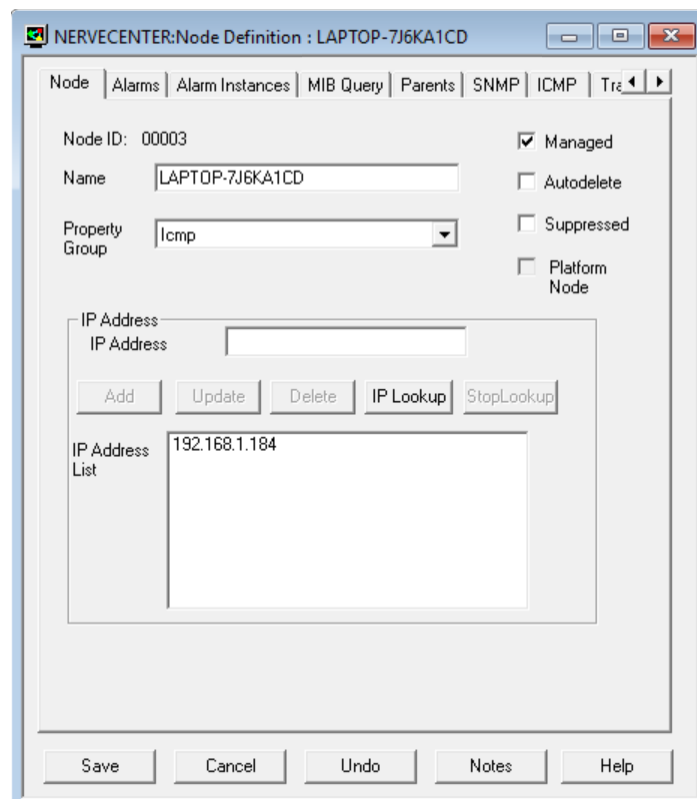
**Note:** The NerveCenter downstream alarm suppression alarms must be turned on before you can monitor a node's parents. Refer [Downstream Alarm Suppression](#) in *Designing and Managing Behavior Models*.

### TO VIEW THE STATUS OF PARENT NODES

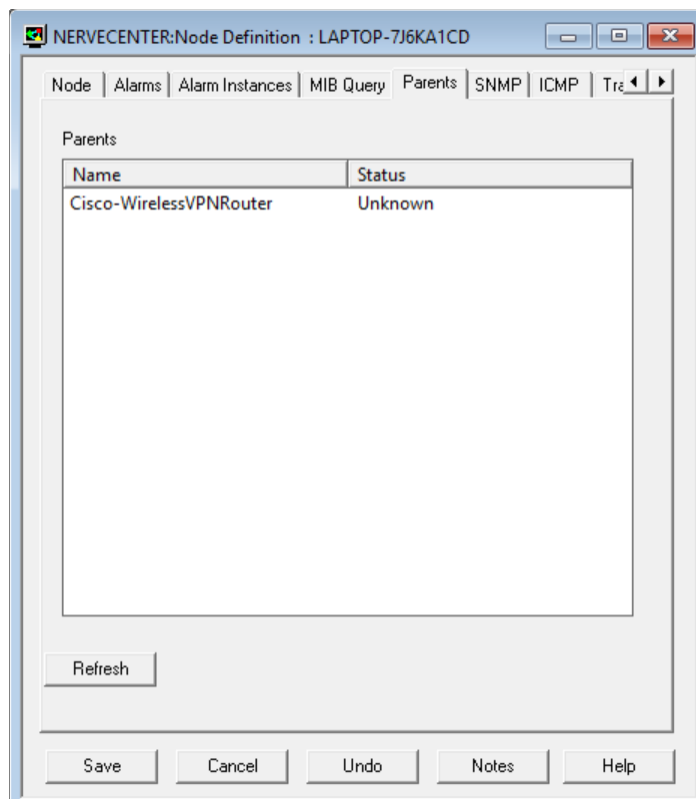
1. From the Client's **Admin** menu, choose **Node List**.  
The Node List window is displayed.



- Double-click the name or IP address of the node in which you're interested.  
The Node Definition window is displayed with the definition of the node you selected.



- Select the **Parents** tab.  
The Parents tab is displayed.



4. Select **Refresh** to update parent status information displayed in the window.

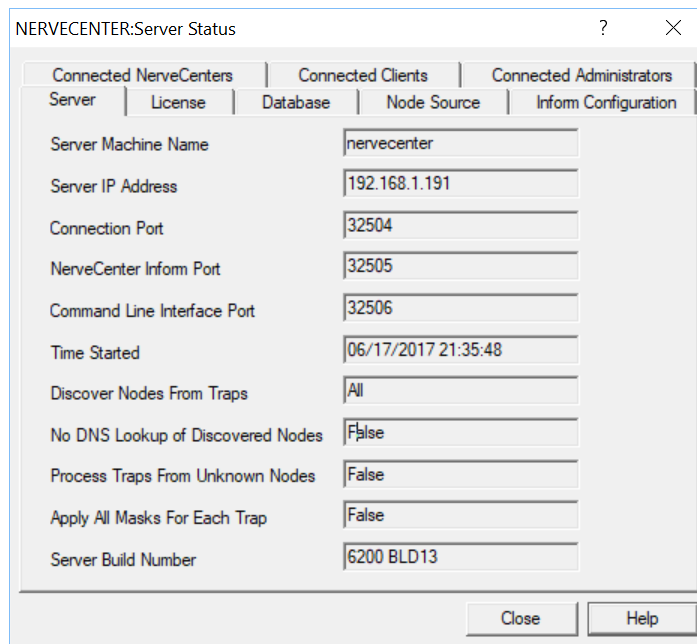




The NerveCenter Client provides a good deal of information about the active NerveCenter server. For example, you can check on the status of the machine the server is using as its node source, or the status of the machines running a network management platform that NerveCenter will notify when an Inform alarm action takes place. Or you can display a list of the NerveCenter clients and administrators that are connected to the active server.

TO DISPLAY INFORMATION ABOUT THE ACTIVE SERVER

- Choose **Server Status** from the **Server** menu.  
The Server Status dialog is displayed.



For an explanation of the information available on a particular page, see the appropriate subsection:

- [Server Tab below](#)
- [License Tab on the facing page](#)
- [Database Tab on page 110](#)
- [Node Source Tab on page 111](#)
- [Inform Configuration Tab on page 112](#)
- [Connected NerveCenters Tab on page 113](#)
- [Connected Clients and Connected Administrators Tabs on page 113](#)

## Server Tab

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 11](#) describes the information on the Server page:

Table 11: 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: <ul style="list-style-type: none"> <li>■ <b>None</b> - Never add an unknown node (NerveCenter discovery is disabled).</li> <li>■ <b>All</b> - Add all unknown nodes.</li> <li>■ <b>IP Filter</b> - Add the node if its IP address matches the IP filter criteria specified in NerveCenter Administrator.</li> </ul>

Label	Explanation
No DNS Lookup of Discovered Nodes	<p><b>False</b>, the default, indicates that NerveCenter will attempt a DNS lookup for any IP address discovered from a trap.</p> <p><b>True</b> 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.</p> <p><b>Note:</b> No DNS Lookup of Discovered Nodes is only valid if Discover Nodes from Traps is selected.</p>
Process Traps from Unknown Nodes	<p><b>True</b> indicates that NerveCenter processes traps from all nodes, regardless of filters imposed by NerveCenter or a network management platform. If <b>False</b>, NerveCenter discards traps coming from nodes outside your filters.</p> <p>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.</p>
Apply All Masks for Each Trap	<p><b>True</b> 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.</p>
Server Build Number	The NerveCenter software version running on the server.

## License Tab

The License page presents information about a server's NerveCenter license. [Table 12](#) provides describes the information on this tab:

Table 12: Fields in the Selected License Key Group Box

Label	Explanation
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.

## Database Tab

The Database page presents information about the NerveCenter database. [Table 13](#) provides brief explanations of the information shown on the Database tab:

Table 13: Fields on Database Tab

Label	Explanation
Database Source Name	The name of your open database connectivity (ODBC) data source.
Machine Name	The name of the host on which the NerveCenter database resides.
Database Name	The full pathname of the NerveCenter database.
Database Status	Indicates whether the server's connection to the database is currently up or down.
Statistics	A list of the number of alarms, polls, masks, nodes, property groups, and properties in the database.

## Node Source Tab

The Node Source page presents information about the network management platform from which NerveCenter is getting node information and describes NerveCenter's node filters. [Table 14](#) provides brief explanations of the information shown on the Node Source tab:

Table 14: Fields on Node Source Tab

Label	Explanation
Machine Name	The name of the host that runs the network management platform from which NerveCenter is to extract managed nodes. If this field is blank, NerveCenter will not retrieve nodes from any platform.
Port	The port number used by the NerveCenter Server to communicate with the platform host; the default is <b>6024</b> .
Wanted Capabilities	The capabilities of the nodes that NerveCenter is managing; these capabilities must be assigned in your platform software before NerveCenter recognize them. If this field is blank, NerveCenter monitors nodes regardless of capability.
System Object IDs	The system object identifiers specify the platform nodes that NerveCenter monitors. For example, .1.3.6.1.4.1.9.1.3.6.1.4.1.11 restricts the nodes retrieved for NerveCenter to those running SNMP agents from Cisco or Hewlett-Packard. A device with OID .1.3.6.1.4.1.9.8 would match the first OID in the list. If this field is left blank, NerveCenter monitors nodes regardless of Object ID.
Resync Parent Rate	The number of seconds between attempts to synchronize parent-child information. The default is <b>600</b> seconds.

## Inform Configuration Tab

The Inform Configuration page presents information about the network management platforms and NerveCenter servers to which your server is sending Inform messages. [Table 15](#) provides brief explanations of the information shown for each Inform destination:

Table 15: Fields on Inform Configuration Page

Label	Explanation
Status	Current state of the connection.
Machine Name	The host name of the machine receiving Informs from NerveCenter.
IP Address	The IP address of the machine receiving Informs from NerveCenter.
Port	The port number that the NerveCenter Server uses to communicate with the Inform action's recipient.
Filter	<p>The types of events that are being sent to the platform host.</p> <p>The filter column summarizes the restrictions on which alarm transitions can cause Inform actions and on what type of information is included in an Inform message:</p> <ul style="list-style-type: none"> <li>■ The first item in this column can be <code>EVENT_ONLY</code>, <code>SYMBOL_ONLY</code>, or <code>EVENT_AND_SYMBOL</code>. This item indicates what type of information will be forwarded to your network management platform: information to be displayed in the event browser, information about map symbol color changes, or both.</li> <li>■ The second item is a number representing a severity level. Only transitions whose destination state has a severity level equal to or greater than this value can cause an Inform message to be sent.</li> <li>■ Following the severity level, there may be a list of properties. The property group of the node associated with an alarm transition must contain one of these properties before the transition can cause an Inform message to be sent.</li> </ul>

## Connected NerveCenters Tab

This tab displays a list of the NerveCenter servers that are connected to the NerveCenter server whose status you're checking. These servers can send Inform messages to your server.

## Connected Clients and Connected Administrators Tabs

The Connected Clients and Connected Administrator pages list the NerveCenter Clients and NerveCenter Administrators that are currently connected to the NerveCenter server whose status you're checking. [Table 16](#) explains what information is presented for each connected Client and Administrator.

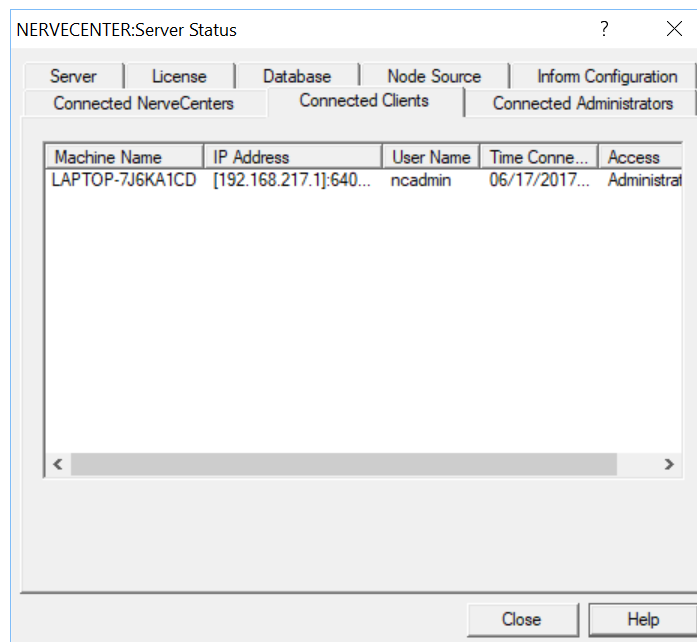


Table 16: Fields on Connected Clients and Connected Administrators Tabs

Label	Explanation
Machine Name	The name of the machine on which the connected Client or Administrator is running.
IP Address	The machine's IP address.
User Name	The name of the user who connected the Client or Administrator to the server.
Time Connected	The date and time at which the user connected to the server.

Label	Explanation
Access	The group to which the connected user belongs: User or Administrator. Only users with Administrator privileges can write to the NerveCenter database.



As a tool that comprehensively monitors and manages your network, NerveCenter uses a variety of data transfers to gather, correlate, disseminate, and store information about network events. This chapter outlines the general flow of data into, through, and out of NerveCenter in the course of its operation.

NerveCenter's primary sources of network information are SNMP traps and device responses to NerveCenter polls. If configured appropriately, LogMatrix NerveCenter responds to trap and poll data by forwarding it to your network management platform and to other NerveCenters. For example, forwarded event data might ultimately land in a network management platform's Event Categories window or trigger an alarm transition in a central NerveCenter. Although this sequence may happen quickly, the actual communication path from initial receipt of trap or poll data to the final event message has many stages.

As [Figure 18](#) shows, a trace of the communication path initiated by a managed device's SNMP trap or poll response might look like this:

1. Traps are relayed directly to the NerveCenter Server if the platform and the server are running on different machines. If they're running on the same machine, traps are detected by the operating system trap service or the management platform's trap service and then forwarded to the NerveCenter SNMP Trap process. The NerveCenter SNMP Trap process, in turn, forwards the trap to LogMatrix NerveCenter.
2. LogMatrix NerveCenter *trap masks* filter incoming traps to see if they are of interest. If a trap is of interest, an internal event, called a *trigger*, is generated and used by active *alarms*. Polls evaluate the poll data returned by managed devices and also use triggers to pass data to alarms.
3. LogMatrix NerveCenter alarms correlate the traps and polls with other related data. For example, an alarm might detect that this is the third trap of the same type from the same machine. The alarm then takes any automated actions that were associated with this trap detection. For example, it could issue a trouble ticket or change the device configuration.

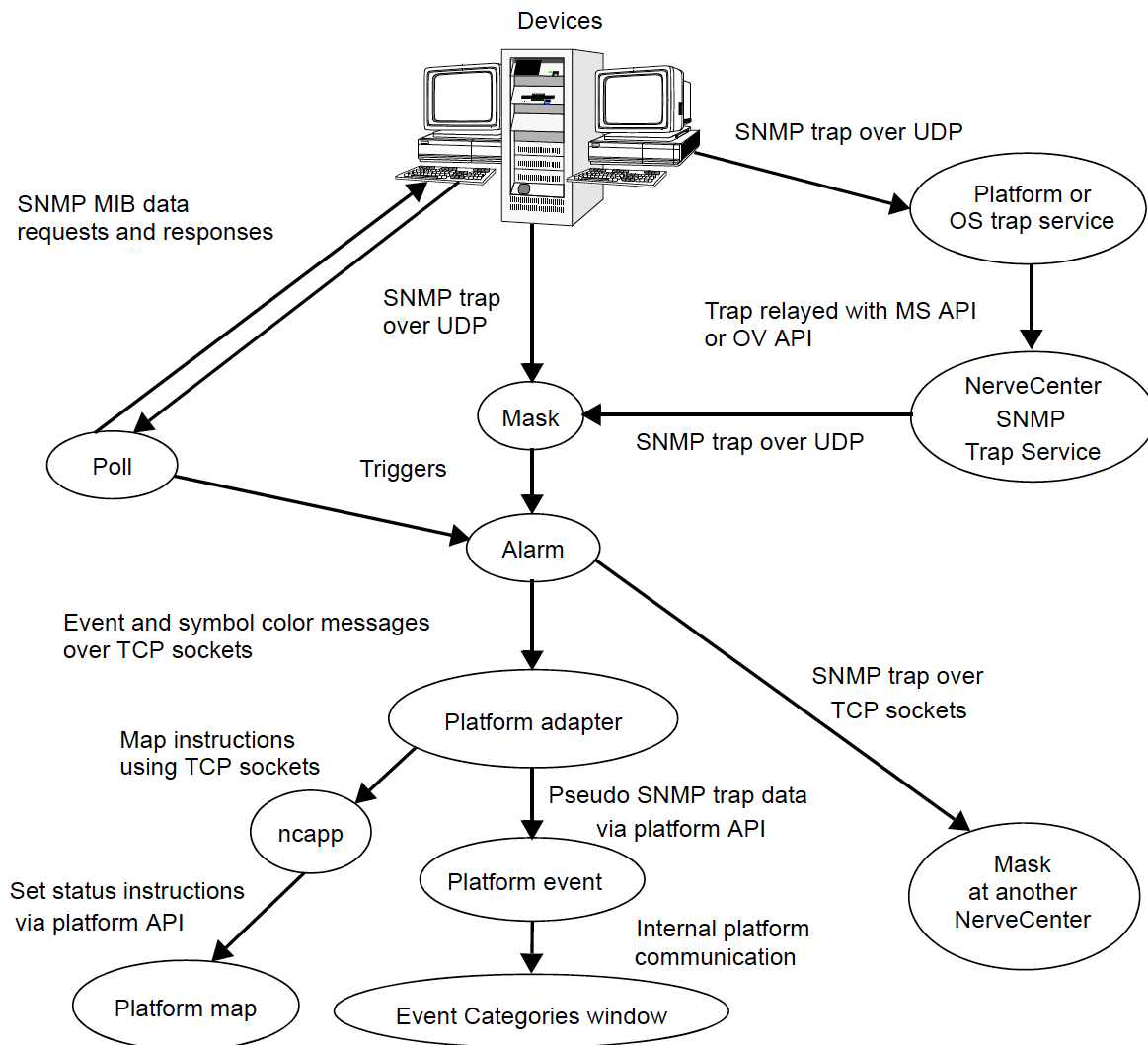


Figure 18: Data Flow

4. If an alarm transition contains the Inform action, the alarm sends a message to the LogMatrix NerveCenter platform adapterprocess, which always resides on the same host as the network management platform, and/or to any listed NerveCenters.
5. The platform adapter determines whether the message requires changing a symbol's color on the map, initiating an event message, or both. Messages to other NerveCenters forward the trap data.
6. If an event is to be posted, the platform adapter uses an API to submit a data structure that resembles an SNMP trap to the platform event facility, which decodes traps, associates text messages with events, and posts them in the Event Categories window.

NerveCenter is a client/server application. The NerveCenter server acts as the hub for the data transfers described in this appendix. As shown in the following illustration, event information moves from managed device to NerveCenter server to management platform. But data also flows between the server and other NerveCenter components in support of this flow.

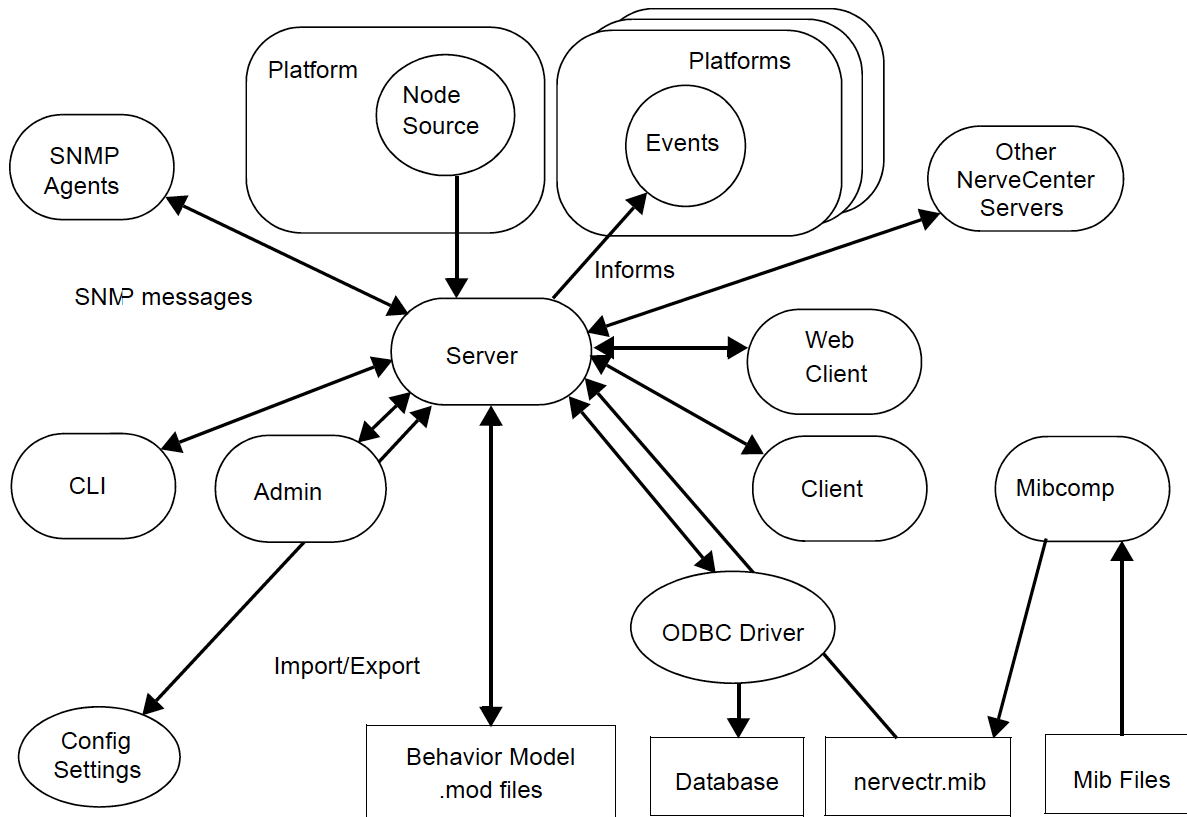


Figure 19: NerveCenter Components

The components shown in the preceding figure are defined in Table 17:

Table 17: NerveCenter Components

Component	Definition
Client	A user interface to the server. Provides facilities for the creation, modification, maintenance, and monitoring of behavior models.
Administrator	A user interface to the server. Provides facilities for NerveCenter configuration.
Command line interface (CLI)	Provides a subset of client commands for use from the command line, programs, and scripts.
Platform/node source	The network management platform that provides and monitors a list of nodes to be monitored by the server.
Platforms/events	The network management platforms that the server informs as an alarm action.
Other NerveCenters	Other NerveCenter servers that can accept Informs from the server, allowing correlation across multiple domains.

Component	Definition
SNMP agents	Agents running on managed nodes that generate traps and respond to NerveCenter polls.
mibTool	Utility to compile and merge MIBs into the NerveCenter master MIB.
Configuration Settings	Repository for NerveCenter configuration parameter values— <b>NerveCenter.xml</b> configuration file (UNIX) and the Registry (Windows).
Behavior model .mod files	ASCII files containing exported behavior models and their components.

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 18: 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 19: 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 122](#)
- ["Poll Manager Error Messages" on page 122](#)
- ["Protocol Manager Error Messages" on page 122](#)
- ["PA Resync Manager Error Messages" on page 123](#)
- ["Server Manager Error Messages" on page 125](#)
- ["Trap Manager Error Messages" on page 127](#)
- ["NerveCenter installation Error Messages" on page 128](#)

## Alarm Filter Manager Error Messages

Table 20: Alarm Filter Manager Error Messages

#	Error	Resolution
1	Lookup failed on line number <i>value</i> in File <i>value</i> .	
3001	Alarm Filter Manager Initialization successfully finished	

## Deserialize Manager Error Messages

Table 21: Deserialize Manager Error Messages

#	Error	Resolution
1	Lookup failed on line number <i>value</i> in File <i>value</i> .	
3001	Deserialize Thread Manager Initialization successfully finished	

## Flatfile Error Messages

Table 22: Flatfile Manager Error Messages

#	Error	Resolution
1	Lookup failed on line number <i>value</i> in File <i>value</i> .	
3001	Flat File Initialization successfully finished	

## Inform NerveCenter Error Messages

Table 23: Inform NerveCenter Manager Error Messages

#	Error	Resolution
1	Lookup failed on line number <i>value</i> in File <i>value</i> .	
3001	InformNC Manager Initialization successfully finished	

## LogToFile Manager Error Messages

Table 24: Log to File Manager Error Messages

#	Error	Resolution
1	Lookup failed on line number <i>value</i> in File <i>value</i> .	
3001	LogToFile Manager Initialization successfully finished	

## Poll Manager Error Messages

Table 25: 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 26: 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 27: 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 28: Server Manager Error Messages

#	Error	Resolution
2	Perl create failed.	N/A
3	Initialization of <i>value</i> 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: <i>value</i> .	N/A
10	Conflict in data. File: SERVER_CS.CPP, Line: <i>value</i> .	N/A
11	Internal Error. File: SERVER_CS.CPP, Line: <i>value</i> .	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 <i>value</i> in map.	N/A
24	Error while reading database. Poll/Mask: <i>value</i> uses a simple trigger that doesn't exist in database.	N/A
25	Please report error number <i>value</i> to technical support.	N/A
26	User validation failed: Unable to communicate with ncsecurity process : <i>value</i> .	~
1002	Initialization failed, cannot find ncp Perl.pl.	Check NCP Perl.pl location.
1003	Failed to open MIB: <i>value</i> .	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: <i>value</i> . The poll will be turned off.	Check the poll condition using the Client Application.
1100	<i>value</i> (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 <i>value</i> .	Use the NerveCenter Administrator to check the configuration settings.
1204	Cannot add value <i>value</i> .	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: <i>value</i>	Use the NerveCenter Administrator to check the configuration settings. Invalid key is likely.

#	Error	Resolution
1208	Cannot write configuration settings Value: <i>value</i>	Use the NerveCenter Administrator to check the configuration settings. Invalid key is likely.
1300	<i>value</i> (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 <i>value</i> failed because the node doesn't exist.	N/A
3002	Failed to find socket in server's map. Line: <i>value</i> .	
3003	Exiting due to a SIGTERM signal.	
3004	Primary thread initialization successful.	

## Trap Manager Error Messages

Table 29: 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 <i>value</i> .	
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 30: NerveCenter Installation Error Messages (UNIX)

Error	Resolution
Space under <i>dirname</i> is INSUFFICIENT to install LogMatrix NerveCenter	Free up space in the file system by removing files, or choose another place for installation.
The directory <i>dirname</i> 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 <i>dirname</i> 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.
<i>ProcessName</i> is running on the system. Please exit from (or kill) <i>processName</i> process.	The installation script found that the <i>nervectr</i> or <i>ovw</i> 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
<i>hostname</i> 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 <i>installation_path</i> /userfiles	Make sure the directory exists and has appropriate permissions.
Can't open <i>hostname.conf</i>	The script couldn't create the file or couldn't open an existing configuration file. Check <i>installation_path</i> /userfiles to make sure that root has permission to write in this directory, that <i>hostname.conf</i> has read permission set, if it exists, and that <i>localhost.conf</i> exists and has read permission set.
Can't create <i>hostname.ncdb</i> Can't create <i>hostname.node</i>	The script was attempting to create the indicated file by copying data from another file. Check <i>installation_path</i> /userfiles to make sure that root has permission to write in this directory, and that <i>localhost.ext</i> 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 <i>installation_path</i> /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 <i>installation_path</i> /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 <i>installation_path</i> /bin/rc.openservice to /etc/rc2.d/K94ncservice.  There's no need to rerun the installation script after this correction.

Error	Resolution
<p>An error occurred in trying to contact the Server "<i>hostname</i>". As a result, the information that you have specified cannot be used to complete this NIS update.</p> <p>Unable to modify <i>filename</i>. It doesn't exist!</p> <p>Unable to modify <i>filename</i>. File size is 0!</p>	<p>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 <code>/etc/services</code> yourself. Make sure that the following lines are included in the file:</p> <pre>SNMP 161/udp SNMP-trap 162/udp</pre> <p>If you're running NIS, be sure to make these changes on the NIS server, change to the NIS directory, and run <code>make services</code>.</p>





## - A -

- Action Router alarm action 13
- Action Router tool 13
- Administrator, NerveCenter 17
- alarm-instance filters 34
- alarm actions 11, 16
  - Action Router 13
  - FireTrigger 12
- Alarm Filter error messages 121
- alarm severity
  - filtering 42
- Alarm Summary window
  - filtering 38, 40
  - resetting 81, 85
  - viewing history for an alarm 72
- alarms 15
  - filtering by IP range 35
  - filtering rules 51

## - B -

- behavior models 5, 15
  - predefined 16
- built-in triggers 69
  - CANNOT\_SEND 69
  - ERROR 69
  - ICMP\_ERROR 69
  - ICMP\_TIMEOUT 69
  - ICMP\_UNKNOWN\_ERROR 69
  - list of 69
  - NET\_UNREACHABLE 69
  - NODE\_UNREACHABLE 70
  - PORT\_UNREACHABLE 70

- RESPONSE 70
- SNMP\_
  - AUTHORIZATIONER 70
  - BADVALUE 70
  - DECRYPTION\_ERROR 70
  - ENDOFTABLE 70
  - GENERR 70
  - NOSUCHNAME 70
  - NOT\_IN\_TIME\_WINDOW 71
  - READONLY 71
  - TIMEOUT 71
  - TOOBIG 71
  - UNAVAILABLE\_CONTEXT 71
  - UNKNOWN\_CONTEXT 71
  - UNKNOWN\_ENGINEID 71
  - UNKNOWN\_USERNAME 71
  - UNSUPPORTED\_SEC\_LEVEL 71
  - WRONG\_DIGEST 71
- UNKNOWN\_ERROR 71

## - C -

- CANNOT\_SEND built-in trigger 69
- CLI 20
- client
  - changing server port 33
- Client, NerveCenter 19
- command line interface 20

- conditions
  - finding sequences 9
  - finding set of network 8
  - network, detecting 6
  - persistent network 7
  - responding to network 11
- corrective actions 12
- correlating conditions 7

## - D -

- database, NerveCenter 14
- defining
  - node sets 6
- Deserialize Manager error messages 121
- detecting condition
  - persistence 7
- detecting conditions 6
- disconnecting 55
- documentation
  - conventions 3
  - feedback 3

## - E -

- ERROR built-in trigger 69
- error messages
  - Alarm Filter 121
  - Deserialize Manager 121
  - Flatfile Manager 121
  - Inform Product Manager 121
  - LogToFile Manager 122
  - PA Resync Manager 123
  - Poll Manager 122
  - Protocol Manager 122
  - Server Manager 125

Trap Manager 127  
 UNIX installations 128  
 user interface 119  
 error status  
   SNMP v3 operations 91

**- F -**

filter 35, 38, 40  
   associating with server 49  
 filtering alarms  
   by IP range 35  
   by property group 46  
   by severity 42  
   IP range 38, 40  
 finding set of network conditions 8  
 FireTrigger alarm action 12  
 Flatfile Manager error messages 121

**- H -**

heartbeat messaging 52  
   deactivating 54  
   interval 53  
 history 72

**- I -**

ICMP\_ERROR built-in trigger 69  
 ICMP\_TIMEOUT built-in trigger 69  
 ICMP\_UNKNOWN\_ERROR built-in trigger 69  
 Inform Product Manager error messages 121  
 integration with network management platforms 22

IP filters 35  
   examples 40  
   subnet filter rules 38  
 ipsweep.exe 35

**- L -**

logging 12  
 logs  
   SNMP v3 operations 87  
 LogToFile Manager error messages 122

**- M -**

main NerveCenter components 13  
 MIB  
   sharing information 31  
 monitoring  
   resetting alarms 81, 85  
   viewing history for an alarm 72

**- N -**

NerveCenter  
   Action Router tool 13  
   Administrator 17  
   Client 19  
   database 14  
   node management 5  
   Server 14  
   servers, multiple 22  
 NerveCenter Client  
   starting 25  
 NerveCenter installation error messages (UNIX) 128  
 NET\_UNREACHABLE built-in trigger 69  
 network conditions  
   detecting 6

  finding set of 8  
   persistent 7  
   responding to 11  
 network management platform  
   filtering by IP subnet 35, 38, 40  
 network management strategy 20  
 node source - server status  
   filtering by subnet 35, 38, 40  
   populating the database 35, 38, 40  
 NODE\_UNREACHABLE built-in trigger 70  
 nodes  
   defining sets 6  
   managing 5  
 notification 11

**- O -**

objects, database 14  
 operations log 87  
   signing for errors 88  
   viewing 90

**- P -**

PA Resync Manager error messages 123  
 Poll Manager error messages 122  
 PORT\_UNREACHABLE built-in trigger 70  
 predefined behavior models 16  
 properties 6  
 property groups  
   filtering alarms 46  
 Protocol Manager error messages 122

**- R -**

reset alarms 81, 85  
 responding to network conditions 11  
 RESPONSE built-in trigger 70  
 rules for alarm filters 51

**- S -**

server  
   associating filters 49  
   automatic connection 29  
   changing port on client 33  
   connecting 26  
   deleting 32  
   disconnecting from 55  
   manual connection 27  
   selecting active 32  
   sharing MIB information 31  
 Server Manager error messages 125  
 servers  
   alarm filtering rules 51  
 servers, multiple NerveCenter 22  
 severities 15  
 smart polling 6  
 SNMP v3  
   built-in triggers 70-71  
   error status 91  
   operations log 87  
 SNMP\_AUTHORIZATIONERR built-in trigger 70  
 SNMP\_BADVALUE built-in trigger 70  
 SNMP\_DECRYPTION\_ERROR built-in trigger 70

SNMP\_ENDOFTABLE built-in trigger 70  
 SNMP\_GENERR built-in trigger 70  
 SNMP\_NOSUCHNAME built-in trigger 70  
 SNMP\_NOT\_IN\_TIME\_WINDOW built-in trigger 71  
 SNMP\_READONLY built-in trigger 71  
 SNMP\_TIMEOUT built-in trigger 71  
 SNMP\_TOOBIG built-in trigger 71  
 SNMP\_UNAVAILABLE\_CONTEXT built-in trigger 71  
 SNMP\_UNKNOWN\_CONTEXT built-in trigger 71  
 SNMP\_UNKNOWN\_ENGINEID built-in trigger 71  
 SNMP\_UNKNOWN\_USERNAME built-in trigger 71  
 SNMP\_UNSUPPORTED\_SEC\_LEVEL built-in trigger 71  
 SNMP\_WRONG\_DIGEST built-in trigger 71  
 standalone operation 21  
 status, SNMP v3 operations 91  
 subnet filter 35, 38, 40

**- T -**

technical support 4  
 tools  
   Action Router tool 13

transitions 15  
   causing 12  
 Trap Manager error messages 127  
 triggers 15  
   built-in 69

**- U -**

understanding NerveCenter 5  
 UNKNOWN\_ERROR built-in trigger 71  
 user interface messages 119

