Tivoli NetView for UNIX Administrator’s Guide

Copyright Notice

© Copyright IBM Corporation 2001. All rights reserved. May only be used pursuant to a Tivoli Systems Software License Agreement, an IBM Software License Agreement, or Addendum for Tivoli Products to IBM Customer or License Agreement. No part of this publication may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any computer language, in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual, or otherwise, without prior written permission of IBM Corporation. IBM Corporation grants you limited permission to make hardcopy or other reproductions of any machine-readable documentation for your own use, provided that each such reproduction shall carry the IBM Corporation copyright notice. No other rights under copyright are granted without prior written permission of IBM Corporation. The document is not intended for production and is furnished “as is” without warranty of any kind. All warranties on this document are hereby disclaimed, including the warranties of merchantability and fitness for a particular purpose.

U.S. Government Users Restricted Rights—Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corporation.

Trademarks

IBM, the IBM logo, Tivoli, the Tivoli logo, are trademarks or registered trademarks of International Business Machines Corporation or Tivoli Systems Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Notices

References in this publication to Tivoli Systems or IBM products, programs, or services do not imply that they will be available in all countries in which Tivoli Systems or IBM operates. Any reference to these products, programs, or services is not intended to imply that only Tivoli Systems or IBM products, programs, or services can be used. Subject to valid intellectual property or other legally protectable right of Tivoli Systems or IBM, any functionally equivalent product, program, or service can be used instead of the referenced product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by Tivoli Systems or IBM, are the responsibility of the user. Tivoli Systems or IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, New York 10504-1785, U.S.A.
Contents

Preface ........................................ xi
Who Should Read This Guide .................... xi
Prerequisite and Related Documents ................. xi
What This Guide Contains ........................ xi
Typeface Conventions ................................ xii
Platform-Specific Information ...................... xii
Online Information ................................ xii
Accessability Information ........................ xiii
Keyboard Access .................................. xiii
Contacting Tivoli Support ....................... xiii

Chapter 1. Understanding Tivoli NetView Processes ............ 1
Foreground Processes ............................ 1
  ovw Application ................................ 1
  ipmap Application ............................ 2
  xxmap Application ............................ 3
  nvevents Application ....................... 3
  xnmtrap Application ....................... 3
  xnmsnmpconf Application ................. 4
  xnmloadmib Application ................. 4
  xnmloadmib2 Application .............. 4
  xnmbrowser Application .............. 4
  xnmbrowser2 Application ............ 4
  xnmcollect Application ............ 5
  xnmbuilder Application ............ 5
  xnmgraph Application ............ 5
  xnmfault Application ............ 5
  xnmrunreport Application .......... 5
  nmpolling Application ............ 5
  backup Application ............ 6
  shpmon Application ............ 6
  xnmappmon Application .......... 6

Process Management ............................ 6
  The ovspmd Daemon .................... 7

Background Processes .......................... 7

Topology Discovery and Database Daemons ............ 7
  mgragentd Daemon .................... 8
  netmon Daemon .................... 8
  nvlockd Daemon .................... 9
  ovtopmd Daemon .................... 9
  owdbd Daemon .................... 9
  nvcoldd Daemon .................... 9
  noniptopod Daemon .................... 9
  gtdm Daemon .................... 10
  otdm Daemon .................... 10
  nvotdd Daemon .................... 11
  C5d Daemon .................... 11

Event and Trap Processing Daemons ............ 11
  pmd Daemon .................... 12
  orsmd Daemon .................... 12
  ovesmd Daemon .................... 13
  ovelmd Daemon .................... 13
  trapd Daemon .................... 13
### Chapter 3. Creating and Customizing Submaps

- Objects ........................................ 41
- Displaying the Object Database .................. 41
- Symbols ......................................... 41
  - Defining Symbol Characteristics ................ 42
  - Indicating Symbol Status ...................... 43
- Maps ............................................ 46
  - Information in the Map Database ............... 46
  - Managing Maps in a Distributed Network Environment 47
  - Customizing Maps ................................ 49
  - Assigning Map Access Levels ................... 50
- Map Snapshots .................................. 51
- Customizing Maps ................................ 51
  - Working with Submaps ......................... 52
  - Creating Child Submaps and Independent Submaps .... 53
  - Understanding Submap Presentation .......... 53
  - Understanding Submap Layouts .................. 54
  - Understanding Submap Planes ................. 55
  - Metaconnection Submap ....................... 55
- Using Tivoli NetView Applications .............. 57
  - ipmap Application .......................... 57
  - Using the xxmap Application ............ 60
  - Displaying Multiple Protocols .......... 60
- Customizing a Graphical Map .................. 61
  - Customizing Symbol Placement .................. 61
  - Partitioning Submaps ......................... 68
  - Automating Internet Submap Partitioning ........ 70
  - Creating a Child Submap ....................... 71
  - Creating an Independent Submap ............. 72
  - Changing a Parent Submap .................... 73
  - Modifying Submap Settings .................. 73
  - Assigning a Home Submap .................... 73
  - Modifying an Object Description ............. 74
  - Managing and Unmanaging Objects .......... 74
  - Acknowledging and Unacknowledging Objects .... 74
- Defining and Managing SmartSets ................ 76
  - Managing and Unmanaging SmartSets ............ 77
  - Types of SmartSets .......................... 77
  - Quick Refresher Course on Boolean Logic .......... 77
  - Pattern Matching .......................... 78
  - Adding a New SmartSet ...................... 79
  - Modifying a SmartSet ....................... 84
  - Listing Objects in a SmartSet ............... 85
  - More Examples of SmartSets ................. 85

### Chapter 4. Customizing the Graphical Interface

- Adding or Removing a Background Graphic .............. 89
- Supported Formats ................................ 89
- Adding a Background ................................ 89
- Removing a Background ................................ 90
- Arranging Symbols ................................ 90
- Using Redo Layout ................................ 90
- Setting Automatic Layout ........................... 91
Preface

This document provides information for operating the Tivoli® NetView for UNIX program. It explains what this program does and how to use it to manage and monitor a multiprotocol network.

Who Should Read This Guide

This document is for system administrators or anyone with a basic familiarity with the Tivoli NetView for UNIX program. You should also have a basic understanding of networking and UNIX. Most of the tasks in this book require a read-write map and root authority.

Prerequisite and Related Documents

The following is a list of Tivoli NetView related publications:

- Tivoli NetView Administrator’s Guide
- Tivoli NetView Administrator’s Reference
- Tivoli NetView Database Guide
- Tivoli NetView Host Connection
- Tivoli NetView Configuration Guide
- Tivoli NetView MLM User’s Guide
- Tivoli NetView Programmer’s Guide
- Tivoli NetView Programmer’s Reference
- Tivoli NetView User’s Guide for Beginners

What This Guide Contains

This document contains the following information:

- **“Chapter 1. Understanding Tivoli NetView Processes” on page 1**
  Describes the processes, applications, and databases used by the Tivoli NetView for UNIX program. This information was in the Tivoli NetView for UNIX Programmer’s Guide Read this chapter if you need an overview of Tivoli NetView.

- **“Chapter 2. Defining and Managing a Security Policy” on page 13**
  This is intended for security administrators or anyone who is responsible for managing Tivoli NetView security. This chapter describes how to use Tivoli NetView security services to control access to Tivoli NetView and includes information about how to create user and group profiles, collect and view audit data, and distribute a central security policy to other servers in your network.

- **“Chapter 3. Creating and Customizing Submaps” on page 41**
  Provides information about creating a customized submap hierarchy, which includes creating and customizing maps, submaps, and objects. This chapter also includes information about how to manage maps in a distributed network environment (client/server). In addition, this chapter describes how to use the SmartSet facility to group objects together into a SmartSet. Defining a SmartSet creates a submap of objects that meet the definition criteria you specify.

- **“Chapter 4. Customizing the Graphical Interface” on page 89**
  Explains how to customize the graphical interface. After your maps and submaps are created, use this information to customize the presentation of information. For example, you can arrange symbols, assign maps, and change the background.
• "Chapter 5. Correlating, Filtering, and Configuring Events" on page 103
  Presents information about creating event correlation rules and defining event filters to control the events that are displayed. This chapter also describes how to configure events.

• "Chapter 6. Managing Network Configuration" on page 151
  Describes how to manage network configuration using some of the tools and menu operations provided by the Tivoli NetView program. This chapter contains information about configuring for manager backup.

• "Chapter 7. Managing Network Performance" on page 177
  Describes how to monitor network performance using some of the tools and menu options provided by the Tivoli NetView program. This chapter contains information about monitoring the system resources on your management system.

• "Chapter 8. Using the Agent Policy Manager (APM)" on page 203
  Describes how to use the Agent Policy Manager to set up and view information about thresholds and file monitoring in a network. You must have the Tivoli NetView Mid-Level Manager or System Information Agent installed in your network to use the Agent Policy Manager.

• "Appendix A. Tivoli NetView Internal Traps" on page 263
  Lists the traps generated by Tivoli NetView.

The glossary at the end of this document can assist you with terminology. To view additional terminology lists, refer to:

**Typeface Conventions**

This guide uses several typeface conventions for special terms and actions. These conventions have the following meaning:

**Bold**
  Commands, keywords, file names, authorization roles, URLs, or other information that you must use literally appear in **bold**. The names of titles of screen objects also appear in **bold**.

**Italics**
  Variables and values that you must provide appear in **italics**. Words and phrases that are emphasized also appear in **italics**.

**Bold Italics**
  New terms appear in **bold italics** when they are defined in text.

**Monospace**
  Code examples, output and system messages appear in a monospace font.

**ALL CAPS**
  NetView for OS/390® commands appear in **ALL CAPS**

**Platform-Specific Information**

Refer to the release notes for platform-specific information:

**Online Information**

The release notes provide the latest information on the Tivoli NetView program. They are available in HTML and PDF versions. The HTML version is accessible from the NetView Console using the **Help...Books Online** menu item. The PDF version is in `/usr/OV/books/$LANG/pdf/readme.pdf`.
The online help facility provides task and user interface information.

The online books are available in HTML and PDF versions (Dynatext is no longer supported). The HTML versions are accessible from the NetView Console using the Help...Books Online menu item, which will bring up the books in the Netscape Navigator or Netscape Communicator browser.

PDF versions are available in the /usr/OV/books/$LANG/pdf directory.

In addition, you can access on-line documents at this web site:
http://www.tivoli.com/support

A user name and password are required.

### Accessability Information

Refer to *Tivoli NetView for UNIX User’s Guide for Beginners* for information about accessibility.

### Keyboard Access

Standard shortcut and accelerator keys are used by the product and are documented by the operating system. Refer to the documentation provided by your operating system for more information.

Refer to *Tivoli NetView for UNIX User’s Guide for Beginners* for more information about keyboard access.

### Contacting Tivoli Support

If you have a problem with any Tivoli product, you can contact Tivoli Customer Support. See the *Tivoli Customer Support Handbook* at the following Web site:

http://www.tivoli.com/support/handbook/

The handbook provides information about how to contact Tivoli Customer Support, depending on the severity of your problem, and the following information:

- Registration and eligibility
- Telephone numbers and e-mail addresses, depending on the country you are in
- What information you should gather before contacting support
Chapter 1. Understanding Tivoli NetView Processes

The Tivoli NetView program uses many processes and databases to perform network management functions. To administer your network effectively, you should be familiar with the basic operation and interactions among the parts of Tivoli NetView.

This chapter describes each part of Tivoli NetView and describes its operation. Read this chapter if you want to understand how the different parts of the Tivoli NetView program interface.

Tivoli NetView uses two types of processes:

- **Foreground processes**, or applications, that run or can be invoked while the graphical interface is running.
- **Background processes**, or *daemons*, that run continuously regardless of whether the graphical interface is running. These processes can be started only by the root user or the root shell and stopped only by the root user. Generally, the daemons provide services that must be available at all times.

## Foreground Processes

The processes, or applications, listed in Table 1 provide the interface through which you manage your network. Many of these processes correspond to selections you can make from the main menu.

**Table 1. The Tivoli NetView Foreground Processes**

<table>
<thead>
<tr>
<th>Process Category</th>
<th>Process Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Principal Tivoli NetView graphical interface</td>
<td>ovw</td>
</tr>
<tr>
<td>Map display</td>
<td>ipmap (IP topology)</td>
</tr>
<tr>
<td></td>
<td>xxmap (non-IP topology)</td>
</tr>
<tr>
<td>Event display</td>
<td>nvevents</td>
</tr>
<tr>
<td>Menu operations</td>
<td>xnmtrap</td>
</tr>
<tr>
<td></td>
<td>xnmsnmpconf</td>
</tr>
<tr>
<td></td>
<td>xnmloadmib</td>
</tr>
<tr>
<td></td>
<td>xnmloadmib2</td>
</tr>
<tr>
<td></td>
<td>xnmbrowser</td>
</tr>
<tr>
<td></td>
<td>xnmbrowser2</td>
</tr>
<tr>
<td></td>
<td>xnmcollection</td>
</tr>
<tr>
<td></td>
<td>xnmbuilder</td>
</tr>
<tr>
<td></td>
<td>xnmfault</td>
</tr>
<tr>
<td></td>
<td>xnmgraph</td>
</tr>
<tr>
<td></td>
<td>xnmrunreport</td>
</tr>
<tr>
<td></td>
<td>nmpolling</td>
</tr>
<tr>
<td></td>
<td>backup</td>
</tr>
<tr>
<td></td>
<td>shpmon</td>
</tr>
<tr>
<td>Dialog box management</td>
<td>xnmappmon</td>
</tr>
</tbody>
</table>

### ovw Application

The `/usr/OV/bin/ovw` application is the principal Tivoli NetView graphical interface for managing TCP/IP-based internets. The ovw application provides map drawing, map editing, and menu management operations. The ovw application is an X11/Motif application based on OSF/Motif user interface guidelines.
To start the ovw application, execute the /usr/OV/bin/netview shell script. The netview shell script supports the same options as those supported by the ovw application. The ovw application automatically starts the ipmap, xxmap, and nvevents applications.

If you are the root user, the netview shell script starts the daemon processes and the ovw application. If you are not the root user, the shell script starts only the ovw application.

After the ovw application is operating, some daemons and applications are dynamically updated through events forwarded by the ovesmd daemon. Table 2 shows where the ovesmd daemon sends events, and how those events are used.

<table>
<thead>
<tr>
<th>Destination</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>The netmon daemon</td>
<td>Tracks changes in the state of the network.</td>
</tr>
<tr>
<td>The ovtopmd daemon</td>
<td>Updates the topology database.</td>
</tr>
<tr>
<td>The ipmap application</td>
<td>Informs the graphical interface of changes in the IP topology database and informs ovtopmd of user-initiated changes to map databases.</td>
</tr>
<tr>
<td>The nvevents application</td>
<td>Displays the event in either the Event Cards or List format.</td>
</tr>
<tr>
<td>The tralertd daemon (AIX only)</td>
<td>Forwards events to the Tivoli NetView for OS/390 program as alerts.</td>
</tr>
</tbody>
</table>

ipmap Application

The /usr/OV/bin/ipmap application is started by the ovw application automatically. The ipmap application ensures that the ovw application (graphical interface) and the ovtopmd daemon behave consistently. For example, when an object is deleted using the graphical interface, the graphical interface tells ipmap which symbols and objects were removed. The ipmap application then tells ovtopmd to make the appropriate changes to the topology database.

In another example, when the netmon daemon discovers a new node, the ovtopmd daemon adds the node to the IP topology database and informs ipmap that a new node has been discovered. The ipmap application uses what it knows about IP devices to tell the graphical interface which icon and connection symbols it needs to create. The graphical interface then displays the correct symbols and modifies the map database accordingly.

When the ipmap application is started, it queries the ovw application to determine when the map was last open. Next, it calls the ovtopmd daemon to find out all changes to the IP topology database since the last time the map was open using the graphical interface. The ipmap application determines what, if anything, has changed since the map was last open, and then tells the ovw application to add, change, or delete the appropriate icon and connection symbols. This process is called synchronization.

If the ipmap application can find an association between an SNMP node’s sysObjectID Management Information Base (MIB) variable and a symbol type in the oid_to_sym file, the graphical interface displays the node by drawing the appropriate symbol on the submap. If the oid_to_sym file does not contain a
matching sysObjectID entry, the ipmap application extracts attributes from the topology database and tells the ovw application to create and display a generic symbol.

After the synchronization phase is completed, the ipmap application is updated based on information received dynamically from the ovtopmd daemon, which receives updates from the netmon daemon. The netmon daemon continuously monitors the state of your network by sending SNMP requests to SNMP-managed nodes and Internet Control Message Protocol (ICMP) requests to non-SNMP IP nodes. The netmon daemon communicates changes to the ovtopmd daemon, which updates the topology database and informs the ovwdb daemon that the object database needs to be updated. If the change affects the IP Map, the ipmap application notifies the ovw application. Otherwise, the change is reflected only in the object database and the IP topology database.

**xxmap Application**

The `/usr/OV/bin/xxmap` application is very similar to the ipmap application with one important difference. The xxmap application processes non-IP topology information that is stored by the gtmd daemon in the general topology database. The ipmap application processes IP information that is stored by the ovtopmd daemon in the IP topology database. The non-IP topology information stored in the general topology database comes from non-IP discovery applications or agents that use the general topology MIB format to send topology information to the gtmd daemon.

The xxmap application ensures that the maps being displayed are synchronized with the contents of both the general topology and object databases. Using the `/usr/OV/conf/oid_to_protocol` file, the xxmap application matches the oid contained in the MIB to the correct submaps and symbols required to display the topology of a particular protocol. For more information about the xxmap application, refer to *Tivoli NetView for UNIX Programmer’s Guide*.

**nvevents Application**

The `/usr/OV/bin/nvevents` application displays events in the main window Control Desk in either the Event Cards or List presentation format. When this process is invoked by the ovw application, it reads the ovevent.log file to recover events that occurred since the nvevents application was last running.

Following startup, nvevents receives SNMP traps that are filtered by the ovesmd daemon and uses those traps to update the event cards or list.

It also monitors the status of the snmpCollect, netmon, or ovtopmd daemons. If one of those daemons is not running, nvevents presents a warning box to notify you.

**xnmtrap Application**

The `/usr/OV/bin/xnmtrap` application is invoked when you select *Event Configuration* → *Trap Customization: SNMP*... from the *Options* pull-down menu. This option helps you control how enterprise-specific events (traps) are handled. For example, you can customize the message displayed through nvevents when a particular event arrives. You can also specify a command or a script that should be executed when a particular event arrives. Event configuration changes are stored in the `/usr/OV/conf/C/trapd.conf` configuration file.
xnmsnmpconf Application

The `/usr/OV/bin/xnmsnmpconf` application is invoked when you select **SNMP Configuration** from the **Options** pull-down menu, which enables you to specify netmon status polling intervals, timeout intervals, number of retries, proxy information, and agent community names.

The SNMP configuration values are stored in the `/usr/OV/conf/ovsnmp.conf_db` file. Processes on the management station, such as the netmon daemon, MIB applications, and SNMP API-based applications, look up an agent’s community name and other SNMP options in the `ovsnmp.conf_db` file when access to agent MIB values through SNMP requests is required.

xnmloadmib Application

The `/usr/OV/bin/xnmloadmib` application is invoked when you select **Load/Unload MIBs: SNMP** from the **Options** pull-down menu. Select this option when you want to load new SNMPv1 Internet-standard, enterprise-specific, or general topology MIBs into the MIB database.

After you have loaded the new MIB into the MIB database, you can use the MIB Browser, MIB Data Collection, and MIB Application Builder options, as well as the applications built by the MIB application builder, to manage your multivendor network.

xnmloadmib2 Application

The `/usr/OV/bin/xnmloadmib2` application is invoked when you select **Load/Unload MIBs: SNMPv1/SNMPv2** from the **Options** pull-down menu. Select this option when you want to load new SNMPv1 or SNMPv2 Internet-standard, enterprise-specific, or general topology MIBs into the MIB database.

After you have loaded the new MIB into the MIB database, you can use the MIB Browser, MIB Data Collection, and MIB Application Builder options, as well as the applications built by the MIB application builder, to manage your multivendor network.

xnmbrowser Application

The `/usr/OV/bin/xnmbrowser` application is invoked when you select **MIB Browser: SNMP** from the **Tools** pull-down menu, which enables you to get and set MIB values for Internet-standard and enterprise-specific MIB objects on SNMPv1 agents. This application hides the actual SNMP requests used to perform these operations, so you need to know only which MIB objects you want to access, not the commands required to do so.

xnmbrowser2 Application

The `/usr/OV/bin/xnmbrowser2` application is invoked when you select **MIB Browser: SNMPv1/SNMPv2** from the **Tools** pull-down menu, which enables you to get and set MIB values for Internet-standard and enterprise-specific MIB objects on SNMPv1, SNMPv2C, and secure SNMPv2USEC agents. This application hides the actual SNMP requests used to perform these operations, so you need to know only which MIB objects you want to access, not the commands required to do so.
**xnmcollect Application**

The `/usr/OV/bin/xnmcollect` application is invoked when you select [Data Collection and Thresholds: SNMP...](#) from the **Tools** pull-down menu, which enables you to configure the manager to collect MIB data from network objects at regular intervals. The configuration information is stored in the `/usr/OV/conf/snmpCol.conf` configuration file. The collected data is stored in files in the `/usr/OV/databases/snmpCollect` directory.

After you have stored the collected data, you can use the xnmgraph application to view the collected data or import the data into your own application using the `snmpColDump` command. You can also define thresholds for the collected MIB data and generate events when the specified thresholds are exceeded.

**xnmbuilder Application**

The `/usr/OV/bin/xnmbuilder` application is invoked when you select [MIB Application Builder: SNMP...](#) from the **Tools** pull-down menu, which enables you to build custom screens to manage multivendor MIB objects. The information you define using the MIB application builder is stored in registration files and help files. The xnmgraph application displays **Graph** applications, while the `xnmappmon` application displays **Form** and **Table** applications.

**xnmgraph Application**

The `/usr/OV/bin/xnmgraph` application enables you to graph the results of monitoring operations for managed SNMP objects selected from the map. The results may be real time or collected historical data.

When you select [Graph Collected Data: SNMP...](#) from the **Tools** pull-down menu, the ovw application forwards the objects previously selected on the map to the xnmgraph application. The xnmgraph application queries the selected objects and displays the results in a line graph.

**xnmfault Application**

The `/usr/OV/bin/xnmfault` application enables you to locate, for any node on the map, all of its component resources that have failed.

When you select a node on a submap, and then select [Tools -> Failing Resource Display](#), the xnmfault application creates a submap showing all the resources that are children of the selected node and that have failed. Be sure to close this submap through the navigation tree. If not, the xnmfault application is not notified that the submap has been closed. The submap remains open and can be accessed from the Navigation Tree.

**xnmrunreport Application**

The `/usr/OV/bin/xnmrunreport` application corresponds to the [Reports: Site Provided...](#) option from the **Monitor** pull-down menu. You can use this application to generate reports that send output directly to the screen, which is useful for problem determination or real-time network monitoring. You can also generate reports that store information in flat files or databases for future reference.

**npmpolling Application**

The `/usr/OV/bin/npmpolling` application is accessed through the [Topology/Status Polling Intervals: IP...](#) option from the **Options** pull-down menu that enables you to configure certain netmon polling intervals.
backup Application

The /usr/OV/bin/backup application is accessed through the Backup... option from the Administer pull-down menu. This process lets you configure managers to backup another manager’s objects if the other manager should go down. You configure objects as manager nodes or managed containers and then specify managers as an active manager or a backup manager.

shpmon Application

The /usr/OV/bin/shpmon application is accessed through the Local Filesystem & Paging Space option from the Monitor pull-down menu. Using this option, you can receive a trap or message letting you know when a threshold condition has reached its limit. The shpmon application monitors the root filesystem and the paging space on the local manager system where Tivoli NetView is installed.

xnmappmon Application

The /usr/OV/bin/xnmappmon application manages dialog boxes that contain the text output of monitoring operations performed on managed SNMP objects that have been selected from the map. This application is also called an application encapsulator.

When you select a monitoring operation from the Tivoli NetView main menu bar, the ovw application forwards the objects selected on the map to the xnmappmon application as input. The xnmappmon application displays the appropriate dialog box, translates the selected operation and object or objects into a command, and executes the command.

When processing is completed, the command returns its text output to the xnmappmon application for display in the dialog box. In the case of the Locate..Route menu option, xnmappmon also returns to ovw a list of map objects to highlight as output of the network management operation.

For example, the xnmappmon application can be used to execute the mibform, mibtable, rnetstat, findroute, rbdf, and rping commands. These commands help you monitor and diagnose problems in your TCP/IP network.

Process Management

As a network administrator, you use several process management commands to control the operation of Tivoli NetView daemons. These commands are:

- **ovstart**: Starts all daemons or selected daemons, depending on the options you specify.
- **ovstop**: Stops all daemons or selected daemons, depending on the options you specify.
- **nvstatus**: Shows the status of all daemons or selected daemons, depending on the options you specify, including the process management daemon itself.
- **nvstatus**: Shows the status of all daemons or selected daemons, depending on the options you specify, that are running on the server. This command can be run only on a client workstation.
The ovspmd Daemon

The process management daemon, ovspmd, coordinates the starting and stopping of daemons that communicate with the Tivoli NetView program.

The ovstart command starts the ovspmd daemon, which in turn starts the other Tivoli NetView daemons in a particular order. The information about startup order is in the startup configuration file, /usr/OV/conf/ovsuf.

The startup configuration file is constructed by the ovaddobj command. The ovaddobj process takes information from the local registration files (LRF) and places it in the startup configuration file.

The ovspmd daemon receives requests from the ovstart command and sends status responses to ovstart. It starts all Tivoli NetView daemons (if they are listed in the ovsuf file) and maintains a communication channel with each of them. These daemons should always be running. On the AIX operating system, the ovspmd daemon starts the host connection daemons, spappld and tralertd, only if the AIX Service Point is installed. A check for the AIX Service Point is performed during installation of Tivoli NetView.

Background Processes

The daemons fall into the following categories:
- Topology discovery and database operation
- Event and trap processing
- Security services
- Host connection (AIX only)

Topology Discovery and Database Daemons

The Tivoli NetView program discovers and updates the topology of IP networks and translates the information into symbols that appear on the views you see of your network’s map. The program also facilitates the discovery of networks that use non-IP protocols for communication, so you can extend your network management coverage.

Figure 1 illustrates the interactions among the topology discovery and database daemons. Each daemon is described in the following sections.
mgragentd Daemon

The `/usr/OV/bin/mgragentd` daemon runs on the manager station to determine the status of the Tivoli NetView program and to respond to queries from other manager stations. In a distributed network environment, the mgragentd daemon is used to determine the status of the Tivoli NetView daemons. This daemon requires no user configuration or manipulation.

netmon Daemon

The `/usr/OV/bin/netmon` daemon polls SNMP agents to discover network topology. During initial discovery, the netmon daemon sends an SNMP trap to the trapd daemon to inform it of newly discovered network entities. Some of these entities may be agents that communicate with non-IP networks or devices. As long as the entities have an IP address, the netmon daemon will discover them.

After initial discovery is complete, the netmon daemon polls the SNMP agents to detect topology, configuration, and status changes in the IP network, and sends corresponding traps to the trapd daemon. However, the netmon daemon does not have a connection with non-IP networks or devices. Topology changes for these network entities must be communicated to the Tivoli NetView program in other ways. See "noniptopod Daemon" on page 9 and "gtmd Daemon" on page 10 for more information about non-IP topology discovery.

In addition to polling SNMP agents, the netmon daemon polls network nodes. To check an SNMP-managed node’s MIB values, the netmon daemon uses an SNMP request. To check the status of all nodes, the netmon daemon uses ICMP echo requests (ping). Based on the discovered information, the netmon daemon generates and updates the topology map.

The netmon polling values and information about network objects, including their relationships, status, and thresholds, are stored in a set of files called the topology database. If the database does not initially exist, the ovtopmd daemon creates it during discovery of the network’s topology and automatic generation of the map. When you start up the graphical interface, the ipmap application compares the contents of the topology database with the contents of the graphical interface’s map databases. The ipmap application tells the graphical interface what has changed since its last invocation, and then the graphical interface updates the map.
The netmon daemon assumes that your initial network management region is composed of the network or networks to which the Tivoli NetView program, the management station, is directly connected. If you want a different initial configuration, you can provide a seed file, which contains a list of nodes you want to appear on the automatically-generated network map.

See "Using a Seed File to Control Network Discovery" on page 153 for information about seed files.

**nvlockd Daemon**

The `/usr/OV/bin/nvlockd` helps the gtmd daemon and xxmap application control access to the ovwdb daemon. This daemon requires no user configuration or manipulation.

**ovtopmd Daemon**

The `/usr/OV/bin/ovtopmd` daemon maintains the network topology database. The topology database is a set of files in the `/usr/OV/databases/openview/topo` directory that store netmon polling values and other information about network objects, including their relationships and status.

The ovtopmd daemon generates and updates the topology database using status information obtained from the netmon daemon. The ovtopmd daemon also checks for existing non-IP objects with which to correlate.

**ovwdb Daemon**

The `/usr/OV/bin/ovwdb` daemon controls the Tivoli NetView object database. This database stores object information that the graphical interface uses to generate output for Describe operations. For example, when you select Modify/Describe..Objects... from the Edit pull-down menu, you can view several attributes for the objects you have selected. The information you see is retrieved from the Tivoli NetView object database.

If the netmon daemon detects a change in the network, it calls the ovtopmd daemon to update the topology database. In turn, the ovtopmd daemon calls the ovwdb daemon to update the Tivoli NetView object database.

To run the ovw application, first start the ovwdb daemon. The object database must be accessible to the ovw application so that the default submap can be generated.

**nvcold Daemon**

The `/usr/OV/bin/nvcold` daemon maintains SmartSets as they have been defined by users or applications.

**noniptopod Daemon**

Initial topology discovery is handled by the netmon daemon, which sends an SNMP trap to the trapd daemon when a new network object is discovered. The trapd daemon then forwards traps, via the bvesmd daemon, to the `/usr/OV/bin/noniptopod` daemon, which accepts any trap that meets the following criteria:

- The discovered object has an IP address.
- The discovered object could be an agent that can communicate with a non-IP device or network.
The noniptopod daemon sends an SNMP get command for each object identifier (OID) listed in the `/usr/OV/conf/oid_to_command` file to the new network object associated with the trap. It checks to see if the OIDs are supported by the agent. If a non-null response is sent back, the commands associated with the supported OIDs are started.

The command in the `oid_to_command` file activates the non-IP protocol's proprietary daemon, which sends a request to the agent to gather all non-IP topology information and forward it to the proprietary daemon for conversion to the general topology MIB format. The MIB information is then sent to the gtmd daemon.

**gtmd Daemon**

The `/usr/OV/bin/gtmd` daemon receives information sent by non-IP discovery applications, agents, and proxy agents that use the general topology MIB format to describe the attributes of devices on a non-IP network. Information can be received in traps or in application programming interface (API) calls. This daemon stores the non-IP topology information in its own database and correlates it with IP topology information stored in the object database to determine whether an IP object can also be identified as having an association with a non-IP protocol. The gtmd daemon makes the topology information available for display through the `xxmap` application.

Non-IP discovery applications can register with the gtmd daemon to receive notifications of changes to the topology information or to receive topology data. The gtmd daemon updates its database each time a trap is received and notifies registered applications of the operation performed. The gtmd and noniptopod daemons are, by default, not started when Tivoli NetView is started. You can use the Tivoli desktop to configure them and indicate that they should be started by the `netview` command.

For more information about the Tivoli NetView general topology function and the gtm API, refer to the *Tivoli NetView for UNIX Programmer’s Guide*.

**otmd Daemon**

The `/usr/OV/bin/otmd` daemon receives and processes dynamic topology and status updates from an openmon topology agent.

The otmd daemon reads the configuration file, in the `/usr/OV/conf/openmon` subdirectory, for each openmon topology application. The otmd daemon uses the `ovstart` command to start any openmon topology application that is configured to start automatically by the otmd daemon.

It obtains the port number for the otmd service from the `/etc/services` file and sets up the TCP socket and waits for the openmon agents to connect. When a connection is received from an agent, the otmd daemon receives and processes all the data from the agent. It stores all topology object information in the general topology manager (GTM) database and the attribute information with the topology object in the OVW object database.

If an update sequence number is received from an agent, the otmd daemon saves the update sequence number in the agent’s configuration file. This sequence number is maintained to keep track of the information that the agent has sent to the openmon application.
nvotd Daemon

The `usr/OV/bin/nvotd` daemon receives non-IP topology events from the gtmd daemon and forwards them to the Event Display application (`nvevents`) if API calls are used to send information to the gtmd daemon. The nvotd daemon, by default, is not started when the Tivoli NetView program is started. You can use the Tivoli desktop to configure the nvotd daemon and indicate that the daemon should be started by the `netview` command.

C5d Daemon

The `usr/OV/bin/C5d` daemon coordinates setting threshold monitor and file monitor definitions for Tivoli NetView Mid-Level Managers (MLMs) and Systems Information Agents (SIAs). The C5d daemon is, by default, not started when the Tivoli NetView program is started. You can use the Tivoli desktop to configure the C5d daemon so that it starts when the `netview` command is used.

Event and Trap Processing Daemons

Events and traps provide information about changes in the status of network elements and alert the Tivoli NetView program to occurrences in the network. When events and traps are received, they must be routed to the appropriate applications and logged for future reference.

The daemons that perform these tasks can be divided into the following three groups:

- **Communications infrastructure daemons**, including the `pmd` daemon and the `orsd` daemon, that control all CMOT and some SNMP communications entering and leaving the Tivoli NetView program
- **Event management services daemons**, including the `ovesmd` daemon and the `ovelmd` daemon, which route and log all events received by the Tivoli NetView program
- **Other event and trap processing daemons**, including the `trappd` daemon, the `nvcorrd` daemon, the `actionsvr` daemon, the `nvpagerd` daemon, the `nvserverd` daemon, the `trapgend` daemon (**AIX** only), and the `snmpCollect` daemon

Figure 2 illustrates the interactions among the event and trap processing daemons. Each daemon is described in the following sections.
pmd Daemon

When an agent sends an event to an application, the event contains no routing information. It's similar to sending a letter without an address. The `/usr/OV/bin/pmd` daemon receives the events from the trapd daemon and forwards them to the ovesmd daemon. The ovesmd daemon forwards the events to the applications that have registered to receive them.

The pmd daemon centralizes the external communications for all applications and processes that use the CMOT protocol, as well as some of those using SNMP. This daemon contains the SNMP and CMOT protocol stacks.

The pmd daemon has two components:
- The locator function, which routes outbound requests to the appropriate agent. The locator function consults the data in the object registration database, which includes agent locations and the protocol used to access them. Note that application programs do not access the object registration database directly; instead, the pmd daemon automatically does so on their behalf.
- The association management function, which provides a way of sharing connections between network management nodes. This function enables application managers to share connections without having to bring up multiple connections between the nodes.

orsd Daemon

The `/usr/OV/bin/orsd` daemon maintains the consistency of the CMIS object information in the object registration database. The object registration database is consulted by the pmd daemon to determine where an agent resides and which protocol to use to communicate with it.

Figure 2. Interactions among Event and Trap Processing Daemons
ovesmd Daemon

The `/usr/OV/bin/ovesmd` daemon is part of Tivoli NetView event management services. Known as the event sieve agent (ESA), the ovesmd daemon distributes events throughout the network based on the filters in effect for a particular application or user.

When events are sent from agents, they do not contain any routing information that the pmd daemon can use to forward them to the correct application. When the pmd daemon receives an event from trapd, it sends the event to the ovesmd daemon, which filters the event and forwards it to applications that have registered to receive it. The ovesmd daemon also forwards the event to the ovelmd daemon, which controls the event log.

ovelmd Daemon

The `/usr/OV/bin/ovelmd` daemon is the event log agent (ELA), which stores SNMP traps, CMIS events, and event log configuration values in the `/usr/OV/log/ovevent.log` file. The ovevent.log and the ovevent.BAK files are binary files that are the source of information for the dynamic and historical event displays. It logs all SNMP traps received from the network and internal processes.

trapd Daemon

The `/usr/OV/bin/trapd` daemon receives traps from agents and internal processes and forwards them to the netmon, tralertd (AIX only), nvcorrd, and the pmd daemons, and to the ipmap application. In return, the netmon daemon sends events to trapd when its polling reveals a change in the status of a network element.

The trapd daemon also forwards network events to other applications that have connected to it through the OVsnmp API.

The trapd daemon logs all received traps in the `/usr/OV/log/trapd.log` file. You can use any editor to look at the contents of the trapd.log file. You can also choose to turn trapd logging off.

nvcorrd Daemon

The `/usr/OV/bin/nvcorrd` daemon receives events from the trapd daemon, correlates or compares the events to event processing decision and actions defined in rulesets registered with nvcorrd, and forwards them to registered applications, one of which is the Event Display application. The nvcorrd daemon passes events to the actionsvr daemon to manage actions defined in event correlation rulesets.

actionsvr Daemon

When an action is to be processed in an event correlation rule, the nvcorrd daemon passes the action to the `/usr/OV/bin/actionsvr` daemon. The actionsvr daemon manages the action, starting a child process, while the nvcorrd daemon continues to process the event correlation ruleset. All actions requested and the events which caused those actions are logged in the `/usr/OV/log/nvaction.alog` and `/usr/OV/log/nvaction.blog` files.

nvpagerd Daemon

The `/usr/OV/bin/nvpagerd` daemon manages the routing of the page command that is issued from the command line or within an event correlation rule.
When the paging action is to be processed in an event correlation rule, the nvcorrd daemon passes the action to the actionsvr daemon, which passes the action to the nvpagerd daemon. All paging actions requested and the events which caused those actions are logged in the /usr/OV/log/pagerd.log file.

nvserverd Daemon

The /usr/OV/bin/nvserverd daemon receives events from the nvcorrd daemon and forwards them to different Event Display applications running on client workstations. The nvserverd daemon enables you to select a set of events in one client application and work with it in all registered applications, clearing these events or changing the status of the events.

trapgend Daemon (AIX Only)

The /usr/OV/bin/trapgend daemon is a subagent (SMUX peer) provided with the Tivoli NetView program that converts AIX alertable errors to SNMP traps.

On RS/6000 processors running AIX Version 3 Release 2 or later, system errors are logged by the operating system’s error logging facilities in the /dev/error special file. An object installed by Tivoli NetView in each system’s object data manager (ODM) directs the AIX error logging daemon (errdemon) to inform the trap-notify process when alertable errors are logged. These alertable errors are forwarded by the trap-notify process to the trapgend daemon, which converts these alertable errors to SNMP traps.

Using the SNMP multiplexer (SMUX) protocol, trapgend forwards the traps to the AIX SNMP daemon, snmpd. The snmpd daemon then forwards the traps to the trapd daemon on the Tivoli NetView manager specified by the trap destination.

The trapgend daemon also provides a trap throttle to suppress identical trap generation, enables remote ping operations from the Tivoli NetView program, and supports CPU utilization and disk space monitoring MIB extensions.

The trapgend daemon must be installed on all nodes running AIX Version 3 Release 2 or later in your network if you want to receive information about CPU and disk space utilization.

snmpCollect Daemon

The /usr/OV/bin/snmpCollect daemon collects, compares, and stores SNMP agent MIB values. It also checks the collected values against user-defined thresholds and generates events if the thresholds are exceeded.

Security Services Daemons

The security services daemons determine whether security is on or off, manage authentication and identification of Tivoli NetView users, and manage communication between the security server and client workstations. Each daemon is described in the following sections.

nvsecd Daemon

The /usr/OV/bin/nvsecd daemon determines if Tivoli NetView security is on or off. If security is on, the nvsecd daemon requires each Tivoli NetView user to login using a valid Tivoli NetView user ID, password, and group ID. The nvsecd daemon checks the user’s profile to verify login and checks the permissions defined in the user’s group profile to control access to Tivoli NetView resources. After login
verification, the nvsecd daemon establishes a shared-key security context for each Tivoli NetView process. The nvsecd daemon also monitors and stores security audit data.

**nvsecltd Daemon**

The `/usr/OV/bin/nvsecltd` runs on client workstations and listens for communication from the security server. The nvsecltd daemon starts only when security is turned on and when at least one user logs into Tivoli NetView on a client workstation. The nvsecltd daemon stops when the last client user logs out of the Tivoli NetView program.

**Host Connection Daemons (AIX Only)**

If your Tivoli NetView program is connected to the Tivoli NetView for OS/390 program by an AIX Service Point program, you are using the services of the host connection daemons. These daemons facilitate both the conversion of SNMP traps to SNA alerts as well as all communication between the Tivoli NetView program and the Tivoli NetView for OS/390 program through the Service Point application. Each daemon is described in the following sections. For more information about the host connection, refer to *Tivoli NetView for UNIX Host Connection*.

**tralertd Daemon**

The `/usr/OV/bin/tralertd` daemon is used in an environment where both TCP/IP and SNA protocols are running. The tralertd daemon receives events and traps generated or received by the Tivoli NetView program. If a trap is so configured, the tralertd daemon converts it to an SNA alert, and sends the SNA alert to the Tivoli NetView for OS/390 program through the AIX NetView Service Point program. If all the converted trap information cannot fit into the SNA alert, the original trap information is saved in the `/usr/OV/databases/tralertd` database and assigned a corresponding Log ID. The Tivoli NetView for OS/390 program uses this Log ID to query the tralertd database to retrieve the rest of the information.

**spappld Daemon**

The `/usr/OV/bin/spappld` daemon provides a command interface between the Tivoli NetView for OS/390 program on the host and the Tivoli NetView program in an environment running SNA and TCP/IP protocols. The spappld daemon receives Tivoli NetView for OS/390 RUNCMDs and executes their contents in the internet environment, and sends responses to the Tivoli NetView for OS/390 program through the AIX NetView Service Point program.

**Databases**

This section describes the following databases:

- Map
- Object
- IP topology
- General
- Object registration service

These databases are controlled by the Tivoli NetView processes; you cannot edit them directly.
Map Database

The map database contains presentation information that is specific to each map. There is one map database per map. Examples of presentation information stored in the map database include the exact symbol placement on a map, the symbol associated with each object, and symbol labels. The ovw application updates the map database based on requests from the user or from other processes, such as the ovmap application. Use the ovmapdump command to view the contents of the map database. The map database is maintained by the ovw application.

Object Database

The object database contains global object information. The information is generic; that is, it is not customized to any specific application. The object database contains information related to fields such as sysObjectID, vendor, and SNMP agent. When you choose the Edit..Modify/Describe..Object menu option, the information you see in the fields comes from this database. If netmon detects a status change in the network, netmon calls ovtopmd, which calls ovwdb to update the object database. Use the ovobjprint command to view the object database. The object database is maintained by the ovwdb daemon.

IP Topology Database

The IP topology database contains topology information used during IP discovery and layout. The information in the topology database spans all maps. Much of the information in the Tivoli NetView object database is duplicated in the topology database. Information in the topology database that is not in the object database includes state information for the netmon daemon.

The most important netmon state information includes time stamps that indicate when the object last changed and should next be polled. This information helps the netmon daemon detect changes so it can communicate the changes to the ovtopmd and tcpd daemons. The topology database is controlled by the ovtopmd daemon and is updated based on information received from the netmon daemon. Use the ovtopodump command to view the contents of the IP topology database. The IP topology database is maintained by the ovwdb daemon.

General Topology Manager Database

The general topology manager (GTM) database stores topology information sent to the bmap daemon. This database also stores information about submap grouping and content that has been defined by the protocol discovery applications or agents.

The xxmap application queries both this database and the Tivoli NetView object database for display and semantic information. Any topology information stored in the general topology database can be deleted only by the agent that originally added the information. You cannot view the GTM database.

For more information about the use of the GTM database, refer to Tivoli NetView for UNIX Programmer’s Guide.

Object Registration Service Database

The object registration service (ORS) database contains location and protocol information for agents that use a protocol other than SNMP. This information helps to provide location transparency, which allows managers and agents to access objects and agents without using hard-coded addresses.
Customizing and Generating Reports on the Object Database

This section describes two utilities that allow you to customize and query the object database from the command line. Using these utilities substantially reduces the effort to customize the object database and produce reports. They also allow integration with Tivoli Inventory to collect and retain hardware and software information across the enterprise.

The nvdbimport Utility

This nvdbimport utility can be used to import object field values into the Tivoli NetView object database from a file. For example, if you wanted to set a new field, called isSite2, to TRUE for all nodes at Site 2, you would perform the following steps:

1. Create an input file with the list of nodes that are at Site2.
2. Add the field isSite2 with the value of TRUE.
3. Create a field called isSite2 in the object database.
4. Run the nvdbimport utility to update the database for these nodes.

You can now create a SmartSet of all nodes with isSite2 set to TRUE.

If you need to recreate the database, you can quickly restore your customization. Refer to the Tivoli NetView Administrator’s Reference, for more information. Example input files (called *.import) are provided in the /usr/OV/conf/nvdbtools directory.

The nvdbformat Utility

The nvdbformat utility can be used to generate custom reports from the Tivoli NetView object database complete with header and footers that provide aggregate information. For more information, refer to the Tivoli NetView Administrator’s Reference. Example input files (called *.format) are provided in the /usr/OV/conf/nvdbtools directory.

Role of the SEAs (Solaris Only)

The two Sun Solstice Enterprise Agents (SEAs) relevant to Tivoli NetView are snmpdx and mibiisa.

The snmpdx agent is the master agent that controls the other agents. It also routes incoming SNMP requests similar to the way snmpd works on AIX. In summary, this agent listens on Port 161 and routes the requests to the corresponding SNMP agent based on the registration files residing in the /etc/snmp/conf directory.

The mibiisa agent provides support for MIB-II.

For more detailed information about these two agents, refer to the HTML documentation supplied on the CD-ROM. Refer to Tivoli NetView for UNIX Installation and Configuration for information about how to install the documentation and the SEAs.

The mrgagentd daemon is registered with the snmpdx agent. When snmpdx receives SNMP requests for the MIB supported by mrgagentd, it passes the requests to mrgagentd.
The configuration files for snmpdx are in the /etc/snmp/conf directory. The snmpdx daemon resides in the /usr/lib/snmp directory. Messages from the snmpdx daemon are written in the /var/adm/messages file.

To run snmpdx, enter the following command:

/usr/lib/snmp/snmpdx

The snmpdx daemon starts the mibiisa daemon.

**Note:** The snmpdx daemon and the mibiisa daemon must be running for Tivoli NetView to work correctly.

When you start your machine, the snmpdx and mibiisa daemons start automatically.
Chapter 2. Defining and Managing a Security Policy

You can use Tivoli NetView security services to define a security policy for your network. This chapter, which is intended for security administrators or whoever is responsible for managing Tivoli NetView security, will help you understand Tivoli NetView security services so you can define a security policy that best suits your needs. This chapter also describes what you need to do to define a security policy. You will need to perform these tasks before turning security on. You might also need to perform some of these tasks after turning security on to add new Tivoli NetView users or applications. In addition, this chapter describes the tasks to be performed in order to manage your security policy, such as, how to distribute the security configuration to other servers in your network and how to view audit data.

The following topics are described:

- ["Understanding Tivoli NetView Security Services"](#)
- ["Defining a Security Policy" on page 24](#)
- ["Managing Tivoli NetView User Profiles" on page 26](#)
- ["Defining the Global Security Settings" on page 32](#)
- ["Managing Security" on page 35](#)
- ["Converting ARFs to SRFs" on page 39](#)
- ["Verifying Security Permission For Shell Scripts" on page 39](#)

---

Understanding Tivoli NetView Security Services

The key to effective security is understanding how the security features work and then enforcing the features. Tivoli NetView security services provides the following controls:

- Network authentication and identification
- Protected network communication
- Password protection
- Continuous, auditable network management
- Network access control of Tivoli NetView resources
- Customized Tivoli NetView graphical interface
- Audit management
- Consistent security controls
- Pager service for event correlation

---

Network Authentication and Identification

Security services authenticates each user, allowing access to Tivoli NetView when the user logs in using a valid Tivoli NetView user ID, group ID, and password. Additional login controls include restricting user access to specific days and times and to specific client and server machines. You create a profile for each user that contains this information.

See ["Managing Tivoli NetView User Profiles" on page 26](#) for more detailed information.

---

Protected Network Communication

Communication between session partners (users, clients, servers) uses a security context to ensure message integrity and identification. When the Tivoli NetView server receives a request for authentication, the server verifies the log in and stores
a security credential or ticket. This ticket enables shared-key security for each Tivoli NetView process when establishing a session. Shared keys are used to validate messages between session partners by checking incoming messages to ensure they are from the correct sender.

Log In Process

Users can log into Tivoli NetView using one of the following methods:

- **Log in** using the NetView Authentication Dialog by entering the `nvauth` command on the command line. The NetView Authentication Dialog is displayed, which contains input fields for the user’s login ID, group ID, and password.
- Log in from the command line using the `nvauth -login nvid nvgid` command. See the man page for more detailed information about the `nvauth` command.
- Start Tivoli NetView. If a user starts Tivoli NetView without first logging in, the NetView Authentication Dialog is displayed. Tivoli NetView is initialized after successfully logging in.
- Use the `Options..Shift_in` operation on the NetView Authentication dialog box menu bar. See [“Continuous, Auditable Network Management” on page 21](#) for a description of the Shift_in and Shift_out operations.

Users can log out using one of the following methods:

- Select the User Security... from the Tools pull-down menu to display the NetView Authentication dialog box. Then select the Options..Logout operation on the NetView Authentication dialog box menu bar.
- Enter the `nvauth -logout nvid` command on the command line.
- Exit Tivoli NetView. When a user exits Tivoli NetView, the user is automatically logged out.

Logging out stops all user-initiated Tivoli NetView processes.

Refer to the Tivoli NetView for UNIX User’s Guide for Beginners for more information about login and logout.

Log In Considerations

Keep these things in mind about the Tivoli NetView login process:

- Only one Tivoli NetView login from the same OS user ID is permitted. Subsequent Tivoli NetView users from the same OS user ID can start Tivoli NetView without having to login. The first user ID is the process owner for audit purposes.
- If you have a security product installed, such as DCE, you can customize your security policy so that a Tivoli NetView password is not required. Your security product manages the login procedure, allowing users to log in using a single password.

By default, Tivoli NetView requires a password unless you change this setting when you define the global security settings. See [“Defining the Global Security Settings” on page 32](#) for more detailed information.

- A user issuing any server-restricted operation from a client workstation does not have to be logged in as a root user and will not be prompted for the root password. The user must, however, have the appropriate security permissions and the same OS user ID on the server as on the client. Server-restricted operations are those operations that involve changing configuration files that exist on the server. These operations include:
– Event configuration (Event Configuration -> Trap Customization:SNMP... option from the Options pull-down menu or the xnsmtptrap command)
– SNMP configuration (SNMP Configuration... option from the Options pull-down menu or the xnmsnmpconf command)
– MIB data collection (Data Collection and Thresholds: SNMP... option from the Tools pull-down menu or the xnmcollect command)
– Loading and unloading MIBs (Load/Unload MIBs: SNMP... option from the Options pull-down menu or the xnmloadmib command)
– Polling interval changes (Topology/Status Polling Intervals: IP... option from the Options pull-down menu or the nmpolling command)
– Converting events to alerts (AIX only) (Event Configuration..Trap to Alert Filter Control: SNMP... option from the Options pull-down menu or the tralertdtfc command)
– Ruleset editor (Ruleset Editor option from the Tools pull-down menu or the nvrsEdit command)
– Security administration (Security Administration... option from the Administer pull-down menu or the nvsec_admin command)

Password Protection

The password, supplied when a user tries to log in, is encoded and compared with the encoded password stored in the security database. If the two match, the user gains access to Tivoli NetView. Passwords are not stored or sent over the network in human-readable format.

You set the user’s initial password, and you can change the user’s password at any time, when you create or change the user's profile.

See “Creating and Changing a User Profile” on page 28 for more detailed information.

The user can also change the password at any time by selecting the Operations..Change Password operation from the NetView Authentication dialog menu bar.

Refer to the Tivoli NetView for UNIX User’s Guide for Beginners for more information.

Continuous, Auditable Network Management

You might require that Tivoli NetView manage your network without interruption. Operators using the same physical display can use the Shift_in and Shift_out operations to accomplish uninterrupted network monitoring. To use the Shift_in and Shift_out operations, operators must be in the same Tivoli NetView group and use the same OS user ID.

After the first operator logs in, the operator uses the Shift_out operation at the end of the shift. The Shift_out operation activates a window lock, allowing Tivoli NetView processes to continue. The session is protected because only an authorized operator can remove the window lock. The next shift operator removes the window lock by clicking on the key displayed on the window lock screen to remove the window lock and then using the Operations.. Shift_in operation from the NetView Authentication Dialog box menu bar. Ownership of currently running Tivoli NetView processes changes to the new operator for audit and identification purposes without having to restart each process.
The operator can regain the screen saver before using the Shift_in operation by selecting the **Reset** button on the NetView Authentication Dialog box.

Refer to the *Tivoli NetView for UNIX User’s Guide for Beginners* for more detailed information about how to use the Shift_in and Shift_out operations.

**Network Access Control of Tivoli NetView Resources**

User permissions or rights to access Tivoli NetView resources are defined based on the Tivoli NetView group to which the user belongs. Each Tivoli NetView application provides a security registration file (SRF), which lists the application’s resources, such as menu items, commands, and tools. The SRF includes the valid permissions (read, write, execute, and so on) for the application’s resources.

Vendor applications can also register with security services to make use of and integrate with security services.

The *Tivoli NetView for UNIX Programmer’s Guide* provides detailed information about integrating an application with Tivoli NetView’s security application.

Each Tivoli NetView group is associated with a security registration files list. The security registration files list is a collection of all the Tivoli NetView applications that a group of users can access and the group’s access permissions for all the elements within each application.

Tivoli NetView provides the following preconfigured groups:

- **Oper** Intended for network operators. Users in this group can perform basic network monitoring tasks.

- **SrAdmin** Intended for system administrators or those with more network experience. Users in this group can perform all network monitoring tasks, including advanced network problem determination, configuration changes, and security administration.

When you [create a Tivoli NetView user profile](#) you specify the groups to which the user belongs. This process also enables you to create a new group, thus, defining a different set of access permissions.

See [Managing Tivoli NetView User Profiles](#) for more detailed information.

As an example how group permissions work, consider the following groups and their associated security registration files:

<table>
<thead>
<tr>
<th>User</th>
<th>Group</th>
<th>Security Registration Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>operator</td>
<td>Oper</td>
<td>ApplA ApplB</td>
</tr>
<tr>
<td>admin</td>
<td>SrAdmin</td>
<td>ApplA ApplB ApplC</td>
</tr>
</tbody>
</table>

The permissions defined for ApplA and ApplB are different depending on whether the user belongs to the Oper group or the SrAdmin group. The user, admin, in the SrAdmin group might have execute permission while the user, operator, in the Oper group might have read permission. A user in the Oper group does not have any access to ApplC; only a user in the SrAdmin group has permissions for ApplC.
Customized Tivoli NetView Graphical Interface

When a user logs into Tivoli NetView, the graphical interface presents only those functions and tools to which that user’s group has access. For example, if a group has no permissions set for a specific menu operation, the menu operation is not displayed for that group of users. If a group has read permission set for a specific menu operation, the menu operation is grayed out for that group of users.

A user can belong to more than one group, allowing you to define access control based on multiple user roles and levels of experience.

Audit Management

Security services collects audit data for the following security-related activities:

- Configuration changes. These include:
  - Event configuration.
  - Changes to polling intervals.
  - SNMP configuration.
  - Configuration changes defined by vendor applications.

  Refer to the Tivoli NetView for UNIX Programmer’s Guide for more information.

- Function access (element access)
- Login/logoff (includes Shift_in and Shift_out)

The audit data is stored in a log file, which you can review. This ability to review the audit data enables you to investigate any unusual activity that might indicate an attempted security breach. It also enables you to review normal usage patterns on your network and reliably trace security-related activities to a user, date, and time.

See "Reviewing Audit Data" on page 37 for information about how to view the audit data.

If you do not want to collect any audit data or if you do not want to collect data for all three categories, you can change this setting when you define the global security settings.

See "Defining the Global Security Settings" on page 32 for more detailed information.

Consistent Security Controls

You can configure your security policy on one server and distribute the same security configuration to other servers in your network. Distribution of a central security configuration provides consistent security controls and reduces security-related administration tasks. You identify your manager station as the distribution server when you define the global security settings.

See "Distributing the Security Configuration" on page 36 and "Defining the Global Security Settings" on page 32 for more detailed information.

Pager Service For Event Correlation Rulesets

You can define pager information when you create a Tivoli NetView user profile that is used to automatically issue a call to a pager. You do not need to turn security on to make use of the pager service.

When you create an event correlation ruleset that includes a paging action, you specify the Tivoli NetView user ID of the person to be paged. The Tivoli NetView
program looks for the pager information in that user’s profile. If the Tivoli NetView user profile does not exist when you create the ruleset, a dialog box is displayed in which you can enter the user ID and pertinent pager information, and the user profile is created.

See "Creating and Editing a Ruleset" on page 124 and "Managing Tivoli NetView User Profiles" on page 26 for more detailed information.

Defining a Security Policy

The following list describes the process you should follow to define your security policy:

1. Create a user profile for each Tivoli NetView user. You might also want to create new Tivoli NetView groups to create different sets of access permissions.
2. If you created new Tivoli NetView groups, define the access permissions for the new groups.
3. Define the following global security settings.
4. Test your security configuration.
5. Turn security on.

Define your security policy using the Security Administration dialog box as shown in Figure 3.

![Figure 3. Security Administration Dialog Box](image)

Accessing the Security Administration Dialog Box

By default, you must be a root user to access the Security Administration dialog box. When security is turned on, you must also be an authenticated Tivoli NetView
user and log in with a valid Tivoli NetView user and group ID. Tivoli NetView permission is required to execute the Security Administration option from the Administer pull-down menu (nvsec_admin command). The SrAdmin group is preconfigured with permission for security administration.

You can change operating system file permissions so that non-root users can execute security administration. To do so, follow these steps as a root user on the Tivoli NetView server that you want to use for security administration:

Note: Umask must be equal to 022 only for a root or non-root user that sets up security using nvsec_admin.

1. Create a new operating system user group or use an existing operating system user group. The remaining steps use a group named secadmin.
2. Set group permissions for the nvsec_admin command as follows:
   ```
   chgrp secadmin /usr/OV/bin/nvsec_admin
   chmod g+x /usr/OV/bin/nvsec_admin
   ```
3. Set security operating system file permissions as follows:
   ```
   chgrp -R secadmin /usr/OV/security
   chmod g+rw /usr/OV/security
   chmod -R g+r /usr/OV/security/C
   chmod g+rwx /usr/OV/security
   chmod g+rwx /usr/OV/security/conf
   chmod g+rw /usr/OV/security/conf/sec.conf
   chmod -R +rwx /usr/OV/security/C
   ```
4. To execute security administration as a non-root user from a client machine, define the same OS user ID on the server machine.
5. Add the security administration group, for example, secadmin, to the groups set for root.

To access the Security Administration dialog, enter the /usr/OV/bin/nvsec_admin command at the command line. When security is turned on, you can also access the Security Administration dialog box by selecting Security Administration from the Administer pull-down menu.

If you want the Security Administration option to be selectable when security is turned off, edit the /usr/OV/registration/C/nvauth file and remove the Security ; line from the "nvsecadmin" Action clause.

**Description of the Security Administration Dialog**

The Security Administration dialog box contains three sections:

**Users**
Lists all users known to Tivoli NetView and the groups to which they belong. Two sample user IDs are provided: operator and admin. The passwords are the same as the user IDs: operator and admin, respectively.

Note: You should change the passwords for the sample user IDs before you turn security on.

You can use the buttons to add, change, delete, and view user profiles. You can also query logged on users.

**Groups**
Lists all groups known to Tivoli NetView and their associated security
registration files. Initially, the Oper and SrAdmin groups are listed. You can use the buttons to add, change, copy, delete, and view group permissions.

**Security Registration Files**
Lists all applications registered with security services. Select an application in the list and then select the **View** button to view the application’s security registration file.

**Note:** You can view only one security registration file at a time. If you have selected more than one security registration file, the **View** button is not available.

Refer to the *Tivoli NetView for UNIX Programmer’s Guide* for information about security registration file syntax.

The menu options available when you select **Options** on the Security Administration menu bar enable you to define global security settings and perform other administrative tasks, such as, querying all logged in users, sending messages to logged in users, forcing off a logged in user, viewing audit reports, and distributing the security configuration to other servers in your network.

### Managing Tivoli NetView User Profiles

Security services authenticates each user who attempts to log into Tivoli NetView and permits access based on the information in the user’s profile. The user’s profile includes login information, such as, the Tivoli NetView user ID, the Tivoli NetView groups to which the user belongs, and the user’s password.

The groups to which the user belongs define the user’s access permissions. When you create a Tivoli NetView user profile, you can add the user to an existing group. To create a unique set of access permissions for a user, you can create a new group. You can update a user’s profile any time, for example, to add the user to another group or include additional login controls, such as permissible login times.

Use the User dialog box as shown in [Figure 4 on page 27](#) to perform the following types of tasks:

- “Creating and Changing a User Profile” on page 28
- “Viewing a User Profile” on page 29
- “Deleting a User Profile” on page 29
- “Adding and Changing a Group” on page 29
- “Copying a Group” on page 31
- “Deleting Group Permissions” on page 31
- “Viewing a Group’s Permissions” on page 31
- “Deleting a Group” on page 31
Description of the User Dialog Box

The User dialog box contains three sections. The top section contains general user profile information and contains the following fields and buttons:

**NV User ID**
Mandatory field that specifies the user’s Tivoli NetView login ID.

**NV Group IDs**
Mandatory field that specifies the group or groups to which the user belongs. Type or click on the arrow button next to this field to specify one or more group names. Separate multiple group names with a comma.

**Client List**
Optional field that specifies the client host names from which the user can log in. The asterisk (*) in this field indicates permission from all clients. Specify one or more host names to restrict login from the specified clients. Separate multiple host names with a comma.

**Manager List**
Optional field that specifies the management stations to which this user can log in. The asterisk (*) in this field indicates permission to log into any server. Specify one or more host names to restrict login to the specified servers. Separate multiple host names with a comma.
Last Profile Update
Informational field that indicates when the user profile was updated.

Set Password
Optional setting to set the user’s password. You can change a user’s password any time without knowing the user’s current password. When you select the Set Password button, a password dialog is displayed. Enter the user’s password in both password fields and select the Ok button. If the password matches in both password fields, the password is set and the dialog box closes.

If you do not set the user’s password, a default password is assigned. The default password is set to the same string as the user ID. For example, if you are creating a user profile for pamb and do not set a password, the password is set to pamb.

Set Pager Information
Optional setting to specify the user’s pager information. You can create an event correlation ruleset that automatically issues a call to a user’s pager. When you create a ruleset that includes a paging action, you specify the Tivoli NetView user ID of the person to be paged. The Tivoli NetView program looks for the pager information in the Tivoli NetView user’s profile.

See “Creating and Editing a Ruleset” on page 124 for more information.

When you select Set Pager Information, a dialog box is displayed. Enter the user’s numeric or alphanumeric pager ID in the appropriate field and use the arrow button to select the carrier name. Click OK to close the dialog box.

Last Password Update
Informational field that indicates when the password was last updated.

User IDs, Group IDs, and passwords can be from 1 to 8 characters long and can consist of the letters a through z and A through Z in addition to the following characters:
. , ; ( ) ' / - _ & + %= <>

Note: Some of the allowed characters have special meaning to the operating system. Avoid using those characters that have special meaning to the operating system, especially at the beginning of an ID or password. For example, the < can be interpreted as input redirection and can cause errors when logging in from the command line.

Use the Permissible Login Time section if you want to restrict user login to a specific time and specific days. Type or use the arrow keys to set the appropriate start and end times and use the toggle buttons to select the appropriate days of the week. The default setting permits the user to log in any time on any day of the week.

Use the Notes section if you want to add comments to the user profile, such as the user’s name and phone number.

Creating and Changing a User Profile
Use this procedure if the groups to which the user belongs are already created. If you want to create a new group for a user, use the procedure described in “Adding and Changing a Group” on page 29 or “Copying a Group” on page 31.
Follow these steps:

1. **Access the Security Administration dialog box.**
2. Do one of the following:
   - If you are adding a user profile, click **Add** which is next to the Users section.
   - If you are changing a user profile, select the appropriate user in the Users section. Select **Change**.

   The User dialog box is displayed as shown in Figure 4 on page 27.
3. Make the appropriate changes to the dialog box. Use the online help if you need information about the dialog box fields. Select **OK**.
   The User dialog box closes, and the new user is added to the Users section of the Security Administration dialog box. The user has the permissions associated with the groups to which the user belongs.
4. Repeat steps 2 through 3 for each user profile you want to add or change.

**Viewing a User Profile**
You can view user profile information by selecting the appropriate user ID in the Users section of the Security Administration dialog box. Then select **View**.

**Deleting a User Profile**
To delete a user profile, select the appropriate user from the Users section of the Security Administration dialog box and then select the **Delete** button. Select **Delete** on the Delete Confirmation box to confirm this action.

**Adding and Changing a Group**
Adding a group enables you to customize access permissions for a particular user or set of users. You can also add a group by **copying** an existing group’s profile.

Minimally, you need to enable permission to run the graphical interface by setting read and execute permission for the ovw_binary and File SRFs. If you want to create a group that has minimal Tivoli NetView security permissions (permission to run the graphical interface), you might find it useful to copy the Oper group and exclude permissions to SRFs that you do not want the group to access. See “**Copying a Group**” on page 31 for those steps.

Changing a group enables you to customize access permissions for all the users who belong to the group. When a new application registers with security services, for example, you would change the groups that should have access to the new application, defining the permissions each group should have. If you changed a group’s permissions, users in that group will not get the change until they log off and then log back on again.

To create or change a group, follow these steps:

1. **Access the Security Administration dialog**
   If you are changing a group, go to step 4 on page 30. If you are creating a group, continue to the next step.

2. Add a user to the group using one of the following methods:
   - If the user profile is already created, select the appropriate user in the Users section of the Security Administration dialog box. Then select **Change**.
   - To create a user profile, select **Add**.

   The User dialog box is displayed as shown in Figure 4 on page 27.
3. Make the appropriate changes to the dialog box fields, adding the new group name to the NV Group IDs field. Then click **Ok**. Refer to the online help for detailed information about these fields.

The User section of the Security Administration dialog box is updated with the information, and the group name is added to the Groups section. At this point, the group does not have any access permissions.

4. Select the group name in the Groups section of the Security Administration dialog box. Then click **Add/Chg**.

The Add/Change Group Security Registration dialog box is displayed.

5. Click the arrow button next to the Applications field and select every security registration file and click **OK**.

**Note:** By default, if you do not select a security registration file, users have access to the elements defined in that SRF and those menu options will be displayed.

6. Click **OK** on the Add/Change Group Security Registration dialog box.

An Add/Change Group Security Registration dialog box is displayed that contains a list of the elements of each application.

![Add/Change Group Security Registration Dialog Box](image)

**Figure 5. Add/Change Group Security Registration Dialog Box**

7. Click on the permissions box.

The Element Permissions dialog box is displayed.

8. Select the appropriate permission for the element and select **OK**:

   - Select r to make the element unavailable. Menu options are displayed but are greyed out.
   - Select rx to make the element available. Menu options are displayed and are available, and command line executables are available.
   - Do not select any permissions to make the element unavailable. Menu options are not displayed.

**Note:** In general, write permission (w) is not used. Write permission is used only if an application’s SRF file includes w as a valid permission.
9. Select **Propagate** to propagate the permissions to all sub-elements (lower level menu options) or individually set the permissions for each sub-element as described in step 8 on page 30.

10. Select **OK** on the Add/Change Group Security Registration dialog box.
    The group profile is updated with the permissions that you set.

**Copying a Group**

If you want to create a group based on an existing group's profile, you can copy a group. Then change the group to meet your requirements.

To copy a group, follow these steps:
1. Access the Security Administration dialog box.
2. Do one of the following:
   - Add or change a user profile, entering the new Tivoli NetView group name in the Tivoli NetView Group IDs field. See "Creating and Changing a User Profile" on page 28 for those steps. Then select the group name in the Groups section of the Security Administration dialog box and click **Copy**.
   - Click **Copy** to copy a group without first adding a user to the group.

The Copy Group Profile and Permissions dialog box is displayed.
3. Use the arrow button next to the Source Group field to select the group name that you want to copy.
4. Enter the name of the new group in the Target field and click **OK**.
   The group you have just created has the same permissions as the group you copied.
5. **Change the group** to meet your requirements.
   See "Adding and Changing a Group" on page 29 for those steps.

**Deleting Group Permissions**

To delete group permissions, select the group name in the Groups section of the Security Administration dialog box, select the appropriate security registration file in the Security Registration Files Section, and select the **Delete** button. The security registration file you selected is deleted from the group's security registration files list.

**Viewing a Group’s Permissions**

To view the permissions set for a group, select the group name in the Groups section of the Security Administration dialog box. Then select the appropriate security registration file in Security Registration Files section and click **View**.

**Deleting a Group**

Before you can delete a group, you must remove all users from the group. To do so, update the appropriate users' profiles, deleting the group name from the NV Group IDs field. See "Creating and Changing a User Profile" on page 28 for those steps. When you delete the last user in the group, a delete confirmation dialog box is displayed. Click **Delete** to delete the group profile.

**Setting User Environment Variables**

You can set the environment variables, NVID and NVGID, in each user's .profile or .kshrc file so that the user's login ID and group ID are automatically displayed in the Authentication login panel.
For example, if Mary’s login ID is mary in the SrAdmin group, add the following lines to Mary’s profile:

```bash
export NVID=mary
export NVGID=SrAdmin
```

### Defining the Global Security Settings

To define the global security settings, use the Global Settings dialog box as shown in Figure 6.

![Global Settings Dialog Box]

**Figure 6. Global Settings Dialog Box**
Description of the Global Settings Dialog Box

The Global Settings dialog box contains the following fields and buttons:

**Security Level**
- Use the **Change** button to select one of the following settings:
  - **ON**: Turns security on for all users.
  - **OFF**: Turns security off for all users and is the default setting.
  - **TESTMODE**: Enables you to test your initial security configuration (before turning security on). Enter an OS user ID and server host name in the Test Mode ID field. Use the server host name that the operating system ID is logging into to test security. (The server host name is optional, the default is the server name you are using to change the TESTMODE global setting). You can log in as any Tivoli NetView user in any Tivoli NetView group from this OS user ID to test the configuration. Security is off for all users except for the test mode ID.

Because TESTMODE has the same effect as turning security off, test future configuration changes by logging on with a Tivoli NetView ID in the Tivoli NetView group you want to test.

**Is Distribution Server**
- Specifies your Tivoli NetView server as the security distribution server.
- When you select this button, you can distribute the security configuration to other servers in your network.

**Prompt for Login**
- Specifies a password prompt when logging into Tivoli NetView and is the default setting. If you have a third-party security product installed, you can turn this setting off so that your security product manages the login procedure, allowing users to log in using a single password.

**Select Audit Categories**
- Specifies the audit data you want to monitor. By default, audit data is collected for the following audit categories:
  - Configuration Changes
  - Function Access
  - Login/Logoff

- Data is logged in the log file you name in the Audit Log File Name field. You can review the audit data by using the Options..Audit Report operation from the Security Administration dialog box menu bar.

**Note**: No other users can be logged into Tivoli NetView when you change the security level. If other users are logged in, a message will be displayed. See "Managing Logged In Users" on page 35 for information about how to query who is logged on and communicate with logged-in users.
Audit Log File Name
Specifies the full path name of the audit log. The default audit log is the /usr/OV/log/sec_audit_log file.

Number of Audit Files
Specifies the number of backup audit log files to keep; the default is 3. When the audit log file reaches the size specified in the Maximum Audit File Size field, it is moved to a backup file in the directory that is specified in the Temporary Directory field. The naming convention for the backup audit log files is as follows:

log_file_name.timestamp

Where log_file_name is the name of the audit log file specified in the Audit Log File Name field, and timestamp indicates the date and time when the audit log file was backed up. The backup process continues until the specified number of audit files is reached. Then, the oldest backup audit log file is removed when the current audit log file is backed up.

Maximum Audit File Size
Specifies the maximum audit log file size in megabytes; the default is 30 MB.

Temporary Directory
Specifies the full path name of the directory in which to store the backup audit log files.

Steps
To define the global security settings, follow these steps:
1. Access the Security Administration dialog box
2. Select Global Settings... from Options pull-down menu on the Security Administration dialog box.
The Global Settings dialog is displayed.
3. Make the appropriate changes to the dialog box.
4. Click Apply to apply the changes and close the Global Settings dialog box.

Changing the Defaults for Interaction with NetSP
The /usr/OV/security/kkcfg file contains default settings that enable Tivoli NetView security services to interact with the IBM Network Security Program (NetSP). The following entries in the /usr/OV/security/kkcfg file enable Tivoli NetView authentication and verification:

comdname=/tmp/.krypto_as_req
cacheroot=/tmp/NetSPkt

You should change these entries to reference a directory other than the /tmp directory, such as /usr/OV/tmp to ensure that security information is not inadvertently deleted. For example, some organizations run cron jobs that periodically delete all the files in the /tmp directory. Change the default settings in the /usr/OV/security/kkcfg file as follows:

comdname=/usr/OV/tmp/.krypto_as_req
cacheroot=/usr/OV/tmp/NetSPkt
Managing Security

This section describes the administrative tasks you can perform that can help you enforce and manage your security policy. The following tasks are described:

- "Managing Logged In Users"
- "Distributing the Security Configuration" on page 36
- "Reviewing Audit Data" on page 37

Managing Logged In Users

You might find it necessary to send administrative messages, find out who is logged on, what processes are running, or force off a logged in user. For example, if a process is consuming system resources, you can determine who is running that process by querying logged in users. You can also send any administrative message to a logged in user.

Suppose you updated a group’s permissions for a new application. Users in that group will not get the change until they log off and then log back on. You might take steps similar to the following:

1. **Access the Security Administration Dialog.**
2. Determine who is logged in using one of the following methods:
   - Select Query All from the Options pull-down menu on the Security Administration dialog box.
   - Select one or more users from the Users section on the Security Administration dialog. Then click Query.

   The Logged In Users panel is displayed containing a list of logged in users.

3. Send a message to the appropriate user informing the user to log off and then log back on to access the new application.
   - a. Select the appropriate user on the Logged In Users dialog box and then click Message.
      A message dialog is displayed.
   - b. Type the appropriate message and click Send.
4. If the user does not respond, you can log the user off by selecting the user ID on the Logged In Users panel. Then click Logoff.
The user is logged off, and a message is displayed informing you that this action is completed.

Distributing the Security Configuration

To distribute your security configuration, you must have defined your server as the distribution server on the Global Settings dialog box. See "Defining the Global Security Settings" on page 33 for those steps.

You can distribute your entire security configuration or selected file sets, depending on the configuration changes you have made. Follow these steps:

1. Access the Security Administration Dialog box.
2. Select Distribution from the Options pull-down menu on the Security Administration dialog box.
   The Security Distribution dialog box is displayed.
3. The section labeled Target NetView Server lists the managers that have been discovered. Select from the list or type the target server host names.
4. Select one or more file sets.
   The Users file set contains user profiles, the groups file set contains group profiles and group permissions, and the Security Registration file set contains the SRFs for each application registering with security services. After initially configuring your security policy, distribute all the file sets. Thereafter, when you make configuration changes, you can distribute only the file sets to which you made changes.
5. Select Send.
   The status window displays the transmission status for each target server, and the dialog box closes.

Security configuration files on the target servers are automatically backed up when you distribute new configuration files should you need to restore the prior configuration.

Figure 8. Security Distribution Dialog Box

3. The section labeled Target NetView Server lists the managers that have been discovered. Select from the list or type the target server host names.
4. Select one or more file sets.
   The Users file set contains user profiles, the groups file set contains group profiles and group permissions, and the Security Registration file set contains the SRFs for each application registering with security services. After initially configuring your security policy, distribute all the file sets. Thereafter, when you make configuration changes, you can distribute only the file sets to which you made changes.
5. Select Send.
   The status window displays the transmission status for each target server, and the dialog box closes.

Security configuration files on the target servers are automatically backed up when you distribute new configuration files should you need to restore the prior configuration.
Each backup directory contains a time stamp (represented by \textit{ddmmyytime}) indicating the day, month, year, and time when the directories were backed up. A list of the file sets and the backup directory for each on the target server is as follows:

\textbf{File Backup Directory}

\begin{itemize}
  \item \textbf{Users} /usr/OV/security/$LANG/Users.ddmmyytime
    
    If you want to restore this directory, copy the directory to the /usr/OV/security/$LANG/Users directory.
  
  \item \textbf{Groups} /usr/OV/security/$LANG/Groups.ddmmyytime
    
    If you want to restore this directory, copy the directory to the /usr/OV/security/$LANG/Groups directory.
\end{itemize}

\textbf{Security Registration}

Application SRFs:

/\textit{usr/OV/security/$LANG/Domains.ddmmyytime/registration}

If you want to restore this directory, copy the directory to the /usr/OV/security/$LANG/Domains/registration directory.

Group SRFs (group permissions)

/\textit{usr/OV/security/$LANG/Domains.ddmmyytime/groupname}

If you want to restore this directory, copy the directory to the /usr/OV/security/$LANG/Domains/groupname directory, where \textit{groupname} represents the name of the Tivoli NetView group.

\section*{Reviewing Audit Data}

To view or print the audit data you are collecting, follow these steps:

1. Select \textbf{Audit Report...} from the \textbf{Options} pull-down menu on the Security Administration dialog box.
   
   The Security Audit window is displayed.

2. View the reports, using one of the following methods:
   
   \begin{itemize}
     \item To view all audit reports:
       
       Select \textbf{All} from the \textbf{View} pull-down menu.
     
     \item To view specific data:
       
       \begin{itemize}
         \item Select \textbf{By Criteria} from the \textbf{View} pull-down menu. Then select \textbf{Set Criteria} from the \textbf{View} pull-down menu.
       
       The Set View Criteria dialog box is displayed.
   \end{itemize}
\end{itemize}
b. Enter the search criteria and click **Apply**.

The Open Audit File dialog box is displayed containing a list of directories that contain audit files.

When you select one or more directories, the list of files contained within those directories is displayed in the Files section.

---

**Figure 9. Security Audit: Set View Criteria Dialog Box**

---

---
3. Select the files you want to view and click **Open**.

You can also open, print, or save a file using the operations available when you select **File** on the Security Audit window.

### Converting ARFs to SRFs

To convert existing Tivoli NetView application registration files (ARF) to Tivoli NetView security registration file (SRF) syntax for registering sensitive security resources, use the `/usr/OV/bin/c_arf2srf` command. See the `c_arf2srf` man page for more information.

Refer to the *Tivoli NetView for UNIX Programmer’s Guide* for information about SRF syntax and integrating applications with security services.

### Verifying Security Permission For Shell Scripts

If you use shell scripts to run Tivoli NetView executables, use the `/usr/OV/bin/vfy_access` command to verify security permission. If you have a shell script named `myscript` that invokes the `ovobjprint` command, add the following line to the script:

```
vfy_access ovobjprint
```

You can specify one or more Tivoli NetView executables with the `vfy_access` command. See the `vfy_access` man page for more information.
Chapter 3. Creating and Customizing Submaps

This chapter describes the major components of the graphical interface and explains how these components work together to help you monitor and manage your network. This chapter also describes how to create a customized submap hierarchy and how to group together objects in your network. These tasks can help you monitor and manage your network more effectively.

The following topics are described in this chapter:

• "Objects" on page 46
• "Symbols" on page 51
• "Maps" on page 54
• "Submaps" on page 54
• "Using Tivoli NetView Applications" on page 57
• "Customizing a Graphical Map" on page 61
• "Defining and Managing SmartSets" on page 66

Objects

An object is an internal representation of a logical or physical entity or resource that exists somewhere in a computer network. An object is made up of a set of fields that specify all the characteristics of the object. Examples of resources represented by objects include:

• A computer node
• A software process on a computer
• An IP network

Most of the objects discovered and displayed by Tivoli NetView are network objects. However, objects can also be created by users or by applications and integrated with the Tivoli NetView program.

For information about how users can add objects, see "Adding Symbols and Objects" on page 61. For information about how applications can add and manipulate objects, refer to the Tivoli NetView for UNIX Programmer’s Guide.

Displaying the Object Database

Object and attribute information is stored in the object database. You can display the contents of the Tivoli NetView object database with the ovobjprint command.

For more information about this command, refer to the man page.

Symbols

A symbol is a graphical representation of an object as it is displayed on a submap of a particular map. Symbols are presentation elements; objects are underlying database elements that describe network entities like workstations, networks, and interface cards. Several symbols can represent the same object, even when the symbols are on different submaps.

Even though symbols represent objects, symbols can have some additional characteristics beyond those of the object they represent. These characteristics, or attributes, can vary among the different symbols representing a particular object.
Defining Symbol Characteristics

The following list describes characteristics of symbols in the Tivoli NetView graphical interface:

Symbol type
The symbol type consists of the symbol class, which specifies the outer shape of the symbol, and the symbol subclass, which specifies the graphic shown within the shape.

The Tivoli NetView program provides a variety of predefined symbols. To see the symbol types provided by the Tivoli NetView program, select Help..Legend from the Tivoli NetView main menu. If necessary, you can define new symbol types using Symbol Type Registration Files. Refer to the Tivoli NetView for UNIX Programmer’s Guide for more information.

Symbol variety
A symbol can be an [icon symbol], which is a two-dimensional picture, or a [connection symbol], which is drawn as a line connecting two symbols. It is important to understand the difference between a connection symbol, which is a line that often represents an interface card object, and a connector symbol, which is a diamond shaped icon symbol that can represent a device like a bridge or router.

Symbol location
A symbol can reside on either the application plane or the user plane of a submap.

Submap planes are described in "Understanding Submap Planes" on page 55.

Symbol behavior
Symbol behavior defines what happens when you double-click on the symbol.

Explodable Symbols
When you double-click an explodable symbol, a child submap is opened. This is the default behavior for symbols.

Executable Symbols
When you double-click an executable symbol, the program represented by that symbol is started.

An executable symbol is displayed as a raised button on the submap. To see an example of an executable symbol, select Help..Legend from the Tivoli NetView main menu.

Symbol label
Each symbol has a label that describes the object represented by the symbol. The label is displayed below the symbol. Because the Tivoli NetView program does not use the symbol label to identify the symbol, the symbol label does not have to be unique. You can choose whether or not to display the label of a symbol.

See "Modifying and Displaying Symbol Labels" on page 91 for steps on displaying symbol labels.

Symbol status
Symbols can display information about the status of the object or connection that the symbol represents.

Colors are used to represent status information, which is described in the Tivoli NetView for UNIX User’s Guide for Beginners.
Indicating Symbol Status

Symbol status conveys the current state of a network entity based on a predetermined set of rules. The graphical interface indicates the status of a symbol by its color on a submap. The status displayed by the symbol stems from the state of certain attributes of the object that the symbol represents. If the symbol represents a container object, for example an Internet symbol, its status color represents the combined status of all the symbols contained within it. See "Understanding Compound Status Source" on page 44 for information about compound status.

The ipmap application determines the status of a node object based on the operational status of all of the IP interfaces installed in the node. If all of a node’s IP interfaces are down, the ipmap application reflects the node’s status as critical. From the perspective of the ipmap application, the node is nonfunctional. However, another application might consider the same node to be running, because that application monitors a different protocol with fully functional interfaces. This is an example where you might have different symbols representing different states of one object.

The following list summarizes the status source used to determine status for the various symbols:

**Object Status**
- Interface card symbols and connection symbols that correspond to interfaces (not networks) derive their status by object status source.

**Symbol Status**
- All node symbols (connectors, servers, and computers) on network and segment submaps derive their status by symbol status source.

**Compound Status**
- All other symbols on your IP submaps derive their status by compound status source. These symbols include, but are not limited to:
  - All network and segment symbols
  - All location and internet symbols
  - All node symbols on the internet and location submaps

You can determine or change the status of a symbol by selecting **Edit Modify/Describe Symbol** from the object context menu.

Each status source is described in the following sections.

**Note:** If a symbol has object status source or symbol status source and users have configured events as status events, the Event Display application can change the symbol’s status. In addition, other applications can change the status of a symbol.

Refer to the *Tivoli NetView for UNIX Programmer’s Guide* for more information.

By default, if a symbol has compound status source, the symbol’s status is updated by the ipmap application. To enable the Event Display application to update a symbol’s status if the status source is compound, change the value for the overrideCompoundStatus resource in the /usr/OV/app-defaults/Nvevents file to TRUE. When the value for the
overrideCompoundStatus resource is TRUE, the Event Display application updates all symbols for the object without having to manually change the status source for each symbol.

**Understanding Object Status Source**
Object status source determines the symbol’s status based solely on the status of an individual object in the object database. If the interface card object in the object database is down, then the status is reflected as critical. If the interface card object is up, then the status is reflected as normal. The status of the interface card symbol is based on that card object only.

**Understanding Symbol Status Source**
Symbol status source determines the symbol’s status based on some algorithm different from object status source and compound status source. The algorithm is contained within the application that manages the symbol. For example, if an IP symbol has symbol status, then the ipmap application determines with its own algorithms what the status of the symbol should be. By default, the ipmap application uses symbol status for the following symbols:
- Any node symbol (connectors, servers, and computers) on a network submap
  The status of the connector is based on the combined status of only the interface cards that the connector has in that network, rather than the combined status of all of the interfaces in the connector.
- Any node symbols (connectors, servers, and computers) on segment submaps
  The status of the node is based on the combined status of only the interface that the node has in that segment, rather than the combined status of all of the interfaces in the node.

Some HP hubs, however, have status propagation rules because these HP hubs contain non-IP interface cards that do not report their status. The status propagation rules for these HP hubs are:
- On network submaps, the status of the hub symbol is based on the combined status of only the IP interface cards in that network.
- On segment submaps, the status of a hub symbol is determined by compound status rather than symbol status.

**Understanding Compound Status Source**
Compound status source determines how status values are propagated from symbols in child submaps to the symbol on the parent submap that represents the child submap and is based on the combined status of all of the symbols in the child submap of that symbol. For example, the status of a segment symbol is based on the combined status of all of the node symbols in the child submap of that segment.

The Tivoli NetView program uses the following rules to determine the combined status of a group of symbols:
- Default compound status
- Propagate most critical compound status
- Propagate at threshold value compound status

**Default Compound Status:** The default compound status scheme determines compound status as follows:
A symbol’s status is: If the symbols in the child submap meet these criteria:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>All symbols are either normal, acknowledged, UserStatus1, unmanaged, or unknown and at least one symbol is normal, acknowledged, or UserStatus1</td>
</tr>
<tr>
<td>Critical</td>
<td>At least one symbol is critical or UserStatus2 and no symbols are normal, acknowledged, or UserStatus1.</td>
</tr>
</tbody>
</table>
| Marginal | Any of the following:  
  - At least one symbol is normal, acknowledged, or UserStatus1 and at least one symbol is marginal, critical, or UserStatus2.  
  - All symbols are marginal, unmanaged, or unknown and at least one symbol is marginal. |
| Unknown | All symbols are unknown or unmanaged. |

A symbol with compound status source will never have its status changed to UserStatus1, UserStatus2, or Unmanaged as a result of compound status propagation.

**Propagate Most Critical Compound Status:** The propagate most critical status scheme causes the graphical interface to propagate the status of the most critical symbol in the child submap to the submap’s symbol in the parent submap.

**Propagate at Threshold Value Compound Status:** The propagate at threshold value (0%–100%) scheme enables you to set two threshold values, marginal and critical. The marginal threshold value determines when a symbol’s status changes from normal to marginal. The critical threshold value determines when a symbol’s status changes from marginal to critical.

- Conditions for marginal status:
  - Marginal Threshold Value < % (Marginal + Critical) < Critical Threshold Value
  - If the percentage of symbols, that are either marginal or critical, in the child submap of a parent object is greater than the marginal threshold value and less than the critical threshold value, then the parent object’s symbols that have compound status source will be marginal.
  - If the percentage of symbols, that are either marginal or critical, in the child submap of a parent object is less than the marginal threshold value, then the parent object’s symbols that have compound status source will be normal.

- Conditions for Critical Status:
  - Critical Threshold Value < % (Marginal + Critical)
  - If the percentage of symbols, that are either marginal or critical, in the child submap of a parent object is greater than the critical threshold, then the symbols of a parent object that have compound status source will be critical. If the critical threshold value is less than the marginal threshold value, then symbols that have compound status source will change directly from the normal state to the critical state.

**Setting Compound Status:** If you have a map open with read-write authorization, you can change the default compound status scheme on the current map or when you create a new map. The status setting applies to the entire map. You can’t set compound status scheme for individual submaps.

When you create a new map, you can set the compound status by selecting one of the following Compound Status buttons on the New Map dialog box:
Maps

A map is a collection of objects stored in a map database that describe a set of network entities. They are displayed by the graphical interface for Tivoli NetView. The graphical interface uses these objects to draw icon symbols and connection symbols on the map’s submaps. A submap is a view, or window, that displays information stored in the map database. You do not view a map directly; instead, you view submaps that contain symbols representing objects within the map. Submaps are described in "Submaps" on page 51.

You can create, delete, or choose a map to be displayed from existing maps. You can even create several maps and control which applications operate on these maps. While you create maps and define their scope, applications dynamically update maps to reflect the state of the management environment.

Although you can create several maps, the graphical interface can have only one map open at a time. The open map is the map currently displayed by the graphical interface. This map can be updated by applications and users.

To view multiple maps simultaneously, invoke multiple instances of the graphical interface (ovw application). If there are multiple simultaneous ovw sessions with the same map open, only one session can have read-write access to the map.

Information in the Map Database

If you have a read-write map, select Modify/Describe -> Map... from the Edit pull-down menu to change information about the map. Each map has the following attributes:

Name
The name of the map, which is assigned when the map is created. Each name must be unique. You can change the map name.

Root submap
The highest-level submap of the map. The root submap cannot be deleted.

Home submap
The submap that is displayed in the initial submap window when you open the map. You can assign any submap of the map as the home submap. The root submap is the default home submap.

Layout algorithm for root submap
The layout algorithm used for the root submap. The default is row/column. Once set, the layout algorithm cannot be changed.

Compound status scheme
The compound status scheme that applies to the entire map. This scheme determines how the graphical interface propagates status from symbols in child submaps to the symbol of the parent object. You can change the compound status scheme for a map. You cannot change it for only a submap.
Configurable applications
Any configurable map applications that are available on your system for that map. You can enable or disable these when you create a new map.

Comments
Any comments or notes about the map. This entry can be used to document the map’s creation date, purpose, or other information you want to store.

Managing Maps in a Distributed Network Environment
In a distributed network environment (client/server), the Tivoli NetView daemons run on the server, and the graphical interface applications run on the client. The client obtains event and topology status information from the daemons running on the server. A client/server configuration enables you to distribute the CPU and memory requirements to the client. You can use several clients, enabling you to divide the management tasks among more operators at one time.

The Tivoli NetView graphical interface application maintains the map database. In other words, a client application maintains the map database. Whether the map database resides on the server or the client depends on how the client was configured. The client can be configured to NFS mount the map database from the server, or it can be configured to store the map database locally.

Refer to Tivoli NetView for UNIX Installation and Configuration for more information.

Each type of configuration has certain advantages, and each presents different network management considerations.

When the Map Database Is NFS Mounted
When the map database is NFS mounted, maps created on the client are stored through NFS on the server, and all users view the same set of map information. This is consistent with previous versions of Tivoli NetView regarding map information. The tradeoff is network performance. Retrieving map database information from the server depends on network speed, bandwidth, and the size of the discovered network. Some map operations will be slower than if the maps were stored locally.

Improving Network Performance for an NFS-Configured Client: To decrease the impact on network performance due to retrieving map database information, use the following methods:

• Create new maps on the server rather than on the client. When the map is fully created, it can be utilized from the client. If you create the map on the client, the client contacts the daemons on the server to determine the network topology and then, through NFS, writes the information back to the server to create the map, symbols, and submaps. Creating maps on the server minimizes the amount of network traffic required to create the map.

• Investigate using read-only maps on the client. NFS write operations will not be performed, and NFS caches read operations, which improves network performance.

Updating the Host Name For NFS Mounted Maps: Because the map names and their host name locations are stored in the object database, if you change the host name of the server, you must update the host name in the object database so you can access the maps.
To update the host name of the server when all maps are on the same machine (NFS mounted), use the following command:

```bash
mapadmin -u newhostname
```

Replace `newhostname` with the new name of the server.

See the mapadmin man page for more information.

**When the Map Database Is Stored Locally**

When the map database is stored locally (on the client), better performance is obtained in retrieving map database information. However, because each client can have its own set of maps, more map administration is required. Consider the following:

- Changes you make to maps on the server are not reflected in maps on the client. If you want to make a change to all the maps, such as deleting a router, you must delete the router from each of the client maps.

- Restarting topology discovery on the server does not affect maps that may exist on the client. If the map on the client is out of date because the server’s discovery has been restarted, the map is invalid and cannot be opened. You should delete invalid maps on the client. See “Deleting a Map that Is Not Valid from the Client” for more information.

Map names must be unique across the server’s domain, because objects get stored in the object database based on a map’s name. If a user tries to create a map with a map name that already exists on another client or on the server, the user is prompted to choose a different name.

In addition, clients and servers need to be configured with similar host name resolution procedures. Clients need to be able to resolve host name servers and other clients. If a domain name server (DNS) is used for a client or server, then DNS should be used for host name configuration for all clients and servers. Otherwise, a client can create a map and have its DNS-configured name recorded as the map owner, but a non-DNS client cannot resolve the host name.

**Deleting a Map that Is Not Valid from the Client:** If a map that is stored locally on the client is out of date with the map on the server, the map is invalid and cannot be opened. To delete the map that is not valid, use the following command:

```bash
mapadmin -r mapname
```

Replace `mapname` with the name of map that you want to delete.

You can use the `mapadmin -l` command to list all maps on the client. See the mapadmin man page for more information.

**Updating the Host Name For Local Maps:** Because the map names and their host name locations are stored in the object database, if you change the host name of the server or the client, you must update the host name in the object database to access the maps.

To update the host name when maps are stored on multiple machines (clients), you can use the `ovwls` command to list all maps that exist and where they are located. Then use the `mapadmin -u mapname:newhostname` command to update the host name.

Assume that the `ovwls` command shows the following maps and locations:
If you changed the host name for cs.networking.tivoli.com to mgr1.networking.tivoli.com, use the following commands to update the host names for the maps named default_cs and mymap:

```
mapadmin -u default_cs:mgr1.networking.tivoli.com
mapadmin -u mymap:mgr1.networking.tivoli.com
```

**Removing Maps from the Client:** Before removing a server connection from a client or deinstalling Tivoli NetView from the client, select Delete Map... from the File pull-down menu or use the `ovw -rmmap mapname` command to remove all maps that are stored on the client. If you do not remove the maps, information is left in the object database on the server regarding the number of maps that exist and where those maps reside. This can provide an incorrect picture of the maps that exist.

Refer to *Tivoli NetView for UNIX Installation and Configuration* for information about removing client access and deinstalling clients.

**Customizing Maps**

You can customize maps to meet the needs of individual users. Customizing a map enables you to:

- Allocate responsibility for managing your network among several people. For example, network administrators expert in managing routers and gateways can open a map that is configured to help manage those devices.
- Use network management applications that perform a specific type of task, for example, data traffic or performance monitoring.
- Create a map that is a projection of an administrator's responsibility or sphere of influence.

You can customize the display of object information for maps you create. Several maps can display information about the same object because maps get their information from the same source, the ovwdb object database.

**Reasons for Creating Several Network Maps**

You can create several maps of your network. The following are reasons for creating more than one map of a network.

- **Sphere of influence**
  You might want to have different maps for different areas of responsibility within your network. In large or complex networks, it can become impractical for one map to display all information about systems in your network. You can create maps that focus on a specific set of system or nodal capabilities. For example, one map might be concerned with software maintenance, and another map with bridge management.

- **Management region**
  You might want to manage a specific portion of your network or partition information about your network. Maps can have specific constraints and characteristics, run different applications, cover different geographic areas, and set compound status differently.

- **Security**
You might want to provide various levels of access to information about your network for security purposes. Using Tivoli NetView permissions, you can create a map of your network that cannot be edited, and another map that is identical to the first map except that it can be edited.

- **Troubleshooting**
  You might want to save a map so you can restore the map later, if necessary. It helps to keep a copy of a map before making changes that may have uncertain results.

- **Customized maps**
  You might want to combine the sphere of influence and management region to create a customized map of your network. Users can share the same map, or you can create a map for each user that combines specific management regions and specific spheres of influence. Perhaps a user wants several maps of the same network, each focusing on a different domain or organization. You can customize a map so that it is a projection of your responsibility for systems on your network. You can customize each map to display certain aspects of your network and avoid sifting through data you do not need. You can also use different background graphics.

- **System-specific maps**
  You might want to have separate maps for different kinds of systems on your network. For example, you might want one map for your IP systems, one map for OSI-based systems, one map for Apollo Domain systems, one map for NFS, and one map for diskless systems.

### Assigning Map Access Levels
Assigning map access levels enables you to limit or deny access on a per-user, per-map basis. Each user of your system will have one of the following types of access to each map:

- **No access**
  The user cannot open this map.

- **Read-only access**
  The user can see status changes, perform locate operations on objects, and update topological changes using the File..Refresh Map operation. However, the user cannot add, delete, or modify symbols, objects, or submaps.

- **Read-write access**
  The user can add objects, add connections, create submaps, and change object attribute values.

### Changing Map Permissions
Only one user can have a specific map open with read-write access at one time. If you have a map open with read-write access and another user displays the same map, the other user’s map is open with read-only access. This is the case even if the other user has read-write permission to the map. Several users cannot have simultaneous read-write access to a specific map. However, if each user creates a copy of this map, they can have read-write access to the copy.

There are several commands that you can use to assign users read-write or read-only access to a map. To change map permissions, use the `ovwperms` command or one of its convenience routines:

- **ovwls** Lists current permissions for specified maps.
**ovwchown**
Changes the owner of one or more maps.

**ovwchgrp**
Changes the group ID of one or more maps.

**ovwchmod**
Changes map permissions by mode. Do not use the file permission commands from your operating system.

For more information about these commands, refer to the *Tivoli NetView for UNIX Administrator's Reference* or the man pages. For steps on changing the map permissions using the Tivoli desktop, see "Setting Map Permissions" on page 92.

---

**Map Snapshots**

A snapshot is a static image of a particular map that preserves the status of all symbols and contains all submaps that existed in the map at the time the snapshot was taken. Although you must take snapshots from a read-write map, the snapshots are read-only and cannot be updated by applications. Select **Map Snapshot → Create...** from the **File** pull-down menu to take a snapshot of a map.

Use snapshots to document your network or to keep a record of your network's current status. It is useful to take snapshots before making major configuration changes. Select **Map Snapshot → Open** from the **File** pull-down menu to open a previously created snapshot of a map.

When you open a snapshot, the home submap at the time the snapshot was taken is displayed. Only one snapshot can be open at a time; however, you can display the open map and a snapshot in submap windows at the same time. The name of the snapshot is displayed in the status line.

Although you can display an open map and a snapshot of the map at the same time, you cannot perform the same operations on them. Operations that highlight a symbol on the open map do not highlight the same symbol on the snapshot. Highlighting an object applies only to the map or snapshot in which it is highlighted.

---

**Submaps**

A submap is a collection of related symbols that are displayed in a single window. Each submap displays a different perspective of the information in the map. Typically, submaps are organized in a hierarchy that enables you to see your network from a distance or to choose a more detailed view. You can customize the organization of submaps in a map to suit your purposes.

The most common method used to navigate through submaps is double-clicking the mouse on explodable symbols. Double-clicking an explodable symbol causes a submap to be displayed if a submap is associated with the object the symbol represents. The object associated with the explodable symbol is called the parent object. The submap that is displayed by double-clicking the symbol associated with the parent object is called a child submap. You can display more than one submap window at one time. To do so, holding the second mouse button, click a symbol and drag the symbol to an area outside the current submap. You can also use the Locate..Submap menu option to view a submap.
Working with Submaps

Submaps enable you to:

- Create a selective view into part of the management domain of a network.
- Choose a collection of symbols to display in a single submap window.

You can create, delete, and modify the characteristics of submaps in the open map. You might want to create a submap to display a detailed view of systems on your network. You can create increasingly detailed submaps of your network map. If you have a submap that becomes too congested, you can create a new submap and partition the information.

Using the Root Submap

The graphical interface creates a root submap that provides a standard, top-level submap on which you can display the symbols that represent different protocol views of the map. For example, on the root submap you will probably have an IP Internet symbol that represents all of the IP entities in your management domain. You can also have symbols that represent other types of topology entities. The root submap enables you to place more than one protocol symbol within one map.

Network and systems management applications can use the root submap to build hierarchies of submaps. The root submap serves as an anchor on which applications can place symbols that represent protocols. You can select one of these symbols and display the highest level of a submap hierarchy.

Using the Home Submap

Each map has a submap designated as the home submap. The home submap is the first submap that is displayed when the map is opened. You can assign any submap in the map as the home submap for all users of that map. By default, when a map is created, the root submap is designated as the home submap. If you delete the home submap, the root submap becomes the home submap until another home submap is assigned.

See “Assigning a Home Submap” on page 73 for more information.

Submap Characteristics

Table 5 describes submap characteristics.

Table 5. Submap Characteristics

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The name of the submap, as assigned when the submap was created. Each name must be unique within the scope of the map.</td>
</tr>
<tr>
<td>Parent Object</td>
<td>The object considered a parent object of a submap. The symbols representing the object explode into the submap. Because several symbols from different submaps can represent the same object, you can navigate to a child submap from several submaps. A submap might not have a parent object.</td>
</tr>
<tr>
<td>Parent Submap</td>
<td>The submap chosen as the parent of the current submap. During submap creation, a user or application can choose a submap to be the parent submap.</td>
</tr>
<tr>
<td>Layout</td>
<td>The layout algorithm used for the submap. Once the algorithm is set, it cannot be changed for an existing submap.</td>
</tr>
<tr>
<td>Presentation</td>
<td>The presentation format used by the Tivoli NetView program to display the submap. The presentation can be either scaling or zooming.</td>
</tr>
</tbody>
</table>
Table 5. Submap Characteristics (continued)

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Background Graphics</td>
<td>The graphic displayed on the background of the submap plane to customize the appearance of the submap.</td>
</tr>
<tr>
<td>Comments</td>
<td>The comments, notes, or keywords about the submap.</td>
</tr>
</tbody>
</table>

Creating Child Submaps and Independent Submaps

In Tivoli NetView, there are child submaps and independent submaps. The method you use to create a new submap determines whether the submap is a child submap or an independent submap (also known as an orphan submap).

A child submap represents a detailed view of its parent object. To create a child submap, double-click an explodable symbol whose parent object has no child submap.

For more information about creating child submaps, see Creating a Child Submap on page 71.

An independent submap has no parent object or parent submap. To create an independent submap, select Create Submap... from the Edit pull-down menu. To display an independent submap, select the submap from the Submaps in Map dialog box and select Open Submap, or select the independent submap from the Navigation Tree window.

Understanding Submap Presentation

Submap presentation enables you to choose how each submap presents symbols and background graphics. You can choose between scaling and zooming for each specific submap.

Scaling

Scaling enables you to display an overall view of the submap. All displayed symbols and the background graphic scale to the size of the submap window. Scaling is the default setting when a submap is created.

Zooming

Zooming enables you to display a close-up view of the submap. Scroll bars are displayed when only a portion of the submap fits into the viewing area. If you are working with a read-write submap and you use the scroll bars to change the visible portion of the submap, the layout is saved.

You can use the zoom feature using the following methods:

- Set a zoom factor to determine the extent to which you zoom into the view of the submap. The default zoom factor is one.
- Use quick zoom to draw a boundary box around the area of the submap to be magnified.

To set a zoom factor:

1. Select Modify/Describe... from the Edit pull-down menu.
2. Select Submap.
3. Select the Zooming button and use the slider bar to select a zoom factor.
4. Click OK.
To use quick zoom:
1. Position the mouse cursor in the upper left corner of the area to be magnified.
2. Press and hold down the Shift key and mouse button 1. Then drag the mouse
to draw a box around the area to be magnified.
3. Release mouse button 1, then the Shift key. If you release the Shift key first, the
objects will be selected instead of magnified. Use the Locate..Selected Objects
List..Deselect All menu operation to deselect them and try again.

To return to the original presentation, press and hold down the Shift key and click
mouse button 1 anywhere on the map.

Displaying a submap with background graphics and zoom presentation uses a large
amount of memory. This is because the entire virtual display has to be buffered,
enabling the user to navigate using the slide bars. The checkZoomAllocation
resource in the /usr/OV/app-defaults/OVw file controls whether a warning dialog box
is displayed every time a user zooms in on a submap with background graphics.
Zoom ratios greater than the value set for checkZoomAllocation cause a warning
dialog box to be displayed. Set the resource to 10 if you do not want a warning
dialog box to be displayed. Set the resource to zero if you want a warning dialog
box to be displayed for any zoom ratio.

See "Changing the Graphical Interface Defaults" on page 93 for more information.

Understanding Submap Layouts
The way the Tivoli NetView program arranges symbols on a submap is called the
submap layout. The method for arranging symbols on the submap is called the
layout algorithm. Symbols can be automatically placed on a submap as determined
by the layout algorithm or they can be manually placed by the user.

Using Layout Algorithms
Each submap has an assigned layout algorithm that determines how symbols are
arranged on the submap. The layout algorithms are based on common network
topologies. Table 6 lists the available layout algorithms.

Table 6. Network Topology Layout Algorithms

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Arrangement on the Submap</th>
</tr>
</thead>
<tbody>
<tr>
<td>Row/Column</td>
<td>Symbols are arranged in rows and columns.</td>
</tr>
<tr>
<td>Point to Point</td>
<td>Symbols are arranged as an arbitrarily inter-connected set of nodes and connections.</td>
</tr>
<tr>
<td>Bus</td>
<td>Symbols are arranged along a backbone representing the linear array of nodes on a segment.</td>
</tr>
<tr>
<td>Star</td>
<td>Symbols are arranged in a star consisting of a circle and a center symbol. You can set the star center using the symbol pop-up menu.</td>
</tr>
<tr>
<td>Ring</td>
<td>Symbols are arranged in a circle.</td>
</tr>
<tr>
<td>Tree</td>
<td>Symbols are arranged in a hierarchical tree structure.</td>
</tr>
<tr>
<td>No layout</td>
<td>Symbols are arranged by the user or are left in the New Object Holding Area.</td>
</tr>
</tbody>
</table>

You can set the layout algorithm for a submap only when the submap is created.
The layout algorithm cannot be changed for that submap after it has been created.
If an application creates a submap, it can specify a layout algorithm. If no layout
algorithm is specified when the map is created, a default layout algorithm is selected. The default layout algorithm for a submap is based on the symbol type of the parent object.

**Note:** You cannot change the layout algorithm for any submap of the system default map. The default layout algorithm for the root submap is row/column.

### Using Automatic Layout
Automatic layout either enables or disables enforcement of the layout algorithm of a submap. You can enable or disable the automatic layout algorithm for a selected submap. After enabling or disabling automatic layout for all submaps in the open map, you can change the automatic layout setting to on or off for all submaps. See "Setting Automatic Layout" on page 91 for steps on setting automatic layout.

### Using the New Object Holding Area
A New Object Holding Area is displayed in the lower portion of each submap window if the submap has no layout algorithm or if automatic layout is disabled for that submap and new objects have been discovered. Symbols in the New Object Holding Area are shown without their connections. To drag symbols from the new object holding area and place them in the associated submap, hold the Ctrl key and using mouse button 2, select the symbol, drag it to the submap, and release the mouse button.

### Understanding Submap Planes
Submaps contain the following three layers, or planes:
- Background plane
- Application plane
- User plane

The **background plane** provides the background against which symbols are viewed. You can add background graphics in the background plane to provide a context for looking at symbols in your submap presentation. Symbols on the application plane and the user plane are displayed on top of the graphic.

For more information about adding background graphics, see "Adding or Removing a Background Graphic" on page 88.

Symbols on the **application plane** represent objects that are managed by at least one network or system management application. If one or more applications manage an object, one or more symbols of that object are displayed on the application plane. The symbols of that object are presented flat against the submap background. If no applications are managing the object, all symbols representing that object appear on the user plane.

Symbols on the **user plane** represent objects that are created by users and not managed by any applications. The Tivoli NetView program distinguishes symbols on the user plane by providing a shadow for symbols to make them appear raised above the submap background.

### Metaconnection Submap
A metacconnection symbol represents more than one connection between two symbols or a symbol and a backbone on a submap. For example, suppose a gateway has more than one interface card in the same network. On the IP Internet submap, the connection between the gateway and the network is a metacconnection. This is because the connection symbol between the gateway and the network
represents two connections (the two interface cards). When you double-click a metaconnection symbol, the metaconnection submap opens.

The metaconnection submap displays the status of each connection between the two symbols. The graphical interface creates a metaconnection submap when a user or an application adds a second connection between two symbols or a symbol and a backbone.

You can add an unlimited number of connections between two symbols in a regular submap. Each of these connections is automatically added to the metaconnection submap with the two symbols or the symbol and the backbone as end points.

It is possible to create a regular child submap for a connection object, if only one connection exists. If that connection becomes multiple, a metaconnection submap is created. You can no longer access the regular child submap by double-clicking the connection. However, you can select the regular child submap from the Submaps in the Map dialog box. You cannot see individual connections between the two symbols on the submap in which the metaconnection symbol is displayed.

**Figure 11. Example of a Metaconnection Submap**

**Characteristics of a Metaconnection Submap**
A metaconnection submap has the following characteristics:
- Displays all the connections represented by the metaconnection symbol.
- Has a row/column layout unless the connections are between a symbol and a backbone.
- Displays the two end points of the connection in the metaconnection submap for each connection in the submap.
Behavior of a Metaconnection Submap

The behavior of the metaconnection submap is similar to a regular submap in some ways:

- You can create a child submap from a metaconnection submap by double-clicking on any of the objects in the metaconnection submap. Therefore, the metaconnection submap can be a parent of other regular submaps.
- You can select objects in the metaconnection submap.
- You can add unconnected objects to a metaconnection submap.

The behavior of the metaconnection submap differs from a regular submap in some ways:

- You cannot add connections to a metaconnection submap.
- You cannot delete the last object from the submap (whether it represents a connection or not) without deleting the metaconnection and metaconnection submap.
- You cannot see propagated status for connected symbols. Compound Status works differently in metaconnection submaps. The metaconnection symbol displays the compound status of the multiple connections in the metaconnection submap. Any unconnected objects in the metaconnection submap also contribute to compound status. However, the connected icon symbol in a metaconnection submap do not propagate their status. Their status is maintained by the symbols in the parent submap above the metaconnection submap.

Using Tivoli NetView Applications

An application is a program that interacts with users through the graphical interface. Applications enable you to perform the following actions:

- Process user requests
- Create or delete objects, symbols, and submaps
- Change the contents of maps
- Provide special display functions

You can write and integrate applications with Tivoli NetView.

Refer to the Tivoli NetView for UNIX Programmer’s Guide for more information about writing applications.

ipmap Application

The ipmap application is the primary application used by the Tivoli NetView program. When the graphical interface is started, the ipmap application is automatically started. The ipmap application ensures that the ovw application (the graphical interface) and the ovtopmd daemon (the IP topology database daemon) behave consistently.

For example, when an object is deleted using the graphical interface, the ovw application tells ipmap which symbols and objects were removed. The ipmap application then tells ovtopmd to make the appropriate changes to the topology database.

When netmod discovers a new node, ovtopmd adds the node to the IP topology database and informs ipmap that a new node has been discovered. The ipmap application uses what it knows about IP devices to tell ovw which icon and connection symbols it needs to create. The graphical interface then displays the correct symbols and modifies the map database accordingly.
For more information about the netmon and ovtopmd daemons, refer to "Background Processes" on page 7 or the man pages.

Map Synchronization
When a map that uses the ipmap application is opened, the application starts its synchronization phase. While the ipmap application is synchronizing, the graphical interface displays the [Synchronizing] message on the status line of all displayed submaps of the open map. During this phase, the ipmap application requests information about changes to the IP topology database since the map was last open. This information is continually updated by the ovtopmd daemon. If there are any new objects in the IP topology database, the ipmap application tells the ovw application which icons and connection symbols to add to the map.

The map will enter the synchronization phase for a short time during operations that change the contents of the map, for example, changing interface labels, adding new objects, or cutting and pasting large numbers of symbols.

Limitations: While the ipmap application is synchronizing, the following limitations exist:

• You cannot delete symbols, objects, or submaps.
• You cannot add objects.
• You cannot cut and paste symbols.
• The ipmap application will not appear in the Configurable Applications list on dialog boxes (Object Description).
• Manage Objects and Unmanage Objects operations will not take effect.
• Acknowledge and unacknowledge operations will not take effect.

When the synchronization phase completes, the ipmap application resumes full operation and the [Synchronizing] message is no longer displayed.

Using ipmap Application Submaps
The ipmap application and the graphical interface have rules that control the placement of symbols on submaps. Only certain symbols can exist on each submap type.

Table 7 describes the submap hierarchy used by the ipmap application and the symbols are supported on each submap.

Table 7. Symbols Supported by ipmap

<table>
<thead>
<tr>
<th>Submap</th>
<th>Symbols Supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root submap</td>
<td>• IP Internet</td>
</tr>
<tr>
<td>Location or Internet</td>
<td>• Location</td>
</tr>
<tr>
<td></td>
<td>• Internet</td>
</tr>
<tr>
<td></td>
<td>• IP network</td>
</tr>
<tr>
<td></td>
<td>• Connector (for example, a gateway)</td>
</tr>
<tr>
<td></td>
<td>• Connection</td>
</tr>
<tr>
<td>Network</td>
<td>• Segment</td>
</tr>
<tr>
<td></td>
<td>• Connector (for example, a gateway)</td>
</tr>
<tr>
<td></td>
<td>• Connection</td>
</tr>
<tr>
<td>Segment</td>
<td>• Node (for example, computers and connectors)</td>
</tr>
<tr>
<td></td>
<td>• Backbone</td>
</tr>
<tr>
<td></td>
<td>• Connection</td>
</tr>
<tr>
<td>Node</td>
<td>• Interface card</td>
</tr>
</tbody>
</table>
Moving through the Submaps
Double-clicking the IP Internet symbol takes you into the IP Internet submap, which is a special internet submap. The internet submap may contain a few network symbols. Double-clicking a network symbol takes you into a network submap. On the network submap, you will probably have segment symbols. Double-clicking a segment symbol takes you into a segment submap. The segment submap will have nodes connected to a backbone symbol. Double-clicking the node symbol displays a node submap. The node submap displays all of the interfaces contained in the node.

Each time you select a symbol, you get a submap that is more specific than the previous submap. You can view your entire network, a part of the network, or a single node. Each time you view the parent of a submap a more expansive view of your network is displayed.

Root Submap: The root submap is the highest level submap. It is at the top of the Navigation Tree and has no parent. The root submap contains the IP Internet symbol created by the ipmap application. The IP Internet symbol is the only symbol on the root submap that the ipmap application manages. If you have other applications that use the graphical interface to draw symbols, you may have other symbols on the root submap, but ipmap does not know of their existence.

Internet and Location Submaps: Internet and Location submaps show logical groupings of IP networks and subnetworks connected by gateways. Location submaps and Internet submaps are considered to be equivalent by ipmap. This means that any symbol type that can be placed on a location submap can also be placed on an Internet submap.

The IP Internet submap is special because it is created by the ipmap application. It is also the highest level Internet submap; it is the only Internet submap that does not have a location or internet object as its parent. The root submap is its parent. The ipmap application places discovered IP addressable gateways and networks on this submap.

Internet and location objects are often referred to as container objects. That is because Internet and Location submaps are the only submaps that can contain their own symbol type. For example, a location submap can contain other location symbols, but a node submap cannot contain other node symbols. It can only contain interface symbols. This function can be used to organize and simplify a map. This is called partitioning. See "Partitioning Submaps" on page 68 for information about partitioning.

Network Submaps: The network submap represents the physical topology of a network at the level of network segments. The ipmap application can discover and display IP-addressable segments, gateways (routers), repeaters, multiports (hubs), bridges, and the connections between them on the network submap.

Note: If you are using IP subnetting, network implies subnet.

Segment Submaps: A segment submap represents the physical topology of a segment of your network at the level of nodes and connectors. It displays the computers and connectors that comprise a segment on your network.

In IP networks, a segment is a group of data communication objects that are interconnected through a common transmission medium. Nodes belonging to the
same segment typically use a common physical medium to communicate with each other (for example, Ethernet, token ring, telephone lines, or satellite links). The following segment topologies can be drawn:

**Bus**
- Represents nodes attached to a single linear cable that transmits data (for example, Ethernet or IEEE 802.3)

**Token ring**
- Represents nodes attached to an SNMP, IP-addressable token ring central wiring MAU through twisted pair wiring, which conforms to the IEEE 802.5 standard

**Star**
- Represents all nodes attached to an SNMP central multiport repeater (a hub)

**FDDI ring**
- Represents nodes attached to an SNMP, IP-addressable Fiber Optic Data Distribution Interface (FDDI)

**Node Submaps:** The node submap displays symbols that represent the components of a node in a row/column layout. The graphical interface draws interface symbols on the node submap. When an interface object is added to a map, the ipmap application tells the graphical interface which node submap to draw the interface card symbol. The ipmap application also tells the graphical interface on which higher level submaps to draw connection symbols.

### Using the xxmap Application

The xxmap application enables you to see submaps showing information about open-topology objects. The xxmap application presents information gathered by the \texttt{gtmd} daemon. The \texttt{gtmd} daemon stores and correlates topology information received in the form of SNMP traps or through API calls based on the Tivoli NetView topology MIB. For more information about the \texttt{gtmd} daemon, refer to the \textit{Tivoli NetView for UNIX Programmer’s Guide} or the man page.

The xxmap application supports user changes to the map. Using the graphical interface, you can add and delete symbols. However, added symbols are not verified or stored in the gtm database. These symbols exist only in the user plane. When you delete a symbol from the map, the underlying object is not deleted from the gtm topology database. Objects can be deleted or added to the gtm database with an SNMP trap command or an API call. This symbol can also reappear during a subsequent map synchronization phase. If you do not want to see the symbol on a map, select **Hide Objects...** from the **Edit** pull-down menu to hide it. The object is still managed even though it is hidden. To view a list of objects that are hidden, select **Hidden Objects List...** from the **Edit** pull-down menu.

### Displaying Multiple Protocols

If you are using protocols other than IP, the xxmap application enables you to see a list of the managed protocols running on a selected object. This option is available only for objects that are interfaces or nodes.

Suppose you select an object that is a computer running the CMIP protocol. Select **Protocols...** from the **View** pull-down menu to display a list that contains the following information about the CMIP protocol interface on the selected node:

- The name or address as defined in the Tivoli NetView General Topology MIB
- The status of the protocol
- Submaps that contain the selected object’s CMIP interface

60 Administrator’s Guide
Customizing a Graphical Map

There are many ways in which maps can be customized to make them easier to use and understand. Objects can be added to and deleted from your map database, and symbols can be moved from one submap to another. You can also group objects together into a SmartSet based on a selection rule that you define, such as a SmartSet of all routers.

Submap characteristics such as submap name, parent submap, layout, and background graphics can be changed. Also, objects can be selectively managed and acknowledged to match your management region responsibilities. The following sections describe how to use these customization functions.

Customizing Symbol Placement

You may want to customize the IP Internet topology to reflect your network layout: geographically, hierarchically, or by some other criteria that is important to your company. For example, you could add location symbols to your IP Internet submap that correspond to your offices in New York, Chicago, and San Francisco and group all networks under these location symbols. You can do this manually or you can create a custom configuration file to automatically customize your network layout. This type of customization is called submap partitioning.

Submap partitioning involves many different actions, including adding objects and moving symbols. First, you should know the rules for manually adding, deleting, moving, and copying symbols and objects. To automate this level of customization in your maps using the location file, refer to "Automating Internet Submap Partitioning" on page 70.

Note: A symbol can be an icon symbol, which is a two-dimensional picture, or a connection symbol, which is drawn as a line connecting two icon symbols. It is important to understand the difference between a connection symbol, which is a line that often represents an interface card object, and a connector symbol, which is a diamond shaped icon symbol that represents a device such as a bridge or router. Throughout this section the term symbol refers to an icon symbol unless stated differently.

Adding Symbols and Objects

Tivoli NetView enables you to add symbols to submaps. The following are some of the reasons why you might want to do add symbols:

- A network entity may not have been automatically discovered by the netmon network discovery daemon.
- You want to add something to your map that is not a part of your network, such as a location or segment symbol.

If you know the object you want to add and on the submap you want to place the symbol, you can drag the symbol from a palette of symbols to the correct submap and then enter some information about the object. See "Steps for Adding Objects and Symbols" on page 63 for the steps on adding objects.

Note: Connection symbols are added differently from icon symbols. See "Adding Connection Symbols and Objects" on page 62 for information about adding connection symbols.

User Plane and Application Plane: A symbol that has a shadow is in the user plane, and a symbol that has no shadow is in the application plane. A symbol in the
application plane is known and controlled by ipmap or some other application, whereas a symbol in the user plane is known and controlled by only the user that added or modified the symbol. Select **User Plane -> For This Submap -> Shadow Off** from the View pull-down menu to turn the shadow off an object in the user plane.

**Symbols Managed by the IP Application:** Certain applications only manage certain kinds of symbols. The ipmap application can manage the following types of symbols, organized by symbol class:

*Table 8. Symbols Managed by ipmap*

<table>
<thead>
<tr>
<th>Symbol Class</th>
<th>Managed Symbols</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cards</td>
<td>IP interface card</td>
</tr>
<tr>
<td>Computer</td>
<td>All</td>
</tr>
<tr>
<td>Connector</td>
<td>All</td>
</tr>
<tr>
<td>Location</td>
<td>All</td>
</tr>
<tr>
<td>Network</td>
<td>Internet, IP Network, Bus, Token Ring, Star, FDDI, Serial, and Frame Relay segment</td>
</tr>
</tbody>
</table>

If you want the ipmap application to manage an IP entity, then use one of the symbols listed above. If you do not use one of the symbols listed above, then the symbols will not have status and connectivity information updated by the ipmap application.

**Symbols That Can Be Added:** The ipmap application enforces rules that control the placement of symbols on submaps. The ipmap application only manages symbols if they are placed on the correct submap. This ensures that submaps can always be traversed. It also ensures that connectivity and other relationships between objects can be conveyed properly.

*Table 9. Symbols You Can Add That ipmap Can Manage*

<table>
<thead>
<tr>
<th>Submap</th>
<th>Symbols That Can Be Added</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root</td>
<td>None.</td>
</tr>
<tr>
<td></td>
<td>The IP Internet symbol is the only symbol controlled by ipmap on this submap. Because this symbol is created for you, there is no reason to add any others.</td>
</tr>
<tr>
<td>Location or Internet submap</td>
<td>• IP network symbol (network class). You must provide unique network name, IP address, and subnet mask.</td>
</tr>
<tr>
<td></td>
<td>• Gateway symbol (connector class). You must provide unique host name, IP address, and subnet mask.</td>
</tr>
<tr>
<td></td>
<td>• All location symbols. You must provide unique selection name.</td>
</tr>
<tr>
<td></td>
<td>• Internet symbol (network class). You must provide unique selection name.</td>
</tr>
</tbody>
</table>
Table 9. Symbols You Can Add That ipmap Can Manage (continued)

<table>
<thead>
<tr>
<th>Submap</th>
<th>Symbols That Can Be Added</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network submap</td>
<td>• All connector symbols. Provide a unique host name, IP address, and subnet mask.</td>
</tr>
<tr>
<td></td>
<td>• Bus, star, token ring, FDDI, serial, frame relay symbols (network class). Provide a unique selection name.</td>
</tr>
<tr>
<td>Segment Submap</td>
<td>• All connector symbols. Provide a unique host name, IP address, and subnet mask.</td>
</tr>
<tr>
<td></td>
<td>• All computer symbols. Provide a unique host name, IP address, and subnet mask.</td>
</tr>
<tr>
<td>Node Submap</td>
<td>IP interface card symbol. Provide an IP address and subnet mask.</td>
</tr>
</tbody>
</table>

Steps for Adding Objects and Symbols

Now that you know what and where symbols can be added, follow these steps to add an object and symbol. You must have the map open with read-write access to add objects.

1. Select **Add -> Object...** from the **Edit** pull-down menu.
2. Select a class from the Symbol Classes on the Add Object Palette.
3. Using mouse button 2, drag the desired symbol subclass icon to the appropriate submap and release the mouse button.
4. When the Add Object dialog box is displayed, complete the following fields:
   - Label
   - Display Label
   - Behavior
   - Object Attributes
   - Selection Name
   - Comments
5. If you want your added object to be updated, controlled, and managed by an application, then you have to:
   a. Select the application name from the Object Attributes list on the Add Object dialog box. If the symbol represents an IP entity, select the IP Map list item.
   b. Click **Set Object Attributes**.
   c. Fill in all the necessary fields.
6. Click **OK** in the Add Object dialog box to add the symbol to the submap and close the dialog box.
7. Click **OK** in the Add Object Palette to close the palette.

If this is done correctly, after you drag the symbol to the submap, you will see that the symbol is displayed flush against the background plane, and the symbol will be drawn without a shadow. The symbol is in the application plane. This indicates that the application you specified is correctly managing the symbol you added.

If you do not select an application and fill in the correct fields, the symbol is displayed raised above the background plane, and a shadow is displayed under the symbol. This means that the symbol is in the user plane and receives no updates from any existing applications.

The application that manages IP entities is called the ipmap application. When adding IP objects to your map, selecting the IP Map entry from the Object Attributes
list and selecting the **Set Object Attributes** button ensures that your IP symbols will correctly reflect status and connectivity information.

**Adding Connection Symbols and Objects**

To draw a connection between two symbols, select **Edit..Add..Connection**. When you connect two IP symbols together, you are specifying that a new IP interface card exists that connects the symbols. For example, when you connect a gateway to a backbone symbol, you are adding an interface card to the gateway. The connection symbol is a graphical representation of the interface object in the object database.

**Symbols That Can Be Connected:** Table 10 describes which IP symbols can be connected together with connection symbols. It is organized by submap type.

<table>
<thead>
<tr>
<th>Submap</th>
<th>Symbols That Can Be Connected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root</td>
<td>None</td>
</tr>
<tr>
<td>Location or Internet</td>
<td>Gateway symbol to IP network symbol</td>
</tr>
<tr>
<td>Network Submap</td>
<td>Gateway symbol to a bus, star, token ring, FDDI, serial, or frame relay segment</td>
</tr>
<tr>
<td>Segment Submap</td>
<td>All connector symbols to backbone symbol</td>
</tr>
<tr>
<td>Node Submap</td>
<td>None</td>
</tr>
</tbody>
</table>

**Steps for Connecting Symbols**

Now that you know what symbols can be connected, follow these steps to add a connection. You must have the map open with read-write access.

1. Select **Add -> Connection...** from the **Edit** pull-down menu.
2. Select one of the connection symbol types from the Add Connection palette.
3. Select a source symbol on the submap that you want to connect.
4. Select a destination symbol. The connection symbol is displayed between the two selected symbols on the map.
5. When the Add Object dialog box is displayed, complete the following fields:
   - Label
   - Display Label
   - Behavior
   - Object Attributes
   - Selection Name
   - Comments
6. If you want your connection symbol to be updated, controlled, and managed by an application, then you must:
   a. Select the application name from the Object Attributes list on the Add Object dialog box. If the connection symbol represents an IP entity, select **IP Map**.
   b. Click **Set Object Attributes** button.
   c. Complete the remaining fields.

   If this procedure is not used, the connection symbol is not updated.
7. Click **OK**.
8. Click **OK** in the Add Object dialog box to set the connection in the map and close the dialog box.
9. Click **OK** in the Add Connection Palette to close the palette.
The application that manages IP entities is the ipmap application. When connecting IP symbols on your map, select the IP Map entry from the Object Attributes list and select the **Set Object Attributes** button. This ensures that your connection symbols correctly reflect the status of the underlying interface card.

### Deleting Objects and Symbols

To delete an object from the map, select **Delete Object -> From All Submaps** from the **Edit** pull-down menu. To delete an object completely from the object database, go into every map that contains that object and delete the object from all submaps. Select **Delete Object -> From All Submaps** to be sure all symbols for an object are deleted from your map.

If the object is still part of your network, and if the netmon daemon is running, a deleted object can be discovered and displayed. There are three ways to prevent this:

- Turn off new node discovery by selecting **Topology/Status Polling Interfaces: IP**... from the **Options** pull-down menu.
- Prevent discovery by starting the netmon daemon with a seed file that contains the IP address or host name of the node preceded by an exclamation mark (!). The ! operator tells netmon not to discover this node. See the netmon man page for more information.
- Let the object be discovered and then select **Hide Objects** from the **Edit** pull-down menu to hide the object. The object is still managed even though it is hidden. To view a list of objects that are hidden, select **Hidden Objects List...** from the **Edit** pull-down menu.

You must have a map open with read-write authorization to delete an object or symbol. You cannot retrieve deleted objects or symbols.

Do **not** use the **Edit -> Delete Object** menu to delete all nodes in the internet view. The complete deletion of the database while the Tivoli NetView graphical interface is running can cause unpredictable results.

To delete all objects in the map, shut down all Tivoli NetView graphical interfaces and use the Server Setup application (serversetup). Select one of the following options:

- **Maintain -> Clear databases -> Clear object/topology/map databases, save customizations**
- **Maintain -> Clear databases -> Clear object/topology/map databases, remove customizations**
- **Control -> Restart automatic map generation**

### Moving Objects and Symbols

The **Edit..Cut** and the **Edit..Paste** functions are used to move symbols from one submap to another. The graphical interface lets you move any symbol you want to any submap you want. Although the graphical interface enables you to move any symbol, some symbols may be managed by particular applications that place limits on where symbols can be placed.

For example, the ipmap application only manages IP network symbols that are located on location or internet submaps. If an IP network symbol that is managed by ipmap is cut from a location submap and pasted onto a node level submap, then the ipmap application will stop managing that symbol. For this reason, it is a good idea to be familiar with the rules that each application places on the location of certain symbols.
As a general rule, if a symbol is in the application plane (the symbol has no shadow and is controlled by an application) and you move (cut and paste) the symbol, you want the symbol to end up in the application plane of the destination submap.

If you want the ipmap application to continue to control an IP symbol, then move (cut and paste) IP symbols to submaps that ipmap supports.

**Symbols You Can Move:** Table 11 describes which IP symbols can be moved and still be managed by the ipmap application. They are listed by submap type.

<table>
<thead>
<tr>
<th>Submap</th>
<th>Symbols That Can Be Moved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root submap</td>
<td>None.</td>
</tr>
<tr>
<td></td>
<td>The IP Internet symbol is the only symbol controlled by ipmap on this submap. This symbol must remain on the root submap.</td>
</tr>
<tr>
<td>Location or Internet submap</td>
<td>• IP network symbols (network class)</td>
</tr>
<tr>
<td></td>
<td>• Gateway symbols (connector class)</td>
</tr>
<tr>
<td></td>
<td>IP Network and gateway symbols can be moved from one Location or Internet submap to another Location or Internet submap. They should not be moved to any other type of submap.</td>
</tr>
<tr>
<td>Network submap</td>
<td>None.</td>
</tr>
<tr>
<td></td>
<td>No symbols should be moved to or from a network submap. That is because the symbols on a Network submap have IP addresses and they would be inconsistent with the IP subnets of other networks.</td>
</tr>
<tr>
<td>Segment submap</td>
<td>• All connector symbols</td>
</tr>
<tr>
<td></td>
<td>• All computer symbols</td>
</tr>
<tr>
<td></td>
<td>All connector and computer symbols can be moved between segments in the same network. Node symbols should not be moved between segments in different networks.</td>
</tr>
<tr>
<td>Node submap</td>
<td>None.</td>
</tr>
<tr>
<td></td>
<td>No symbols should be moved to or from a Node submap.</td>
</tr>
</tbody>
</table>

When using the cut function to move symbols from one submap to another, always select the option **Cut..From This Submap**. Cutting from all submaps will cause multiple copies of your symbol to be moved, and that could cause links between symbols to be drawn incorrectly.

The cut and paste options enable you to move symbols from one submap to another. They should not be used to copy symbols.

**Cut Buffer:** The cut buffer holds symbols you have cut or copied until you do one of the following:

- Store another symbol into the cut buffer.
- Open another map.
- Exit Tivoli NetView.
Steps For Moving an Object: Now that you know what can be moved and where it can be moved to, follow these steps to move an object. You must have the map open with read-write access to move objects.

1. Open the submap that has the symbol representing the objects you want to move.
2. Select the symbols.
3. In that submap, select Cut..From This Submap from the Edit pull-down menu.
4. Open the destination submap.
5. Select Paste from the Edit pull-down menu.

When symbols are pasted into a submap, they are first put in the user plane. Any applications that control or manage the symbols determine if they have been moved to a legal location. Depending on the machine speed and the number of symbols moved, this could take from 1 to 15 seconds. Once necessary applications determine that the move is supported, the symbols are moved into the application plane and the shadow disappears.

Copying Symbols
Like cutting and pasting, copying is a function that is provided by the graphical interface. Applications that use the graphical interface to control and display symbols provide various levels of support for the copy function.

The ipmap application does not support the copy function. When manipulating IP topology symbols, it is recommended that the Edit..Copy function not be used. If you want to move symbols between submaps, it is best to use the Edit..Cut function. IP symbols that are copied will be placed in the user plane rather than the application plane. These symbols will not show a connection to any other symbol. In general, the copy menu option on the graphical interface exists for use with other applications, and not with the ipmap application.

Steps for Copying Symbols: There are two parts to this procedure:
• Copying a symbol
• Pasting a symbol

1. Copy the symbols.
   a. Select the symbol that you want to copy.
   b. Select Copy from the Edit pull-down menu.
   c. Select one of the following from the Copy pull-down menu:
      • From This Submap
        Performs the operation on this submap.
      • From All Submaps
        Performs the operation on all submaps.

   When you select Copy, the symbol remains on the submap.

2. Paste the symbols.

Select the submap on which you want to paste the symbol. Select Paste from the Edit pull-down menu to paste the symbol onto the submap.

Note: You can paste the symbol onto any submap of the open map. Pasting the copied symbol onto the same submap from which you copied it creates multiple symbols of the object for the same submap.

All symbols are pasted in the New Object Holding Area for each submap in which automatic layout is disabled.
Partitioning Submaps

You can partition a submap to subdivide submaps into smaller, more manageable units. For example, you can add location symbols that correspond to your offices in New York, Chicago, and San Francisco to your IP Internet Submap. You can then move the network and gateway symbols from the IP Internet submap to the New York, Chicago, and San Francisco submaps, making network entities much easier to find.

There are two kinds of submaps that you can partition:
- Internet and location
- Segment

The rules for each are slightly different.

This section describes how to partition submaps manually. You can automate this process for the Internet submap using the location configuration file. Refer to "Automating Internet Submap Partitioning" on page 70 for more information on automating this customization.

Partitioning Internet/Location Submaps

Partition Internet or Location submaps when you want to take symbols from an Internet or Location submap and place them in other Internet or Location submaps. To partition an Internet or Location submap manually, follow these steps:

1. Add either a location object or an internet object to either a location submap or Internet submap. If you are working with IP symbols, don't forget to fill out the correct IP Map information in the Add Object Dialog Box. See "Steps for Adding Objects and Symbols" on page 63 for steps on adding objects.
2. After the location or internet object has been properly added, select the network or gateway symbols that you want to move into the new location.
3. Select Cut -> From This Submap from the Edit pull-down menu. When you have selected the symbols and cut them, they should disappear from the original submap.
4. Open the submap of the internet or location symbol you added in the first step.
5. Select Paste from the Edit pull-down menu.

Example of a Partitioned Internet Submap

Figure 12 on page 69 shows an Internet submap in which four container objects were added. All symbols of objects discovered by the ipmap application have been cut from the Internet submap and placed in lower-level partitioned Internet submaps. If you clicked on the United States symbol, you would open the submap shown in Figure 13 on page 70.
Figure 12. Example of a Customized Internet Submap

Figure 13 on page 70 shows the United States submap. Six container objects have been created in this submap. From this submap, you can double-click any symbol and open a child submap to display other container objects or symbols that represent IP networks and gateways.
Partitioning Segment Submaps

Partition segment submaps when you want to move node symbols from one segment submap to another segment submap. To partition a segment submap, follow these steps:

1. Open the network submap that contains the segment that you want to partition.
2. Add a segment object to the Network submap. If you are working with IP symbols, don’t forget to fill out the IP Map dialog box when adding the segment. See “Steps for Adding Objects and Symbols” on page 63 for steps on adding objects.
3. When the new segment has been added, open the segment submap that contains the symbols you want to move.
4. Select the symbols.
5. Select Cut — From This Submap from the Edit pull-down menu. When the cut is complete, the symbols should disappear.
6. Open the new segment submap and select Paste from the Edit pull-down menu.

Automating Internet Submap Partitioning

You may also customize the IP Internet topology to reflect your network layout automatically using the location file (/usr/OV/conf/location.conf). The location file contains a list of networks, or ranges of networks, and the locations under which they should be displayed; locations can be nested to create a hierarchy. All other symbols, such as segments, are placed under the network in which they belong. Locations cannot be used to divide a subnet.
Using this file, instead of manually customizing the submap, automates the customization process and will preserve the customizations when the map is regenerated.

Refer to Using a Location File to Customize the Map Layout in Tivoli NetView for UNIX Installation and Configuration for information on the format of the location.conf file and how to use it.

Creating a Child Submap

Create a child submap if you need a more detailed view of an object. You can use the default settings or modified settings. Open the submap from any explodable symbol of the parent object. The submap is part of the map’s hierarchy of submaps.

If the symbol you select already has a child submap, then select Add -> Object from the Edit pull-down menu to add an explodable object. This function adds a parent object from which you can add a child submap. Use the Location symbol type to represent the object if you are adding network views.

Using Default Settings

If you have a submap that consists of three groups of objects, organize your network by creating three child submaps, then select Cut and Paste from the Edit pull-down menu to cut and paste each group of objects into a different child submap. Because you are only reorganizing your network, you do not need to change any of the default settings for the child submap.

Steps: To create a child submap with default settings, follow these steps. You must have the map open with read-write access.

1. Double-click an explodable symbol that does not contain a child submap. A Tivoli NetView Question box displays a message that the object does not have a child submap.

2. To create a child submap with default settings, click OK. The new submap is created and displayed in a separate submap window. Because this is a new submap, the submap does not contain symbols or objects.

When you double-click an explodable symbol of the parent object, the submap is opened. When you select Show Parent on the background context menu of the child submap, the parent submap is reopened in the separate submap window.

Default Settings: The default settings for the child submap are:

• The name of the submap is the same as the selection name of the object from which the submap was created.
• The presentation is scaling.
• The submap contains no background graphic.
• The submap contains no comments.

Using Modified Settings

If you have a submap that consists of three groups of objects that represent three parts of your network and you want to change the layout of one of the groups, you might consider creating three child submaps and changing the layout algorithm for one of the child submaps. After you create the child submaps and change the settings as necessary, select Cut and Paste from the Edit pull-down menu to cut and paste each group of objects into the appropriate child submap.
Steps
To create a child submap with modified settings, follow these steps. You must have the map open with read-write access.

1. Double-click on an explodable symbol that does not contain a child submap. A Tivoli NetView Windows Question box is displayed, telling you that the object does not have a child submap.

2. To create a child submap with modified settings, click Modify. The New Submap dialog box is displayed. The name of the new submap is displayed in the Name field. The name of the parent object is displayed in the Parent Object field. The button to the right of the Parent Submap displays the name of the parent submap.

3. You can modify the following settings:
   - Name
   - Parent submap
   - Layout
   - Presentation
   - Background graphics
   - Comments

   For more information about modifying these settings, see "Modifying Submap Settings" on page 73.

4. After you complete the dialog box, click OK. The new submap is created and displayed. Because this is a new submap, the submap does not contain symbols or objects.

Creating an Independent Submap
You can create an independent submap (also known as an orphan submap) that does not have associated parent objects. This type of submap is independent of the existing submap hierarchy. For example, you might create an independent submap that contains all your routers. This enables you to monitor router status from one submap.

To open an independent submap, select Open Submap... from the View pull-down menu. You can open a submap created without a parent object through the Navigation Tree window only if you have already opened that submap during the current Tivoli NetView session. Use the horizontal scroll bar on the Navigation Tree window to see the independent submap symbols.

Steps
To create an independent submap, follow these steps. You must have the map open with read-write access.

1. Select Create Submap... from the Edit pull-down menu.

2. A Tivoli NetView Question box is displayed. Select one of the following buttons:
   - Click OK to create a submap with default settings. For more information about default settings, see "Creating a Child Submap" on page 71.
   - Click Modify to create a submap with modified settings.
     For more information about modified settings, see "Using Modified Settings" on page 71.
   - Click Cancel to undo the submap creation.

Note: Because this submap is an independent submap, you cannot assign it a parent submap.
### Changing a Parent Submap

You can change the parent submap of the open submap. The parent submap is the submap whose icon is displayed last in the submap stack.

**Steps**

To change the parent of a submap, follow these steps. You must have the map open with read-write access.

1. Decide which open submap you want to modify. Then make it the current window.
2. Select **Modify/Describe → Submap...** from the **Edit** pull-down menu.
3. The Submap Description dialog box is displayed. Select the Parent Submap option button to display a list of possible parents for this submap.
4. Select a parent from the list. If no list is displayed, the current parent submap is the only choice at this time.
5. After you select a parent submap from the list, click **OK** to apply the changes to the submap and close the dialog box.

### Modifying Submap Settings

You can modify certain characteristics or information about a submap such as its presentation, the background graphic, or its parent.

**Steps**

To modify a submap, follow these steps. You must have the map open with read-write access.

1. Decide which open submap you want to modify. Then, make it the current window.
2. Select **Modify/Describe → Submap...** from the **Edit** pull-down menu.
3. The Submap Description dialog box is displayed. You can modify any of the following characteristics:
   - Name
   - Parent object
   - Parent submap
   - Presentation
   - Background graphics
   - Comments

   **Note:** You cannot modify the submap layout. The layout algorithm can only be set during submap creation.

4. After you modify the characteristics on the dialog box, click **OK** to apply the changes to the submap and close the dialog box.

### Assigning a Home Submap

To assign the submap to be the **home submap**, that will be displayed when the map is opened, follow these steps. You must have the map open with read-write access.

1. Select **Set Home Submap...** from the **Options** pull-down menu.
2. Select the submap you want to be the Home Submap on the Submaps in Map dialog box. You can use the Find Submap field to locate a specific submap by matching a string or substring with a submap entry.
3. Click **Set as Home**.
4. Click **Close** in the Submaps in Map dialog box to apply the change and close the dialog box.
The selected submap is the submap displayed when this map is opened.

**Modifying an Object Description**

To modify an object description, follow these steps. You must have the map open with read-write access.

1. Select one or more objects in the submap.
2. Select Modify/Describe -> Object... from the Edit pull-down menu.
3. An Object Description dialog box is displayed for each selected object. You can modify the following attributes in the Object Description dialog box:
   - Selection name
   - Object attributes list
   - Comments
4. Click OK on the Object description dialog box to apply the changes and close the dialog box.

**Managing and Unmanaging Objects**

An object can be managed or unmanaged. A managed object is being monitored for topology, status, and configuration changes. The symbol for the managed object reports the status changes by changing to the color that represents the status. If an object is not being managed, the symbol for the object does not report the status because it is not known. However, the symbol is still visible. Managing objects uses network resources. If you have objects that don't need to be managed, unmanage them.

If you want an object to continue being managed but you do not want to see it, select Hide Objects from the Edit pull-down menu. The object receives and reports status but the symbol for the object does not appear on the submaps. You can also access the Hide operation by selecting Hide Objects from the Edit pull-down menu or Edit -> Hide -> Symbol from the object context menu. To view a list of objects that are hidden, select Hidden Objects List... from the Edit pull-down menu.

**Steps**

To manage or unmanage one or more objects, open the map with read-write access and follow these steps:

1. Select one or more objects to be managed or to be unmanaged.
2. Select Manage Objects or Unmanage Objects from the Options pull-down menu.

All selected objects and child submaps are managed or unmanaged, depending on which option you selected.

**Acknowledging and Unacknowledging Objects**

If you know that an object has stopped functioning, but you do not want Tivoli NetView to notify you continuously about this problem, use the acknowledge operation.

When you acknowledge the object, the object changes to a dark green color and remains in the acknowledged state until you select the object and unacknowledge it. Unacknowledging an object causes Tivoli NetView to resume normal processing.

Tivoli NetView offers two modes of acknowledging objects: a map-based mode and a global-based mode.
Map-based mode is the default. However, you can set an environment variable to activate the global-based behavior for individual clients. You may want the default map-based behavior for some clients and the global-based behavior for others. For instance, you may have NetView client users who do not need to see objects being acknowledged by the network operations center.

**Setting Map-Based Acknowledge Status**

The default map-based Acknowledge mode only affects the current map. Operators of client machines must refresh the map to be notified that objects have been acknowledged or unacknowledged. Also, acknowledging objects on read-only maps is not persistent; restarting the console clears all the acknowledged statuses. Note that the Acknowledged status can not be set from the command line.

To acknowledge or unacknowledge one or more objects, follow these steps. You must have the map open with read-write access.

1. Select one or more objects to be acknowledged or unacknowledged.
2. From the Options pull-down menu, select **Acknowledge** or **Unacknowledge**.

All selected objects and child submaps are acknowledged or unacknowledged, depending on which option you selected.

**Setting Global-based Acknowledge Status**

In the global-based Acknowledge mode, the Acknowledged status is an object attribute. All NetView clients are notified immediately when an object is acknowledged or unacknowledged, regardless of which map is open and even if the action is performed on a read-only map. In addition, you can set the Acknowledged status of an object from the command line, permitting the transfer of the Acknowledged status to backed-up NetView databases.

To activate the global-based mode, set the environment variable `NVMAPGLOBALACK` equal to 1 on the NetView machine (`NVMAPGLOBALACK=1`). This must be set before you invoke the NetView console.

You can acknowledge an object using one of three methods. The first, from the map, uses the graphical user interface. The other two methods, from an event or through `nvdbimport`, allow users greater control of the Acknowledged state outside of the graphical user interface.

- **From the map.** To acknowledge or unacknowledge map objects:
  1. Select one or more objects.
  2. Choose Options–> **Acknowledge** to set the isAcknowledged field to TRUE for that object and generate an event to update all open maps (local or NFS) connected to the NetView server.

     If a NetView console running in global-based mode starts up later, the object is displayed as Acknowledged.

- **From an event.** To acknowledge or unacknowledge an object from the command line, type:

```
event -b openview -e event -a object ID
```

Where `event` is either ACK_EV (acknowledge event) or UNACK_EV (unacknowledge event), and `object ID` is the object ID of the object you are acknowledging in the OVw database. For more information on this command, see the `event` man page.

This event sets the status of the specified object to Acknowledged on all open maps where the status of the object is CRITICAL or MARGINAL. It also sets the
Acknowledged field to TRUE in open NetView maps. If no NetView sessions are running in global-based mode, this field is not set. If NetView consoles running in global-based mode start up later, the object shows as Acknowledged.

- **Using nvdbimport.** To directly set the isAcknowledged field for chosen objects in the Ovw database, use nvdbimport. See the [nvdbimport man page](#) for more details.

**Import/Export Utility for Acknowledged Status**

The sample import/export utility script, `/usr/OV/bin/acknowledgeUtil`, provides an example of setting the acknowledged state from an event. To use the script, type:

```
acknowledgeUtil -export -map mapname > filename
acknowledgeUtil -import < filename
```

The beginning of the script includes a complete description of the arguments.

Only Critical and Marginal objects are changed to Acknowledged. Attempts to Acknowledge normal objects during the import operation are silently ignored.

---

**Defining and Managing SmartSets**

Network administrators often discover that, as their networks grow, distributing files and customization changes to nodes in the network is a time-consuming and error-prone task. The Tivoli NetView SmartSet facility provides a mechanism for you to group objects together. This group of objects is called a **SmartSet**. You can define SmartSets using the SmartSet Editor or the nvUtil command. You can also use the SmartSet facility APIs to have your applications create and use SmartSets.

The collmap application provides the display of SmartSets and is automatically started when you start the graphical interface. You can use the **Administer..Start Application..collmap** menu option to start the collmap application without having to close and restart the graphical interface. You might find this useful if the collmap application ends abnormally after you have started the graphical interface.

Note that collmap is a viewing tool only. It cannot be used to update SmartSets in the SmartSet facility. Thus, cut and paste are not supported within SmartSet submaps.

SmartSets are aggregated under a SmartSet icon and are displayed on the Root submap. If you double-click on the SmartSet icon on the Root submap, you see a submap containing symbols for all the SmartSets that are defined. Double-clicking on one of the SmartSet symbols opens a submap containing all the objects that are currently in that SmartSet. As objects move in and out of the SmartSet, the submap is dynamically updated. A user in read-only mode needs to select **Refresh Map** from the **File** pull-down menu to see new objects that have been dynamically updated.

Defining a SmartSet can be useful for creating a submap of devices that you want to monitor closely. For example, you can define a SmartSet of all critical routers. This enables you to view a submap of all routers that are inactive at any time.

SmartSets enable you to do policy-based management. For example, you can establish a **SmartSet** policy for all routers, establish distribution policies through the **Agent Policy Manager** for specific domains as defined by SmartSets, and establish polling policies for groups of machines. You can also create event rulesets that check to see if the node is a member of a specified SmartSet.
If you have the Tivoli NetView MLM program installed in your network, you can use the Agent Policy Manager to set thresholds and set up file monitoring for SmartSets you have defined. For example, you can define a threshold to monitor CPU utilization for all objects in a Fileserver SmartSet. The Agent Policy Manager is closely integrated with the SmartSet facility. Management by policy in this manner facilitates your task of system management by centralizing control. If you want to change the thresholding on a group of objects, you do not have to change each object in the SmartSet; the change is applied automatically to all objects in the SmartSet. Similarly, if an object that fits the defined SmartSet is added to or taken out of the network, no additional changes are necessary. The SmartSet facility automatically updates the SmartSet.

See Chapter 8, Using the Agent Policy Manager (APM) on page 203 for more information.

Managing and Unmanaging SmartSets

The SmartSet icon and SmartSet symbols are unmanaged until you double-click the icons. Double-clicking a SmartSet symbol starts the monitoring process, and status is propagated upwards from objects within the SmartSet. Because monitoring status uses network resources, you should unmanage SmartSets that do not need to be managed.

To stop monitoring SmartSet status, select the SmartSet symbol and then select Edit -> Delete -> Symbol from the context menu to delete the SmartSet symbol. Because you have not deleted the SmartSet definition, the symbol is automatically recreated and is unmanaged. If you delete the SmartSet icon on the IP root map, all the SmartSets under the SmartSet icon become unmanaged. If you want to unmanage a specific SmartSet and continue to manage other SmartSets, delete the SmartSet symbol for the SmartSet that you want to unmanage.

Types of SmartSets

To create a SmartSet, you can specify the selection name of an object, or you can define a rule (much like a filter rule), using the Tivoli NetView object capability definitions. If you specify a rule (such as `isRouter=True`), the SmartSet facility locates objects that fit that description. A SmartSet can also be a combination of a node list, rules, and other SmartSets you have defined.

The Agent Policy Manager automatically creates one SmartSet for you, and you will see it appearing on your root map with the label, MLM subnets. This special collection is made up of the MLMs in your network that Tivoli NetView knows about, and all the objects managed by the MLMs. If you double-click this symbol, Tivoli NetView displays a map showing each of the MLMs, and double-clicking an MLM symbol displays a star configuration of the MLM and the managed objects in its subnet.

See Chapter 8, Using the Agent Policy Manager (APM) on page 203 for more information.

Quick Refresher Course on Boolean Logic

When you define a SmartSet, you have several opportunities to use logical AND and OR statements to join different rules you have specified. These logical operators have a different meaning than and or in everyday speech. Consider this example:
• If I said to you, “Please bring me an apple and an orange,” you would return with two pieces of fruit, an apple and an orange.
• If I tell the SmartSet facility, “Locate a mainframe computer AND a PC,” it will find nothing, because there probably is no individual device that has both of those characteristics assigned to it.

The logical operator OR also operates differently:
• If I said to you, “Please bring me an apple or an orange,” you would return with either an apple or an orange.
• If I tell the SmartSet facility, “Locate a PC OR a mainframe,” it will find all the PCs and all the mainframes in the network.

When you use the SmartSet facility, keep these simple rules in mind:
• When you want to find the union of two characteristics, use OR.
• When you want to find the intersection of two characteristics, use AND.

Pattern Matching

Pattern matching enables you to use regular expressions as the value of a SmartSet rule. A valid regular expression consists of printable characters. However, in a regular expression, the following symbols have special meaning:

• . (period)
  Matches printable and nonprintable character except <newline> (unless it is used inside brackets). For example, node.in matches:
  
  node3interface
  nodeXinterface

• * (asterisk)
  Means zero or more occurrences of the preceding character. For example, node* could match the following strings:
  
  nod
  node
  nodeTwo
  nodee

The expression node.* could match node1 and node, but the expression would not match nod.

The rule sysLocation˜Building* would match the following strings:
  
  Building 002
  Building 500
  Building 062

Note: The ˜ character is the "like" operator.

• ` (caret)
  If the ` character is the first symbol, the following character is the first character in the string, as shown in the following example:
  
  `any Matches the string: any of them.
  `any Does not match the string: many of them.

• $
  If $ character is the last symbol, the preceding character is the last character in the string, as shown in the following example:
  
  long$ Matches the string: long
  long$ Does not match the string: longer
Pattern matching on IP addresses is treated differently (not as a regular expression). If the value of a SmartSet rule looks like an IP address, the syntax for using wildcards is as follows:

IP address: `<byte>.<byte>.<byte>.<byte>

byte: int | int-int | [int-int] | *

int: 0 - 255

The rule `IP Address˜146.83.[120-255].*` specifies all nodes with IP addresses 146.83.120.* through 146.83.255.*, where * represents any integer in the range 0–255.

Adding a New SmartSet

To define a new SmartSet, use the SmartSet Editor or the `/usr/OV/bin/nvUtil` command. Refer to the man page for information about the nvUtil command.

You can use a list of IP addresses, capability rules, or a combination of both to define a SmartSet. You can use pattern matching in the values of capability rules as explained in “Pattern Matching” on page 78. You can also use the name of another SmartSet, thus building a hierarchy of SmartSet.

The following example demonstrates how you can set up a SmartSet for critical routers in your network. For this example, we will set up a SmartSet that includes all of the Cisco and IBM routers. We will use the Tivoli NetView predefined object capabilities to find objects that are classified as IBM routers or Cisco routers. We will add other routers by specifying their IP addresses. In addition, we will find those routers whose IP status is critical.

1. SmartSets are defined using the SmartSet Editor. Select SmartSet Editor from the Tools pull-down menu. The SmartSet Editor window is displayed.
2. Click the Add button to display the Add SmartSet dialog.
3. Enter the name of the SmartSet, with a brief description of what the SmartSet contains. This SmartSet is called CriticalRouters.
4. Define the first SmartSet rule. Next to Definition 1, click Modify. The Modify Definition dialog box is displayed.
5. On this dialog box, click Definition Type. You will see a list of the types of rules you can use to define a SmartSet:
6. Select **Attribute**.
   You will see a list of Object Attributes. Use the scroll bar to move down through the list until you find **vendor**.
7. Select the vendor object attribute. A list of selectable values is displayed:
8. Select the IBM value and click **OK**. The definition is added to the Add SmartSet dialog box:
9. You now have a rule to select objects that have IBM specified as the value for their vendor attributes. To complete the first rule, you need to OR this rule with a rule selecting objects that have a vendor attribute of Cisco. (In other words, we will find objects that are made by IBM as well as objects that are made by Cisco.) Between Definition 1 and Definition 2, make sure the Or radio button is selected.

10. Click Modify next to Definition 2. Repeat the steps for selecting the vendor attribute, but this time select Cisco as the value for the attribute.

11. Click OK to add the definition.

12. You now have a rule to find IBM OR Cisco objects. You need to AND this rule with a rule that will find objects that are classified as routers. Ensure the And radio button between the top two definitions (1 and 2) and the bottom two definitions (3 and 4) on the Add SmartSet dialog box is selected.
13. Select the Modify button next to Definition 3. Select the Definition Type option button, and select Attribute from the menu that is displayed.

14. Scroll through the list until you find the attribute isRouter. Select this attribute.

15. This time, you will see that the attribute can be set to True or False. Select the True radio button and click OK.

16. You now have a rule to find IBM and Cisco routers. You need to AND this rule with a rule that will find routers that are down.

   Ensure the And radio button between Definition 3 and Definition 4 is selected.

17. Click Modify next to Definition type 4. Select the Definition Type option button, and select Attribute from the menu that is displayed.

18. Scroll through the list until you find the attribute IP Status. Select this attribute.

19. Select the Critical value and click the OK.

20. You now have a rule to find all critical routers. Here is how the completed rule looks:
21. Click **OK** to add this definition. The SmartSet you defined appears on the SmartSet Editor main window.

**Modifying a SmartSet**

Suppose you need to add one more routers to your SmartSet of critical routers. This router is a vendor that is not listed as a selectable value for the vendor attribute, but you do know the router’s IP address. You can create another SmartSet, using the CriticalRouters SmartSet you already defined, and add the new router by specifying its selection name. Here are the steps:

1. Start the SmartSet Editor if it is not running. Select **SmartSet Editor** from the **Tools** pull-down menu. The SmartSet Editor window is displayed.
2. Click **Add** to display the Add SmartSet dialog box.
3. Enter a name and description for this new rule. We will call this example SmartSet CriticalRoutersPlus1.

4. Next to Definition 1, click **Modify**. The Modify Definition dialog box is displayed. Select Include SmartSet Rule.

5. A list of defined SmartSet rules is displayed. Select CriticalRouters and click **OK**. The SmartSet name is displayed in Definition 1.

6. Next to Definition 2, click **Modify**. The Modify Definition dialog box is displayed. This time, we will use Object List as the Definition Type. It is probably already selected; if not, select Object List.

7. At the bottom of the Modify Definition window, enter the selection name of the object to be added in the Object Selection Name field and click **Add**. The object is added to the List of Objects.

8. Click **OK**. The object is displayed in Definition 2.

9. Click **And** between Definitions 1 and 2 and the **Or** radio button between Definition Definitions 3 and 4.

10. Next to Definition 3, click **Modify**. The Modify Definition dialog box is displayed. Select Attribute as the Definition Type.

11. Select the attribute IP Status and the value **Critical**.

12. Click **OK**. The IP status is displayed in Definition 3.

13. Click **OK** to add this definition. The SmartSet you defined is displayed on the SmartSet Editor main window.

### Listing Objects in a SmartSet

To find out which objects in the network are included in a SmartSet, select a SmartSet on the SmartSet Editor window and then click **Resolve**. A list of objects in the selected SmartSet is displayed. You can also use the `/usr/0V/bin/nvUtil` command to list objects in a specific SmartSet. Refer to the man page for more information.

### More Examples of SmartSets

The following examples illustrate other SmartSets you might develop. Each example has a statement of the rules under which an object is to be included in the SmartSet and a picture of how the rule is to be defined in the SmartSet facility.

**Example 1**
Create a SmartSet of all printers running OS/2* TCP/IP SNMP agents that are not located in building 123 or building 456.
Example 2
Make a SmartSet of all nodes contained in subnet 195.88.31.0.
Figure 19. Nodes Contained in Subnet 195.88.31.0
Chapter 4. Customizing the Graphical Interface

Customizing the graphical interface can mean as little as adding backgrounds or it can include specifying maps for startup, authorizing file permissions, setting event filters, and customizing menu bars.

This chapter describes the tasks involved in customizing the graphical interface. You can change the graphical interface for each user so they see only what they are responsible for. For example, if you have three users, each responsible for a different geographical region, you can create or customize submaps for each user so they see only the region for which they are responsible. You can give them access to applications that pertain to only them. If you want to add a different background to each submap, you can do that also.

This chapter includes the following tasks:
- "Adding or Removing a Background Graphic"
- "Arranging Symbols" on page 90
- "Assigning Maps" on page 93
- "Setting Map Permissions" on page 92
- "Customizing the Menu Bar and Tool Palette" on page 92
- "Changing the Graphical Interface Defaults" on page 93
- "Customizing the Failing Resource Display" on page 95
- "Customizing Event Filters for Users" on page 96
- "Customizing the Tivoli NetView Grapher" on page 96

Adding or Removing a Background Graphic

You can add a graphic to the background of a submap or replace a graphic with a different graphic. Background graphics can provide contextual information, such as:
- A floor plan of your business
- A geographic map showing diverse sites
- A diagram representing some characteristics of a portion of the managed network

Supported Formats

The graphical interface supports the following file formats for background graphics:

<table>
<thead>
<tr>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GIF</td>
<td>CompuServe Graphics Interchange Format Background graphics must be in GIF87a format.</td>
</tr>
<tr>
<td>XBM</td>
<td>X11 monochrome bitmap format</td>
</tr>
</tbody>
</table>

Note: To support the operation of sizing a window, the owv application zooms graphics down to a minimum size (about icon size) first and then attaches the graphics to windows. If the graphic is large and contains small text and thin lines, some information may be lost when the graphic is sized down. If you can clearly view the graphic when it is zoomed down to icon size, Tivoli NetView should clearly display the graphic.

Adding a Background

To add a background graphic or replace an existing background graphic of a submap, follow these steps. The map must be opened with read-write access.

1. Choose **Select Background Picture** from the **Edit** pull-down menu.

   The graphical interface displays the Submap Description dialog box for the open submap.
2. Complete or change the Background Graphics field by entering a line with the complete path and name of the graphic file. For example, to select the background graphic for a map of the United States, enter the following:

/usr/OV/backgrounds/usa.gif

Or, click Select and choose a graphic from the list that is displayed.

3. To display the new background graphic, click OK.

If you do not select a background graphic, the background plane remains empty.

Removing a Background

To remove a background graphic, delete the input file name from the Background Graphics field. The full path name must be deleted. The map must be opened with read-write access.

1. Choose Select Background Picture from the Edit pull-down menu.

The Submap Description dialog box for the open submap is displayed.

2. Select the Background Graphic field to make it the active field. Make sure the cursor is at the beginning of the text.

3. Press Ctrl and then Delete.

The graphic name is removed and the field is blank.

4. Click OK to close the Submap Description dialog box.

Arranging Symbols

You can arrange the symbols that are displayed on the submaps either automatically or manually. Whichever method you choose, you must have a read-write map to save the new layout. You can also change the symbol labels. This section provides the steps for arranging and labeling symbols.

Using Redo Layout

Redo layout enables you to arrange the symbols in a submap according to the assigned layout algorithm. Redo layout works regardless of whether automatic layout is enabled or disabled.

To save the new layout for the submap, you must have read-write access. New symbol positions are not saved for read-only maps.

Steps
To redo the layout for a submap, follow these steps:

1. Select Redo Layout from the View pull-down menu.

2. If you redo the layout for a read-only map, the graphical interface displays a warning message that the new symbol positions will not be saved. If you redo the layout for a read-write map, the graphical interface displays a warning message that all symbols will be repositioned and their current positions cannot be restored.

3. Click OK button to redo the map. The graphical interface repositions the symbols on the submap according to the submap layout algorithm.
Setting Automatic Layout

Automatic layout enables the system to automatically arrange the symbols in a submap according to the assigned layout algorithm. You can turn automatic layout on or off for the current submap, or for all submaps. You must have the map open with read-write access.

Steps
To set automatic layout, follow these steps:
1. Select Automatic Layout from the View pull-down menu.
2. The graphical interface displays a cascade menu. You can turn automatic layout on or off for the current submap or for all submaps.
3. After you make your selection, the graphical interface applies the change.

Note: If you turn automatic layout off, for either the current submap or for all submaps, the graphical interface requires a holding area to place the newly discovered objects. The graphical interface displays an area at the bottom of the current submap or all submaps called the New Object Holding Area. The New Object Holding Area is displayed only when there are symbols to be placed.

To manually move an object from the New Object Holding Area, select the object using the Ctrl button and mouse button 2 and drag it to the current submap.

To enable the graphical interface to move all objects from the New Object Holding Area to the current map, select Redo Layout from the View pull-down menu or turn automatic layout on.

Modifying and Displaying Symbol Labels

If you have changed the name of a resource and you want to change the label for the symbol, follow the procedure to modify and display symbol labels.

Steps
To modify symbol labels, follow these steps. You must have the map open with read-write access.
1. Select the symbol of the object with mouse button 3 to display the context menu.
2. Select Edit → Modify/Describe → Symbol... from the context menu.
3. Enter the text in the Label field on the Symbol Description dialog box.
4. Click Yes or No to the right of Display Label to select whether to display the symbol label in the submap.
5. Click OK on the Symbol Description dialog box to apply the change and close the dialog box.

Note: If there are too many symbols on a submap, the labels are not displayed.

To control the display of selected symbols, select Show/Hide Labels from the Edit pull-down menu.

If you want to change the symbol on a submap, select Change Symbol Type from the Edit context menu.
Assigning Maps

You can provide read-write access for each user by creating a unique map for each user. The map can be a copy of the map named default. Users can make changes to their own map without effecting what is displayed on other maps. When you create a map, you become the owner and the only one with read-write access to the map. Use the ovwchown command to change the owner of the map from you to the user.

Setting Map Permissions

To prevent maps from being deleted, set up different map permissions on the various map databases.

Use the ovwperms command or the Configure -> Change Maps owner/group/mode option available through the Server Setup application to set permissions on maps. Setting map permissions to read-only for users prevents them from deleting specific maps. Only the root user can then delete a map. You must be a root user to change map permissions.

See "Assigning Map Access Levels" on page 50 for information about the different types of map access.

The ovwperms command changes the permissions for all files and directories associated with the map. See the ovwperms man page for more information about changing map permissions with a command.

Steps

To change map permissions using the Server Setup application, follow these steps:
1. Enter serversetup on the command line to access the Server Setup application.
2. Select Configure -> Change Map(s) owner/group/mode.... A dialog box is displayed.
3. Enter the necessary information in the fields.
4. Click OK.

If you have a number of users that share the same maps and applications or perform similar tasks, use the UNIX mkgroup command to create a group for them. Groups can be formed for users that share access authority to protected resources. After your group is established, you can change map permissions for the group rather than for individuals.

Customizing the Menu Bar and Tool Palette

If you are not using Tivoli NetView security services, which provides a customized graphical interface based on Tivoli NetView group permissions, you can use the OVwRegDir environment variable to customize each user’s menu bar and tool palette with the applications that pertain to only that user.

See "Chapter 2, Defining and Managing a Security Policy" on page 19 for information about Tivoli NetView security services. The OVwRegDir environment variable points to the location of the application registration files. You can set this variable in the user’s .profile or .kshrc file. Placing the desired registration files in each user’s directory gives each user access to a different set of operations. You
can point to individual directories, then from those directories, set up symbolic links to the registration files you want in /usr/OV/registration/C.

If there are specific menu items you do not want users to access, edit the ovw application registration file to remove those entries. However, doing so prevents anyone from using those options because they no longer exist.

An alternative to deleting menu items is to split the ovw application registration file. For example, you can split the file in two, ovw1 and ovw2, and separate the options. Add the OVwRegDir environment variable to the user's .profile to point to either the ovw1 or ovw2 registration file.

For example, let's say we split the ovw file into ovw1 and ovw2. You have two users, Lou and Judy. If you want Lou to have access to the menu items in ovw1 and Judy to have access to the menu items in ovw2, add the following line to each of their .profiles:

```
export OVwRegDir=/u/lou/reg/C
    For Lou
export OVwRegDir=/u/judy/reg/C
    For Judy
```

Copy ovw1 to Lou's directory:

```
/u/lou/reg/C/ovw1
```

Copy ovw2 to Judy's directory:

```
/u/judy/reg/C/ovw2
```

Before editing the ovw registration file, make a copy of the original for a backup file.

### Changing the Graphical Interface Defaults

There are certain graphical interface characteristics that you might want to change. For example, you might want the navigation tree and tool palette to display as icons, or you might not want them to be active at all when Tivoli NetView is started. The resources and defaults for these characteristics are defined in the /usr/OV/app-defaults directory for all users.

You can change a default for each user by copying the line containing the resource you want to change into the users $HOME/.Xdefaults file. Changing the resource in the app-defaults files affects all users. You might prefer to put the entries in the .Xdefaults file because customized settings in the app-defaults files are overwritten if you apply a service fix, but the settings in the .Xdefaults file are not. Settings in the .Xdefaults file override the settings in the app-defaults files.

If you change any of the app-defaults files after the application you are customizing is started, such as Tivoli NetView or the Event Display application, use the command `xrdb -merge .Xdefaults` to load the new resource and restart the application. If you have not started application you are customizing, the new resource will be loaded when you start the application.

To change the files in the /usr/OV/app-defaults directory, you must be a root user.
Window Resources

Resources are found in the /usr/OV/app-defaults/OVw file, which defines the resources for the NetView Windows server. Although there are too many to list them all here, if you browse the file, you’ll find resources for the following:

**Fonts**
You can change all fonts in the graphical interface, such as button fonts, label fonts, and window title fonts.

**Colors**
You can change all colors in the graphical interface, such as background colors, symbol colors, and connection colors.

**Sizes**
You can change the default sizes for all windows in the graphical interface, such as message windows, submap windows, and the tools window.

Other Resources

Table 12 lists files, resources, and defaults found in the /usr/OV/app-defaults directory. This is not a complete list. Browse the files to see a complete list. These are resources that affect the appearance of applications displayed in the graphical interface. If you are changing the resources for an individual, you must have read-write access to the .Xdefaults file in the $HOME directory you are changing. If the .Xdefaults file does not exist, you can create this file.

**Table 12: Xdefault Resources**

<table>
<thead>
<tr>
<th>What You Want Changed</th>
<th>File</th>
<th>Resource</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event card format presentation</td>
<td>Nvevents</td>
<td>nvevents.initialPresCard</td>
<td>True</td>
</tr>
<tr>
<td>Event cards text color</td>
<td>Nvevents</td>
<td>nvevents.<em>card</em>cardTextColor</td>
<td>black</td>
</tr>
<tr>
<td>Event cards color</td>
<td>Nvevents</td>
<td>nvevents.cardColor</td>
<td>#f7ded3d2d2</td>
</tr>
<tr>
<td>Event application text color (text that doesn’t appear on the cards)</td>
<td>Nvevents</td>
<td>nvevents.*foreground</td>
<td>black</td>
</tr>
<tr>
<td>Number of events in a workspace</td>
<td>Nvevents</td>
<td>nvevents.maxLoadEvents</td>
<td>500</td>
</tr>
<tr>
<td>Number of workspaces per session</td>
<td>Nvevents</td>
<td>nvevents.maxNumWS</td>
<td>500</td>
</tr>
<tr>
<td>Save event workspaces</td>
<td>Nvevents</td>
<td>saveEnvOnExit</td>
<td>False</td>
</tr>
<tr>
<td>Start event application with saved workspaces</td>
<td>Nvevents</td>
<td>loadEnvOnInit</td>
<td>False</td>
</tr>
<tr>
<td>Include static workspaces and workspaces that were loaded using the File..Load option for saving</td>
<td>Nvevents</td>
<td>considerStaticWrkSpcs</td>
<td>False</td>
</tr>
<tr>
<td>Start main window and control desk as icon</td>
<td>OVw</td>
<td>OVw.*shellIconify</td>
<td>False</td>
</tr>
<tr>
<td>Start the tool palette as icon</td>
<td>OVw</td>
<td>OVw.*toolShellIconify</td>
<td>False</td>
</tr>
<tr>
<td>Start the navigation tree as icon</td>
<td>OVw</td>
<td>OVw.*navTreeShellIconify</td>
<td>False</td>
</tr>
<tr>
<td>Start the navigation tree</td>
<td>OVw</td>
<td>OVw.*navTreePresent</td>
<td>True</td>
</tr>
<tr>
<td>Start the tool palette</td>
<td>OVw</td>
<td>OVw.*toolPalettePresent</td>
<td>True</td>
</tr>
<tr>
<td>Start control desk with events minimized</td>
<td>OVw</td>
<td>OVw.*thereAreInternalTools</td>
<td>True</td>
</tr>
<tr>
<td>Start control desk with network view</td>
<td>OVw</td>
<td>OVw.*controlDeskHasBox</td>
<td>True</td>
</tr>
<tr>
<td>Graph line width</td>
<td>XNm</td>
<td>xnmgraph.lineWidth</td>
<td>2</td>
</tr>
<tr>
<td>Graph line colors</td>
<td>XNm</td>
<td>xnmgraph.graphLineColors</td>
<td>See file</td>
</tr>
</tbody>
</table>
Steps for Changing the Graphical Interface Defaults

To customize the graphical interface defaults for a user ID, follow these steps:

1. Examine the files in the /usr/OV/app-defaults directory to determine which items you want to customize.

2. Edit the .Xdefaults file in the user’s $HOME directory and implement your changes. For example, if you want to change the default font for the Tree button on a submap window, enter a line with the following format:

   OVw*treeButtonFont: fontname

   Where fontname is the name of the font that you want. For more information about fonts, colors, and sizes, refer to the system documentation.

3. After you change the .Xdefaults file, save it. To see the changes, you must exit the Tivoli NetView program and restart it or use the command `xrdb -merge .Xdefaults` to load the new resource.

Preventing the Control Desk from Automatically Starting

By default, the Control Desk is active when you start the graphical interface. If you don’t want the Control Desk to start, edit the /usr/OV/registration/C/ovsnmp/nvevents file, and remove the –Initial flag from the following line:

   Command -Shared -Initial "$[nvevents:-/usr/OV/bin/nvevents]" ;

   Note that the Control Desk will start by running any application, such as the Event Display application, that uses the Control Desk, and you can start the Control Desk from the tool palette. When you apply Tivoli NetView service, a new /usr/OV/registration/C/ovsnmp/nvevents might replace your customized file, and you might need to remove the -Initial flag from the new file.

Customizing the Failing Resource Display

If you have a read-write map you can display all the failing resources for an object by selecting Tools..Failing Resources Display. This option opens a new submap with symbols that represent the failing resources of a selected object. This new submap is not connected to the submap hierarchy for the currently opened map. The failing resource submap starts its own hierarchy which you can see in the Navigation Tree.

The failing resource symbols reflect the current status of all the interfaces of an object. If an interface status changes, the symbol representing that interface also changes. When you double-click on a failed resource, another submap displays the hierarchy of the failed resource within the network.

You can customize the Failing Resource Display by editing the /usr/OV/registration/C/xnmfault file and setting the following flags:

- **-t** Changes the text in the title bar.
- **-r** Changes the capability field so that only certain objects are displayed. For example, you might want to display only those objects that have the field isConnector set to True.
- **-s** Changes what the status of the resources must be to be displayed. For example, you can change the status so that only marginal status is displayed.
- **-u 011** A zero (0) indicates that if symbols matching the criteria are not found,
continue checking and display a submap when one is found. A one (1) removes the symbols if the status does not match the criteria.

For changes to take effect, restart ovw. The following is an example of the command line in the xnmfault file:
Command "/usr/OV/bin/xnmfault -t 'title' -r isConnector -s 2 "$OVw Selection1";

**Customizing Event Filters for Users**

You can start the nvevents application with activated filters. During startup, the nvevents application reads an event filter file located in the user’s $HOME directory or in the directory indicated by the resource profileDir in the /usr/OV/app-defaults/Nvevents file. The filter file is named after the user with the extension of events. For example, if the user name is shannon, and you did not modify the .Xdefaults file, the filters are defined in $HOME/shannon.events.

You can define which filters you want activated for each user. The events file name and the filter rule name must be defined in the user’s events file using the following syntax:
FilterFileName filterFileRule

For example, on the AIX operating system, if you create an events file to activate filters Trap_to_Alert_Threshold and Receive_from_6611_router, the contents of the events file will look like the following example:
/usr/OV/filters/filter.samples Trap_to_Alert_Threshold
/usr/OV/filters/filter.samples Receive_from_6611_router

Where filter.samples is the filter file name and Trap_to_Alert_Threshold is the filter file rule.

When you activate filters during the nvevents operation, they are saved in this file. The next time the application is started the last filter(s) activated will be automatically registered. See "Chapter 5. Correlating, Filtering, and Configuring Events" on page 103 for more information about filtering events.

**Customizing the Tivoli NetView Grapher**

Many of the Tivoli NetView program’s graph applications present their results in graphs that you can save, print, or customize to better suit your needs. Although each application graphs different information, the general presentation format is the same.

Graph applications have certain display default values. These defaults might be suitable for your purposes, but there might be other times when you need to modify the default display to meet your needs. This section describes the ways you can customize the graph display.

Figure 20 on page 97 shows the graphical interface’s standard layout for graph applications, using the **Network Activity --> Interface Traffic...** option from the **Monitor** pull-down menu.
This time, the application was started from the Tivoli NetView main menu, so it is displayed in the control desk. You might choose to start it from the Tools window and not put it in the control desk.

**Entering Numeric Values**

The following rules apply when entering numeric values in dialog box fields:

- A number beginning with x or 0x is treated as hexadecimal. For example, 0x0010 becomes decimal 6.
- Any other number beginning with 0 is treated as octal. For example, 010 becomes decimal 8.
- Any number beginning with a numeric digit other than 0 is treated as decimal. For example, 10 becomes decimal 10.
- If you enter a value containing a digit that is not valid for its format, the number is truncated at the place that was not valid. For example, 123A56 becomes 123.

**Setting Time Intervals**

Select **Time Intervals** from the **View** pull-down menu to see the time statistics associated with the graph application. The Display Interval at the top of the dialog box shows the date and time the application was started and is dynamically updated with the current date and time. Move the slider box below the Display Interval to change the interval for which graphed data is displayed.

By default, the graph display shows the most recent 5 minutes of results. The vertical bars on the display mark the minutes. You can change the display width by typing a new interval in the Display Width text field and selecting the **Apply** button. Notice that a beginning date and time and an ending date and time are displayed directly above the Display Width text field.
You can change the resolution of the graph by selecting the Resolution by data/Resolution user defined option button. Resolution by data means that the graph displays the data exactly as it is collected. If you select Resolution user defined, you can increase the resolution, which normalizes the data and enables you to see trends. Refer to the Help system for more information about changing the graph resolution.

The **SNMP Polling On / SNMP Polling Off** option button enables you to set the frequency with which a graph of real-time data is updated. This button has no effect on historical data. The update frequency determines how often the device is queried and any new data displayed in the graph.

**Changing the Line Configuration**

You can change the characteristics of the lines on the graph. Select **Line Configuration...** from the **View** pull-down menu to display the Line Configuration dialog box, as shown in [Figure 21](#).

![Figure 21. The Line Configuration Dialog Box](#)

The Line Configuration dialog box displays a row of fields for each line on the graph. The fields of the Line Configuration dialog box are described in the following list.

**Data Label**

This section lists the names of all MIB objects for which the application is collecting and graphing information. By selecting a name from the list that is displayed when you select the option button, you can choose which MIB objects you want to display. This feature is useful when an application gathers data on more MIB objects than the maximum number that can be displayed at one time in the graph.
On/Off
If you have many lines on your graph and want to simplify the presentation, turn some lines off by clicking the **On/Off** toggle button. You can also turn lines off by selecting the lines on the graph with the Ctrl button and mouse button 2. The lines’ data labels are also removed from the display. To turn the lines back on, click the **On/Off** toggle button to On.

Color
You can change the color of each line by selecting a color from the list that is displayed when you select the Color toggle buttons. If it is difficult for you to distinguish certain colors and, therefore, differentiate the lines, you can display the label for the line by selecting the line on the graph with mouse button 2.

Line Width
You can change the line width by selecting a line width from the selections that appear when you select the **Line Width** toggle button.

Multipliers
Sometimes the collected data includes values that are reported in different units of measurement. You can display all of them on one graph for comparison by changing multipliers, so they are all based on the same unit of measurement. The default is to display data with no multiplier.

Displayed Values
By default, the average values are displayed. You can select any of the following values for display:

- Minimum
- Average
- Minimum and Average
- Maximum
- Minimum and Maximum
- Average and Maximum
- Minimum, Average, and Maximum

For example, if you select Minimum and Average, the graph will display two lines for the same MIB object, one that reflects the Minimum value and one reflecting the Average value.

Getting Application Statistics
Select **Statistics** from the **View** pull-down menu in the graph application to see traffic statistics about each line on the graph. Information about the minimum, average, maximum, and last values is displayed in a table where each line of graphed data forms a row of the table. The statistics are updated based on the value specified for SNMP polling intervals. These statistics are used for analyzing trends about performance peaks and valleys.

By default, the raw values are shown. You can select a multiplier to change the y-axis increments. Doing so also changes the shape of the lines on the graph. To change the multiplier, select **Line Configuration...** from the **View** pull-down menu. Choose a value from the list that is displayed when you select the **Multiplier** option button.

Checking Application Messages
You can look at all the messages that have accumulated for the graph application by selecting **Messages...** from the **View** pull-down menu. The Messages dialog box
displays error and informational messages associated with the application’s processing. Output from the File..Memory Usage and File..Line Info operations is also stored in this dialog box.

**Paging through the Graph**

If you are collecting data over a long period of time, the graph cannot display all of it at once. Select **Screen Paging...** from the **View** pull-down menu for a Help dialog box that explains how to view different parts of the collected data. You can page backward or forward, one screen at a time, or center the graph around a selected point in time.

**Scaling the Y-Axis**

You can scale the y-axis of the graph in one of the following ways:

- On all data
- On displayed data

The default display scales the y-axis on displayed data. You can use the default if you are not running the application for a long period of time. However, if you run the application for a long time, there might be a greater fluctuation from high to low values, which might not be reflected in the portion of the graph currently being displayed. If you need to base subsequent decisions on the overall pattern of data, consider changing the scaling.

**Changing the Display from Color to Monochrome**

The default graph presentation uses a different color for each line that represents a monitored MIB object. The colors help you track which line goes with which MIB object.

You can change to monochrome mode and still differentiate each line on the graph. Select **Color/Monochrome** from the **View** pull-down menu and then select **Monochrome**. Monochrome mode uses different types of lines, for example, solid or dotted, to represent each MIB object’s values. You might want to use this option to see how the graph would look if it were to be printed on a monochrome printer. You can use the Print Tool to see the results.

**Displaying or Hiding the Grid**

By default, a grid is displayed for all graph applications. You can choose to hide the grid by selecting **Show/Hide Grid** from the **View** pull-down menu. The x-axis and y-axis remain, but the vertical lines that mark each minute disappear, as do the horizontal lines that extend the numerical divisions on the y-axis across the width of the graph.

**Showing Counter Values**

If you are graphing MIB values of type Counter, you can select **Show counters As...** from **View** pull-down menu and choose one of the following ways to display the data:

- **Rate of Change**
  The default value, which shows the new counter value as a time-averaged value since the last query for the MIB object.

- **Actual Sampled Value**
  Shows the actual value returned from the MIB Counter variable.
**Delta Value**

Shows the actual change in the MIB variable since the last query for the MIB object. This value is not time-averaged.

For example, suppose you are graphing a MIB variable with the following statistics:

Value of MIB variable at time 0 -- 100
Value of MIB variable at time 10 -- 300

The value from time 10 would be graphed in the following ways:

**Rate of Change**

20 (derived from (300–100)/10)

**Actual Sampled Value**

300

**Delta Value**

200 (derived from (300–100))

**Note:** MIB expressions containing counter values will not be affected by this setting. These MIB expression values will still be represented as a rate of change.

**Adding a Line**

An application sometimes has more lines to graph than the maximum number that are specified for the graph when the application is created. Select **Add Line** from the **View** pull-down menu to temporarily increase the number of lines the graph can display. This selection will be grayed if the number of lines to be graphed does not exceed the maximum number of lines that can be graphed, as defined in the application’s app-defaults file.

**Using the Context Menu**

The context menu in a graph application enables you to zoom in and out so you can look at the graph from different perspectives. You can also use the context menu to page forward or backward through the collected data, or to display the beginning of the data, the end of the data, or all of the data.

**Printing Graphs**

You can use the Print Tool application to print graphs. See [Printing Graphed Data](#) for these steps.
Chapter 5. Correlating, Filtering, and Configuring Events

To help you manage a network effectively, Tivoli NetView must receive information about changes that affect objects in the network. Events generated by agents that monitor network objects convey this information to Tivoli NetView.

Large networks with many objects and agents can generate so many events that the manager is flooded with traffic and must devote an excessive amount of time to processing incoming events. In addition, the manager generates events when it polls agents for the status of network objects. Event correlation rules and event filters can help you control the amount of event traffic to be displayed on the Tivoli NetView graphical interface or forwarded to the Tivoli NetView for OS/390 program for further handling.

This chapter contains the following topics:
- "Events: General Information"
- "Starting the Event Display Application" on page 106
- "Viewing the Event Log" on page 109
- "Correlating Events" on page 110
- "Creating Event Filters" on page 131
- "Activating and Deactivating Event Filters" on page 136
- "Configuring Events" on page 143
- "Displaying a Warning Window for Events" on page 146
- "Converting Events to Alerts (AIX Only)" on page 147
- "Sending Alerts to the Host Program (AIX Only)" on page 149

Events: General Information

Tivoli NetView uses the following types of events:

Map events
Notifications issued because a user or application does something that affects the status of the current map or of the Tivoli NetView graphical interface. For example, if you add a connection between a workstation and a server on a submap, an event is generated and logged in the event log file. The contents of the submap change to include the added connection.

Network events
A message sent by an agent to one or more managers to provide notification of an occurrence affecting a network object. These events are not necessarily reflected in the map. For example, if an SNMP agent is not in your management region, but is configured to send traps to the manager, you will receive events for that agent.

Logging Events

All of the events and SNMP traps received by the Tivoli NetView program are logged in the /usr/OV/log/ovevent.log file by default. From this file, the nvevents application reads the events filtered for display and displays them through the Event History application.

See "Viewing the Event Log" on page 109 for information about using the Event History application to view events stored in this file.

SNMP traps can also be logged in the /usr/OV/log/trapd.log file. This file is in ASCII format, so you can edit it to view logged traps, or print the contents of this file. If the
Tivoli NetView program is configured to work with a relational database, you can transfer trapd.log data to a relational database and use the relational database tools to create reports.

Refer to the Tivoli NetView for UNIX Database Guide for more information about transferring trapd.log data to a relational database.

**Information Provided by SNMP Traps**

SNMP defines six generic types of traps and allows definition of enterprise-specific traps. The trap structure conveys the following information to the Tivoli NetView program:

- Agent’s object that was affected
- IP address of the agent that sent the trap
- Event description (either a generic trap or enterprise-specific trap, including trap number)
- Time stamp
- Optional enterprise-specific trap identification
- List of variables describing the trap

The agent knows which manager system to send traps to by use of a user-configurable trap destination. The manager system can then retrieve more information to isolate a problem by polling the agent system.

**Information Provided by Tivoli NetView Internal Events**

Tivoli NetView internally generates events, which are treated as enterprise-specific traps. These events include the following information:

- Description of the event.
- Name of the node associated with the event in the system name (sysName) MIB variable.
  
  A node name with the value <none> refers to the manager station running the Tivoli NetView program.

To look at a list of the Tivoli NetView program’s internally generated events, enter the `/usr/OV/bin/event -l` command at the command line. See “Appendix A. Tivoli NetView Internal Traps” on page 263 for detailed information about the internal traps.

**Forwarding Events to the Tivoli Enterprise Console**

You can configure Tivoli NetView through the Tivoli desktop to forward events to the Tivoli Enterprise Console. Using the Tivoli desktop, you supply the host name on which the Tivoli Enterprise Console resides and a Tivoli NetView event correlation rule. The `nvserverd` daemon forwards only the events that pass through the specified rule to the Tivoli Event Server.

Refer to Tivoli NetView for UNIX Installation and Configuration for information about how to configure Tivoli NetView to forward events to the Tivoli Enterprise Console. See “Correlating Events” on page 110 for information about event correlation rules.

**Format of Events Forwarded to the Tivoli Enterprise Console**

Event information is formatted as a set of attributes, or slots. Each attribute is a predefined information slot that contains the attribute name and attribute value. The event adapter separates event information into event classes, formats this
information into slots, and sends this information to the Tivoli Event Server, which subsequently forwards the event information to the Tivoli Enterprise Console.

The default event class for these events is Nvserverd_Event. The class definition is contained in the /usr/OV/conf/nvserverd.baroc file. The Tivoli administrator must import the event class into the current rule base after the tecad_ov.baroc, root.baroc, and tec.baroc event classes.

The default event slot mappings are described in the following list:

<table>
<thead>
<tr>
<th>slot</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>nvserverd</td>
</tr>
<tr>
<td>sub_source</td>
<td>Tivoli NetView source character corresponding to the internal component of Tivoli NetView that generated the event</td>
</tr>
<tr>
<td>origin</td>
<td>IP address of trap origin</td>
</tr>
<tr>
<td>sub_origin</td>
<td>&lt;empty&gt;</td>
</tr>
<tr>
<td>hostname</td>
<td>Host name of Tivoli NetView</td>
</tr>
<tr>
<td>msg</td>
<td>Text description of the trap</td>
</tr>
<tr>
<td>date</td>
<td>Date Tivoli NetView received the event</td>
</tr>
<tr>
<td>severity</td>
<td>Configured severity for the trap</td>
</tr>
<tr>
<td>status</td>
<td>OPEN</td>
</tr>
<tr>
<td>nv_enterprise</td>
<td>Enterprise of the trap</td>
</tr>
<tr>
<td>nv_generic</td>
<td>Generic trap number</td>
</tr>
<tr>
<td>nv_specific</td>
<td>Specific trap number</td>
</tr>
<tr>
<td>nv_var1 through nv_var15</td>
<td>Variable bindings 1 through 15; remaining variable bindings are not forwarded.</td>
</tr>
</tbody>
</table>

You can change the event class and event slot mappings so that the event includes information that is meaningful. See "Configuring Events" on page 142 for more information.

Note: Any traps containing variable bindings that may have values equal to any of the following keywords will fail to parse correctly by the Tivoli Enterprise Console event parser. Do not set nv_var12="INTEGER". Because the nv_var1 through nv_var15 default to their corresponding variable binding, any variable binding that could possibly contain a keyword should not match an event attribute. For example, a MLMThresholdArm trap may contain "INTEGER" in variable binding 12. This, by default, maps to nv_var12. Because this may cause a parsing error in Tivoli Enterprise Console, map nv_var12 to something else.

Reserved Tivoli Enterprise Console keywords are as follows:

- DEBUG
- DEFINES
- END
- ENUMERATION
- INTEGER
- INT32
- ISA
- I_NAME
- LIST_OF
- POINTER
- REAL
Controlling the Event Adapter

When Tivoli NetView is configured to forward events to the Tivoli Enterprise Console, the event adapter is automatically started. If you want to stop forwarding events to the console, use the `/usr/OV/bin/ nvtectia -stop` command.

Note: If you restart Tivoli NetView, the event adapter will automatically restart. To permanently stop forwarding events to the console, reconfigure Tivoli NetView through the Tivoli desktop.

If you modify the ruleset you are using to forward events to the Tivoli Enterprise Console, use the `nvtectia -reload` command to reload the ruleset.

Refer to the man page for more information.

Starting the Event Display Application

The Event Display application can be started in any of the following ways:

- The Event Display application starts automatically when the Tivoli NetView program starts. It displays all events received during the current Tivoli NetView session that have been filtered for display. Only one instance of this application can be open per session to display all the filtered events of that session. However, you can open other instances to display the filtered events of that session for selected network objects.

- You can also select events for display in a separate workspace window. Events are synchronized in client machines. For example, if an event is cleared, notes are added, or the event severity and category is changed in a workspace on a client machine, the change is reflected in the workspaces on all the clients.

- Select a symbol on a submap. Then select Events -> Current Events from the Monitor pull-down menu. In this case, the application displays all uncleared events (events currently displayed in the main workspace window) associated with the selected symbol.

- Click mouse button 3 on a symbol in a submap to display the object context menu and select Monitor -> Events -> Current Events. The Event Display application displays all uncleared events associated with the selected symbol.

- Select an object or objects on the submap and drag the Events icon from the Tools window to the Control Desk window or another area of the desktop. Events for each object will be displayed in separate dynamic workspaces.

You can start Tivoli NetView without starting the Event Display application. If you are a root user, you can edit the `/usr/OV/registration/C/ovsnmp/nvevents` file and remove the `-Initial` flag from the command that initiates the Event Display application. Although you make this change, you can still start the Event Display application from the main menu, an object context menu, or the Tools window.
When Workspaces Are Automatically Created

The Tivoli NetView program automatically creates a workspace in the following circumstances:

- The first time the Event Display application is started, the workspace contains the dynamic event display for the current Tivoli NetView session. It is located in the Control Desk window. You can also start this application by dragging its icon from the Tools window.
- Each time you select a node from the submap and then select Events -> Current Events... from the Monitor pull-down menu or drag the Events icon from the Tools window, a new workspace is opened for the node you selected.
- The first time you select Events -> Event History... from the Monitor pull-down menu a new workspace that contains events from the /usr/OV/log/ovevent.log file is opened instead of events being received during the current Tivoli NetView session. See "Viewing the Event Log" on page 103 for more information about the event history application.
- When you drag an event card from either the Event Display or Event History applications.
- Each time you select Dynamic Workspace... from the Create pull-down menu in the main workspace window.

Refer to the Tivoli NetView for UNIX User’s Guide for Beginners for information about creating a dynamic workspace.

Saving the Event Workspace Configuration

You can change the following resources in the /usr/OV/app-defaults/Nvevents file so that you can save your event workspace configuration and start the Event Display application with the saved configuration:

**saveEnvOnExt**
Defines whether the Event Display application saves all workspaces. The default value is False. When you change the value for saveEnvOnExt to True, the current event workspace configuration is saved in a configuration file in the $USER/$HOME/NvEnvironment directory when you exit the Event Display application.

**loadEnvOnInit**
Defines if the Event Display application starts with a saved event workspace configuration. The default value is False. When you change the value for loadEnvOnInit to True, all saved workspaces are opened inside the Control Desk when you start the Event Display application.

**considerStaticWrkSpcs**
Defines if the Event Display application saves and loads static workspaces in addition to dynamic workspaces. When you change the value for considerStaticWrkSpcs to True, static workspaces and workspaces that were loaded using the File..Load option are saved and opened when you start the Event Display application. The default value is False, which saves and loads only dynamic workspaces. Save and load the configuration for dynamic workspaces to avoid starting the Event Display application with saved static workspaces which can cause a delay in loading the event workspace configuration.

You can override the value set for saveEnvOnExt by selecting the **Save Environment** from the **Options** pull-down menu in the Event Display application main workspace. When the value for saveEnvOnExt is True, a radio button is displayed next to the Options..Save Environment option indicating that this option is
active and the event workspace configuration will be saved when you exit the Event Display application. Select **Save Environment** from the **Options** pull-down menu to deactivate the option, and the current event workspace configuration will not be saved. If the value for loadEnvOnInit is True, the Event Display application will start with the previously saved configuration, if one exists.

Similarly, when the value for saveEnvOnExit is False, a radio button is not displayed before the **Options**..**Save Environment** option indicating that this option is not active and the event workspace configuration will not be saved when you exit the Event Display application. Select the **Options**..**Save Environment** option to activate the option and save the current event workspace configuration.

Whether the configuration is saved because the value for saveEnvOnExit is True or because you selected the **Save Environment** from the **Options** pull-down menu, the Event Display application is not started with the saved workspace configuration unless the value for loadEnvOnInit is True.

### Controlling How Events Are Displayed

You can change the following resources in the `/usr/OV/app-defaults/Nvevents` file, which control how events are displayed:

**workListDetailMode**

Controls how events are displayed when events are in list format. The valid values are:

- **0**: Double-clicking an event opens the card in a static workspace. This is the default.
- **1**: Double-clicking an event opens the card in a static workspace that is resizable.

**workCardDetailMode**

Controls how events are displayed when events are in card format. The valid values are:

- **0**: Double-clicking a card brings the card to the top of the stack. This is the default.
- **1**: Double-clicking a card opens the card in a static workspace.
- **2**: Double-clicking a card opens the card in a static workspace that is resizable.

Changing the values for the workListDetailMode and workCardDetailMode resources to **1** and **2**, respectively, enables you view all the information on the card. You can resize the workspace window to resize the card in the same proportion as the window.

### Suppressing Events from Unmanaged Nodes

To suppress events from IP nodes that are unmanaged in the open map, select **Suppress Traps from Unmanaged Nodes** from the **Options** pull-down menu in the Event Display window. This menu option is a toggle button. To resume seeing the traps, select this menu option again.

### Searching for Events

You can search for events based on predefined filtering criteria or on any or all of the following criteria:

- A search string
• An event source
• An event category
• An event severity level

Searching by Criteria
To search for events by criteria, follow these steps:

1. Select By Criteria from the Search pull-down menu in the Event Display Application.
2. Enter the search criteria in the Search String field on the Search by Criteria dialog box. You can also search for a specific event source, category, or severity.
3. If you want the search results displayed in a separate workspace, click the Create Workspace. Otherwise, the events are selected in the Event Display Application Main Workspace. Selected event cards are a darker shade than the others.
4. Click OK to start the search and close the dialog box. Events meeting the search criteria are either selected or displayed in a Static Workspace.

Searching by Filter
To search for events by filtering criteria, follow these steps:

2. Select the name of the filter from the list of Available Filters in File. You can click File List to enter changes to the list of available filters. You can also edit the selected filter by clicking Display/Edit.
3. Select Activate to activate the filter. The dialog box is closed. A workspace is created in the Control Desk window containing the filtered events.

Viewing the Event Log

The Event History application, which displays events from the /usr/OV/log/ovevent.log file, starts in one of the following ways:

• When you select Events -> Event History... from the Monitor pull-down menu.
• When you drag the Event History icon from the Tools window and drop it either in the Control Desk window or another location on the desktop

The displayed events come from the /usr/OV/log/ovevent.log file, which contains events from the current Tivoli NetView session. The number of events that can be stored in this file at one time, and potentially displayed by the event history application, depends on the maximum allowable size of the file.

See "Changing the Size of the Event Log" on page 110 for more information.

When the Event History application is started, events are not displayed until you select Display Events from the Query pull-down menu. This delay in displaying events gives you the opportunity to apply filter criteria to the query of the /usr/OV/log/ovevent.log file, so the display does not contain an unmanageable number of events. If there are too many objects selected for inclusion in the filter, the filter will not be activated, and the following message will be displayed:

Sieve creation failed. Error in Object creation.

To filter the Event History display, select Filter Control... from the Options pull-down menu. From the Filter Control dialog box, you can activate and deactivate
filters to control the event history display. You can also use the Filter Editor to create a filter if the list of available filters does not contain one that suits your needs.

See "Using the Filter Editor" on page 132 for more information.

Refer to the online help for information about the other Event History menu options.

**Changing the Size of the Event Log**

To change the size of the /usr/OV/log/ovevent.log file, select **Options..Set Log Size** from the Event History menu. The default log size is 128 KB. The maximum log size is 2 MB. Changing the default log size through the Event History menu changes the size for the current editing session only. You can configure the oveld daemon to change the size permanently through the Tivoli desktop.

Refer to *Tivoli NetView for UNIX Installation and Configuration* for those steps. The following conditions affect changing the size of the log file:

- If you specify a new size that is less than the current size of the file, the current file becomes the backup log file and a new log file is started. You might do this if you want to clear the current log file and record only events from a given time in the current session.
- If you specify a new size that is greater than the current size of the log file, events continue to be added to the current file.
- If you set the size to zero (0), event logging is turned off until you enter a positive integer. Events already logged will not be deleted unless the log file is deleted.

**Correlating Events**

You can create a ruleset that correlates or compares incoming events to event processing decisions and actions. The ruleset editor enables you to graphically create a rule comprised of event-processing decisions and actions that are represented by icons (nodes).

You can create rulesets to:

- Enhance the filtering capabilities for dynamic event displays. You can filter for MIB variables inside traps and define thresholds based on event data.
- Use object database fields to correlate events.
- Automatically remove resolved events from the Event Display application.
- Override severity and object status associated with an event.
- Automatically issue a call to a pager.

**Processing Events within a Ruleset**

The following list describes how the decision and action nodes process an event through a ruleset:

**Decision**

If the decision is determined to be true, the event is passed on to the next node (or nodes if there are multiple connections) in the ruleset. If the decision is determined to be false, processing halts on that path through the ruleset. When all processing of an event halts and the event is not passed on to the next node, the event is deleted.

**Action**

The action is passed to the `actionsvr` daemon, which starts a new process
for the specified action, and the event is passed on to the next node in the
ruleset. If the action does not complete successfully, the ACTF_EV
(59179071) action failed trap is generated. All actions requested and the
events which caused those actions are logged. Actions include:
- Forwards the event to the Event Display application
- Overriding the severity or object status associated with the event
- Resolving the event
- Issuing any operating system command, script file or executable, or Tivoli
  NetView command
- Issuing a call to a pager
- Setting a global variable to some value
- Setting a Tivoli NetView object database field
- Setting a MIB variable

Configuring the Paging Utility

To use the Tivoli NetView paging utility through an event correlation rule (using the
Pager node) or from the command line (using the nvpage command), an analog
line for modem communications. Then follow these steps for the modem that is
attached to your system:
1. Add and configure a tty device for modem communications using the UNIX
   mkdev command.
2. Test the modem communication through the tty device using a communications
   program, such as ate, provided with your operating system.
3. Customize the paging utility configuration files, which are located in the
   /usr/OV/conf directory:
   - nv.carriers
     Lists the defined carriers. Add the appropriate entries for all paging
carriers used at your site. The Numeric IDs accepted on Modem line:
     Y/N field indicates the pager type. If numeric IDs are accepted, the
     pager type is numeric. If numeric IDs are not accepted, the pager type
     is alpha.
     See the nv.carriers man page for more information.
   - *.modem
     Contains the default information for the modem. The asterisk (*)
     represents the name of the modem file. The following modem files are
     provided:
     - ibm5853.modem
       For the 2400 baud IBM Model 5853 modem
     - ibm7855.modem
       For the IBM Model 7855
     - newhayes.modem
       For most Hayes compatible modems
     - oldhayes.modem
       For Hayes compatible modems that do not understand the
       extended AT command set
     - qblazer.modem
       For Hayes compatible modems
**blank.modem**

For you to copy and customize

Usually, you do not need to change the values in the modem file. If a modem file is not provided for the modem you are using, use the blank.modem file as a template.

See the `modem` man page for more information.

**nvpager.config**

Lists the defaults that are the physical characteristics of the modem. Specify the tty device that you already configured and tested and change the modem characteristics to reflect the values configured on the tty device. Add the name of the modem file that corresponds to the modem dedicated to paging. See the `nvpager.config` man page for more information.

**nvpaging.protocols**

Defines the characteristics of the following paging protocols: TAP, IXO, PET, and PAKNET. The Protocol field in the nv.carriers file specifies the paging protocol being used and points to the nvpaging.protocols file for configuration information. If you are using a paging protocol similar to TAP, IXO, PET, and PAKNET, copy the information provided for one of these protocols and modify it with the appropriate information for the protocol you are using.

See the `nvpaging.protocols` man page for more information.

After updating the configuration files, stop and restart the `nvpagerd` daemon to make the changes available to the paging utility.

4. Create Tivoli NetView security user profiles for those individuals who you want to page automatically through an event correlation ruleset. See "Creating and Changing a User Profile" on page 28 for those steps.

When you use the Pager node in an event correlation ruleset, you specify the user ID of the person you want to page. The Tivoli NetView security user profile defines the user ID and the paging information. It is not necessary to activate security to access the paging information in a user’s profile.

You can also send a page from the command line using the `nvpage` command. See the man page for more information.

**Types of Ruleset Nodes**

The ruleset editor contains the following nodes as shown in Figure 22 on page 124:

**Action**

Specifies the action to be performed when an event is forwarded to this node. Fields from the trap being processed are available as environment variables. The specified action can be any operating system command, the full path name of any shell script or executable, or any Tivoli NetView command. Usually, the output from the specified action is displayed on the screen. If the output is not displayed on the screen, it is written in the `/usr/OV/log/nvaction.log` file. You can use this node to execute the `/usr/OV/bin/ ovxecho` command to display a dialog window when a specific event occurs.
The dialog box contains one relevant field: Action. Enter any operating system command, the full path name of any shell script or executable, or any Tivoli NetView command.

**Block event display**
Prevents events from being forwarded to the Event Display application. Use this node if you have changed the default processing action to pass (forward) events to the Event Display application and you do not want to forward events that meet specific conditions. A trap that is processed through this node is marked so that it will not be handled by the default processing action specified for the ruleset.

The dialog box does not contain any relevant fields.

**Check Route**
Checks for communication between two network nodes and forwards the event based on the availability of this communication. For example, you can use this node to check the path from the manager to a device before forwarding a node down trap.

*Note:* The check route node does not check the status of the node; it checks only the availability of the path to the node.

The dialog box contains the following relevant fields:

- **Source**
  Specifies the node where the check route starts.

- **Destination**
  Specifies the node for which you are checking the route

- **Forward Event when**
  Specifies forwarding of the event to the next node if the path is available (communication is successful) or unavailable (communication fails)

- **SNMP Defaults**
  Specifies the following SNMP defaults: community name, number of retries, remote port, and local port

**Compare MIB Variable**
Compares the current value of a MIB variable against a specified value. When a trap is processed by this node, the ruleset processor issues an SNMP GET request for the specified MIB variable.

The dialog box contains the following relevant fields:

- **MIB Variable Name**
  Specifies the fully-qualified name, including the instance number, of the variable you want to compare.

- **Object ID Source**
  Specifies the trap attribute to be used to determine the object from which to get the specified MIB variable.

- **Community Name**
  Specifies the community name to be used in the SNMP request.

- **Value to Compare**
  Specifies a literal value that will be compared to the MIB variable
value. This value must correspond with the type of MIB variable. If the type of the MIB variable is integer, for example, this value must be numeric.

**Comparison Type**
Specifies the type of comparison to be made.

You can use the **Browse MIB** button to start the MIB browser and then cut and paste information from the MIB browser into the Compare MIB Variable dialog box.

**Event Attributes**
Compares any attribute of the incoming event to a literal value. You can use this node to check for events generated by a particular device.

The dialog box contains the following relevant fields:

- **Attribute**
  Specifies the name of the attribute to be compared

- **Comparison Type**
  Specifies the type of comparison to be performed

- **Value**
  Specifies the literal value to be compared to the specified trap attribute, such as a host name

See [“Event Attribute Values” on page 122](#) for more information.

**Forward**
Forwards the event to applications that have registered to receive the output of the ruleset. A trap that is processed through this node is marked so that it will not be handled by the [default processing action](#) specified for this rule.

The dialog box does not contain any relevant fields.

**Inline Action**
Specifies the action to be performed when an event is forwarded to this node. Unlike a command specified in an [Action node](#), a command specified in an Inline Action node is not sent to the actionsvr daemon. Instead, the command is executed immediately, and processing continues to the next node if the return code of the action matches the return code you specify within the specified time period.

The dialog box contains the following relevant fields:

- **Command**
  Specifies any operating system command, the full path name of any shell script or executable, or any Tivoli NetView command.

- **Wait Interval**
  Specifies the time period, in seconds, that the ruleset processor should wait for the specified action to return. Values can be in the range of 0–999 seconds. If the wait interval is 0, the return code from the action is ignored and processing immediately proceeds to the next node. If a wait interval is specified, and the return code from the action is not received in the wait interval, it is considered to be a failure and processing does not proceed to the next node. If the action is not completed within the specified time period, processing will not proceed to the next node.
Command exit code comparison

Specifies the type of comparison you want to make.

Exit Code

Specifies the return code value from the specified action that you want to use in the comparison.

Override

Overrides the object status or severity assigned to a specific event and updates applications that have registered to receive the output of the ruleset. The Event Display application is registered to receive the output. For example, you can use this node to change the severity to Major when a node down event is received for a router. Use this node with the Query Database Field node to override status or severity for specific device types.

The dialog box contains the following relevant fields:

Status

Specifies the new object status to be associated with this event or select no override if you do not want to change the status. The Event Display application updates the object status to this value.

Severity

Specifies the new severity level to be used for this event or select no override if you do not want to change the severity level.

A trap that is processed through this node is marked so that it will not be handled by the default processing action specified for this rule.

Pager

Issues a call to a pager that has been defined in a Tivoli NetView user profile. You should have already configured the paging utility. See "Configuring the Paging Utility" on page 111 for those steps.

The paging utility will use the pager number and carrier information defined in the user profile.

The dialog box contains the following relevant fields:

User ID

Specifies the Tivoli NetView user ID of the person to be paged. If pager information is not found in the Tivoli NetView user profile or there is no Tivoli NetView ID for the user, a dialog box is displayed in which you can enter the User ID and pager information. Then a user profile is created or updated.

Message Text

Specifies message text to be delivered with the page. The message can include trap data passed in environment variables. See "Environment Variables for Trap Data" on page 123 for more information.

Pass on Match

Compares some attribute of the event being processed with an attribute of all traps received in a specified period of time.

You can specify one or a maximum of ten attribute comparisons. The Attribute List section contains the list of attribute comparisons that you want to evaluate. When you define an attribute comparison in the dialog box and click Add the attribute comparison is added to the list. The attribute comparisons will be examined in the order in which they are listed in the Attribute List. If the first pair of attributes matches, then the second pair of
attributes is examined, and so on through the list. Click Reorder to change the order in which the attribute comparisons are evaluated.

You can stop processing when the first incoming event matches the criteria you defined, or you can continue processing to find matches in all incoming events received in the specified time period. Specifying multiple attribute comparisons on multiple events can be used with the Resolve node to resolve all interface down or node down events for a specific device when an interface up or node up event is received for that device. When the attribute specified in Event 1 Attribute, in the current trap, matches the attribute specified in Resetting Event Attribute for all attribute comparisons in the Attribute List, in any trap received during the specified time period, processing continues to the next defined node.

The dialog box contains the following relevant fields:

**Event 1 Attribute**
Specifies the name of the attribute in the first event.

**Comparison Type**
Specifies the type of comparison to be performed.

**Resetting Event Attribute**
Specifies the name of the attribute in the second event. Click the Add button to add the attribute comparison to the Attribute List.

**Event Retention**
Specifies the length of time the first event will be held to wait for the second event. The maximum event retention value is 999 hours, 59 minutes, 59 seconds. If an attribute match is found between two events within the specified period of time, processing continues to the next node in the ruleset.

**Match on multiple events**
Controls whether processing stops when the first incoming event received in the specified time period matches all the criteria in the Attribute List or whether processing continues to find matches in all incoming events received in the specified time.

To modify an attribute comparison, follow these steps:

1. Select the attribute comparison you want to change in the Attribute List. The comparison values are entered in the Event 1 Attribute, Comparison Type, and Resetting Event Attribute fields.
2. Change the fields as necessary and click Modify. The comparison values in the Attribute List are updated.

To change the order in which the attribute comparisons are evaluated, select the comparisons you want to move in the Attribute List and click the up or down arrows on Reorder to move the comparisons up or down one position.

To delete an attribute comparison, select the appropriate attribute comparison in the Attribute List and click Delete.

See "Event Attribute Values" on page 122 for more information.

**Query Database SmartSet**
Tests whether the node is a member of the specified SmartSet.

The dialog box contains the following relevant fields:
**Object Source ID**
Specifies the source of the object ID to be used in querying the database.

**SmartSet Name**
Specifies the name of the SmartSet to be tested. Click on Select... button to display a list of all currently defined SmartSets. You can also specify a name of a SmartSet that does not yet exist but you intend to create. See "Defining and Managing SmartSets" on page 76 for more information.

**Forward event when**
Specifies forwarding of the event to the next node if the node is contained in the specified SmartSet or if the node is not contained in the specified SmartSet.

**Query Database Field**
Compares a value from the Tivoli NetView object database to a literal value or to a value contained in the incoming event. You can use this node to check if the originating device is a router.

The dialog box contains the following relevant fields:

**Field Name**
Specifies the name of the database field to be queried. Enter the name of the database field or click the Select.

**Object ID Source**
Enter the fully-qualified object name of the device whose field is to be queried or click Select.

**Comparison Type**
Specifies the type of comparison to be performed.

**Compare Field to**
Specifies either a literal value or an attribute value to be used in the comparison.

**Reset on Match**
Compares some attribute of the event being processed with an attribute of all traps received in a specified period of time. This node is similar to the Pass on Match node, except that if a match is found, the event is not passed on to the next node in the ruleset and processing stops.

You can specify one or a maximum of ten attribute comparisons. The Attribute List section contains the list of attribute comparisons that you want to evaluate. When you define an attribute comparison in the dialog box and click Add, the attribute comparison is added to the list. The attribute comparisons will be examined in the order in which they are listed in the Attribute List. If the first pair of attributes matches, the second pair of attributes is examined, and so on through the list. Click Reorder to change the order in which the attribute comparisons are evaluated.

You can stop processing when the first incoming event matches the criteria you defined, or you can continue processing to find matches in all incoming events received in the specified time period. Specifying multiple attribute comparisons on multiple events can be used to check for all interface down or node down events for a specific device when an interface up or node up event is received for that device. When the attribute specified in Event 1 Attribute, in the current trap, matches the attribute specified in Resetting...
Event Attribute for all attribute comparisons in the Attribute List, in any trap received during the specified time period, processing through the ruleset stops.

You can use this node to discard events before they are forwarded to the Event Display application. You might find this node useful for events that are generated from a device that frequently goes up and down. The dialog box contains the following relevant fields:

**Event 1 Attribute**
Specifies the name of the attribute in the first event.

**Comparison Type**
Specifies the type of comparison to be performed.

**Resetting Event Attribute**
Specifies the name of the attribute in the second event.

**Delay Time**
Specifies the length of time the first event will be held to wait for the second event. The maximum event retention value is 999 hours, 59 minutes, 59 seconds. Click **Add** to add the attribute comparison to the Attribute List.

If an attribute match is found between two events for all attribute comparisons in the list within the specified period of time, the event is not forwarded to the next node in the ruleset and processing stops.

**Match on multiple events**
Controls whether processing stops when the first incoming event received in the specified time period matches all the criteria in the Attribute List or whether processing continues to find matches in all incoming events received in the specified time.

To modify an attribute comparison, follow these steps:
1. Select the attribute comparison you want to change in the Attribute List. The comparison values are entered in the Event 1 Attribute, Comparison Type, and Resetting Event Attribute fields.
2. Make the necessary changes to these fields and click **Modify**. The comparison values in the Attribute List are updated.

To change the order in which the attribute comparisons are evaluated, select the comparisons you want to move in the Attribute List and click on the up or down arrows on the **Reorder** button to move the comparisons up or down one position.

To delete an attribute comparison, select the appropriate attribute comparison in the Attribute List and click **Delete**.

See [Event Attribute Values on page 122](#) for more information.

**Set Database Field**
Sets the value of any Tivoli NetView non-Boolean object database field. Fields that have TRUE or FALSE values cannot be changed.

The dialog box contains the following relevant fields:
Field Name
Specifies the name of the field in the object database that you want to change

Object ID Source
Specifies the source of the object ID to be used in selecting an object from the object database

Set Value to
Specifies either a literal value or an event attribute value to be used for the database field setting

See “Event Attribute Values” on page 122 for more information.

Query Global Variable
Queries the value of the global variable that has been previously set using the Set Global Variable node.

The dialog box contains the following relevant fields:

Variable Name
Specifies the name of the variable you are checking. The variable is created and assigned a value using the set global variable node.

Comparison Type
Specifies the type of comparison to be performed.

Compare Variable to
Specifies either a literal value or an event attribute value to be used in the comparison

See “Event Attribute Values” on page 122 for more information.

Set Global Variable
Sets a variable for use within the ruleset. For example, use this node to set a flag whose value will be checked later in the ruleset using the Query Global Variable node. When the ruleset is finished processing, the global variable is no longer in effect.

The dialog box contains the following relevant fields:

Variable Name
Specifies a user-defined text string associated with the value of the variable, such as, flag.

Set Variable
Specifies one of the following settings:

- Increment Value by One
  If the global value has already been set, the value will be increased by one. If the global variable has not yet been set, the value will be set to one.

- Decrement Value by One
  If the global value has already been set, the value will be decreased by one. If the value has not yet been set, the value will be set at negative one.

- Set to Literal Value
  Enter a text value in this field.

- Set to Attribute Value
  Click Select to select a trap attribute.
See “Event Attribute Values” on page 122 for more information.

Set MIB Variable
Issues an SNMP SET command to set the value of a variable in the MIB representing any network resource. For example, you can use this node to change the system contact for a particular device.

The dialog box contains the following relevant fields:

MIB Variable Name
Specifies the fully-qualified name, including the instance number, of the variable you want to change.

Variable Data Type
Specifies the type of data to be placed in the MIB field, such as integer, string, and so on.

Object ID Source
Specifies where to get the object ID whose MIB is to be changed.

Community Name
Specifies the community name of the object whose MIB is to be changed.

Value to be Set
Specifies a literal value to be used as the data for the MIB field value. The data type of this value must match the type specified in the Variable Data Type field.

See “Event Attribute Values” on page 122 for more information.

Click Browse MIB to start the MIB browser and then cut and paste information from the MIB browser into the Set MIB Variable dialog box.

Resolve
Forwards a message to all registered applications indicating that a previous event has been resolved. By default, the Event Display application is registered to receive the output from rulesets. The receiving application determines how to handle a trap that has been forwarded from this node. This node is frequently used in conjunction with the Pass on Match node. You can use the Resolve node to delete an interface or node down event from the Event Display application when an interface or node up event is received. A trap that is processed through this node is marked so that it will not be handled by the default processing action specified for the ruleset.

The dialog box does not contain any relevant fields.

Set State
Sets the correlation state of an object in the Tivoli NetView object database. The current state is updated in the corrstat1 field in the object database, and the previous value in the corrstat1 field is moved to the corrstat2 field. This process continues until the current state and as many as four previous states are stored in the object database. You can view the correlation state by selecting the object and then selecting the Display Correlation Status option from the context menu.

The dialog box contains the following relevant fields:

State Value
Specifies the text string that you want to store in the corrstat1 field of the specified object.
Object ID Source
Specifies the source of the object ID to be used in selecting the object in the object database.

See "Event Attribute Values" on page 122 for more information.

Thresholds
Checks for repeated occurrences of the same trap or of traps with one or more attributes in common. You can use this node to forward an event after receiving the specific number of the same event received within a specific time period. Use this node with the Trap Settings node to identify a specific trap number. The dialog box contains the following relevant fields:

Type  Specifies when the event should be forwarded by selecting one of the following values:

First  When a threshold condition is reached, forwards the first \( n \) traps to the next node, where \( n \) is the number specified in the Count field.

At  When a threshold condition is reached, forwards the \( n \)th trap to the next node, where \( n \) is the number specified in the Count field.

After  When a threshold condition is reached, forwards all traps after the \( n \)th trap to the next node, where \( n \) is the number specified in the Count field.

Count  Specifies the number of traps required to reach the threshold condition.

Time Period  Specifies the length of time within which the number of events specified in the Count field must be received to reach the threshold condition. Use this field in conjunction with the Time Unit field.

Time Unit  Specifies the unit of measure (minutes, seconds, hours, or days) for the number specified in the Time Period field.

Threshold by Attribute 1—Threshold by Attribute 9
By default, the threshold depends on the trap ID. If you want to define on one or more trap attributes, select the Threshold by Attribute 1 button and then click on the Select...button to specify the first attribute in the trap to be used. Use the scroll bar in the Threshold by Attribute panel to define up to nine trap attributes. For each attribute you want to define, click the appropriate Threshold by Attribute button and then select the trap attribute to be used.

Threshold by Attribute 1  Specifies the first attribute in the trap to be used. By default, the threshold depends on the trap ID. Select Threshold by Attribute 1 and then click Select to choose the first attribute to be used.

Threshold by Attribute 2  Specifies the second attribute in the trap to be used. Select Threshold by Attribute 2 and then click Select to choose the second attribute to be used.
**Threshold by Attribute 3**
Specifies the third attribute in the trap to be used. Select **Threshold by Attribute 3** and then click **Select** to choose the third attribute to be used.

See [Event Attribute Values](#) for more information.

Assume that you specify a time of 5 minutes in the Time Unit field and 10 for the number of traps in the Count field. The following list describes the use of First, At, and After in the Type field.

**First**  A burst of 20 traps is received in 5 minutes. Only the first 10 traps will be forwarded.

**At**  A burst of 10 traps is received in 1 minute. Only the 10th trap is forwarded. If traps continue to arrive at one per minute thereafter, no other traps will be forwarded because traps arriving at one per minute do not meet the threshold again. The threshold is calculated by considering the current event plus past events that are received within the same time period.

**After**  A burst of 10 traps is received in 1 minute. One trap is received every minute thereafter. Each trap received in minute 2, 3, 4, and 5 will be forwarded. Similar to above example, no more traps will be forwarded after minute 5 because traps arriving at one per minute do not meet the threshold again.

**Trap Settings**
Specifies a specific trap to be processed and is identified by a pair of generic and specific trap numbers.

The dialog box displays a list of enterprise names and IDs. When you select an enterprise ID, a list of generic and specific trap numbers for that enterprise is displayed in the Event Name and Specific fields. Select one or more traps from this list. The description of the trap you select is displayed in the Trap Description field. Click **Comparison Type** to specify the type of comparison to be performed (equal to or not equal to).

**Event Attribute Values**
Use the following event attribute values to identify the event to be processed:

**Severity**  Specifies the severity of the event, such as Critical.

**Category**  Specifies the type of event, such as a status event.

**Source**  Specifies the internal component of Tivoli NetView that generated the event, such as netmon.

**EnterpriseID**  Specifies the enterprise that sent the trap.

**Origin**  Specifies the host name generating the trap. The management station is the origin for traps generated as a result of Tivoli NetView polling operations.

**Generic**  Specifies the generic trap value defined by SNMP.
Specific
   Specifies the specific trap value defines by SNMP.

sysObjectID
   Specifies the MIB object describing the agent hardware, software, and so
   forth.

sysUpTime
   Specifies the MIB system running time since the agent has been started.

Community Name
   Specifies the community name.

1-50   Trap variable which varies by enterprise or trap.

You can use one word for a trap attribute. A word is a unit of text separated by
blanks. To specify the word to be used, add a period (.) and the number of the
word. For example, to use the second word of the third variable binding, specify
3.2. The ruleset processor will use the string that starts after the first blank and
ends with the second blank.

Tivoli NetView internal traps use variable bindings 1 through 5 as follows:

1   Specifies the source ID and is a integer value that corresponds to the
    internal component of Tivoli NetView that generated the event, such as
    netmon.

2   Specifies the host name to which this trap applies.

3   Specifies the event description and is a string value containing a description
    of the event that was generated.

4   Specifies specific trap data and is a string value containing internal data
    that is specific to the type of trap that is generated.

5   Specifies the database name and must be openview.

See [Appendix A. Tivoli NetView Internal Traps on page 263 for more information
about Tivoli NetView internal traps.

Environment Variables for Trap Data
You can specify trap data using the following environment variables:

NVE   Specifies the enterprise ID

NVA   Specifies the agent address

NVG   Specifies the generic trap number

NVS   Specifies the specific trap number

NVT   Specifies the time stamp

NVC   Specifies the community name

NVATTR_<1-50>
   Specifies the MIB attribute where 1-50 is the variable binding number.

These environment variables are frequently used with the Action node and the
Pager node. For example, you might include a pager message similar to the
following:

Multiple authentication failures for $NVA.
Sample Rulesets

The following sample rulesets are provided:

**corrNdNu.rs**
Forwards a node down trap to nvevents and clears the event if a node up trap is received for the same device within 10 minutes.

**corrIdIu.rs**
Forwards an interface down trap to nvevents and clears the event if an interface up trap is received for the same device within 10 minutes.

See [Activating a Ruleset on page 127](#) for information about how to activate a ruleset.

Creating and Editing a Ruleset

To create a ruleset, use the ruleset editor as shown in **Figure 22**.

![Figure 22. Ruleset Editor](image)

You must be a root user to start the ruleset editor. To start the ruleset editor, use one of the following methods:

- Select **Ruleset Editor...** from the **Tools** pull-down menu.
- Enter the **nvrsEdit ruleset_name** command at the operating system command line, where *ruleset_name* is the name of the ruleset you want to create or change.
- Double click on the Ruleset Editor icon on the tool palette.
- Drag and drop the Ruleset icon from the Tools window onto the desktop.
The ruleset editor is divided into two windows:

**Ruleset**
Contains the Event Stream icon, which represents all incoming events and is the work area for creating rulesets. The window title bar contains the name of the ruleset you are currently editing. If you started the ruleset editor without specifying a ruleset name, the window's title bar contains the name `default.ruleset`. You can change the name of the ruleset when you save the ruleset or you can edit an existing ruleset by selecting the File..Open option from the ruleset editor menu bar. Other menu bar operations enable you to create, modify, or delete a ruleset.

**Templates**
Contains the nodes that you can use to create a ruleset. Each node represents either a decision node or an action node.

### Changing the Default Processing Action
You can change the default processing action, which defines what is done with a trap after it has been processed through a ruleset. The default processing action is not used for traps that have been marked by being processed through one of these nodes:

- **Block event display**
- **Forward**
- **Override**
- **Resolve**

If the trap is not processed through a Block event display, Forward, Override, or Resolve node, the trap can be passed on to interested applications or discarded. The default processing action is **Block**, which means that the trap is not forwarded (passed) to applications that have registered to receive the output of the ruleset, such as, the Event Display application.

To change the default processing action so that events are forwarded to registering applications, double click on the Event Stream icon and click **Pass**. You can override forwarding specific events that have passed through a ruleset by using the Block event display node, or you can reset the default action for all events that pass through the ruleset by selecting the **Block** button.

### Adding a Node
To add a node to a ruleset, drag and drop the appropriate node from the template area onto the work area and then connect the nodes. You can also select the appropriate node from the New pull-down menu bar option. If you maximize the work area window size, select **Focus to Templates** from the Edit pull-down menu in the Ruleset Editor main window to keep the Templates window in front of the work area window. The Focus to Templates option is also available from the context menu that is available in the background area of the work area. When two or more decision nodes are connected sequentially (in a straight line), the logical operator AND is used. When two or more decision nodes are connected in parallel from a single decision node, the logical operator OR is used.

When you drop a node into the work area, a dialog box is displayed that contains relevant data fields for the decision or action to be performed. Complete the dialog box fields and click **OK** to add the node to the ruleset. The description field in each dialog box is optional. You can use this field to document the decision or action taken at the node.
You can double-click a node at any time or select **Edit** from the context menu on a node to display its dialog box and view or modify the data.

**Connecting Two Nodes**
Connect the nodes in the ruleset to define the logic path through the ruleset. Connect decision nodes sequentially (in a straight line) to use the logical operator **AND**. Connect two or more decision nodes in parallel from a single decision node to use the logical operator **OR**.

Use one of the following methods to connect nodes:
- Select **Connect Two Nodes** from the **Edit** pull-down menu in the Ruleset Editor main window. Then select the nodes you want to connect in the work area.
- Select the node in the work area to which you want to connect another node. Then drag and drop another node into the work area. The connection will be drawn automatically.

Select the "from" node first and then the "to" node so that the event flow through the ruleset is from left to right.

**Deleting a Node**
To delete a node from a ruleset, use one of the following methods:
- Select **Delete Node** from the **Edit** pull-down menu in the Ruleset Editor main window and then select the node.
- Select the node and then select **Delete Selected** from the **Edit** pull-down menu.
- Select the node and press the **Delete** key.
- Select **Delete** from the context menu on the node.

The node and the connection to the node is deleted.

**Deleting a Connection**
To delete a connection, use one of these methods:
- Select **Delete Connection** from the **Edit** pull-down menu in the Ruleset Editor main window and then select the connection.
- Select the connection and then select **Delete Selected** from the **Edit** pull-down menu.
- Select the connection and press the **Delete** key.
- Select **Delete** from the context menu on the connection.

**Inserting Another Ruleset**
You can insert another ruleset into the ruleset you are currently editing. You might find this useful for combining rulesets or building complex rules out of existing rulesets. To insert another ruleset, follow these steps:

1. If you do not want the inserted ruleset connected to the Event Stream icon, select the node to which you want to connect the inserted ruleset. If you do not select a node, the ruleset is connected to the Event Stream icon.
2. Select **Insert...** from the **File** pull-down menu. A dialog box is displayed containing a list of existing rulesets.
3. Select the name of the ruleset you want to insert and click **OK**. The ruleset is inserted into the current ruleset, and the layout is recalculated.

**Saving a Ruleset**
When you are finished editing a ruleset, save it by selecting **File..Save** from the ruleset editor menu bar. If you want to save the ruleset with a different name than
the one that is displayed in the title bar of the ruleset window, select **Save As...** from the **File** pull-down menu in the Ruleset Editor main window. Then enter a name for the ruleset and click **OK**. The name of the ruleset must include a .rs extension. You might name a ruleset that resolves node down and node up traps from the save device as `resolve.rs`. Rulesets are stored in the `/usr/OV/conf/rulesets` directory.

**Activating a Ruleset**

To activate a ruleset, create a new dynamic workspace. For each dynamic workspace, you can activate only one ruleset, and you can activate one or more filters.

Select **Dynamic Workspace...** from the **Create** pull-down menu in the main workspace and enter the ruleset name and any other appropriate information. The new workspace uses the ruleset and filters, if any, to determine which events are displayed. If you edit a ruleset while it is active, close and reopen the dynamic workspace window to put the changes into effect. Select the **Help** button on the Dynamic Workspace dialog box for information about the dialog box fields.

You can activate one or more rulesets for automatic action when you start Tivoli NetView by editing the `/usr/OV/conf/ESE.automation` file, adding the names of the rulesets on separate lines. Use this method for rulesets that perform a specific action, such as a call to a pager when a particular device goes down, and you do not want to display the events.

**Attention:** Do not set the default processing action as Pass in the Event Stream icon or include a Forward node in rulesets that you start from the ESE.automation file. Forwarded events are passed to the `actionsvr` daemon, which has no events display and no mechanism to dequeue the events. Thus, the socket connection to the `nvcorrd` daemon will fill up and cause the nvcorrd daemon and all `nvevents` windows to hang.

**Testing a Ruleset**

When you have created and activated a new ruleset, use the **Diagnose** → **Send event to trapd daemon...** option available through the Tivoli desktop to send the appropriate traps and test the results of the ruleset in the workspace you created.

**A Threshold Example**

Suppose you want to monitor a specific router. You can create a ruleset to display authentication failure events after a minimum of five events have been received from the router within one minute. You also want to display a message box indicating that multiple authentication failure events have been received from the router. Figure 23 on page 128 shows how the completed ruleset looks.
Here are the steps to create and activate this ruleset:

1. Select **Tools..Ruleset Editor** from the Tivoli NetView menu bar. The ruleset editor is displayed.
2. Drag and drop the Trap Settings node into the work area.
3. Complete the dialog box as follows:
   a. Select an Enterprise ID of ENTERPRISES, because this is a generic trap
   b. Select Generic Trap 4 - Authentication Failure
   c. You can add a description similar to the following to describe the purpose of this node:
      
      Watch for authentication failure traps.
   d. Click **OK**.
4. Connect the Trap Settings node to the event stream node.
5. Drag and drop the Event Attributes node into the work area.
6. Complete the dialog box as follow:
   a. Select **Origin** in the Attribute field.
   b. Select **Equal To** for the comparison type.
   c. Type the fully-qualified host name of the router, for example: route1.raleigh.ibm.com.
   d. You can add a description similar to the following to describe the purpose of this node:
      
      Watch for authentication failures from router1.
   e. Click **OK**.
7. Drag and drop the threshold node into the work area.
8. Complete the dialog box as follows:
   a. Select **After** in the Type field, enter **5** in the Count field and **1** minute in the Time field.
   b. Click **Threshold by attribute** and select **Origin** for the attribute, because you are checking for events from the same device.
   c. You can add a description similar to the following to describe the purpose of this node:
Watch for 5 authentication failures from the router1 within 1 minute.

d. Select the OK button.

9. Connect the Threshold node to the Trap Settings node.

10. Drag and drop the forward node into the work area.

11. Complete the dialog box as follows:
   a. You can add a description similar to the following to describe the purpose of this node:
      
      Display event after five authentication failures are received from the same device within 1 minute.
   
   b. Click OK.

12. Connect the forward node to the threshold node.

13. Drag and drop the action node into the work area.

14. Complete the dialog box as follows:
   a. In the action field, enter the following:
      
      /usr/OV/bin/ovxecho Multiple Authentication Failures for $NVA
   
   b. You can add a description similar to the following to describe the purpose of this node:
      
      Use ovxecho to display a notice when more than 5 authentication failures are received within 1 minute.
   
   c. Click OK.

15. Connect the action node to the threshold node.

16. Click File..Save As to save the ruleset. Enter a name for the ruleset, such as router1.threshold.rs and click OK.

17. Activate the ruleset by creating a new dynamic workspace. Select Create..Dynamic Workspace from the main workspace menu bar. Enter the name of the ruleset and any other appropriate information.

More Examples of Rulesets

Here are a few more examples of rulesets you might create. Each example includes the objective of the ruleset and a picture of how the completed ruleset would look.

**Example 1:** Create a ruleset that forwards a node down trap to the Events Display application and clears the event if a node up trap is received for the same device within 30 minutes.
Example 2: Create a ruleset that changes the system contact for all routers.

Example 3: Create a ruleset that pages the appropriate individual when an IBM router or a Cisco router goes down.

This ruleset does not display the node down traps. To activate this ruleset, add the name of the ruleset to the /usr/OV/conf/ESE.automation file.
Creating Event Filters

Many events arrive at the management station on a network. Event filters are sets of criteria that determine the following conditions:

- Which events are dynamically displayed by the Event Display application.
- Which logged events are displayed by the Event History application.
- Which events are received by applications that register to receive them.
- Which events are forwarded to a host program as alerts (AIX only). These filters are called trap-to-alert filters.

See [Activating a Trap-to-Alert Filter (AIX Only) on page 139](#) for more information about using these filters.

Filtering events involves setting up criteria that an event must meet before it can be displayed or sent to another application. The data is then stored in the default directory, /usr/OV/filters, or in a directory that you create. To access filters, the Tivoli NetView program must know the location of the directory containing the filter files.

You can create more than one filter file, and each filter file can contain one or more filters. Criteria for the same event can also be placed in several filters.

Types of Filters

Tivoli NetView enables you to create the following types of filters:

**Simple filters**
Expressions that include SNMP criteria and can be stand-alone. They can be edited using the Simple Filter Editor.

**Compound filters**
Expressions that are composed of several simple filter expressions. They use nested parentheses to group simple expressions and combine them with the logical operators AND, OR, and NOT. They can be edited using the Compound Filter Editor.

When multiple simple filters are activated, the logical OR is used. That means that traps meeting either set of filter criteria will be received. Trap exclusion is more effectively implemented by combining the two simple filters using the AND operator to create one compound filter. Then, activate the compound filter.

Accessing the Filter Editor

You can access the filter editor in any of the following ways:

- Select [Filter Editor](#) from the [Tools](#) pull-down menu.
- Enter the `filtered` command as follows:
  ```
  filtered -f <filename> /\*name of filter file */
  -r <rulename> /\*name of filter to edit*/
  -e /\*edit option; default=display*/
  ```

  If you want only to display a filter, you can omit the `-e` option. To edit a filter, use the `-e` option.

- Click [Options..Filter Control](#) in the Event Display application window. The Filter Control dialog box enables you to activate, deactivate, display, and edit filters.
Using the Filter Editor

The filter editor helps you perform the following tasks:

- Create the criteria for events you want to see.
- Organize filters in files that you specify.
- Define a threshold that enables you to modulate the number of events received over a period of time.
- Define filters that can be used independently by different applications.

When you select **Filter Editor...** from the **Tools** pull-down menu, the Filter Editor dialog box is displayed, as shown in [Figure 24](#).

![Figure 24. Filter Editor Dialog Box](image)

The name of a filter file is displayed at the top. You can work with the filters in this file or select the **File List** button to display a list of filter files from which you can choose. Once you have selected a filter file, the names and brief descriptions of all filters in the file are displayed in the bottom part of the window.

To see the contents of a particular filter, select it and select the **Display** button to the right of the list of filters. You might want to look at a filter to determine whether it is one that you want to use as a base for creating another filter or to combine it with other filters.

Click the **Add Simple** to display the **Simple Filter Editor** dialog box, or click **Add Compound** to display the **Compound Filter Editor** dialog box.

See [Creating Simple Filters](#) on page 133 and [Creating Compound Filters](#) on page 135 for information on these filter operations.

You might decide, after looking at a filter, that you no longer need it. In that case, click **Delete** to discard the filter.

To change a filter, select it from the list and select the **Modify** button. If the selected filter can be edited using the Simple Filter Editor, that editor is displayed. Otherwise, the Compound Filter Editor is displayed.
You can organize your files on a per-application basis by storing the same filter in more than one file. Use the Copy to File operation to copy a selected filter to a different file.

**Note:** When you create a filter, make sure that you do not include more than 250 criteria. Filters used by the Event History application are limited to 40 criteria. This maximum is the sum of all choices for the following criteria:

- Enterprises selected
- Traps selected
- Objects in the IP address list
- Selection of the event logged time criteria
- Selection of frequency parameters
- Logical operators (AND, OR, and NOT)

**Creating Simple Filters**

Suppose you want to create a simple filter that sends an enterprise-specific event from the Tivoli NetView program to your events display. Follow these steps:

1. Select **Filter Editor...** from the **Tools** pull-down menu.
2. In the Filter Editor window, click **Add Simple** to display the Simple Filter Editor dialog box, as shown in [Figure 25](#).
3. Complete the Filter Name and Description fields.
4. You want to receive one particular event, so select **Events Equal to Selected** in the Event Identification section of the Simple Filter Editor dialog box.

![Simple Filter Editor Dialog Box](#)

[Figure 25. Simple Filter Editor Dialog Box]
5. Click **Add/Modify** to display the Enterprise Specific Trap Selection dialog box. From the list of available enterprises.

The enterprises in this list are those that are configured in the /usr/OV/conf/C/trapd.conf file. Once you make a selection from the enterprise list, all generic and specific traps associated with that enterprise are displayed in the Available Trap Types field. The default is that all traps from the selected enterprise are included as part of the filter. However, you can modify this list.

6. If you know which enterprise-specific trap you want to filter, select it from the Available Trap Types list by clicking on it and then clicking **Select**. The selected trap is displayed in the Selected Trap Types field. If you don’t know which trap you want, go back to step 5.

If you want to add an enterprise-specific trap, enter an enterprise-specific trap number in the Specific Trap Number field and click **Add To List**. Click **OK** button to apply your changes and close the dialog box.

7. In the Object Identification section of the Simple Filter Editor dialog box, select From Objects Equal to List. If you selected your workstation’s symbol on a submap before you opened the Filter Editor, you can click **Add From Map** to add your workstation to the List of Objects field. Otherwise, enter either the name or the IP address of your node to the Name or IP Address field, and select the **Add to List** button.

8. You can specify time and date ranges during which this event is to be sent from your workstation to the Tivoli NetView program. For example, if you want to receive this event between 8:30 and 10:30 a.m. today, complete the Time Range section as follows:

```
TIME RANGE
Time (HH:MM:SS) Date (DD:MM:YY)
Start 08:30:00
Stop 10:30:00
```

The default date is today’s date. If you do not specify any date or time ranges, and the filter is activated, this event is displayed every time it occurs. If you only specify a time, the date defaults to today’s date, and you have to reactivate the filter each day.

9. To set a threshold that specifies how many times this event may occur before notification is sent to the Tivoli NetView program, complete the text fields in the Threshold section of the Simple Filter Editor dialog box.

For example, suppose you want 5 to be the maximum number of occurrences per minute of this event that can be generated without displaying the event in the nvevents application. Enter 5 in the Frequency field and select **Less Than or Equal To**. Enter 60 in the Time Interval (Seconds) text field.

10. Click **OK** to close the Simple Filter Editor dialog box.

11. If you want to look at the filter you just created, select the filter name from the list in the Filter Editor window and click **Display**.

12. To activate your filter for the Event Display application, select **Filter Control...** from the **Options** pull-down menu in the Event Display application window. Select your filter from the list of available filters and click **Activate**. Now, only those events that match your filter criteria will be displayed. When activating more than one simple filter, the logical operator OR is used. If you want the criteria of several filters to be used when filtering events, use the Compound Filter Editor to combine the simple filters.

13. To select the filtered events, first select **Filter** from the **Search** pull-down menu in the Event Display application window. Next, select the filter name and click
Activate. The search operation highlights all events that match the filter criteria. You can also create a separate workspace in which to store these events.

For more information about creating simple filters or the different sections of the Simple Filter Editor dialog box, refer to the Help system.

### Creating Compound Filters

Compound filters are created by joining several existing simple filters or several simple expressions with logical operators. These filters can also be used to specify CMIS expressions that cannot be specified using the Simple Filter Editor. To create a compound filter, use the Compound Filter Editor, as shown in Figure 26.

Follow these steps:

1. Select **Add Compound** from the Filter Editor window.
2. Specify a filter name and description for the compound filter.
3. In the Filter Expression field, either type a filter expression or click **Get Filter...**.
4. If you click **Get Filter...**, the Get Filter Dialog box is displayed. Select a filter in one of the following ways:
   - To retrieve the name of a filter, **Get Filter Name**.
   - To retrieve the contents of a filter, **Get Filter Contents button**.

   If you need to modify the filter in any way, retrieve its contents from the Get Filter dialog box. Otherwise, if you plan to use the filter without making changes, get only its name.

   Click **OK** to close the Get Filter Dialog box. The selected filter or filter name is displayed in the Filter Expression field of the Compound Filter Editor dialog box. Now you can edit this filter expression or combine it with others.

5. Select one of the following logical operators to combine filter expressions:
• AND, which places && at the end of the current filter expression. Combining two filter expressions with AND means that events must match all criteria of both expressions.
• OR, which places || at the end of the current filter expression. Combining two filter expressions with OR means that events must match the criteria of at least one filter expression, but need not match the criteria of both expressions.

6. Either select another filter or enter some filter criteria in the Filter Expression field. You can combine several expressions in one filter.
7. Click OK to close the Compound Filter Editor dialog box.
8. If you want to save your compound filter to another filter file, click Save as... and either select another file from the list or enter a new path name in the Selection field. Click OK to close the File Selection dialog box. Otherwise, click the OK button to close the Compound Filter Editor dialog box.
9. Click Close to close the Filter Editor window.

For more information about creating and using compound filters, refer to the online help.

Activating and Deactivating Event Filters

Filters must be activated to affect the destination and the display of events. Once a filter is activated, all events matching that filter pass through the active filter to the registered applications. Events that do not match the criteria are not permitted to pass through the filter. Therefore, filtering the events will reduce the number of events being displayed.

You can activate and deactivate filters for the Event Display application, the Event History application, a dynamic workspace, and the trap-to-alert conversion process. Being able to change the activation status of a filter gives you increased control over the number and type of events that are received and displayed. However, make sure that you understand the effect of activating or deactivating a particular filter before you make the change.

Filters can also be activated and deactivated programmatically. Refer to the Tivoli NetView for UNIX Programmer’s Guide for information about the event filtering API.

Filtering Events for Display in the Main Workspace

You can select which events are displayed in the main workspace or the Event History workspace by activating filters.

To activate a filter, follow these steps:
1. Select Filter Control... from the Options pull-down menu in the Event Display or Event History application’s window.
   The Filter Control dialog box is displayed.
The Filter Control dialog box contains two sections. On the left is the **Available Filters in File** section, which lists each filter in the selected file and provides a short description. On the right is the **Active Filters List**, which displays the name, description, and full path name of all filters that are currently active.

2. If you want to select a different input file, click **File List** and select another filter directory or file from the File Selection dialog box, shown in Figure 28 on page 138.

To see the file in the specified directory, click **Filter** in the File Selection dialog box. After you have made your selection, click **OK** to close this dialog box.
3. To activate a filter, select a filter from the Available Filters in File Filter/Description list in the Filter Control dialog box and click Activate. The filter information is copied to the Active Filters List, and the filter is activated immediately.

4. If you want display the filtered events from the /usr/OV/log/ovevent.log file for display by the Event History application, select Display Events from the Query pull-down menu in the main workspace window. See "Viewing the Event Log" on page 105 for more information.

If there are too many objects selected for inclusion in the filter, the filter will not be activated, and the following message will be displayed:

Sieve creation failed. Error in Object creation.

If filters that retrieve logged events have more than 40 host names defined as one particular object, for example, IP_ADDRESS, you will receive the following error message:

Error receiving logged event

To resolve the error, edit the filter to reduce the number of objects (IP_ADDRESS).
When activating more than one simple filter, the logical operator OR is used. If you want the criteria of both filters to be used, use the Compound Filter Editor to combine them and then activate the compound filter.

See “Creating Compound Filters” on page 135 for more information.

To display a selected filter from the Available Filters in File Filter/Description list, click the Display/Edit below the list of available filters on the Filter Control dialog box. The Filter Editor dialog box is displayed. From this dialog box, you can make changes to the following filter criteria:

- The filter name
- The filter description
- The enterprise name
- The generic and specific event numbers
- The object from which the trap is to be sent
- The time range, including a starting date and time and a stopping date and time
- The threshold values, including frequency and time interval

Note, however, that you cannot edit a filter from the Available Filters in File list if that filter has been placed in the Active Filters List. If you need to make changes to an active filter, first deactivate the filter, then click Display/Edit to access the Filter Editor.

See “Using the Filter Editor” on page 132 for more information about editing filters.

All filters you activate in a workspace can be saved in a configuration file in the $USER/$HOME directory. See “Saving the Event Workspace Configuration” on page 107 for information about saving workspace configuration.

**Deactivating an Event Filter**

To deactivate an event filter, follow these steps:

1. Select Filter Control... from the Options pull-down menu in the Event Display or Event History application’s window.
   The Filter Control dialog box is displayed.
2. Select the filter name from the Active Filters List and click Deactivate.
   The filter is immediately deactivated, and the number of events being displayed might be affected.

**Activating a Filter in a Dynamic Workspace**

To activate a filter for a dynamic workspace, select Dynamic Workspace from the Create pull-down menu in the main workspace.

Refer to the Tivoli NetView for UNIX User’s Guide for Beginners for more information about creating a dynamic workspace.

**Activating a Trap-to-Alert Filter (AIX Only)**

If you are using the Tivoli NetView host connection, you can specify that selected events are to be converted to alerts and forwarded to the Tivoli NetView for OS/390 program. Trap-to-alert filters permit selected events to pass to the tralertd daemon, which converts events and traps to alerts. You can create and activate filters that
limit the number of traps to be processed by the tralertd daemon. All trap-to-alert filters are stored in the `/usr/OV/conf/tralertd_default.filter` file.

**Note:** You must have root authority to activate trap-to-alert filters because these filters affect the configuration of the tralertd daemon.

To activate a trap-to-alert filter, follow these steps:

1. Select the Event Configuration -> Trap to Alert Filter Control: SNMP... from the Options pull-down menu.

   The Trap-to-Alert Filter Control dialog box is displayed as shown in Figure 29.

2. If the file name in the Input File field at the top of the Trap to Alert Filter Control dialog box is not the one you want to use, change it by using one of the following methods:
   - Type a new file name in the Input File field
   - Select the File List button and select a directory or file name from those displayed in the File Selection dialog box. Select the OK button to close this dialog box.

3. From the Available Filters in File list, select the name of the filter you want to activate and select the Activate button.

   The filter file name is copied to the Active Filters List and the filter is immediately activated.

   If there are too many objects selected for inclusion in the filter, the filter will not be activated, and the following message will be displayed:
Displaying a Trap-to-Alert Filter (AIX Only)
Suppose you select a filter from the Available Filters in File list, but before you activate it you want to ensure that it is the one you want to use. Select the name of the filter and click Display/Edit. Depending on whether the selected filter is simple or compound, the appropriate Filter Editor dialog box is displayed. The filter criteria are displayed in the fields of this dialog box. If you need to change a field, make the changes here, then click either Save As... button to save the changes under a different filter name or OK to close the filter editor.

Deactivating a Trap-to-Alert Filter (AIX Only)
To deactivate a trap-to-alert filter, select its name in the Active Filters List, then click Deactivate. The Event Display application will no longer filter events based on this filter. Deactivating a filter might affect the number of events being displayed.

Creating Cron Table Entries for Trap-to-Alert Filters (AIX Only)
If you want certain filters to be activated and deactivated on a regular and frequent basis, the buttons in the top section of the Trap to Alert Filter Control dialog box. However, you might spend much of your time keeping track of which filters to activate and deactivate, and when.

To automate filter activity, use the bottom section of the dialog box, Cron Table Filter Control. This operation enables you to create and modify cron table entries. The cron daemon uses these entries to control the tralertd daemon’s use of selected filters. Using this operation gives you more precise control of the alerts forwarded to the host program.

**Activating and Deactivating Cron Table Entries:** In the Cron Table Filter Control section of the Trap to Alert Filter Control dialog box, you can enter activation and deactivation parameters for filters that appear in the Active Filters List in the top half of the dialog box. For example, to activate a filter called EveryWednesday on Wednesdays at 7 a.m. and deactivate it at 3 p.m. on Wednesdays, follow these steps:

1. Ensure that the filter is in the Active Filters List. If it is not there, select it from the Available Filters in File list and click Activate.
2. Select the filter in the Active Filters List.
3. In the Cron Table Filter Control section, enter 07:00 in the Activation field and click check for Wednesday. Then, enter 15:00 in the Deactivation field, and select the check button for Wednesday. The 24-hour clock is used for entries in these fields.
4. Click Add to Cron. The activation and deactivation entries for the selected filter appear in the Trap-to-Alert Cron Table Entries field.
5. Click Close to close the Trap to Alert Filter Control dialog box.

**Modifying a Cron Table Entry:** You can modify a cron table entry by selecting it in the Trap-to-Alert Cron Table Entries field, specifying new activation or deactivation parameters, and then selecting Modify.

**Sorting Cron Table Entries:** To sort the entries in the Trap-to-Alert Cron Table Entries field, click Sort. Entries can be sorted in any of the following ways:
- By activation day of the week
- By deactivation day of the week
- By activation hour
- By deactivation hour
- By filter name (the default)
To remove an entry from the list, click Trap-to-Alert Cron Table Entries field, then click Remove.

**Using the `selectfilter` Command (AIX Only)**
You can use the `selectfilter` command in a terminal window to accomplish the same tasks you can perform by selecting **Options..Event Configuration..Trap to Alert Filter Control: SNMP** from the Tivoli NetView main menu. These tasks include the following:

- Activate a filter immediately.
- Deactivate a filter immediately.
- Add an entry to the cron table that will issue the `selectfilter` command at regular intervals to activate or deactivate filters.
- Remove a selectfilter entry from the cron table.

The following example activates a filter called `newFilter` every Wednesday at 5:30 p.m.
```
selectfilter -f /usr/OV/filters
-r newFilter
-s /usr/OV/sockets/tralertd.socket
-a 1 -t 17:30 3
```

For more information about the parameters of the `selectfilter` command, refer to the man page.

---

**Configuring Events**

Every enterprise has a Management Information Base (MIB) that describes operations that can be performed on that enterprise’s devices in the network. Enterprises can specify traps that they expect to receive from agents that support their MIBs. You can configure the events supplied with the MIB to provide additional, more specific information about the status of network objects.

**Advantages to Configuring Events**

Event configuration offers the following advantages:

- You can format a trap to display information that is meaningful to you.
- You can format a trap that is forwarded to the Tivoli Enterprise Console to display meaningful information.
- You can associate an action with a trap by specifying the commands that are to be executed when the management station receives an event. Configuring actions enables you to automate some fault management procedures and to restrict the amount of event information to be displayed.
- You can associate severities with an event. These severity levels are displayed in the upper-left corner of event cards. You can search on events by severity level.
- You can create new event notification categories for filtering events.
- You can create new, additional actions to specify further processing that the operator should manually perform when an event is received.
- You can provide a message window to be displayed when selected events are received. This option is available when you add, copy, or modify an event. The maximum number of windows you can display is ten.
- You can access the trap-to-alert filter control to convert events to alerts that are sent to the Tivoli NetView for OS/390 program (**AIX** only).
Customizing Traps

To use Event Configuration -> Trap Customization: SNMP... from the Options pull-down menu, the following conditions must be true:

- You must be logged in as a root user.
- The node for which you want to configure an event must support SNMP.
- The enterprise-specific MIB for which you want to configure events must be loaded into the Loaded MIBs database.
- You must understand the definition of and purpose for the MIBs on which you want to configure events. Review the documentation provided by the MIB vendor about the enterprise-specific events included with their product.

When an event is configured, it is added to the /usr/OV/conf/C/trapd.conf file. When this event is received from an agent, the information in the /usr/OV/conf/C/trapd.conf file is used to format the trap information that is logged in the /usr/OV/log/trapd.log file.

In addition, the Event Display application reads the event information from the /usr/OV/conf/C/trapd.conf file and formats it for display in the event cards or list. The Event Display application updates symbol status based on status events. In addition to defining the event as a status event, the object must have object or symbol status source.

By default, the Event Display application cannot update objects that have compound status source. You can determine or change the symbol's status source by selecting Edit -> Modify/Describe Symbol from the object context menu. To enable the Event Display application to update a symbol's status if the status source is compound, change the value for the overrideCompoundStatus resource in the /usr/OV/app-defaults/Nvevents file to TRUE. When the value for the overrideCompoundStatus resource is TRUE, the Event Display application updates all symbols for the object without having to manually change the status source for each symbol.

See "Indicating Symbol Status" on page 43 for more information.

To delete configured events, select the event and click Delete. Another way to delete events is to edit the trapd.conf file and delete, or comment out, selected configured events.

Steps

To configure an event that provides you with a more readable message, follow these steps:

1. Select Event Configuration -> Trap Customization: SNMP... from the Options pull-down menu.

   The Event Configuration dialog box is displayed, as shown in Figure 30 on page 144. Click Help for information about the fields and buttons in this dialog box.
2. If desired, use the Configure Event Categories dialog box to define event categories. To display this dialog box, select the Configure Categories button.

3. In the Event Identification section, select an enterprise by selecting an item in the selection list. If the enterprise whose event you want to configure is not in this list, select Add New Enterprise... and enter the enterprise name and object ID in the Add New Enterprise dialog box. Click Add to add the new enterprise and close the dialog box.

To view a list of the Tivoli NetView program’s internally generated events, enter the /usr/OV/bin/event -l command at the command line.

4. Select the event that you want to configure from the selection list in the Event Identification section. If the event already exists, click Modify to display the Modify Event dialog box. Otherwise, you can create a new event by clicking Add or Copy. Clicking Add displays the Add Event dialog box. Clicking Copy displays the Copy Event dialog box.

5. Complete the fields in the Add Event, Copy Event, or Modify Event dialog box. Click Help in each dialog box for detailed information on completing the fields. Click OK button to return to the Event Configuration dialog box.

6. Repeat steps 3 through 5 until all events have been configured.

7. Use the Configure Additional Actions for Operator dialog box to define additional actions that the operator should manually perform when an event is received. Click Configure Additional Actions to display the dialog box.

8. Click OK or Apply in the Event Configuration dialog box to apply any changes you have made. Click Cancel if you want to undo the changes.
Note: Behaviors such as list sizes and default values, can be modified in the /usr/OV/app-defaults/XNm file.

Verifying Trap Customization

If you want to verify the change, look at the /usr/OV/conf/C/trapd.conf file. The next time this event is received from an agent, the log message will contain the message you specified.

If you want to ensure that the message will be displayed as you specified, you can issue the `snmptrap` command at the command line. The `snmptrap` command issues an SNMP trap based on the parameters you specify in the command. For example, to send the Tivoli NetView enterprise-specific trap number 59160427 to a host named `host1` from an agent named `agent1`, enter the following command:

```
snmptrap host1 "" agent1 6 59160427 ""
```

For more information about the `snmptrap` command, refer to the man page.

Another way to see the event you just created is to use the `event` command. This command sends an event to the trapd daemon. Specify the specific number of the event as follows:

```
event -E 59160428
```

For more information about the `event` command, refer to the man page.

Using the addtrap Command to Configure Events

You can type the `addtrap` command at the command line to configure an event without using the Options..Event Configuration..Trap Customization: SNMP menu item. The `addtrap` command creates a trap and adds the new trap to the /usr/OV/conf/C/trapd.conf file.

You can also use the `mib2trap` command to retrieve special comments from a MIB definition file and create a shell script that contains a series of `addtrap` commands. Then, you can execute the shell script to add the new traps to the /usr/OV/conf/C/trapd.conf file.

For more information about the `mib2trap` command, refer to the man page.

If there is no enterprise definition for the trap, the new enterprise definition is added. If a trap exists with identical enterprise-object-ID, generic-trap, and specific-trap values, the `addtrap` command updates the existing trap with the new information.

Example

The following example illustrates the `addtrap` command that adds a trap for the IBM 6611 Router to the /usr/OV/conf/C/trapd.conf file.

```
addtrap -n ibm6611
  -l mytrap
  -l 1.3.6.1.4.1.2.6.2
  -g 6 -s 16 -o A -t 3
  -c "Status Events"
  -f !
  -F "$E $G $S $T"
  -S 4
  -c xecho
  -A 'Status Event received from 6611 agent $E$G $S'
  -e nodeDown
  -E msg
  -V 'PRINTF ("Node %s down", $V2)'
```
This command specifies the following information:

- **-n** The enterprise name is `ibm6611`.
- **-i** The enterprise ID is `1.3.6.1.4.1.2.6.2`
- **-g** The generic trap number is 6.
- **-s** The specific trap number is 16.
- **-o** The trap is sent from an agent, in this case, the 6611 router agent.
- **-t** The object that generates the trap is to be assigned a status of Critical on the map.
- **-c** This is a status event.
- **-f** A specified action (see -C and -A below) will be performed by the management system when this trap is received.
- **-F** The enterprise name ($E), generic ($G) and specific ($S) event numbers, and the time-stamp ($T) are displayed in the event cards or list.
- **-S** The trap is a Severity 4 (Critical) trap.
- **-C** The `xecho` command is activated when this event is received.
- **-A** The following arguments are passed to the Tivoli NetView program with this event:
  - Event text ('Threshold Event received from 6611 agent')
  - The enterprise name ($E)
  - The generic trap number ($G)
  - The specific trap number ($S)
- **-e** This event is forwarded to the Tivoli Enterprise Console with an event class of nodeDown.
- **-E** This event is forwarded to the Tivoli Enterprise Console containing the event text specified by the -V flag.
- **-V** Trap variable 2, the host name, is substituted in the event text for the %s format specifier when this event is forwarded to the Tivoli Enterprise Console.

For more information about the `addtrap` command, refer to the man page.

### Displaying a Warning Window for Events

To display a pop-up window, specify a shell script when a specific event occurs that executes the `ovxbeep` or `ovxecho` command when a specific event occurs. If you execute the ovxbeep command in the shell script, an error dialog box is displayed with an audible alarm. If you execute the ovxecho command in the shell script, an error dialog box is displayed without an audible alarm. The shell script must export the display to the appropriate workstations before executing the ovxecho or ovxbeep commands, and the `xhost` command must have been run on the workstations where the pop-up window is to be displayed.

You specify the name of the shell script in the Optional Command and Argument format section of the Event Configuration dialog box.

For example, let’s say you want to display a pop-up window when NodeA or NodeB fail. For NodeA you want to include an alarm. You also want to send an electronic-mail notice of the failure. Here are the steps:
1. Select **Event Configuration → Trap Customization: SNMP...** from the **Options** pull-down menu.

2. On the Event Configuration dialog box, select or enter the following:
   - Enterprise name: netview6000
   - Event: Specific 58916865

3. Click **Modify**. The Modify Event dialog box is displayed.

4. Enter the following in the Command for Automatic Action field:
   - `<ShellScriptPath> $2`

5. Click **OK** to close the Modify Event dialog box.

6. Click **OK** to close the Event Configuration dialog box.

**Note:** If you filter an event for which you have configured a command for automatic action, the actions specified in the shell script will still be executed. If the shell script executes the ovxbeep or ovxecho command, for example, an error dialog box is displayed even though the event has been filtered.

### Example Shell Script

Following is the shell script used in the previous example.

```bash
#!/bin/ksh
# example.sh
#
# Shell script for node down trap from the netview6000 enterprise
# (specific = 58916865). Displays warning messages and sends e-mail.

export DISPLAY=NodeA.austin.tivoli.com:0
export DISPLAY=NodeB.austin.tivoli.com:0

if [ $1 = NodeA.austin.tivoli.com ]; then
    /usr/OV/bin/ovxbeep $1" is down"
    echo $1" is down" | mail oper1@manager.austin.tivoli.com
fi

if [ $1 = NodeB.austin.tivoli.com ]; then
    /usr/OV/bin/ovxecho $1" is down"
    echo $1" is down" | mail oper2@manager.austin.tivoli.com
fi
```

The $2 passes to the script the name of the device that generated the alert. The shell script checks the $2 flag to see whether it is NodeA or NodeB that generated the alert. If it is NodeA, the shell script calls a program, /usr/OV/bin/ovxbeep that displays a window and an audible alarm. If it is NodeB, the shell script calls the program, /usr/OV/bin/ovxecho that displays a window without a sound. For either node, an electronic-mail notice is sent to the addresses specified in the shell script.

See the /usr/OV/prg_samples/nnm_examples/beeper/beep_951x sample shell script for more examples.

### Converting Events to Alerts (AIX Only)

If you are using the [Tivoli NetView host connection](#) you can edit the events, or traps, that are converted to SNA alerts and forwarded to the host program.

Use the [Alert Editor](#) to define the SNA alert for a trap that must be forwarded to the host program. You can start the Alert Editor from the Options..Event Configuration..Trap Customization: SNMP menu item, which displays the Event Configuration dialog box. The Alert Editor button is on the right side of this dialog box.
Use the Alert Editor to perform the following tasks:

- Configure the trap-to-alert mapping for selected events
- Define the alert for each expected trap by specifying the following information:
  - Type of event
  - Description
  - Probable or failure causes
  - Qualifiers
  - Recommended actions to take
- Delete an alert
- Write alerts to the tralertd.conf file, which can be edited
- Check errors to make sure that the defined alert is valid

The Alert Editor also enables you to modify existing information on alerts through text fields and buttons. When the events are converted and sent to the host program, pertinent alert information is displayed on Tivoli NetView for OS/390 screens.

Using the addalert command (AIX Only)

You can use the `addalert` command to add an alert definition to the tralertd.conf file without using the Tivoli NetView graphical interface. Enter the `addalert` command and its parameters at the command line to perform the same function as selecting Options..Event Configuration and clicking Alert Editor.

The following example of the `addalert` command adds an alert definition for a Bay Networks agent.

```
addalert -o .1.3.6.1.4.110
  -l bayalert
  -g 6
  -s 0
  -t 1
  -d 1400
  -p 0202
  -q 13
  -m "$1"
  -f 0202
  -a 1320
```

This command specifies the following information:

- **-o** The enterprise ID is .1.3.6.1.4.1.10
- **-l** The alert label is bayalert.
- **-g** The generic trap number is 6.
- **-s** The specific trap number is 0.
- **-t** The alert type is Permanent (1).
- **-d** The Generic Alert subvector (X'92') contains a description for generic alert 1400, Loss of Electrical Power.
- **-p** The Probable Causes Alert subvector (X'93') contains probable cause 0202, Internal Power Control Unit.
- **-q** The Detailed Data Alert subvector (X'98') contains the detailed data entry Status Code.
- **-m** The message passed to the host program resides in $1.
The Failure Caused Alert subvector (X'96') contains failure cause 0202, Internal Power Control Unit.

The Recommended Actions subvector (X'81') contains recommended action 1320, Check Cable Connection and Retry.

For more information about the addalert command, refer to the man page.

Refer to Tivoli NetView for UNIX Host Connection for instructions on using the Alert Editor.

## Sending Alerts to the Host Program (AIX Only)

When a trap is received by the trap-to-alert conversion process and it matches an active filter, the trap is converted to an SNA alert and forwarded to the Tivoli NetView for OS/390 program.

The goal is to send all information about a trap to the host program. However, sometimes the trap contains too much information to be sent in one piece. In that case, the trap is saved in the tralertd database and assigned a Log ID, which is sent to the host program in the alert. Tivoli NetView for OS/390 uses the Log ID to issue the gettrap command within a RUNCMD to request complete trap information for the incomplete alert.

The trap information sent to the host program differs based on whether the trap is IBM enterprise-specific, non-IBM enterprise-specific, or generic. Refer to Tivoli NetView for UNIX Host Connection for more information about sending alerts to Tivoli NetView for OS/390.
One of the challenges of network management is keeping track of all the devices on a network and ensuring that you have current information about how they are configured. Current configuration information can help you perform the following tasks:

- Make sure that all devices are configured correctly.
- Resolve network connectivity, performance, and service problems.
- Customize polling intervals to regulate network traffic and collect necessary information.
- Configure SNMP proxies to manage non-SNMP devices.

This chapter describes the following configuration management tasks:

- “Discovering the Network”
- “Monitoring the Network Using Polling” on page 156
- “Monitoring Network Configuration” on page 160
- “Retrieving MIB Configuration Information” on page 162
- “Enabling and Disabling Polling” on page 156
- “Configuring SNMP Nodes” on page 163
- “Configuring a Backup Manager” on page 167

Discovering the Network

The Tivoli NetView program provides two ways to discover IP networks and enables the discovery of open topology networks and Cisco routers. Network discovery provides you with a database of network configuration information. This section describes the following:

- “Automatic Network Discovery”
- “Discovering Open Topology Networks” on page 153
- “Discovering Cisco Routers” on page 153
- “Configuring Symbol Creation Time and Buffer” on page 153
- “Increasing the ovwdb Cache Size” on page 153

Automatic Network Discovery

The Tivoli NetView program uses an automatic network discovery process to generate and maintain a network topology database. The more nodes on the network that support an SNMP agent, the more efficient this discovery process will be, and the more complete and accurate the resulting configuration information will be. Discovery starts with the management station, then proceeds to discover everything up to the first set of routers. Subnets beyond that are unmanaged. Use a seed file to include additional devices in the initial discovery process.

Information Retrieved

When a new node is discovered, it is added to the topology database and also to the list of nodes that is being monitored. If the newly discovered node supports an SNMP agent, information about its system configuration is retrieved and stored in the database. Table 13 on page 153 shows the information that is retrieved:
### Table 13. Configuration Information Retrieved From Nodes During Discovery

<table>
<thead>
<tr>
<th>Information</th>
<th>MIB Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>System description</td>
<td>sysDescr</td>
<td>Includes the full name and version of the system's hardware type, software operating system, and networking software.</td>
</tr>
<tr>
<td>System object ID</td>
<td>sysObjectID</td>
<td>Identifies the network object's place in the MIB hierarchy.</td>
</tr>
<tr>
<td>Forwarding status</td>
<td>ipForwarding</td>
<td>Indicates whether this entity is acting as an IP gateway to forward datagrams received by, but not addressed to, this entity.</td>
</tr>
<tr>
<td>IP address table</td>
<td>ipAddrTable</td>
<td>Lists addressing information relevant to this entity's IP addresses.</td>
</tr>
<tr>
<td>Interface table</td>
<td>ifTable, ifNumber, ifSpeed</td>
<td>Lists interface entries by number.</td>
</tr>
<tr>
<td>System location</td>
<td>sysLocation</td>
<td>Indicates the physical location of this network object.</td>
</tr>
<tr>
<td>System contact</td>
<td>sysContact</td>
<td>Lists the person to contact for this network object and tells how to contact that person.</td>
</tr>
</tbody>
</table>

Once the node has been discovered, these MIB values are polled periodically. Any changes are reflected in the topology database.

If the Tivoli NetView program is configured to work with a relational database, you can store IP topology data in a relational database and use the relational database tools to create reports.

Refer to the *Tivoli NetView for UNIX Database Guide* for more information about transferring IP topology data to a relational database.

You can store additional, enterprise-specific information with each node and network. This information, used with the predefined data generated by the Tivoli NetView program, can give you a clearer picture of your network's configuration.

**Note:** Tivoli NetView sometimes thinks that devices with multiple interfaces are routers and displays the router symbol.

### Turning Off Automatic Discovery

To turn automatic discovery off, select Topology/Status Polling Intervals: IP... from the Options pull-down menu. Click Discover New Nodes to turn polling off.

### Discovering Cisco Routers

Tivoli NetView takes advantage of the Cisco CDP Cache MIB (if available) on routers for next hop information. This greatly improves discovery in a Cisco environment and provides a significantly better chance at discovering the unnumbered serial links promptly. When a CDP MIB is available, netmon will not read the IP Routing table on routers, which can take up to 4 or 5 minutes on a busy router.
What netmon does with the CDP Cache MIB information:

- The default discovery mode for netmon is *Local*. That is, netmon discovers and manages the local subnet and displays the next hop network as Unmanaged. If the CDP Cache MIB information is available, netmon automatically attempts to discover the other side of a serial link without having to rely on a seed file entry specifying it. The other end of the serial link is discovered as Unmanaged, and the serial link is not displayed until both ends are managed.
- You can then extend the discovery as usual by managing the networks and routers. This will initiate a New Node discovery poll, and after a few minutes, you will see the serial links added in. To expedite this behavior, you can issue demand poll requests to the routers.
- Isolated islands will exist when the next hop routers cannot be queried using SNMP.
- For non-Cisco routers, NetView attempts to discover the next hop for unnumbered serial links using other means.

Using a Seed File to Control Network Discovery

When Tivoli NetView is started for the first time, the default management region is the management station on which the Tivoli NetView program is operating and any networks to which it is attached. The discovery process generates the topology map by working outward from the management station up to the first set of routers.

You can define a management region by using a *seed file*. A *seed file* contains a list of host names or IP addresses of SNMP nodes within your administrative domain. Using a seed file forces or restricts the discovery process to generate the topology map beginning from nodes other than the management station. Refer to *Tivoli NetView for UNIX Installation and Configuration* for information about configuring a seed file.

Discovering Open Topology Networks

The Tivoli NetView program uses specially created applications to manage networks that use protocols other than IP. These applications pass topology information to the Tivoli NetView program in the form of enterprise-specific SNMP traps or API calls.

An open-topology discovery application can be started whenever the Tivoli NetView program is started, or it can be started only when a node that supports a protocol other than IP is discovered on the network. The information acquired by the open-topology discovery application is stored in the general topology database on the manager, and can be displayed on a submap along with information about IP network nodes.

Discovering Topology Using the Openmon Application

You can use the Tivoli NetView openmon application to discover and load topology information from an openmon agent into the Tivoli NetView program. Select *Openmon Application* from the *Administer* pull-down menu to start the openmon configuration program to create an application that interacts with an agent that represents a specific topology. The application stores the information obtained from the agent in the GTM database. The topology is displayed along with other network topologies, such as IP, on the Tivoli NetView graphical interface. The topology information is correlated with IP topology information, wherever possible, to determine whether an IP object can also be identified as having an association with the topology stored in the GTM database. Select *Openmon Application* from the *Administer* pull-down menu to start, query, or stop the application.
When the openmon functions receives additional attribute information for the topology objects, openmon stores the attribute information along with the objects in the OVW object database. To view the attribute information select Display Object Information... from the Tools pull-down menu or use the `ovobjprint` command.

Refer to the `openmon` man page for additional information.

The Novell NetWare topology is an example of a topology that openmon creates. The openmon application interfaces with one or more Novell ManageWise Export Services to discover and load the Novell Netware topology into the GTM database.

The NetWare topology is represented on the root map with a Novell icon. When you double-click on the Novell icon, a separate icon is displayed for each ManageWise Export Service. Each ManageWise Export Service contains one object for the IPX network view and one object for the servers view. Here is a description of the NetWare topology views:

**IPX network**
Contains all the IPX network segments connected by the routers that ManageWise discovered.

**Servers**
Contains the icons for all the NetWare servers that ManageWise discovered.

**Router**
Contains the icons for the router adapters and the running software functions, such as IP, IPX routing functions, file servers, and so forth.

**Segment**
Contains all the NetWare servers, requesters, hubs, and routers in the segment.

**Server**
Contains the icons for the adapter and all the running functions such as file server, printer server, and so forth.

**Requester**
Contains the icon for the adapter.

**Hub**
Contains the icons for the adapter and the running functions such as the hub function, file server, and so forth.

**Discovering Cisco Routers**

Tivoli NetView takes advantage of the Cisco CDP Cache MIB (if available) on routers for next hop information. This greatly improves discovery in a Cisco environment and provides a significantly better chance of discovering the unnumbered serial links promptly. When a CDP MIB is available, netmon will not read the IP Routing table on routers, which can take up to 4 or 5 minutes on a busy router.

What netmon does with the CDP Cache MIB information:

- The default discovery mode for netmon is `Local`. That is, netmon discovers and manages the local subnet and displays the next hop network as Unmanaged. If the CDP Cache MIB information is available, netmon automatically attempts to discover the other side of a serial link without having to rely on a seed file entry specifying it. The other end of the serial link is discovered as Unmanaged, and the serial link is not displayed until both ends are managed.
• You can then extend the discovery as usual by managing the networks and routers. This will initiate a New Node discovery poll, and after a few minutes, you will see the serial links added in. To expedite this behavior, you can issue demand poll requests to the routers.
• Isolated islands will exist when the next hop routers cannot be queried using SNMP.
• For non-Cisco routers, Tivoli NetView attempts to discover the next hop for unnumbered serial links using other means.

Configuring Symbol Creation Time and Buffer
The ipmap application draws the IP topology maps that represent your network in the graphical interface. When a node is discovered, ipmap stores it in a buffer. When the buffer reaches its threshold, ipmap draws the symbols for the nodes in the buffer onto the map. In other words, it dumps the buffer to the map. Depending on the size of your network and the speed at which nodes are discovered, you can improve the performance of ipmap by altering the size of the buffer and how often it is dumped.

Using the Describe Map... from the File pull-down menu, you can configure the number of symbols to be created at one time and how often they are to be created. When a node is discovered, it remains in the buffer until the maximum number of nodes arrives or the set time has expired. Normally, the batch size is never reached before the time expires.

The default for the buffer size is 150 nodes. The maximum value allowed is 1000 nodes and the minimum is 1. The synchronization buffer uses the same value set for the buffer size (when no timer is present).

The default and recommended setting for the timer is 3 seconds. More than 3 seconds results in poor response time and less time results in slower overall performance of ipmap. The maximum value allowed is 3600 seconds and the minimum is zero (0).

The size of the buffer should vary in proportion to the speed of the processor.

Steps
To change the buffer size or timer, follow these steps:
1. Select Describe Map... from the File pull-down menu.
2. Select IP Map from the Configurable Applications selection list.
3. Click Configure For This Map.
4. Enter the number of symbols in the field How many symbols should be created at one time?.
5. Enter the time in the field How Often (in seconds) should IP Map create symbols?.
6. Click Verify to check what you entered.
7. Click OK to apply the change and close the dialog box.

Increasing the ovwdb Cache Size
The ovwdb daemon acts as a caching daemon for the object information stored in the object database. You can control the number of objects maintained in the cache. Increasing the cache size will improve your CPU performance, particularly for networks containing more than 5000 objects.
Changing the Size
Change the size of the ovwdb’s cache from the default of 5000 objects to a number larger than the number of objects in the ovwdb database. To get the number of objects in the ovwdb database, use the following command:

```
ovobjprint | head -1
```

To change the size of the ovwdb’s cache, follow these steps:

1. Exit all Tivoli NetView windows. All opened sessions of the ovw application are closed during execution of the last step.
2. Enter `tivoli` on the command line to access the Tivoli desktop.
3. Double-click the policy region that contains the appropriate Tivoli NetView server. The Policy Region window is displayed.
4. Click and hold down the right mouse button on the server icon to display the pull-down menu of the icon.
5. Select `Configure --> Set options for daemons --> Set options for topology, discovery, and database daemons --> Set options for ovwdb daemon...`. The Set options for ovwdb dialog box is displayed.
6. Complete the dialog box fields as follows:
   - Enter a number in the Number of objects to hold in cache field. Enter a number larger than the number of objects defined in the ovwdb database. That enables the cache to grow to the maximum size if needed.
   - Click `Yes` or `No` in the Use port to receive requests over tcp? field.
     - Click `Help...` on the dialog box for additional information.
   - Click `OK`.

Monitor the size of the database and adjust the cache size as necessary. If RAM size and paging space are not a problem, using a cache size of zero allows the cache size to grow to an unlimited size and shrink as needed.

If Tivoli NetView discovers a network that exceeds the available paging space, the operating system may stop a process, including ovwdb, to relieve paging space. See "Monitoring File System and Paging Space" on page 193 for information about monitoring your paging space.

---

Monitoring the Network Using Polling

Tivoli NetView has comprehensive polling capabilities that enable you to view the up-to-date status of your network. Tivoli NetView can use PING (ICMP echo) requests to poll devices for status, or SNMP GET requests to determine the status of all interfaces on a device.

Many aspects of the Tivoli NetView polling configuration can be customized to fit your network environment. This section describes the following:

- "Enabling and Disabling Polling"
- "Using SNMP for Status Polling" on page 156
- "Understanding Router Fault Isolation" on page 158

Enabling and Disabling Polling

Select `Topology/Status Polling Intervals: IP...` from the `Options` pull-down menu to turn polling on or off for all IP nodes in the management region. When you select
the Options..Topology/Status Polling Intervals: IP option, the Topology/Status Polling Configuration Dialog Box is displayed as shown in Figure 31.

![Topology/Status Polling Configuration Dialog Box](image)

**Figure 31. The Topology/Status Polling Configuration Dialog Box**

**Note:** To turn polling on or off, or to set polling intervals, root authority or write access to the /usr/OV/databases/openview/topo/polling file is required.

**Enabling Polling and Discovery**

The Topology/Status Polling Configuration dialog box contains four check buttons that enable you to turn polling and discovery on or off. The Enable Polling and Discovery Settings check button must be selected for the settings of the other three buttons in the dialog box to take effect and for the values in the SNMP Configuration Dialog Box to take effect.

Click the Enable Polling and Discovery Settings check button to turn polling and discovery on or off according to the following variations:

- Turn all polling intervals and discovery on by first selecting **Enable Polling and Discovery Settings**, then selecting **Poll for Status**, **Discover New Nodes**, and **Poll for Configuration Changes** (all three) in the dialog box. All polling intervals and discovery are turned on.

- Individually, turn polling intervals and discovery on by first selecting **Enable Polling and Discovery Settings**, then selecting **Poll for Status**, **Discover New Nodes**, or **Poll for Configuration Changes** by clicking the associated check button. For example, if you click the Enable Polling and Discovery Settings check button then set Poll for Status on, Discover New Nodes off, and Poll for Configuration Changes on, status polling and configuration checks occur, but new node discovery does not occur.

- Turn Poll for Status off, Discover New Nodes off, and Poll for Configuration Changes off by deselecting the **Enable Polling and Discovery Settings** check button. The individual settings of the other three check buttons in the dialog box do not take effect if the Enable Polling and Discovery check button is turned off. If this check button is not selected, the values set in the SNMP Configuration Dialog Box do not take effect.

**Polling for Status, New Node Discovery, and Configuration Changes**

After clicking **Enable Polling and Discovery Settings** as described in the previous section, the other three check buttons enable you to poll for status, discover new nodes, and poll for configuration changes. You can individually turn on the following:

- Poll for Status
When you poll for status, nodes are polled for status states. If they do not respond to status polling during the amount of time specified in the Node Down Delete Interval field in the SNMP Configuration Dialog box, they are automatically deleted from the object database. The default is 7 days. To change the time period specified in the Node Down Delete Interval field, the Poll for Status button must be selected.

- Discover New Nodes
  When you discover new nodes, Tivoli NetView polls existing SNMP nodes to determine if new nodes exist. The frequency of this polling is determined by whether you use an auto-adjusting polling interval or specify a fixed polling interval in the SNMP Configuration Dialog box. Use an auto-adjusting polling interval to send less polling traffic to the manager once most of the network is discovered. Auto-adjusting polling is the default.

- Poll for Configuration Changes
  SNMP nodes can be polled at specified time intervals to check their configuration status. You can regulate this polling by selecting Poll for Configuration Changes and specifying a value in the Fixed Polling Interval field in the SNMP Configuration Dialog box.

  By default, Tivoli NetView checks the configuration once per day. It checks the /usr/OV/conf/ovsnmp.conf database to determine how frequently to poll the node for status. This file contains default configuration values that you can change; see "Configuring SNMP Nodes" on page 163 for more information.

  Configuration checking provides the following information for selected network nodes:
  - Change in contact or location
  - Forwarding IP packets change
  - Interface added
  - Interface deleted
  - Incorrect routing by a node
  - Link address change
  - Mismatch of link address
  - Network mask change
  - Node name change
  - Object identifier change
  - Undetermined link address

If you want to set or change polling intervals for individual nodes, select either SNMP Configuration... from the Options pull-down menu or Data Collection & Thresholds: SNMP... from the Tools pull-down menu.

Using SNMP for Status Polling

Agents in the network are regularly polled by NetView to update the status of the objects in the topology map. Normally, Tivoli NetView sends PING requests to all interfaces on a device to determine their status. Certain network devices have interfaces whose status cannot be determined from a PING. These include devices with unnumbered interfaces, such as ATM devices.

For these devices, Tivoli NetView can be configured to determine the status using SNMP queries for the ifAdminStatus and ifOperStatus values of the interfaces. This behavior is set in one of three ways:

1. Automatically, if the node has at least one unnumbered interface.
2. Explicitly, by setting the IP address or range of addresses in your netmon seed file.
This may be done using the Server Setup application (Configure -> Set options for daemons -> Set options for topology, discovery, and database daemons -> Set options for netmon daemon) by creating an SNMP Status entry using the Network Monitor Seed File Editor. You may also edit the netmon.seed file using a text editor; the prefix $ denotes an SNMP Status entry. The same rules apply for IP address ranges and pattern matching characters (wildcards) as those for negative seed entries.

3. Explicitly, using the P switch in the /usr/OV/conf/oid_to_type file against classes of devices

If Tivoli NetView is configured to use SNMP to poll for status, the demand poll will report the values of the ifAdminStatus and ifOperStatus for each interface of the node. If you still want to use a PING request to poll for status on these nodes during a demand poll, specify the -E switch in the /usr/OV/lrf/netmon.lrf file.

Refer to the Tivoli NetView for UNIX Installation and Configuration for information on the formats of the netmon.seed and oid_to_type files. Refer to the Tivoli NetView for UNIX Administrator’s Reference or the man pages for more information on the netmon daemon.

Understanding Router Fault Isolation

The Router Fault Isolation (RFI) feature reduces the diagnostic cost of locating a network problem, especially when the network becomes partitioned as a result of the failure. RFI simplifies the notification action by issuing only one summary alert identifying the router nearest the fault.

When you use Tivoli NetView to manage a network with a high proportion of nodes to routers, Router Fault Isolation can significantly reduce the number of Node Down events that are false alarms. Router Fault Isolation enables you to detect which nodes are actually down and which nodes are simply unreachable because the router fault is occluding them from the management station. Router Fault Isolation relies on connectivity tests and responds instantly to dynamic routing changes.

During a network failure, the Tivoli NetView management workstation cannot reach devices or interfaces in the failed portion of the network. For example, a failed router interface can make other portions of the network, such as subnets, invisible or inaccessible. That portion of the network is occluded and considered unreachable. Without Router Fault Isolation, Tivoli NetView creates numerous events during a network failure and again after the failure is corrected.

This event proliferation:
• Increases the difficulty of determining the original cause of the network failure
• Slows network traffic considerably with the large number of status polls to the occluded area
• Creates performance problems and unreliable status reports if the events are forwarded to the Tivoli Enterprise Console

When a node or interface is down, Router Fault Isolation first checks the status and reachability of the router interfaces connected to that subnet. During the router check, each interface and its subnet are analyzed. A non-responding interface triggers checks of the interface and any connecting routers. Router Fault Isolation generates an event for each Router Down or Router Marginal event. Tivoli NetView maps display unreachable networks and router nodes or interfaces as white symbols. Note that non-router nodes and interfaces in unreachable subnets are not changed to Unreachable (white).
When active, the Router Fault Isolation feature generates the following events to alert users to important status changes:

Table 14. Events generated by Router Fault Isolation.

<table>
<thead>
<tr>
<th>Events</th>
<th>Network Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Router Marginal</td>
<td>At least one router interface is down. At least one other interface on that router is up.</td>
</tr>
<tr>
<td>Router Down</td>
<td>All interfaces are not responding, but at least one connected subnet is reachable. (The router is not occluded.)</td>
</tr>
<tr>
<td>Router Unreachable</td>
<td>The network management workstation cannot query the router because it is occluded.</td>
</tr>
<tr>
<td>Router Up</td>
<td>All the interfaces have responded successfully. This event is issued on initial discovery and following a recovery from one or more interfaces being down.</td>
</tr>
<tr>
<td>Network Unreachable</td>
<td>All router interfaces in the subnet have stopped responding.</td>
</tr>
<tr>
<td>Network Reachable</td>
<td>After one interface is successfully polled, the network is once again reachable.</td>
</tr>
</tbody>
</table>

Router Fault Isolation also suppresses polls and status events for all non-router nodes and interfaces in unreachable subnets. After a partition is repaired, the first successful status poll from inside an unreachable subnet triggers a recovery. To speed the initiation of recovery, you can also manually PING any node in an unreachable region.

The Router Fault Isolation feature, also known as Event Suppression, is active by default in Tivoli NetView. To turn this feature off use the -K 0 option for netmon. To control the suppression of polling traffic to routers (including unreachable routers), use the -k option for netmon. Refer to the netmon man page in Tivoli NetView for UNIX Administrator’s Reference for more information. For a detailed explanation of Router Fault Isolation, see the description in /usr/OV/doc/RouterFaultIsolation.htm.

Monitoring Network Configuration

There are many reasons to keep track of the configuration of objects on your network. The following list provides some configuration information you might need:

- What the IP and non-IP addresses are for a node or nodes
- What types of interfaces a node supports, and the status of each
- How a node is connected to the network
- Whether two nodes show the same address for a third node
- Whether a particular service has been installed on a remote node

The following sections describe how you can obtain this information.

Listing IP Addresses for Remote SNMP Nodes

To list IP and non-IP addresses associated with a remote SNMP node, select a node on a submap, then select the Network Configuration → Addresses... from the Monitor pull-down menu. This operation collects information that you might otherwise have to obtain by looking at numerous configuration files. The following information is provided about each interface this object has with the network:

- The index, or MIB instance, of the interface on the selected node. You must have this value if you plan to set up MIB data collection.
- The name of the interface
• The IP address of the interface
• The network mask
• The network address
• The link address, if any

Checking Configured Interfaces

Select Network Configuration -> Interfaces... from the Monitor pull-down menu to list information about interfaces on remote SNMP nodes. This information can help you resolve performance problems because it provides statistics on incoming and outgoing SNMP node traffic and associated errors. It can also help you resolve connectivity problems, because it lists the status of interfaces.

Selecting Network Configuration -> Interfaces... from the Monitor pull-down menu provides the following information:

• The index, or MIB instance, of the interface on the selected node. You must have this value if you plan to set up MIB data collection.
• The name of the interface.
• The type of interface, for example, loopback, Ethernet, FDDI.
• The maximum transmission unit (MTU) size. This is the largest packet size that can be sent unfragmented.
• The status of the interface, which can be up, down, or testing (no operation packets can be passed through the interface).
• The total number of input packets and the number of erroneous input packets received.
• The total number of output packets and the number of erroneous output packets sent.

For hubs and bridges, each entry in the table corresponds to a port on the hub or bridge.

Viewing Routing Table Information

To obtain routing table information for selected remote SNMP nodes, use the Network Configuration -> Routing Table... from the Monitor pull-down menu. This operation can help you resolve connectivity problems. The following information is provided by this operation:

• Destinations. The default destination is a route used by the system when it cannot find a specific route.
• The name of the next gateway between the selected node and the destination.
• The type of connection, as follows:
  – Direct—To a directly connected local area network (LAN)
  – Remote—Through a remote gateway
  – Other
• The name of the interface that is used to reach the destination.

You can also list information about gateway routing tables by entering the following command in at the command line:

```
rnetstat -r <host name>
```

Refer to the man page for more information about the rnetstat command.
Obtaining ARP Cache Information

The Address Resolution Protocol (ARP) cache is helpful in resolving connectivity problems, because it can tell you whether two nodes have a different link address than a third node. To obtain this information for selected remote SNMP nodes, select **Network Configuration → ARP Cache...** from the **Monitor** pull-down menu. The following information is displayed:

- The name or IP address of the destination node
- The link address associated with the destination node
- The interface name of the selected node that is used to access the destination node

You can list the ARP Cache table for a selected SNMP node by entering the following command at the command line:

```
rnetstat -A <host name>
```

For more information about the **rnetstat** command, refer to the man page.

Listing Configured Services

If you want to know the network services a node is configured to support, a user might be having trouble accessing a particular service on a remote SNMP node, select **Network Configuration → Services...** from the **Monitor** pull-down menu to learn the following information:

- The service protocol, either TCP or UDP
- The port to which the service is bound
- The service for which the node is listening, such as SNMP, telnet, or NFS. If this field is blank, the service is unknown.

You can get a listing of configured services for a remote SNMP node by entering the following command at the command line:

```
rnetstat -S <host name>
```

For more information about the **rnetstat** command, refer to the man page.

Retrieving MIB Configuration Information

Certain MIB variables store information that provides a summary of the configuration of a selected network node or nodes. The operations described in this section enable you to conveniently retrieve the values of frequently accessed MIB variables.

Displaying MIB Interface Information

To display the current values of a network object’s interface MIB variables, select a symbol on the submap, then select **MIB Values..Interface Info...** from the **Monitor** pull-down menu. The following information is displayed:

- **ifDescr**: Information about the manufacturer, product, and version of the hardware interface
- **ifType**: Type of interface
- **ifMtu**: Size of largest datagram that can be sent or received over this interface, specified in octets
ifSpeed
Estimated current bandwidth of the interface, in bits per second (bps)

ifPhysAddress
Address of protocol layer at the protocol layer immediately below the network layer in the protocol stack, in octets.

ifAdminStatus
Desired state of the interface (up, down, or testing)

ifOperStatus
Current operational state of the interface

Displaying MIB System Information
To display the current values of a network object’s system MIB variables, select a symbol on the submap, then select MIB Values -> System Information... from the Monitor pull-down menu. The following information is displayed:

sysDescr
Information about the name and version of the hardware, software operating system, and networking software

sysObjectID
The node that is being managed

sysUpTime
Time, in hundredths of a second, since the network management portion of a system was last initialized

sysContact
Contact person for this managed node, and how to contact that person

sysName
Administrative name (fully qualified domain name) assigned to this node

sysLocation
Physical location of the node

Configuring SNMP Nodes
The Tivoli NetView default configuration values are stored in the /usr/OV/conf/ovsnmp.conf_db file. You can make the following changes to this file by selecting the SNMP Configuration... from the Options pull-down menu:

- Change the default SNMP configuration.
- Change the netmon daemon’s status polling intervals.
- Configure different community names for different nodes in your network. The community name used by the management application must match the name that is configured on the agent of the managed node.

Note: Be careful when configuring a Get community name for a SmartSet. Remember that the netmon daemon needs a valid community name to discover nodes, but a node cannot be in a SmartSet until it is discovered. For discovery purposes, the community name should be set in the Specific Nodes or IP Address Wildcards section on the SNMP Configuration dialog box if it is not the value specified in the default configuration.

- Configure SNMP proxies to manage non-SNMP devices.
- Configure a specific node or a group of nodes to have different values than the default configuration.
• Control polling for a specific node or a group of nodes according to the following variations:
  – To turn new node discovery on or off, click on Discovery Poll.
  – To turn auto-adjusting polling on or off, click on Auto Adjust.
  – Control the configuration check polling interval.
• Discover new nodes as managed or unmanaged by clicking Discover Node(s) Managed.

In a distributed network environment, make sure that you configure managed network devices to enable SNMP communication from client machines as well as from the manager workstation. If you do not properly configure the network device, all SNMP requests from the clients, such as MIB applications, MIB browsers, and so forth, will time out with no response from the device.

Note: The polling configuration values are set to the default settings when automatic map generation is restarted or the databases are cleared.

How an Application Uses SNMP Configuration

When an application initiates an SNMP request, the SNMP APIs look for a configuration entry in the node list to query the node. If the application cannot find an entry, it looks for the first IP address wildcard entry in the network list that matches the IP address of the node. If an entry exists, then it’s used. If a network list entry does not exist for the node, the application uses the default SNMP configuration.

In a distributed network environment, SNMP requests issued from the client are directly transmitted on the network from the client. That is, the requests do not go through the server first. Therefore, clients must have their community names configured to enable SNMP requests on a node.

When an SNMP request is made for a node name, the node name must resolve to an IP address. You must have the node entry in the /etc/hosts file or the domain name server. When the node name does not resolve to an IP address, the community name public is used to send the SNMP request.

Description of the SNMP Configuration Dialog Box

Figure 32 on page 165 shows the SNMP Configuration dialog box.
The SNMP Configuration dialog box contains two main sections. The top section contains three selection lists that display current SNMP configuration values for:

- Specific nodes
- A group of nodes
- SmartSets
- The default SNMP configuration for your network

The bottom section contains text entry fields in which you can specify new values for entries in the top section.

**Note:** If you change the value for configuration checking to a shorter interval, you must perform a demand poll on a node to make that change effective. Otherwise, you would need to wait 24 hours or to update the rate of the daily configuration check.

To modify polling intervals, you must have either root authority or write permission for the configuration file `/usr/OV/databases/openview/topo/polling`. 
When you select an item from any of the three selection lists in the top section, the current parameters for that entry are displayed in the bottom section. You can then view, delete, or modify the parameters.

The order of the three selection lists, and the order of entries within each list, illustrates the precedence order in which entries are searched when a node is queried.

**Note:** The parameters displayed in the selection lists can be tailored by the X11 resource xnmsnmpconf.summaryList. If this resource is set to True, only the Target, Community, Set Community and Proxy parameters are displayed in the selection lists. Otherwise, all parameters are displayed in the selection lists.

---

**Steps**

The values you enter for individual nodes override the values set in the Global Default section. The Enable Polling and Discovery Settings button must be selected in the Topology/Status Polling Configuration dialog box for the values you enter in the SNMP Configuration dialog box to apply. To change the configuration for an SNMP node, follow these steps:

1. Select **SNMP Configuration...** from the **Options** pull-down menu.
2. Select the item you want to change from one of the selection lists. The configuration for the item you selected is displayed in the SNMP Parameters section.
3. Change the following configuration values as appropriate.
   - Proxy
   - Target
   - Community
   - Set Community
   - Timeout
   - Retry count
   - Remote Port
   - Status Polling
   - Node Down Delete Interval
   - Fixed Polling Interval
   - Configuration Polling Interval
   - Number of Route Entries
   - Discovery Poll
   - Auto Adjust
   - Discover Node(s) Managed

   Click **Help** for information about these fields.
4. Click **Replace** to display the new values in the appropriate selection list.
5. Click **OK** or **APPLY** to apply the changes.

To configure a specific node, a group of nodes, or a SmartSet to have different configuration values than the default configuration, follow these steps:

1. Select **SNMP Configuration...** from the **Options** pull-down menu.
2. Enter the values in the text entry fields in the SNMP Parameters section. If you specify a specific node, you only need to fill in the fields that have different configuration values than the default configuration.
3. Click **Add** to add the new values to the appropriate selection list. If the target is a regular host name or an IP address, the configuration is added to the Specific
Nodes selection list. If the target is an IP address with wildcards, the configuration is added to the IP Address Wildcards selection list. If the target is a SmartSet, the configuration is added to the SmartSets selection list.

4. Select either Apply or OK to apply the changes.

If you cannot set the isSNMPProxied general attribute in the map database, add the target to the map. The target must exist in the map database and have a selection name.

---

**Configuring a Backup Manager**

With Tivoli NetView, you can segment a large network and create individualized spheres of control for several management stations. Objects outside an operator’s sphere-of-control are unmanaged by that operator’s management station. When an object is unmanaged, Tivoli NetView no longer polls that object for status and configuration changes. You can have numerous Tivoli NetView programs on the network. Each one can be configured so that there is little duplication of management network traffic.

Using the [Backup Configuration dialog box](#), you can configure objects as manager nodes (Tivoli NetView programs) or as managed containers, and then specify which manager is managing each container. A managed container is a collection of objects and is designated as being managed by one or more managers or not managed at all.

The entire network can be separated into various containers using the concept of a [partitioned Internet map](#). Each manager can be designated to manage a subset of the containers. Managed containers that are defined and managed by a remote Tivoli NetView managing system are unmanaged.

Each manager checks on the status of the other known managers and notifies the backup manager when a manager is disabled or when the manager is restored. The operator can then [manage the backup session](#).

See [“Managing a Backup Session” on page 173](#) for more detailed information. Depending on the configuration of the local manager, it can manage all or part of the disabled manager’s containers.

A submap labeled ManagerSubmap is displayed on the root submap. The Manager submap contains the symbol for each manager node that is discovered on the network. The Manager submap’s only function is to group all managers together in one place, allowing you to quickly determine current backup sessions for the selected remote manager.

See [“Determining Current Backup Sessions” on page 174](#) for more detailed information.

---

**Configuring Manager-Container Associations**

To configure nodes as managers and specify the containers they manage, do one of the following:

- Create a seed file that specifies which associations to make
- Use the [Backup Configuration dialog box](#)

There is no manager-to-manager communication. Therefore, to configure manager-container associations across multiple managers, create a definitive seed.
file and distribute it to all managers. This ensures the same configuration file is used for all managers. If you change a map or a manager-container association on a manager, change all managers. The manager-container configurations are global only on the local system. Use the Backup Configuration dialog box to display and modify the associations already configured.

Manager-Container Characteristics
Managers and containers must have the following characteristics to be valid:

- The name specified must be the selection name of an object in the object database.
- The specified manager name is a node (the isNode capability must be set to True).
- The specified container name must be a Location, Internet, Network, or Segment object.

Using a Seed File to Configure Managers
You can create a seed file to pass to the backup process. A seed file is used by Tivoli NetView to determine which nodes are managers, as well as how manager-container associations should be made. It defines the relationship between managers, containers, and backup managers. The seed file is passed to the backup process by modifying the command line for the backup process in the application registration file, /usr/OV/registration/C/backup. Find the line:

```
Command -Shared -Initial -Restart "${BackupDir:-/usr/OV/bin}/backup";
```

Add the seed file name to the end of the line as follows:

```
backup [-s /path/seed_file_name]";
```

Save the seed file in the /usr/OV/conf directory. Include the full path name. Invoke the seed file through the registration file and not from the command line.

Seed File Format: The seed file consists of three fields per line:

```
active manager  container  backup manager
```

Where:

<table>
<thead>
<tr>
<th>active manager</th>
<th>Is the name of a manager. This is the active manager for the container object</th>
</tr>
</thead>
<tbody>
<tr>
<td>container</td>
<td>Is the container object name.</td>
</tr>
<tr>
<td>backup manager</td>
<td>Is the name of the backup manager.</td>
</tr>
</tbody>
</table>

You can designate more than one backup manager as shown in the following example. Names must be enclosed in double quotation marks:

```
"bos.tivoli.com"  "192.10.30"  "rudd.tivoli.com"  "cowan.tivoli.com"
```

Where:

<table>
<thead>
<tr>
<th>bos.tivoli.com</th>
<th>Is the manager node.</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.10.30</td>
<td>Is the container node.</td>
</tr>
<tr>
<td>rudd.tivoli.com and cowan.tivoli.com</td>
<td>Are backup manager nodes.</td>
</tr>
</tbody>
</table>

If bos should go down, both backup managers will start managing the specified container objects.
**Seed File Process:** The seed file is processed line-by-line. The manager name and the container name are read. These objects are validated to make sure that they actually exist and that they have the appropriate characteristics. If the objects are valid, the manager object's capability field, isManager, is set to `True`. Associations between the manager and container are activated. The container object is added to the manager's selected containers list and the manager object is added to the container's selected managers list. The container is checked to see if it should appear as managed or unmanaged by the ipmap application. After those associations are made, the associations between the backup managers and the containers are made.

If the seed file has been updated and changed or the necessary managers and containers were not discovered when the seed file was initially read, select **Backup -> Read Seedfile** from the **Administer** pull-down menu to re-read the seed file. This option adds associations to those that already exist. Associations are not deleted.

**Seed File Example**
In the following example, M1, M2, and M3 are manager nodes in a network. Each manager has the following backup seed file configured:

```
"M1" "Backbone" "M2"
"M1" "North_America_Network" "M2"
"M1" "Latin_America_Network" "M2"
"M1" "South_America_Network" "M3"
"M1" "Antarctica_Network"
"M2" "Africa_Network" "M1" "M3"
"M2" "Eastern_Europe_Network" "M1"
"M2" "Western_Europe_Network" "M1"
"M3" "Asia_Network" "M1" "M2"
"M3" "Australia_Network" "M1"
"M3" "Middle_East_Network" "M1" "M2"
```

When M1 is started, Backbone, North_America_Network, Latin_America_Network, South_America_Network, and Antarctica_Network are automatically managed. Africa_Network, Eastern_Europe_Network, Western_Europe_Network, Asia_Network, Australia_Network, and Middle_East_Network are automatically unmanaged.

When M2 is started, Africa_Network, Eastern_Europe_Network, and Western_Europe_Network are automatically managed. All the other containers are unmanaged.

When M3 is started, Asia_Network, Australia_Network, and the Middle_East_Network are automatically managed. All the other containers are unmanaged.

In this example, each manager has a subset of the whole network to manage. M1 is acting as the backup for all containers of M2 and M3. M2 and M3 are backing up specific containers. The Antarctica_Network is not backed up at all.

**Using the Backup Configuration Dialog Box**
Use the Backup Configuration dialog box, shown in Figure 33 on page 170, to designate nodes as managers and add, display, and modify the configuration of backup managers and containers. Define and configure remote manager nodes on the local manager node so they can be backed up.
The Backup Configuration dialog box displays a list of all configured manager nodes (nodes that have the isManager capability field set to True) and a list of all the container nodes known by Tivoli NetView.

You must have a read-write map to configure backup managers. Only one Backup Configuration dialog box can be opened for each object database.

Designating Nodes as Managers: To designate nodes as managers, follow these steps:

1. Select Backup -> Backup Configuration... from the Administer pull-down menu. The Backup Configuration dialog box is displayed.
2. Enter the name of the node in the Selection field or select an object on the submap and click Select From Map. When you select an object from the map, the object characteristics are checked. If it is a valid manager, its selection name is placed in the manager selection box.
3. Click OK. The Manager Configuration pane is displayed.
4. Click True next to the isManager field. The manager node name is added to the list of managers in the Managers field. When using a seed file, the isManager field is automatically set to True.
5. Click OK or Apply.
6. Repeat the steps for each node you want to designate as a manager.

Once you have designated nodes to be managers, you can make your manager-container associations.

Modifications to the isManager capability field causes Tivoli NetView to re-evaluate the managed or unmanaged state of all containers in the Selected Containers list.

**Adding Manager-Container Associations:** Configure on the local managing system the remote managers and which containers they are actively managing. Also add the containers you want the local manager to backup if the active manager for those containers should go down. The manager-container associations that you configure apply to the current system's database only.

You can make a manager-container association even though the selected node does not have the isManager capability set to true.

Dividing the backup management of containers between managers prevents a sphere of control, which could be an entire network, from being placed on one manager.

**Steps:** To configure manager-container associations, follow these steps:

1. Select **Backup --> Backup Configuration...** from the **Administer** pull-down menu. The Backup Configuration dialog box is displayed.
2. Do one of the following:
   a. Select a Manager or a Container from either list.
   b. Enter the name of the node in the Selection field.
   c. Select an object on the submap and click on the **Select From Map** button. When you select an object from the submap, the object characteristics are checked. If it is a valid manager or container, its selection name is placed in the appropriate selection box.
3. Click **OK** if you select an item from the list.
   
   **If a manager was selected:**
   a. The Manager Configuration pane is displayed.
   b. Do one of the following:
      a. Select a container from the Available Containers list. This list displays all of the available containers that are not associated to the selected manager. Click **Select** to create an association between the manager and the selected container. The container name is removed from the Available list and added to the Selected Containers list.
      b. Type a container object name in the field below the Selected Container list and click **Add to List**. The container name is added to the Selected Containers list.
      c. Select an object from the map, click **Select from Map**, then click **Add to List**. The container name is added to the Selected Containers list.

   The Selected Containers list displays all of the containers associated with the selected manager. Selecting containers from the Selected Containers list can affect the managed or unmanaged state of that container on the manager.

   c. Select the type of association to be made for the selected manager and container.
Click **Active** if you are configuring associations between a manager and the containers that the manager is managing. There can be more than one manager configured as active for each container.

Click **Backup** if you are configuring associations between a manager and the containers that this manager is going to manage if the active manager should go down. Multiple backup managers for a single container and multiple backup containers per backup manager are permitted.

d. Repeat the steps for each manager-container association.

**If a container was selected:**

a. The Container Configuration pane is displayed.

b. Do one of the following:

   • Select a manager from the Available Managers list. This list displays all of the available managers that are not associated with the selected container. Click **Select** to create an association between the container and the selected manager. The manager name is removed from the Available list and added to the Selected Managers list, which displays all of the managers associated with the selected container.

   • Type a manager name in the field below the Selected Managers list and click **Add to List**. The manager name is added to the Selected Managers list.

   • Select an object from the map, click **Select from Map** and then **Add to List**. The manager name is added to the Selected Managers list.

c. Select the type of association to be made (active or backup).

d. Repeat the steps for each manager-container association you want to add.

4. Click **Apply** to activate your associations.

5. Click **OK** to close the Container Configuration pane.

**Deleting Associations:** Associations can be deleted by selecting a container from the Selected list and clicking **Remove**. Deleting containers from the selected list can affect the managed or unmanaged state of that container on the local manager.

To delete managers from the Managers list, select a manager and click **False** next to the isManager capability field.

If you remove associations that contain non-manager nodes, the manager is moved from the Selected Managers to the Available Managers list when you select **Remove**. If you exit the Container Configuration pane and then return, the node is no longer on the Available Managers list because the capability was not set to isManager.

**Determining Remote Manager Status**
The status of remote managers is determined by Node Up and Node Down traps sent by netmon to the local manager.

When the local manager receives a Node Down trap for a remote manager, all active containers for that remote manager are examined. If a container is also part of the local manager’s sphere of control, no further action is taken because the local operator already has control of that container. By default, the local manager receives notification of a disabled manager. A Manager Down message box is displayed, and the operator can initiate actions from the message box to manage the backup session.
When the local manager receives a Node Up trap for a remote manager, a Manager Restored message box is displayed stating the remote manager’s name and the fact that it is now active. The operator can initiate actions from the message dialog box to manage the backup session. No further action is taken by the backup process until the operator returns the container to unmanaged state by closing the container submap from the Navigation Tree window, the Manager Submap, or through the Manager Restored message box. At that time the managed or unmanaged state of the container is re-evaluated.

Note: Unmanaging remote managers or submaps and containers in which the remote manager objects reside prevents the local manager from being notified when the remote manager is disabled, because the local manager no longer polls those objects.

Manage-Unmanage Rules
When the configuration changes for a container, for example, a container is removed from the list of Selected Containers associated with a manager, Tivoli NetView determines whether to automatically manage or unmanage the container. When the configuration changes for any manager, for example, the isManager capability changes from true to false, the managed or unmanaged condition for all of that node’s managed containers is updated. Here is how Tivoli NetView handles status changes:

- If the local manager and the given container are associated, the container is managed.
- If the container is not associated with any manager, the container is managed by the local manager.
- If an association exists between this container and any remote node where the isManager capability is set to True, the container is unmanaged by the local manager.
- If all remote associations are with nodes where isManager is set to False, the container is managed by the local manager.

Managing a Backup Session
This section describes the actions you can take when you receive notification of a remote manager status change. This section also describes how you can customize the backup default actions, and how you can determine what containers a backup manager is managing.

Responding To a Manager Down Notification
When a remote manager is disabled, a Manager Down message box is displayed. You can perform one of the following actions:

- Click OK to manage all the backup containers and open submaps for those containers.
- Click Manage to manage all the backup containers but do not open submaps for those containers.
- Click Cancel to take no action.

Responding To a Manager Up Notification
When a remote manager is restored, a Manager Restored message box is displayed. You can perform one of the following actions:

- Click OK to take no action and close the message box. This gives the operator the opportunity to complete tasks being performed in the disabled manager’s
submaps. When the tasks are completed, the submaps must be closed from the Navigation Tree or from the Manager Submap to make sure that the containers are automatically unmanaged.

• Select Close All to unmanage all the backup containers and close any submaps for those containers that are open.

Changing Backup Default Actions
You might want to change backup default actions to prevent notification of remote manager status changes and to automatically perform a specified action. You can change entries in the /usr/OV/app-defaults/Backup file or copy the appropriate entries in the user’s .Xdefaults file to change backup default actions.

The following lines in the /usr/OV/app-defaults/Backup file contain the default settings for the actions taken for remote manager status changes:

backup*ManagerDown.displayPopup: TRUE
backup*ManagerDown.defaultAction: OK

backup*ManagerUp.displayPopup: TRUE
backup*ManagerUp.defaultAction: OK

You can set both displayPopup values to FALSE to prevent notification of remote manager status changes. The values for ManagerDown.defaultAction (OK, Manage, or Cancel) and ManagerUp.defaultAction (OK or Close All) correspond to the actions initiated from the Manager Down and Manager Restored message boxes.

See "Responding To a Manager Down Notification" on page 173 and "Responding To a Manager Up Notification" on page 173 for a description of those actions.

To automatically manage and unmanage backup containers without displaying notification of remote manager status, do the following:
• Change both displayPopup values to FALSE.
• Change the ManagerDown.defaultAction to Manage.
• Change the ManagerUp.defaultAction to Close All.

If you make changes to any of the app-defaults files after Tivoli NetView is started, use the command `xrdb -merge .Xdefaults` to load the new resource. If you have not started Tivoli NetView, the new resource will be loaded when you start the graphical interface.

Determining Current Backup Sessions
You can determine what containers a selected manager is managing on behalf of another manager using the Backup Session window. From the Backup Session window, you can stop managing some or all of the backup containers, and you can open submaps for selected containers. To determine current backup sessions, follow these steps:

1. Select a manager symbol from either the Manager submap or another submap that contains a manager symbol.
2. Select Administer –> Backup –> Display Sessions... from the context menu.
   The Backup Session Window is displayed.
3. Do one of the following:
   • Select specific containers from the list and select the Close button to stop managing some, but not all, of the containers in the list.
   The selected containers will be unmanaged, and any open submaps for those containers are closed.
• Click **Close All** to stop managing all the containers in the list.
  All containers in the list will be unmanaged, and any open submaps for those containers are closed.

• Select specific contains in the list and click **Open Submap** to open the submap for one or more containers in the list.
  Submaps are opened for those containers. If a submap is already open for any selected container, it is placed on top of the other windows that are displayed.

• Click **Cancel** to exit the session window without performing any action.
Chapter 7. Managing Network Performance

The Tivoli NetView program helps you manage network performance by providing several ways to track and collect performance information for objects on the network. You can use performance information to help you:

- Monitor the network for signs of potential problems
- Resolve network problems
- Collect information for trend analysis
- Generate regular performance reports
- Allocate network resources
- Plan future resource acquisitions

This chapter describes MIBs and explains how to load, unload, and browse them. It also describes how to use Tivoli NetView predefined performance applications and how to create your own applications to monitor network performance. The following topics are covered in this chapter:

- "Loading and Unloading MIBs" on page 177
- "Browsing MIBs" on page 179
- "Using the Tivoli NetView Performance Applications" on page 181
- "Monitoring Real-Time Network Performance" on page 183
- "Collecting Historical Performance Information" on page 187
- "Monitoring File System and Paging Space" on page 193
- "Using the Tivoli NetView Graph Applications" on page 197
- "Generating Performance Reports" on page 199

Loading and Unloading MIBs

The Options..Load/Unload MIBs and Options..Load/Unload MIBs..SNMPv1/SNMPv2 operations lets you include your enterprise-specific MIB, or the enterprise-specific MIB for a device that you use on your network, in the Tivoli NetView program’s MIB description file. These menu operations work for SNMPv1 MIBs and SNMPv1 or SNMPv2 MIBs, respectively. You can also use these operations to load other MIBs.

The purpose of loading a MIB is to define the MIB objects so the Tivoli NetView applications can use those MIB definitions. The MIB you are interested in must be loaded on the system where you want to use the [MIB Browser] [MIB Data Collector] [MIB Application Builder] and the applications built by the MIB Application Builder. In a distributed network environment, load enterprise-specific MIBs on the Tivoli NetView server. Because the directory where the MIBs are stored, /usr/OV/conf, is automatically NFS mounted onto the client machines during the Tivoli NetView installation process, loaded MIBs are available to the clients. If you have unloaded all the MIBs in the MIB description file, you must load MIB-I or MIB-II before you can load any enterprise-specific MIBs.

Loading MIBs

To load an enterprise-specific MIB, you can first copy the MIB into the default directory, /usr/OV/snmp_mibs. Otherwise, if you know the full path name of the directory where the MIB is located, you can enter it in the MIB File to Load text field in the Load MIB From File dialog box.

Each MIB that you load adds a subtree to the MIB tree structure. You must load MIBs in order of their interdependencies. A MIB is dependent on another MIB if its
highest node is defined in the other MIB. For example, the MIB ibm-alert.mib is
dependent on the MIB ibm.mib, so ibm.mib must be loaded before ibm-alert.mib is
loaded.

**Steps**
To load a MIB, follow these steps:

1. From the Tivoli NetView main menu, select either **Load/Unload MIbs -> SNMP...** for SNMPv1 MIbs or **Load/Unload MIbs -> SNMPv1/SNMPv2...** for SNMPv1 or SNMPv2 MIbs from the **Options** pull-down menu.
   The Load/Unload MIbs window is displayed, containing a scrollable list of all
   loaded MIbs, as shown in Figure 34.

![Load/Unload MIbs Dialog Box](image)

2. Click **Load...** to display the Load MIB From File dialog box, which contains a
   scrollable list of the MIbs files in the default MIb directory, /usr/OV/snmp_mibs.

3. Select the MIb you want to load.

4. Click **OK**.

The Load/Unload MIbs operation stores loaded SNMPv1 MIbs in the
/usr/OV/conf/snmpmib database and SNMPv2 MIbs in the /usr/OV/conf/snmpv2mib
database, which is known as the Loaded MIb Database. Do not try to edit this file
directly; instead, make any changes to this file through the Load/Unload MIbs
operation.

When the MIb is loaded, you can traverse the MIb tree and select objects from the
enterprise-specific MIb to use in the following operations:

- Browsing MIbs
- Collecting MIb data
- Building MIb applications
- Running applications you build with the MIb Application Builder

**Unloading a MIb**
To unload enterprise-specific MIbs, select the MIb from the list of loaded MIbs in
the MIb Load/Unload MIbs window and select **Unload...**. Click **OK** in the Unload
MIb—Confirmation dialog box to unload the MIb.
Browsing MIBs

Use the MIB Browser to query and set MIB values for both Internet-standard and enterprise-specific MIB objects.

You can also use the MIB Browser to graph MIB objects and their specific instances. Some MIB objects can occur several times per network object, each time with a different value. Each such occurrence of the MIB object is called an *instance*.

For example, the interfaces MIB object ifDescription has as many instances as its associated network object has interfaces, as shown in the following example:

1 : lo0; Software Loopback
2 : tk0; trty0; IBM 6611 Token-Ring Network Interface
3 : tk1; trty1; IBM 6611 Token-Ring Network Interface

However, the system MIB object sysContact has only one instance per network object, because generally there is only one person so designated. This is MIB instance 0, as shown in the following example:

sysContact.0: J. J. Wanscott 555-1234

To use the MIB Browser, follow these steps:

1. Select an SNMP network object whose MIB objects you want to view on a submap.
2. From the Tivoli NetView main menu, select either *MIB Browser* –> *SNMP*... for SNMPv1 agents or *MIB Browser* –> *SNMPv1/SNMPv2*... for SNMPv1, SNMPv2, or SNMPv2USEC agents from the *Tools* pull-down menu.

The Browse MIB window contains the name or IP address of the network object you selected, as shown in Figure 35 on page 180.
3. In the Community Name field, enter the community name of the agent that is running on the selected object. If left blank, the community name defaults to public. The community name functions as a password for different levels of access to MIB objects. For example, to retrieve the value of a MIB object, you need to know the community name that permits SNMP Get operations. Generally, this community name defaults to public. If you want to change the value of a MIB object, you must know the community name that permits SNMP Set operations.

4. Select nodes along the path of the MIB tree and select either Up Tree or Down Tree to traverse the tree. Generally, the MIB objects you will be working with are contained in either the mgmt branch, which contains standard MIB definitions, or the private branch, which contains enterprise-specific MIB definitions.

5. When you have reached a leaf node, or actual MIB object, select Describe to see a description of the object to determine whether it is the one you want. The Describe MIB Variable dialog box displays the name of the MIB object, its object ID, and the type of MIB object it is, and gives a brief description of its meaning.
Figure 36 shows the description of the snmpOutTraps MIB object.

Figure 36. MIB Object Description-snmpOutTraps

6. To query the MIB object for its current value, click Start Query. You do not have to be at a leaf node in order to use the Start Query operation. You can begin a query at any intermediate node to retrieve all the MIB objects in that node’s subtree at the same time. Once you see the list displayed in the MIB Values field, you can select one and look at its description. The current value of the MIB objects will be displayed in the MIB values fields. To stop the query, click Stop Query.

7. You can graph real-time data on the MIB object you have selected, provided it is numeric. Click Graph to display the Tivoli NetView Grapher window and look at the real-time values of the selected MIB object.

Using the Tivoli NetView Performance Applications

The Tivoli NetView program provides applications that enable you to monitor both real-time and historical network performance. The predefined applications monitor and graph real-time network performance. These predefined applications are described in "Using Tivoli NetView Predefined Applications" on page 182. The MIB Application Builder enables you to create your own applications to collect, display, and save real-time MIB data.

The MIB Data Collector provides a way to collect and analyze historical MIB data over long periods of time to give you a more complete picture of your network’s performance.

The MIB Application Builder is described in "Building MIB Applications" on page 185.
Comparison of MIB Applications

Table 15 compares the MIB Application Builder and the MIB Data Collector.

<table>
<thead>
<tr>
<th>Task</th>
<th>MIB Application Builder</th>
<th>MIB Data Collector</th>
</tr>
</thead>
<tbody>
<tr>
<td>Starting</td>
<td>Must always be started by selecting <strong>Tools..MIB Application Builder</strong> from the main menu</td>
<td>Data collection automatically started when the Tivoli NetView program is started, unless the collection is in suspended mode</td>
</tr>
<tr>
<td>Selecting MIB Variables</td>
<td>• Can select more than one MIB variable</td>
<td>• Can collect data for one or all MIB instances</td>
</tr>
<tr>
<td></td>
<td>• Can apply selection rules to MIB variables</td>
<td>• Can specify a single interface, but data is collected on the MIB index (instance)</td>
</tr>
<tr>
<td>Specifying Polling Intervals</td>
<td>Can specify polling intervals only for Graph format</td>
<td>Can change polling intervals for individual devices</td>
</tr>
<tr>
<td>Selecting Agents</td>
<td>Agents must be selected from map</td>
<td>Define agents by using pattern-matching characters, selecting from map, or by entering the agent name in the field.</td>
</tr>
<tr>
<td>Setting Thresholds</td>
<td>Not applicable</td>
<td>Can choose one of the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Exclude Collection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Store, Check Thresholds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Store, No Thresholds</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Don’t Store, Check Thresholds</td>
</tr>
<tr>
<td>Generating Events</td>
<td>Not applicable</td>
<td>Can generate events and define specific trap numbers</td>
</tr>
<tr>
<td>Selecting Output Format</td>
<td>Choose table, form, or graph</td>
<td>Shows data in table format. You can select <strong>Graph</strong> from the table display.</td>
</tr>
<tr>
<td>Customizing Output</td>
<td>• Can customize the “name” of the MIB variable displayed in the title bar</td>
<td>Uses applications in the <strong>/usr/OV/reports/C</strong> directory to graph output and generate reports</td>
</tr>
<tr>
<td></td>
<td>• Cannot customize line names</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Customization of graphic display lost between invocations</td>
<td></td>
</tr>
<tr>
<td>Saving Output</td>
<td>Can save output to /usr/tmp/xnmappmon.save or choose another file name</td>
<td>Collection file always stored in /usr/OV/database/snmpCollect. You cannot specify the file name</td>
</tr>
<tr>
<td>Exporting to ASCII Format</td>
<td>• Cannot export graph data to ASCII format</td>
<td>• Can convert to ASCII format</td>
</tr>
<tr>
<td></td>
<td>• Can export summary instance table format</td>
<td>• Time points provided through the Tivoli desktop (For <strong>AIX</strong> and <strong>Solaris</strong>)</td>
</tr>
<tr>
<td></td>
<td>• No time points are provided (for <strong>AIX</strong> and <strong>Solaris</strong>)</td>
<td></td>
</tr>
<tr>
<td>Performing Other Calculations on MIB Variables</td>
<td>Not applicable</td>
<td>Apply expressions in /usr/OV/conf/mibExpr.conf file to MIB variables</td>
</tr>
</tbody>
</table>

Monitoring Real-Time Network Performance

Tivoli NetView provides applications that enable you to monitor network real-time performance. It also provides a tool, the MIB Application Builder, that enables you to build applications. This section describes the Tivoli NetView predefined applications and the MIB Application Builder.

Using Tivoli NetView Predefined Applications

Many of the performance applications described in this section display their data in graphs. Some performance applications do not graph the data they collect; see the
individual application description for more information. Refer to the Help system entries for each application for complete information.

**Monitoring CPU Performance:** If you suspect that some network nodes are handling more tasks than others and might become overloaded, you can check this by selecting System Activity → CPU Performance from the Monitor pull-down menu. To use this application, one of the following conditions must be true:

- The trapgend daemon must be installed on a RS/6000
- The node must be an HP node.
- The Tivoli NetView MLM feature of Tivoli NetView must be installed.

This application monitors and graphs the average number of jobs performed in the last minute. Use this application to determine whether the system workload needs to be balanced.

**Monitoring Disk Space:** You can check the status of disk space on a remote SNMP agent node by selecting System Activity → Disk Space... from the Monitor pull-down menu. This operation provides the following information:

- Name of the file system
- Total KB of space
- Number of KB used
- Number of KB available
- Percent of total capacity being used
- Directory name on which the file system is mounted

To use this application, one of the following conditions must be true:

- The trapgend daemon must be installed on a RS/6000 node.
- The node must be an HP node.
- The Tivoli NetView MLM feature of Tivoli NetView must be installed.

**Monitoring Interface Traffic:** If you suspect network performance problems, you can select Network Activity → Interface: Traffic... from the Monitor pull-down menu to see a graph of packet statistics for selected SNMP nodes. If the graph reveals that packets are not being received or transmitted by certain nodes, you can then begin fault management procedures for those nodes.

**Monitoring Ethernet Performance:** You can monitor and graph performance for Ethernet networks by using the operations listed under Network Activity → Ethernet from the Monitor pull-down menu.

**Monitoring Ethernet Traffic:** Select Network Activity → Ethernet → Traffic... from the Monitor pull-down menu to monitor and graph interface statistics for a Cisco gateway server that supports SNMP. The following information is displayed in the graph:

- The number of packets successfully transmitted by the interface hardware
- The number of packets successfully received by the interface hardware
- The number of bits per second successfully transmitted
- The number of bits per second successfully received

**Monitoring Ethernet Errors:** To view interface error statistics for a gateway server that supports SNMP, select Network Activity → Ethernet → Errors from the Monitor pull-down menu. This operation graphs the average number per second as computed over the previous polling interval for the following statistics:
• The number of packets received with a Cyclic Redundancy Checksum (CRC) value that is not valid
• The number of packets received with a Frame Check Sequence (FCS) value that is not valid
• The number of packets received that are smaller than the physical media permits
• The number of packets received that are larger than the physical media permits
• The number of packets that arrive too quickly for the interface hardware to receive.

**Monitoring TCP Connections:** Select Network Activity -> TCP: Connections... from the Monitor pull-down menu to list the TCP connection table for selected remote SNMP nodes. The list shows the local and remote addresses of the node and the state of each session. This operation can help you determine whether performance problems are caused by the TCP layer instead of the SNMP layer.

**Monitoring SNMP Network Activity:** You can monitor and graph performance for SNMP networks by using the operations listed under Network Activity -> SNMP from the Monitor pull-down menu.

**Monitoring SNMP Traffic:** Select Network Activity -> SNMP -> Traffic... from the Monitor pull-down menu to monitor and graph the SNMP network traffic to and from selected nodes. You can compare this graph to general interface traffic graphs to determine the percentage of traffic to and from the selected node that is attributed to the SNMP management of the selected node. The following information is displayed in the graph:
• The number of packets per second received by the selected node that carry SNMP traffic
• The number of packets per second transmitted by the selected node that carry SNMP traffic

**Monitoring SNMP Operations:** Select Network Activity -> SNMP -> Operations... from the Monitor pull-down menu to monitor and graph the SNMP operations requested of and performed by the SNMP agent on the selected nodes. The number of operations reported is the number of operations for the given category that have occurred since the SNMP agent on the selected node was last started. The following information is displayed in the graph:
• The number of MIB objects retrieved successfully from the selected node through Get and Get Next requests
• The total number of MIB objects modified successfully on the selected node through Set requests
• The number of SNMP Get requests received and successfully processed by the selected node
• The number of SNMP Get Next requests received and successfully processed by the selected node
• The number of SNMP Set requests received and successfully processed by the selected node
• The number of SNMP Traps (unsolicited notifications) transmitted by the selected node

**Monitoring SNMP Errors:** Select Network Activity ->SNMP -> Errors from the Monitor pull-down menu to display a summary of the SNMP protocol errors detected by the SNMP Agent on the selected nodes. The number of errors reported...
is the total number of errors for the given category that have occurred since the
SNMP Agent on the selected node was last started. The following information is
displayed in the graph:

- ASN parsing errors
- SNMP requests by an unsupported version of SNMP
- SNMP requests received with an incorrect community name
- SNMP requests received with a community name that is inappropriate for the
  operation requested
- SNMP requests that could not be processed because the request message is
  larger than the maximum message size supported by the selected node
- SNMP requests that contained a MIB object not supported by the selected node
- SNMP set requests that contained values not valid for the specified MIB object
- SNMP requests that could not be processed for some other reason

**Monitoring SNMP Authentication Failures:** An authentication failure occurs
when a management system sends an SNMP request to an agent but does not
send the correct community name, or password, with the request.

If you have a map open with read-write access, you can obtain a list of the
management systems that have caused authentication failures on a selected node.
The node you select must be an HP node.

Select **Network Activity -> SNMP -> Authentication Failures** from the **Monitor**
pull-down menu. From the resulting display, you can learn that:

- Certain management systems are not permitted to communicate with this
  particular agent.
- The agent might have a configuration problem, which you can take steps to
  resolve.

**Building MIB Applications**
The MIB Application Builder enables you to build MIB applications without
programming. This is helpful when you want to monitor real-time performance of
specific MIB objects. You can build MIB applications that poll certain MIB objects on
a regular basis and produce output such as forms, tables, or graphs.

The new MIB application is placed in the Monitor menu by default. To look at MIB
values returned by the new application, select the appropriate item under the
Monitor option on the Tivoli NetView main menu.

Select **MIB Application Builder...** from the **Tools** pull-down menu. The MIB
Application Builder dialog box is displayed, as shown in **Figure 37 on page 186**.
The MIB Application Builder window contains a list of all existing MIB applications and the location of their selections in the Tivoli NetView menu structure. You can select an existing application to modify or delete, or you can add a new application.

Adding a MIB Application: To add a MIB application, click Add... button. The Add MIB Application dialog box contains four sections, each of which is described in the following paragraphs.

The top section contains text fields where you can enter an Application ID, or name, for your MIB application. This name should be the same as the name you specify in the menu path that enables users to access your application. Your MIB application will be available from the Tivoli NetView main menu and from the object context menu.

The Application Title field enables you to specify the text that is displayed on the title bar in the application once it has been selected. You can select one of the following formats for your application's output:

- **Form**: Use only MIB objects that have a single instance per system, that is, those that occur only once.
- **Table**: No restrictions identified.
- **Graph**: Use only MIB object types Integer, Counter, and Gauge. Variables with these types can have more than one instance per system.

The Display Fields section contains data about the MIB objects you select to be monitored by this application. Select the Add... button to display the MIB Application Builder/Add MIB Objects dialog box. Use this dialog box the same way you used the Browse MIB dialog box to find the MIB objects you want to add to your application.
The Tivoli NetView Integration section is where you decide what menu path users must traverse to select your application. The default main menu selection is Monitor; you can choose selections under Monitor that are most appropriate for your application, or add selections if necessary. You can also change the selection rule, which directs the application to begin only if the selected objects meet the criteria found in the rule. For example, if the selection rule is:

(isSNMPSupported || isSNMPProxied)

The MIB application will run only if the objects selected on the map support SNMP directly or through a proxy agent. If you select an object that does not meet the selection criteria, the MIB application’s menu selection is grayed.

In the bottom section, Help Text, you can enter some basic information about what your application does and how to use it. It is good practice to enter at least a minimum amount of information in case someone wants to know what your application does. A user can access Help for your application by clicking Help on the application menu bar and selecting On Application.

Once you have completed all sections of the Add MIB Application dialog box, click OK to add the new MIB application to the list of available MIB applications. Click Close to exit from the MIB Application Builder.

**Running MIB Applications through the Graphical Interface:** To run the application, select an object or objects on the network map, select the Monitor option from the Tivoli NetView main menu, and follow the menu path you entered for your application. If your application graphs MIB values, it runs in the Control Desk window until you stop it by selecting File..Exit from the application menu bar. If your application produces a table or form, it runs in a separate window.

If a MIB application does not run, check the following conditions:

- Ensure that the MIB is loaded into the Loaded MIB Database. Sometimes, a MIB is loaded for the specific purpose of creating a MIB application, and then it is unloaded to conserve space in the database.
- Check that the MIB object ID you have selected is supported by the device you want to monitor.
- Ensure that the application was created to monitor the selected device and that the community name is properly set. Check the /usr/OV/conf/snmp.conf file to verify the community name.

**Running MIB Applications Using the runapp Command:** You can run an existing MIB application without using the MIB Application Builder. The `runapp` command executes the MIB application, using the application name and host name you specify, as shown in the following example:

```
runapp -a mibObjApp -h host1
```

For more information about the `runapp` command, refer to the man page.

**Collecting Historical Performance Information**

You can compile historical performance data about your network by using the Tivoli NetView MIB Data Collector. This tool enables you to manipulate data in several ways, including:

- Collect MIB data from network nodes at regular intervals.
• Store MIB data in a file. If Tivoli NetView is configured to work with a relational database, you can transfer collected data to a relational database and use the relational database tools to create reports. Refer to the Tivoli NetView for UNIX Database Guide for more information about transferring collected data to a relational database.

• Define thresholds for MIB data and generate events when the specified thresholds are exceeded. Setting MIB thresholds enables you to automatically monitor important network and system parameters to help you detect and isolate problems.

Using the MIB Data Collector
Before you configure your system to collect MIB data, you need to understand the definitions of the MIB objects and what they do. To look at descriptions for selected MIB objects, select MIB Browser: SNMP... from the Tools pull-down menu. Refer to the vendor documentation for information about their enterprise-specific MIBs.

Also ensure that there is enough room to store data in the /usr/OV/databases/snmpCollect directory. It might be necessary to remove some files in this directory to make space available. You can set a crontab command to periodically remove files, or use the snmpColDump command to edit the directory files and delete selected lines. See the man pages for more information about using these commands.

Select Data Collection & Thresholds: SNMP... from the Tools pull-down menu. The MIB Data Collection dialog box is displayed, as shown in Figure 38 on page 189.
The MIB Data Collection dialog box contains three sections:

- **MIB Objects Configured for Collection**
  Use this section to select the MIB objects or MIB expressions on which to collect data.

- **MIB Object Collection Summary**
  Use this section to identify the nodes from which to collect MIB data and to view the current collection defaults.

- **Collection Details**
  Use this section to identify details about the data collection such as how often to collect data and whether to send threshold events.

**MIB Objects Configured for Collection:** The MIB Objects Configured for Collection section of this dialog box contains the following fields:
Status
Indicates whether data collection is suspended or active. You can change
the status by clicking either Suspend or Resume.

Label
Displays the name of the file to which data will be stored.

MIB Object ID
Displays the object identifier of the MIB object configured for collection. If
the value in this field does not begin with a dot (.), it is a MIB expression.
This is the path through the MIB tree that ends in this particular leaf node,
or object.

MIB Object Collection Summary: The MIB Object Collection Summary section of
this dialog box contains the following fields:

Interval
Specifies how often data is collected from a source. The s, m, h, or w after
the number indicates seconds, minutes, hours, or weeks.

Store
Indicates whether or not data is being stored.

Threshold
Displays two values: the first is the threshold, the second is the rearm. If
the threshold and rearm values are not defined, the Threshold column
displays 0.00 0.00.

Source
Displays the name of the collection source. The source name can be an
individual node or a set of nodes based on an IP address wildcard, for
example, 15.212.*.*.

Instances
Displays the MIB instances on which data is to be collected. The instance is
an internal counter on the system. For example, assume you have multiple
disks on a node and the name of the MIB object is disks. The instance tells
the data collector on which disk to collect data. An instance value that is an
asterisk (.* ) signifies that data is to be collected on all instances. A
numerical instance value specifies a particular instance. An instance can
also be a regular expression.

Collection Details: The Collection Details section of this dialog box contains the
following fields:

Collection Mode
Determines the collection mode. Use one of the following four values:
• Exclude Collection
• Store, Check Thresholds
• Store, No Thresholds
• Don’t Store, Check Thresholds

Polling Interval
Determines how often data is collected from a source. Enter a positive real
number followed by an s, m, h, or w that indicates seconds, minutes, hours,
or weeks. If you do not enter a letter following the number, the data
collector uses the default of seconds. For example, 1.5h indicates one and
a half hours, and 30 indicates 30 seconds.

Trap Number
Specify an enterprise-specific trap number. The default trap is 58720263,
the enterprise-specific trap that the MIB Data Collector sends when a
Threshold or a rearm value is exceeded. The MIB Data Collector sends the trap using the enterprise ID of the management station.

Threshold
Enter a threshold value that specifies when you want to be notified of traffic patterns that are outside the normal expectations. When the threshold value is passed, the specified threshold event is generated.

Rearm
Enter an appropriate rearm value to control the frequency of threshold events generated. When a MIB value drops below or is equal to the rearm value, a rearm event is generated. Another threshold event will not be generated until the rearm event occurs and the collected value again exceeds the threshold value after being rearmed.

Instances
Use the option list to select the type of instances you want to enter in the text field to the right of the option button. Specify the instance of the MIB object on which you want to collect data. If the object on which you want to collect data does not support multiple instances, the instance is zero. If you have multiple instances of a MIB object on a node, you must specify the instance on which you want to collect data. See the dialog box help for more information.

Example of Collecting MIB Data
Suppose you want to collect information about the number of inbound SNMP packets received by objects on a particular network. Follow these steps:

1. Data Collection & Thresholds: SNMP... from the Tools pull-down menu.

2. Select the snmpInPkts item from the default list of configured collections. The current status of snmpInPkts data collection is Suspended.

   Note the information about this list item that is displayed in the MIB Object Collection Summary area of the dialog box. In this area, you can click Add... if you want to add other nodes from which to collect this data.

3. Select the list item in the MIB Object Collection Summary area of the MIB Data Collection dialog box. The bottom area of the dialog box, Collection Details, becomes active and displays the values from the Collection Summary area in fields that you can change.

   The following values apply to this particular collection:
   • The polling interval is 1 hour.
   • The data collected is to be stored and checked against a threshold value.
   • When the collection frequency exceeds 15 occurrences per hour, a threshold event will be generated, sending trap number 58720263 to the manager.
   • Once the trap has been sent, the rearm value, 70%, controls the frequency with which subsequent traps signifying a threshold event will be sent.

4. If this information suits your needs, you can click Resume and then Apply to apply the change in status.

Notes:

   a. Although the Status field for snmpInPkts changes to Collecting after you click Resume, the change does not take effect until you click Apply at the bottom of the dialog box.

   b. Data collection is restarted each time you click Apply which means that polling is interrupted. For example, suppose that you have set polling intervals on a MIB object to 1 hour. Clicking Apply causes the data collector...
to begin polling, even if the last poll took place only 5 minutes ago. Clicking **Apply** several times will increase network traffic and store more data than you might have intended.

5. To see the collected data, click **Show Data...** in the MIB Objects Configured for Collection section of the MIB Data Collection dialog box. Collected data will be displayed after the first polling interval you specified has passed. In this example, no data will be displayed until one hour has passed from the time you clicked **Apply** to resume data collection for snmpInPkts.

The data is displayed as a table that lists the polling interval, time of collection, the source node from which the data was collected, and the value of snmpInPkts for that node. If you want to see a graph of the collected data, click **Graph** at the bottom of the MIB Data Collection/Show Data dialog box.

6. You can exit from the Tools..Data Collection & Thresholds: SNMP operation and view the collected data at a later time by selecting **Tools..Graph Collected Data: SNMP** from the Tivoli NetView main menu. Again, you must wait until after the specified polling interval has passed before there will be any collected data to view.

If no data is being collected, make sure that the snmpCollect daemon is running on the manager. This daemon stops running when file system space is not available. If you have root authority, you can use the Tivoli desktop to restart the daemon. Then, you can restart the data collection.

Refer to the man page for more information about the snmpCollect daemon.

**Graphing MIB Variable Expressions**

You can graph the result of expressions applied to MIB variables. Expressions for manipulating MIB variables are stored in the /usr/OV/conf/mibExpr.conf file. These expressions are in postfix format. You might want to try using the example expressions before you create others.

**Steps:** To create and graph the results of an expression, take the following steps:

1. Use one of the example expressions or add an entry to the /usr/OV/conf/mibExpr.conf file.
2. Select an object or objects on a submap.
3. Select **Data Collection & Thresholds: SNMP...** from the **Tools** pull-down menu.
4. Click **Add...** in the MIB Data Collection dialog box.
5. Select **Expression** in the MIB Data Collection / MIB Object Selection dialog box.
6. Select an expression from the list that is displayed in the Expression ID text field and click **OK**. The MIB Data Collection / Add Collection dialog box is displayed.
7. Select the type of instances you will enter in the Instances text field to the right of the Instances option button. Or, enter . to collect all instances of the selected MIB variables.
8. Click **Add From Map** to display the selected object or objects in the List of Collection Sources field.
9. To change the collection mode, click **Collection Mode**. The selections are as follows:
   - Exclude Collection
   - Store, Check Thresholds
   - Store, No Thresholds
• Don’t Store, Check Thresholds

You can also change the threshold and rearm values if necessary. Click OK to close the dialog box.

10. The Status field in the MIB Objects Configured for Collection area of the MIB Data Collection dialog box now shows Collecting for this MIB object. However, you must click the Apply to start the data collection.

11. Once data has been collected, you can display it by clicking Show Data.. in the MIB Data Collection dialog box. The data is displayed in table format, but you can convert it to a graph by clicking Graph at the bottom of the MIB Data Collection/Show Data dialog box. You can also graph the collected data at a later time by selecting Tools..Graph Collected Data: SNMP from the Tivoli NetView main menu.

Using the setthresh Command
You can set a threshold for MIB data collection without using the Tools..MIB Data Collection operation. The setthresh command sets up data collection configurations and stores the values in the /usr/OV/conf/snmpCol.conf file. If any entry already exists with matching MIB object ID and source name, that entry is updated with the new information.

Example of Collecting Thresholds: You can collect data or monitor thresholds only on numeric MIB variables, that is, those that are defined as type Counter, Gauge, or Integer. The following example shows how the setthresh command is used to collect the same data that was collected in the previous example:

```
setthresh -o snmp.snmpInPkts -s R
    -n *.*.*.*
    -c W -m s -p 1h
    -v 10 -r 70 -t %
    -i ALL -T 58720263
```

For more information about the setthresh command, refer to the man page.

Monitoring File System and Paging Space

Collecting network and performance data can quickly deplete key system resources. If file system or paging space become full, some processes might stop. However, it is not always convenient to continually monitor the system for this problem.

When you want to monitor file system or paging space, select Local Filesystem and Paging Space... from the Monitor pull-down menu to receive a trap or a message that informs you when a threshold condition has reached its limit. This option monitors the root file system and the paging space on the local manager system where Tivoli NetView is installed (usually /usr/OV). If /usr/OV/databases and /usr/OV/log are defined as separate file systems, the program also monitors them.

The file system and paging space monitoring process uses the values set in the Monitor File System & Paging Space dialog box.

The Monitor File System & Paging Space Dialog Box
When you select Local Filesystem and Paging Space... from the Monitor pull-down menu, the Monitor File System & Paging Space dialog box is displayed as shown in Figure 39 on page 194.
You can start file system monitoring (with or without dynamic increase of file system space) and accept or change the default values for file system and paging space thresholds, the polling interval, and the notification method. The Monitor File System & Paging Space dialog box contains the following settings:

- **File System & Paging Space Monitor Status**
  - **Monitor With Dynamic Increase**
  - **Monitor Without Dynamic Increase**
  - **No Monitor**

- **File System Threshold (% used):** 97
- **Paging Space Threshold (% used):** 87
- **Polling Interval (seconds):** 300

**Notification:**
- **Message Windows**
- **Events**

**Messages:**

![Monitor File System & Paging Space Dialog Box](image)

*Figure 39. Monitor File System & Paging Space Dialog Box*

You can start file system monitoring (with or without dynamic increase of file system space) and accept or change the default values for file system and paging space thresholds, the polling interval, and the notification method. The Monitor File System & Paging Space dialog box contains the following settings:

- **File System & Paging Space Monitor Status**

  For **AIX**:
  - Select **Monitor with Dynamic Increase** if you want to start file system and paging space monitoring and if you want file system size or paging space to be increased automatically when the threshold is reached.

  File system size is increased by 4 MB, and paging space is increased as follows:
  - If only one page space is defined and there are free partitions on the hard disk, the paging space is increased by 12 MB.
  - If more than one page space is defined and the page space with the smallest percentage of use has reached the threshold, the page space defined on the hard disk with the most free partitions is increased by 12 MB.
  - If more than one page space is defined on a single hard disk, then the page space with the smallest percentage of use is increased by 12 MB.
If no free partitions exist, users are notified by the notification method you specify (a message window or a trap).

The monitoring process also logs a message to the /usr/OV/log/shpmon.log file, indicating when the process was started, stopped, and the amount of increase that was made to the file system space or paging space.

- Select **Monitor Without Dynamic Increase** to start file system and paging space monitoring and to notify users when file system or paging space needs to be increased.

For **Solaris**:
- Select **Monitor** to start file system and paging space monitoring and to notify users when file system or paging space needs to be increased.
- Select **No Monitor** when you want to stop the monitoring process. This is the default.

The monitoring process also logs a message to the /usr/OV/log/shpmon.log file, indicating when the process was started and stopped.

- **File System Threshold**
  The default threshold for file system space is **97** percent. You can use this value, or you can type a different value in this field. Users are notified when the threshold is reached.

- **Paging Space Threshold**
  The default threshold for paging space is **87** percent. You can use this value, or you can type a different value in this field. Users are notified when the threshold is reached.

- **Polling Interval**
  The default polling interval is **300** seconds. You can use this value, or you can type a different value in this field.

- **Notification**
  You can notify users with an audible message box or by generating a trap. The default is to generate a trap. You can configure the trap to be displayed and logged using the Options..Event Configuration..Trap Customization: SNMP operation.

**Starting and Stopping the Monitoring Process**

To start or stop the file system and paging space monitoring process, follow these steps:

1. Select **Local Filesystem and Paging Space...** from the **Monitor** pull-down menu.
   - The Monitor File System & Paging Space dialog box is displayed as shown in [Figure 39 on page 194](#).

   **Note:** For **Solaris**, the Monitor File System & Paging Space dialog box is slightly different than [Figure 39 on page 194](#).

2. Make the appropriate changes to the dialog box. See [The Monitor File System & Paging Space Dialog Box](#) on page 193 if you need more information about the dialog box settings.
3. Click **Apply** to accept the changes and close the dialog box.

   **Note:** If you stop the monitoring process when the process is inactive (depending on the polling interval), it seems as though the process is still running.
Because the termination signal has not yet been received, if you attempt to start the monitoring process again, a message is displayed telling you that the process is already running. When the monitoring process becomes active, the monitoring process receives the termination signal and stops.

**Starting from the Command Line**

You can start the file system and paging space monitoring process from the command line and specify what options to use. Use the following command to start the monitoring process:

**For... Enter...**

**AIX**

```
shpmon [-m 1|0] [-e] [-t update_time][-Q]
```

**Solaris**

```
shpmon [-e] [-t update_time] [-Q]
```

Where:

- **-m 1|0** Starts with or without dynamic increase. The value 1 starts monitoring with dynamic increase. You must be a root user to use this option. A 0 (zero) value starts monitoring without dynamic increase.
- **-e** Executes the program one time and exits. The monitoring process checks the thresholds set in the Monitor File System & Paging Space dialog box and responds depending on what other flags are used. See **The Monitor File System & Paging Space Dialog Box** on page 193 for more information.
- **-t** Specifies a polling time interval in seconds. If this option is not specified, the value set in the Monitor File System & Paging Space dialog box is used. See **The Monitor File System & Paging Space Dialog Box** on page 193 for more detailed information.
- **-Q** Suppresses all messages sent to the user when the threshold value has reached its limit. Messages about changes to the file system and paging space sizes are logged in the shpmon.log file. Events are generated when threshold values are exceeded and cannot be increased.

**Monitoring Specific Events**

To configure the Local Filesystem and Paging Space option to execute only when a specific event occurs, use the Event Configuration dialog box. When a threshold event occurs for the configured MIB object ID, the program executes, takes the appropriate actions, and exits.

Follow these steps to configure system events. You must be a root user.

1. Select **Data Collection & Thresholds: SNMP...** from the **Tools** pull-down menu. On the **MIB Data Collection** dialog box, configure the threshold value and trap number for the disk utilization MIB object. See **Using the MIB Data Collector** on page 188 for those steps.
2. Click **OK**.
3. Select **Event Configuration..Trap Customization: SNMP...** from the **Options** pull-down menu. On the **Event Configuration** dialog box, select the same trap number you configured on the MIB Collection dialog box. The trap numbers are listed in the Event Identification section under Event. Use the Help option if you are not familiar with the Event Configuration dialog box.
4. Click **Add**. The Add Event dialog box is displayed.
5. Enter the following command in the Command for Automatic Action field:
For... Enter...

AIX and Solaris
/usr/OV/bin/shpmon -m 1 -e -Q

6. Click OK to close the Add Event dialog box.
7. Click OK to close the Event Configuration dialog box.

This command executes the shpmon process only one time when the specified event occurs. It checks the file system and page space, increases them if necessary, and exits. Paging space is increased on the AIX operating system only.

Using the Tivoli NetView Graph Applications

Graph applications provide a convenient way to display performance information. This section describes how to use and customize graph applications.

Starting a Graph Application

To start a graph application, select an object or objects on a submap, then select the graph application whose results you want to view. All graph applications can be started from either the Tivoli NetView main menu or the object context menu. The graphs are displayed in a control desk window.

MIB Graph Applications

The following MIB graph applications can be started from an object context or main menu:

• Monitor -> System Activity -> CPU Performance
• Monitor -> System Activity -> Disk Space
• Monitor -> Network Activity -> Interface: Traffic
• Monitor -> Network Activity -> Ethernet -> Traffic
• Monitor -> Network Activity -> Ethernet -> Errors
• Monitor -> Network Activity -> TCP: Connections
• Monitor -> Network Activity -> SNMP -> Traffic
• Monitor -> Network Activity -> SNMP -> Operations
• Monitor -> Network Activity -> SNMP -> Errors
• Monitor -> Network Activity -> SNMP -> Authentication Failures

If you start one of these applications from the Tools window, you can place it in a control desk or drop it anywhere else on the desktop to make it a stand-alone application.

If you start one of these applications from an object context menu, it is displayed in the control desk. Starting one of these applications from an object context menu means it applies only to the selected object. If you want to collect and graph data for more than one network object, select the objects on the submap, then select the appropriate application from either the Tivoli NetView main menu or the Tools window.

Saving Performance Data

To save the performance data that is displayed in the graph application, select File..Save As.. from the menu bar in the application window. You can save the data to the default directory and file name, /tmp/xnmgraph.data, or specify a different directory and file name. Click Apply to save the data. Look in the Messages area for any messages generated as a result of the Apply operation. For example, if the save is successful, the message tells you how many data points were saved for each monitored MIB object.
The Save As.. operation dumps an ASCII file containing the data to the directory and file name you specify. The ASCII file displays the collected data in table form for each monitored MIB object.

To re-create the graph from the contents of the ASCII file, enter the following command at the command line:
```
```

### Printing Graphed Data

To look at some of the graph information in printed format, print the contents of the graph window. Select the Print Tool icon in the Tools window and drag it to another area on the desktop to open a Print Tool window. Follow these steps to capture and print the contents of the graph window:

1. Rearrange the windows on your desktop so that the window containing the graph display does not overlap with any others.
2. In the Print Tool window, click **Capture**.
3. Move the pointer, which has changed shape from an arrow to a hand, to the graph window you want to capture and click.
4. In the View Identification dialog box that is displayed, enter a file name for the captured information. Click **OK** to close the View Identification dialog box.
5. The file name you specified is displayed in the Selected field of the Print Tool window. You can either print the captured information immediately or save it to print later.

Before you print, it might be helpful to save the captured information in a directory in case you need to print it again.

6. To print the captured information immediately, select the name of the file and specify the name of a printer and the number of copies, then click **Print**. The Print Tool window closes.
7. To save the captured information immediately, select the name of the file and then select **Save...** from the **File** pull-down menu in the Print Tool window. In the File Save dialog box, enter the directory path where you want to store the file. You do not need to enter a file extension because the Print Tool automatically adds .ps to the file name if you save the captured image in a directory. Click **OK**.
8. To exit the Print Tool, select **Exit** from the **File** pull-down menu in the Print Tool window.

Note that the Print Tool captures and prints only the information currently contained in the graph window. It does not print the entire contents of the graph. If you need to print the entire graph, you can page forward or backward and use the Print Tool to print each page of graphed information. To page forward or backward, click anywhere on the graph and select **Page Forward** or **Page Backward** from the context menu that is displayed.

### Showing Counter Values

If you are graphing MIB values of type Counter, you can select **Show counters As...** from the **View** pull-down menu in the Print Tool window and choose one of the following ways to display the data:
Rate of Change
The default value shows the new counter value as a time-averaged value since the last query for the MIB object.

Actual Sampled Value
Contains the value returned from the MIB Counter variable.

Delta Value
Contains the change in the MIB variable since the last query for the MIB object. This value is not time-averaged.

Example
For example, suppose you are graphing a MIB variable with the following statistics:

Value of MIB variable at time 0 --> 100
Value of MIB variable at time 10 --> 300

The value from time 10 would be graphed in the following ways:

Rate of Change
20 (derived from (300−100)/10)

Actual Sampled Value
300

Delta Value
200 (derived from (300−100))

Adding a Line
An application sometimes has more lines to graph than the maximum number that are specified for the graph when the application is created. Select Add Line from the View pull-down menu to temporarily increase the number of lines the graph can display. This selection will be grayed out if the number of lines to be graphed does not exceed the maximum number of lines that can be graphed, as defined in the app-defaults file of the application.

Using the Context Menu
The context menu in a graph application enables you to zoom in and out so you can look at the graph from different perspectives. You can also use the context menu to page forward or backward through the collected data, or to display the beginning of the data, the end of the data, or all of the data.

Generating Performance Reports

Tivoli NetView provides sample shell scripts that use real-time and historical information to generate performance reports. In addition to using these shell scripts, you can create others that collect and display the performance information you most need to see. These reports are stored in the /usr/OV/reports/C directory.

Select Reports: Site Provided from the Monitor pull-menu in the Tivoli NetView main window. The sample shell scripts provided with the Tivoli NetView program are displayed in the Run Report File window.

Figure 40 on page 200 shows the reports available with Tivoli NetView.
You can create your own reports to gather real-time or historical information about MIB objects. Your reports can be stored in the /usr/OV/reports/C directory, or in another directory of your choice. When you want to run a report that is stored in another directory, enter the full path name of the report in the Report File to Run field of the Run Report File dialog box.

Contents of the Report Directory

This section describes the files and shell scripts that are provided in the Tivoli NetView reports directory.

- The README file provides a description of each report in the /usr/OV/reports/C directory, including the type of object each report requires as a selection. It is good practice to update the README file each time you create a report and add it to the report directory.
The 0graphDemo shell script provides an example of using the Tivoli NetView graph facility, called xnmgraph, to graph information that is collected in a standard file.

The 0sumByDN shell script uses data from the SNMP Data Collector to provide an approximate estimate of which nodes have the most total traffic. This shell script must be run with data of type Counter.

The NgraphSumByDN shell script uses data collected by the SNMP Data Collector to estimate which nodes have the most total traffic by time of day. Only nodes that are currently selected will be graphed.

The NifInOctets shell script shows how to use the Tivoli NetView graph facility, xnmgraph, to graph a specific MIB object for selected nodes.

The NifOutOctets shell script shows how to use the Tivoli NetView graph facility, xnmgraph, to graph a specific MIB object for selected nodes. The values must have been collected by the SNMP Data Collector.

Writing Reports

You can write reports that send their output directly to the terminal so you can see the results immediately. Getting immediate feedback can help you with monitoring and troubleshooting. You can also write reports that store their results in a file or in a database, where they can be used to chart performance trends over a period of time.

Some reports are run only if an object is selected from the map before the report is selected. Other reports require that more than one object be selected, or that no objects be selected. You can tell which reports require selected objects, and how many selected objects, by looking at the first character of the report name.

Table 16. The number of Selected Objects a Report Requires

<table>
<thead>
<tr>
<th>First Character of Report Name</th>
<th>Number of Selected Objects</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>None</td>
</tr>
<tr>
<td>1</td>
<td>One</td>
</tr>
<tr>
<td>N</td>
<td>One or more</td>
</tr>
</tbody>
</table>

You might want to follow this convention when you create reports.
Chapter 8. Using the Agent Policy Manager (APM)

If you have Systems Monitor Version 2 (V2) Mid-Level Manager (MLM) or Tivoli NetView MLM Version 5 Mid-Level Manager (MLM) installed in your network, read this chapter to understand how to configure and to use the APM to more easily set up and view information about thresholds and file monitoring in a network.

This chapter is organized to accommodate both users who are familiar with Tivoli NetView MLM and those who are new to it and contains the following information:

- “What the APM Can Do for You” on page 203
- “Types of APM Policies” on page 205
- “Configuring and Starting the APM Daemon” on page 208
- “Starting the APM Configuration Application” on page 210
- “Creating, Changing, and Distributing an APM Policy” on page 211
- “Fixing an APM Policy” on page 217
- “Copying an APM Policy from One System to Another” on page 219
- “Completing the APM Policy Dialog Boxes” on page 219
- “Diagnosing Problems Using the Problem Determination Assistance Facility” on page 248
- “Example of Defining and Distributing a Policy” on page 253

This chapter also includes the APM Reference on page 256. This reference provides detailed information about how APM defines itself and its domains in the network, how community names are configured, and other reference information.

What the APM Can Do for You

If you use Systems Monitor V2 in your network, you already know that it is a powerful application. Through the MLM, you can offload some systems management from Tivoli NetView with the MLM’s thresholding capability. An MLM can identify a developing problem and alert you to it or take corrective action before your users ever become aware of it.

Tivoli NetView MLM is powerful, but it can require that you gain many new skills, especially if you are new to SNMP and the concepts of MIBs, community names, and sets and gets. APM, running in conjunction with Tivoli NetView MLM and the SmartSet Facility, allows you to accomplish these same tasks more easily by:

- Simplifying the task of defining threshold or file monitoring conditions by eliminating the need to define trap destinations.
- Providing the ability to distribute a threshold or file monitor configuration to groups of nodes as a single operation.
- Automatically creating submaps and an icon on your Tivoli NetView root map for MLMs and their managed nodes.
- Automatically creating icons on your root map that represent active threshold and file monitoring settings.
- Providing a way to filter file monitor traps so that only the ones you care about are forwarded to the Tivoli NetView Control Desk.

This management of agents in the network is accomplished by setting up policies. The policies you define are made up of two pieces:

- Defining rules about which object will be acted on (such as all routers, all machines in a building, or all devices in a certain subnet)
• Defining rules about **what** action will be taken (such as threshold on a MIB-II variable, monitor a log for an error message)

The process for adding a new policy is made up of two steps:

1. **Creating the new policy**
   How to create the new policy is described in "Creating or Copying an APM Policy" on page 211.
2. **Distributing the policy** to the nodes in the SmartSet.
   How to distribute the policy is described in "Distributing a Policy to Remote Nodes" on page 213.

APM simplifies the task of distributing changes to your network. Rather than NFS mounting Tivoli NetView MLM configuration files, or editing configurations on every MLM machine in your network, you use the SmartSet Facility to set up SmartSets to include all the nodes that will get a new policy. After you have defined the SmartSet to which the policy is to be distributed, it is a simple matter of clicking **Distribute** to update all the nodes in your network. The SmartSet Facility maintains a list of objects that fit SmartSet rules. It updates the list if changes in the network topology result in the addition of nodes that fit a SmartSet rule (or deletion of nodes already in a SmartSet). APM will act on these changes by configuring new nodes that fit the SmartSet rule or removing policies from nodes that no longer fit SmartSet rules. This policy maintenance takes place automatically.

See "Defining and Managing SmartSets" on page 76 for more information.

**APM SmartSet Icons That You Get Automatically**

APM and the SmartSet Facility create several new icons on the root map:

• **SmartSets** shows all SmartSets that have been defined. It is added to the root map by the SmartSet Facility, regardless of whether or not APM is configured to run.
  Double-clicking this icon displays a submap showing all of the defined SmartSets. Initially, these SmartSets do not reflect status. When you double-click a SmartSet icon, status is set and maintained for that SmartSet.

• **MLM Managers** displays all the MLMs in your network, with all the IP nodes in each MLM’s domain.
  Double-clicking this icon displays a submap of all MLMs in the network.
  Double-clicking an MLM displays the MLM and the nodes in its domain.
  Double-clicking a particular node shows details about the node itself, including IP interfaces and all distributed policies.

• **APM Monitors** is displayed after you define an APM policy. This icon indicates that APM policies are active.
  Double-clicking this icon displays all the SmartSets to which policies have been distributed.
  Double-clicking one of these SmartSets shows the nodes in the SmartSet against which the policies are set.
  Double-clicking an individual node displays details about the node itself, including IP status and icons for the individual APM policies. The icons for APM policies are executable icons. Double-click on an icon to access the **Problem Determination Assistance** dialog box.

  The Problem Determination Assistance dialog box is described in "Diagnosing Problems Using the Problem Determination Assistance Facility" on page 248.
Propagation of Policy Status to Map Icons

The color of each icon reflects the status of the policy as described in the following list:

**Yellow**
The APM Monitors icon on the root submap turns yellow when any threshold has been exceeded or when a file monitor condition has been met.

**Green**
Policy icons are green by default.

**Blue**
Policy icons turn blue when the distribution of the policy to the node failed. Threshold policy icons turn blue if the MLM session for thresholding the node goes down.

**Red**
File monitor policy icons turn red when a file monitor condition is met. Threshold policy icons turn red when arm conditions are met. Threshold policy icons turn green again when rearm conditions are met.

The MLM icon on the root map and its submaps propagates only IP status. The APM Monitors map aggregates IP status and APM policy distribution status. File monitor and threshold status are also aggregated into the APM Monitors map.

Because the SmartSets submap aggregates status, APM policy icon status also propagates to the SmartSets icon.

File monitor functions do not have a rearm definition. If the condition being monitored for is found and the APM Monitors icon changes color, reset the color manually through the Problem Determination Assistance dialog box.

The Problem Determination Assistance menu is described in "Diagnosing Problems Using the Problem Determination Assistance Facility" on page 248.

Types of APM Policies

This section describes the types of APM policies that you can define and distribute using the APM configuration application.

- "Threshold Policy"
- "File Monitor Policy" on page 206
- "Command Policy" on page 207
- "Filter Policy" on page 207
- "Trap Destination Policy" on page 208
- "Analysis Policy" on page 208
- "Alias Policy" on page 208
- "Administration Policy" on page 208

Threshold Policy

Through a threshold policy, you can collect important MIB data and can set thresholds to send a trap or to run a command when the threshold is tripped. Thresholds can be set on many types of MIB objects:

- MIB objects in the Tivoli NetView MLMMIB
- MIB-I or MIB-II objects
- MIB objects in standard or private MIB extensions that can be retrieved using SNMP gets (such as the MIB shipped with a router or other device)
With APM, you are setting a threshold against a group of nodes. The APM finds each of the nodes in the SmartSet and then determines which MLM in the network has monitoring responsibility for each of the nodes. It will be an MLM in the same domain as the node. The APM then does an SNMP set against the Threshold Table for each MLM to define the threshold. For example:

You define a SmartSet MYCOLL that includes nodes A, B, C, and D.

Nodes A and B are in the same domain as MLM 1.
Nodes C and D are in the same domain as MLM 2.

Through APM, you define a threshold to be distributed to SmartSet MYCOLL. When you click Distribute, an SNMP set command is done against the Threshold Tables on MLM 1 and MLM 2. Each of these MLMs will begin polling their respective managed nodes for the MIB data specified in the threshold. MLM 1 now thresholds on nodes A and B, and MLM 2 thresholds on nodes C and D.

Note: If you are familiar with thresholding using the Tivoli NetView MLM configuration interface, you should note that APM differs from the MLM in that you cannot prepend an alias or node name to the beginning of the MIB object ID on which you are setting a threshold. Because you are setting thresholds against nodes in a SmartSet, there is no need for you to prepend a target to the front of the threshold variable. APM automatically creates the appropriate aliases and prepends them to the variable. APM also maintains those aliases so that they stay synchronized with the SmartSets and thresholds they represent.

Data about threshold and file monitor traps are recorded in files in the /usr/OV/databases/C5 directory for graphing by Problem Determination Assistance applications. See "Diagnosing Problems Using the Problem Determination Assistance Facility" on page 248 for more information.

### File Monitor Policy

Through a file monitor policy, you can set up file and log monitoring on any node that has the Systems Monitor SIA installed. When you define a file monitor condition through the APM configuration application and distribute it, SNMP set commands are done against the actual SIA running on the nodes defined in the SmartSet for which you are defining a file monitor condition. Thus, the community names on the Tivoli NetView host where you are running the APM daemon and on the remote SIA nodes must be defined so that the management station can do SNMP sets on the SIA nodes.

Files of any type can be monitored, and you can monitor for many different conditions or combinations of conditions. Each condition returns a different trap. The conditions that you can monitor and the traps returned are:

<table>
<thead>
<tr>
<th>Condition Being Monitored</th>
<th>Trap Returned</th>
</tr>
</thead>
<tbody>
<tr>
<td>Existence of a text string</td>
<td>Specific trap 21 - String found</td>
</tr>
<tr>
<td></td>
<td>Specific trap 22 - File data modified</td>
</tr>
<tr>
<td>Changes to characteristics of the file, such as owner, group, or permissions</td>
<td>Specific trap 23 - File status changed</td>
</tr>
<tr>
<td>Existence of a file</td>
<td>Specific trap 24 - File does not exist</td>
</tr>
<tr>
<td></td>
<td>Specific trap 25 - File exists</td>
</tr>
</tbody>
</table>
See "Matched, Armed, and Disarmed Command Environment Variables" on page 253 for information about environment variables that correspond to MIB variables in the traps.

Tivoli NetView has a predefined filter that displays only Tivoli NetView MLM traps in the Control Desk. You create a dynamic workspace and select the predefined filter named C5filters to display the Tivoli NetView MLM traps only. Step 9 on page 255 in "Example of Defining and Distributing a Policy" on page 253 shows the steps for opening this dynamic workspace.

**Command Policy**

Through a command policy, you can perform the following two types of command execution:

- Execute commands in the Korn shell environment. The results are placed in a MIB variable.
- Access kernel memory or a shared memory segment. The results are placed in a MIB variable.

You can execute any command and, optionally, return the results of the command in the form of a MIB variable. The results can be a display string, integer, counter, or gauge. An SNMP set or get to the variable results in the command being executed, and the output of the command is placed in the associated Result variable (depending on what type of output resulted from running the command).

A command policy also provides access to the kernel memory or a shared memory segment of a user application. Accessing the kernel or shared memory is much faster than executing a command and returning the result. With memory operations, the command table can set or get a value from a shared memory segment that an application has opened, or perform get operations from kernel memory. The results of a memory operation can be returned as a display string, integer, counter, or gauge.

Commands set up through a command policy can be run from either a set or get request. Set commands are done against the actual SIAs running on the nodes defined in the SmartSet for which you are defining a command policy.

**Filter Policy**

Through a filter policy, you can define the action to be taken by the local MLM when certain traps are received. When a trap is received, the MLM compares the incoming trap information with the information in the filter policy. If the trap matches a filter policy, the action defined in that filter policy is taken.

If the incoming trap does not match filter policy, a defined default action is taken. The default action is defined in the Trap Destination table, which you can customize by defining a trap destination policy through APM. To control the protocol and the port that is used for trap reception, you must use the Tivoli NetView MLM configuration application to update the Trap Reception table.

If you want to reference an alias with a filter policy, create and distribute the alias policy separately.
Trap Destination Policy

Through a trap destination policy, you can specify the hosts that will receive traps from MLMs. A trap destination policy is used by the MLM to determine the destination for traps that fit one of the following criteria:

- The trap does not match filter policy and the defined default action is set to sendTraps.
- The trap matches a filter policy and the matched filter policy specifies to forward the trap, but there is no trap destination specified (the Destination field is blank).

APM automatically sets one APMHOST entry for each MLM to forward its traps to Tivoli NetView running APM. The trap destination is predefined to be all Tivoli NetView managers that discover MLMs that are thresholding and forwarding traps.

Analysis Policy

Through an analysis policy, you can evaluate complex mathematical expressions on local or remote MIB variables for the MLM. You can use the analysis result as the input for another operation (for example, as input to a threshold policy). Using an analysis result as input to a threshold policy, permits the data collection and verification of the final expression result. The expression is evaluated using floating point arithmetic and converted to the type of the final result.

If you want to reference an alias with an analysis policy, create and distribute the alias policy separately.

Alias Policy

Through an alias policy, you can associate a group of nodes with a single alias name, or a group of alias names with a single alias name. You can assign a node to multiple alias groups. This grouping feature enables you to monitor groups of nodes by specifying a single alias.

An alias policy enables you to define threshold, analysis, and filter policies for groups of similar nodes by using an alias for the group.

You will not see aliases that APM created (used for thresholding) in the list of aliases. Nor will you see aliases that are created by Tivoli NetView for the netmon daemon.

Note:

These aliases should not be changed manually. Only aliases that are created using APM alias policies will be in the list of aliases.

Administration Policy

Through an administration policy, you can store MLM administrative data in a tabular form. For example, you can use this policy to track available configurations, maintenance performed on a node, or the latest levels of software.

Configuring and Starting the APM Daemon

As shipped, the APM function is not automatically configured to run. To use APM, configure and start the APM daemon for the first time through the Tivoli desktop.
Configuring the APM Daemon through the Server Setup Application

To configure the APM daemon, follow this procedure:

1. Enter `serversetup` on the command line to access the Server Setup application.
2. Select **Configure --> Set options for daemons --> Set options for Agent Policy Manager daemon...**
   
   The Set options for Agent Policy Manager daemon dialog is displayed. By default, logging and tracing is on.
3. Make the appropriate changes in the Set options for Agent Policy Manager daemon dialog. Click the **Help...** button on the dialog if you need more information.
   
   - Change the default log and trace files if desired.
   - Optionally, change the elapsed time between daemon attempts. This value determines how often APM will attempt to distribute threshold and file monitoring policies that it could not distribute previously (for example, if a node was down). By default, APM retries all failed distributions for all nodes once every 60 minutes. To turn off automatic retries, set this value to zero (0).
   - Optionally, change the number of threshold events stored in the history file. These events are used as plot points by the `xnmgraph` application. You can graph threshold history through the **Problem Determination Assistance** application.
     
     See ["Diagnosing Problems Using the Problem Determination Assistance Facility" on page 248](#) for more information.

4. Click **OK**.

Now that the APM daemon is configured and registered, it starts automatically each time you start Tivoli NetView.

Starting the APM Daemon from the Command Line

After the APM daemon is configured, you can start it from the command line. The daemon is called `C5d`. Root authority is required to start the daemon. To start the daemon, enter the following command on the command line:

```
/usr/OV/bin/ovstart C5d
```

You can specify the following options when you start the C5d daemon:

- **-t tracefile**
  
  Turns on tracing when the C5d daemon is started. Trace information is saved in the file specified.

- **-l logfile**
  
  Log file is saved in the file specified.

- **-g**
  
  Specifies the number of threshold events to be saved.

- **-r**
  
  Specifies the retry interval.

When the daemon is running, you can use the -T and -L options to toggle tracing and logging, respectively, without having to stop and restart the daemon.

**Note:** The log and trace files can continue to grow without bounds. Periodically, delete or prune the files to prevent running out of file system space. By default, the log and trace files are stored in the `/usr/OV/log` directory.

See the C5d man page for more information.
Starting the APM Configuration Application

You can start the APM configuration application in two ways:

- Select APM Configuration from the Tools pull-down menu.
- Double click APM on the Tools Window.

When you start APM, the Agent Policy Manager Configuration main menu is displayed as shown in Figure 41.

![Figure 41. Agent Policy Manager Configuration Main Menu](image)

From the Agent Policy Manager Configuration main menu, you can look at a list of current APM policies and their distribution status. ["Distribution Status States" on page 211] describes the various status states.

By default, File Monitor policies are displayed. To switch to another type of APM policy, click Policy Type. You can add, delete, or modify a policy and distribute the policy to other nodes in the network. Much like the Tivoli NetView MLM Configuration Application, the APM configuration application gives you fields for defining configuration information about policies. You specify SmartSets to which a policy will be distributed. You can also view the distribution status of a selected policy to the individual nodes in the SmartSet.

Note that the first time you start APM, it might take some time before it is displayed because it must set status on all nodes in the associated SmartSets Agent Policy Manager uses and create submaps for the domain SmartSets. After the initial synchronization is not delayed.

You can use the Administrator..Start Application..APMmap menu option to start the APM maps without starting the APM configuration application and without having to close and restart the Tivoli NetView graphical interface. You might find this useful if the APM configuration application ends or if you start the APM daemon after you have started the Tivoli NetView graphical interface. If you recycle the APM daemon, you will have to restart APM in this manner to update the maps. The same process that manages maps also displays the configuration application.
Distribution Status States

Table 17 describes the various policy status states.

**Table 17. Distribution Status States For APM Policies**

<table>
<thead>
<tr>
<th>Status</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>NeverDistributed</td>
<td>An attempt has never been made to distribute this policy.</td>
</tr>
<tr>
<td>Distributed</td>
<td>The policy was successfully distributed to all nodes in the SmartSet.</td>
</tr>
<tr>
<td>PartiallyDistributed</td>
<td>An attempt was made to distribute the policy, but one or more nodes could not be modified. The nodes might not have been reached. If the failure is due to a timeout, you might want to let APM continue to redistribute on its own.</td>
</tr>
<tr>
<td>PartiallyDeleted</td>
<td>An attempt was made to delete the policy, but one or more nodes could not be reached to delete its definition. The nodes might not have been reached.</td>
</tr>
<tr>
<td>PendingModifyDistribute</td>
<td>A modification was made to the policy, but it has not yet been distributed.</td>
</tr>
<tr>
<td>PendingDeleteDistribute</td>
<td>The user selected a policy in the APM Configuration main menu and clicked Delete. The delete request has not been distributed to the SmartSets yet.</td>
</tr>
<tr>
<td>DistributeInProgress</td>
<td>APM was doing a distribution when the C5d daemon went down or a logic error occurred. If you see this error after C5d recycles, allow the C5d daemon to continue to distribute. If you are seeing several items with this status, recycle the daemon and call support.</td>
</tr>
<tr>
<td>ModifyDistributeInProgress</td>
<td>APM was distributing a modified policy when the C5d daemon stopped or a logic error occurred. If you see this error after C5d recycles, allow the C5d daemon to continue to distribute. If several items have this status, recycle the daemon and call support.</td>
</tr>
<tr>
<td>DeleteDistributeInProgress</td>
<td>APM was distributing a deletion when the C5d daemon stopped or a logic error occurred. If you see this error after C5d recycles, allow the C5d daemon to continue to distribute. If several items have with this status, recycle the daemon and call support.</td>
</tr>
</tbody>
</table>

Creating, Changing, and Distributing an APM Policy

This section describes the following tasks which you can perform from the Agent Policy Manager Configuration main menu:

- Creating or Copying an APM Policy
- Viewing Policy Details on page 212
- Distributing a Policy to Remote Nodes on page 213
- Managing Policy Distribution Using the Node Distribution Status Dialog Box on page 214
- Modifying an Existing Policy on page 215
- Viewing Pending Policies on page 216
- Deleting an Existing APM Policy on page 216
- Undoing a Change on page 217

Creating or Copying an APM Policy

You can create a new APM policy or you can base a new APM policy on an existing policy. If you are creating a file monitor or threshold policy, it is best to use the default policies provided as the basis for your new policy. If you create a new file monitor or threshold policy, and you leave some fields blank on the Threshold or File Monitor dialog, you might get unexpected results, such as not receiving traps or the color of icons not changing.
To create a new APM policy, follow this procedure:

1. Select **Policy Type** on the Agent Policy Manager Configuration main menu. By default, File Monitor is selected as the type of policy that is to be created.
   
   A list of currently defined policies is displayed.

2. Do one of the following:
   
   - To create a new policy, click **Add/Copy** without selecting a policy.
   - To base a new policy on one that you have already defined, select the policy you want to copy, and click **Add/Copy**.

   If you are creating a file monitor or threshold policy, you should use the default policies that are provided to ensure that all fields are filled in. This ensures that traps will be returned correctly.

   The dialog box for defining the policy type you selected is displayed.

3. Fill in the fields in the dialog box.
   
   Note that the new policy you are creating must have a unique name across all APM policies. In other words, you cannot have two thresholds or two file monitor policies called Test, nor could you have one threshold policy called Test and one filter policy called Test. If you are creating a new file monitor or threshold policy, make sure you fill in the Specific Trap and Arm Enterprise fields in the Threshold Actions dialog box and the Rearm Specific Trap and Rearm Enterprise fields in the Rearm Actions dialog box if you have a rearm condition. Use the values from the default policies as follows:
   
   - In the Threshold Actions dialog box, specify 1 in the Specific Trap field and .1.3.6.1.4.1.2.6.12.5.1 in the Arm Enterprise field.
   - In the Rearm Actions dialog box, specify 2 in the Rearm Specific Trap field and .1.3.6.1.4.1.2.6.12.5.1 in the Rearm Enterprise field.

   See "Completing the APM Policy Dialog Boxes" on page 219 for additional information about dialog box fields. If you are new to Tivoli NetView MLM, refer to the **Tivoli NetView MLM User's Guide**.

4. Specify the SmartSets to which the policy is to be distributed clicking **Assign** in the dialog box.
   
   The SmartSet Assignments dialog box is displayed.

5. Select the SmartSets on the SmartSet Assignment dialog box, click **Assign**, and then click **OK**.
   
   Note that if you do not have the SmartSet defined before bringing up the APM configuration application, you can start the SmartSet Editor from the SmartSet Assignments dialog box.

   The SmartSet Assignments dialog box closes.

6. Click **Apply** in the policy dialog box.
   
   You will see a message indicating that the policy was saved and that you can distribute it from the Agent Policy Manager Configuration main menu.

   See "Distributing a Policy to Remote Nodes" on page 213 for more information.

### Viewing Policy Details

To view a policy, select the appropriate policy from the Agent Policy Manager Configuration main menu and click **View Current**. The policy dialog box, which you completed when you created or modified the policy, is displayed and contains the details of the selected policy.
Distributing a Policy to Remote Nodes

After you have successfully created a policy, you can distribute it to the nodes specified in the SmartSet assigned to the policy. You can distribute a policy from the APM main menu in two ways:

- **To all nodes** in the SmartSet. Use this method when you distribute a policy or change for the first time.
- **To individual nodes** in the SmartSet. Use this method if you have already distributed a policy or change and you want to manually retry the distribution to selected nodes.

For thresholds, the distribution process consists of SNMP set commands on the specified MIB object on the MLMs assigned to nodes in the SmartSets. Thus, you must have the community names on the node where you are running APM and the target MLMs set up so that the APM node is allowed to do SNMP sets on the target nodes.

Note that you do not have to redistribute a policy if nodes have been added or deleted in target SmartSets or if you have modified the SmartSet rule in some way. The information is automatically redistributed by APM. You also do not have to redistribute to a failed node. APM retries distribution on a regular interval, by default, every hour.

### Distributing a Policy to All Nodes

From the Agent Policy Manager Configuration main menu, select the policy you want to distribute by clicking it, and then Distribute. The dialog that is displayed lists the name of the policy and only those nodes to which the policy is to be distributed. For policies being distributed to MLMs, the target SmartSets resolve to a list of nodes. Click Start to distribute the policy. Messages about the success or failure of the distribution appear in the Messages area on the Distribute dialog box.

### Distributing a Policy to Individual Nodes

To distribute a policy to individual nodes, you must have distributed the policy to all nodes at least once. Then you can distribute the policy to selected failed nodes. To distribute a policy to individual nodes, follow this procedure:

1. From the Agent Policy Manager Configuration main menu, select the policy you want to distribute by clicking it, and then Node Status. The Node Distribution Status dialog box is displayed.
2. Select the nodes to which you want to distribute the policy, and then click Distribute. The dialog box that is displayed lists the selected target nodes.
3. Click Start to distribute the policy. Messages about the success or failure of the distribution appear in the Messages area on the Distribute dialog box. APM distributes the policy to only the selected nodes and does not automatically retry failed distributions to nodes that are not selected.

### Successful Distribution

If all nodes in the SmartSet successfully receive the policy, the policy status in the Agent Policy Manager Configuration main menu is changed to Distributed to indicate that the policy was distributed.
Distribution Failures

If any or all of the distributions fail, the policy status is changed to PartiallyDistributed on the Agent Policy Manager Configuration main menu. Failures can occur for several reasons:

- An incorrect community name (causes an authentication error). See "Configuring Community Names" on page 256 for more information.
- A network failure.
- An SNMP agent or application not responding.
- The target MIB is not loaded.
- The distribution attempt times out.
- The node is no longer present in Tivoli NetView and not yet deleted from the topology database.

If you receive a message that targets are not defined, it indicates one of two conditions:

- On a file monitor policy, the target SmartSet is empty.
- On a threshold policy, APM either could not find an MLM in the network whose domain contains any of the nodes in the SmartSet, or it could not find any MLMs.

You can continue to attempt to redistribute policies manually, or you can let APM continue to try to redistribute. APM continues to periodically retry the distribution in the following ways:

1. Periodically, depending on the interval set when the C5d daemon was started. This option can be set using the -r option. All failed distributions are tried at the same time. The default interval is to retry distribution once an hour. To stop retries, use the -r option with zero (0) as the retry interval (-r0).
2. When a policy is redistributed (either manually by a user or automatically when a SmartSet is changed)
3. When a previously unresponsive node responds to a set request for another policy. When the node responds, APM checks to see if there are any other outstanding distributions for that node and attempts to redistribute any it finds.

Managing Policy Distribution Using the Node Distribution Status Dialog Box

After you have distributed a policy, you can use the Node Distribution Status dialog box to perform the following tasks:

- View the distribution status to the individual nodes.
- Distribute the policy to failed nodes. See "Distributing a Policy to Individual Nodes" on page 213 for these steps.
- Delete an obsolete node from the distribution list.
- Reset all the APM policies to selected nodes.

To display the Node Distribution Status dialog box, select the appropriate policy on the Agent Policy Manager Configuration main menu and then click Node Status. The Node Status dialog box is displayed as shown in Figure 42 on page 215.
The Node Distribution Status dialog box lists the target nodes, the target MLM and the policy distribution status (whether the add or delete failed or was successful). If the selected policy is a threshold policy, the dialog box also lists the MLM that is doing the thresholding on behalf of the target node.

You can select one or more nodes and then click one of the following buttons:

- **Distribute**: Retry distribution of just this policy to the selected nodes. See "Distributing a Policy to Individual Nodes" on page 213 for more detailed information.

- **Delete**: Delete the selected obsolete nodes from the distribution list. An obsolete node is a node that has been previously distributed to but is no longer a member of any target SmartSet for the policy. When a node is removed from the target SmartSet, APM automatically tries to delete the policy from the node. If the delete operation fails, the node remains in Delete Failed status until the delete is successful, the node is deleted from the ovwdb database, or you manually delete the node from the distribution list using this function. If you try to delete a node that is still in a target SmartSet, an error message is displayed.

- **Resynch**: Reset all the APM policies for the selected nodes. The Resynch operation looks up all the valid APM policies for the selected nodes and redistributes all the APM policies.

**Modifying an Existing Policy**

To modify an existing policy, select the policy you want to modify on the Agent Policy Manager Configuration main menu and click **Modify**.

After a policy has been defined or distributed, you can modify the policy in two ways:

- **Change who** by adding or deleting target SmartSets from the policy.

  When you use the SmartSet Editor to change the SmartSets used by an APM policy, the change to the SmartSet is distributed to the nodes in the SmartSet automatically. You do not need to redistribute the policy to the nodes in the SmartSet to implement the changes to the SmartSet.
- Change **what** by modifying the fields in the policy dialog box. See "Completing the APM Policy Dialog Boxes" on page 213 for additional information about what to put in the dialog box fields.

After you modify a policy, you must **distribute** the modified policy to the target nodes. You can choose to **distribute** the modified policy, **undo** the changes to the policy, or **delete** the policy completely.

After you modify a policy, the policy’s status is PendingModifyDistribute until it is distributed. When you distribute the modified policy, the changes are analyzed. Only the changes to the existing policy are distributed to each node that is impacted by the policy change.

### Viewing Pending Policies

If you have distributed or partially distributed policies and then modify the policy, you can view the modified policy before redistributing it. The modified policy has a status of PendingModifyDistribute. Select the **View Pending** button to see the modified policy. The **View Current** button shows the currently distributed version of the policy.

### Deleting an Existing APM Policy

To delete an APM policy, select the policy you want to remove, and then click **Delete** on the APM Configuration main menu. The status of the policy is changed to PendingDeleteDistribute. If you change your mind, click **Undo** to return the policy to its previous status. To distribute the delete operation to the nodes in the SmartSet, click **Distribute**.

On the Distribute dialog that is displayed, click **Start** to delete the policy on all nodes. This action removes the policy from the nodes. Messages are displayed to indicate the success or failure of the operation.

If any or all of the nodes cannot be reached, the policy has a status of PartiallyDeleted. If the problem was a timeout, such as if an agent was not available, you can redistribute manually by clicking **Distribute** again until all nodes are deleted. However, APM continues to periodically redistribute the delete operation as well. The policy remains in the database until APM is able to delete the policy from all nodes in the SmartSet. APM automatically deletes the policy from the database when the last node responds to the delete operation.

If a policy is in the PartiallyDeleted state, you can force deletion of the policy from the database by selecting it and clicking **Delete** again; however, this action may leave the policy still running in some of the nodes to which it has been distributed successfully and not deleted successfully. It may then be necessary to manually delete the policy from the hosts affected using the Tivoli NetView MLM Configuration interface on those Tivoli NetView MLM hosts.

Forcing a deletion from the database should only be done in cases where the hosts or agents that have failed deletion requests are removed from the network and will not return to receive another deletion request.

To keep a policy in the database, but remove the policy from all nodes in the network, follow this procedure:

1. Select the policy on the Agent Policy Manager Configuration main menu.
2. Click **Modify**. The Modify Policy dialog box is displayed.
3. Select all the SmartSet names under SmartSet Assignments.
4. Click **Unassign** and click on **Apply**.
5. On the Agent Policy Manager Configuration main menu, the policy now has a status of PendingModifyDistribute. Select the policy and click on **Distribute**. The status of the policy changes to Distributed.

To delete individual nodes that have failed deletion from the distribution list, follow this procedure:
1. Select the policy on the Agent Policy Manager Configuration main menu.
2. Select **Node Status** The Node Distribution Status dialog box is displayed.
3. Select the nodes to which you no longer want to retry the deletion request, and then click **Delete**. The nodes are deleted from the distribution list.

**Undoing a Change**

If you modify or delete a policy and then change your mind before you distribute it, you can undo the change by clicking **Undo** on the Agent Policy Manager Configuration main menu.

**Fixing an APM Policy**

If you have created an APM policy that prevents you from starting the APM configuration application or is causing performance problems, you can fix the policy in two ways:

- Use the **C5Maint** command to fix the fields in the policy definition.
- Use the **APMSetStatus** command to change policy status.

**Using the C5Maint Command**

To fix a problem APM policy using the **C5Maint** command, follow this procedure:
1. Stop the C5d daemon using the following command:
   ```bash
   /usr/0V/bin/ovstop C5d
   ```
2. Enter the following command:
   ```bash
   /usr/0V/bin/C5Maint -d
   ```

   The /usr/0V/conf/C5.conf file is created with all your APM policy definitions in it.

   The C5.conf file has the following format:

   ```plaintext
   operation:objID:list_flag:field_type_indicator:field_length\field_name:field_value_lenth,field_value
   ```

   **Note:** Do not change the objID, list_flag, field_type_indicator, field_length, or field_name fields. You can delete entire lines to remove node lists or variable lists, and you can change the field_value field and the field_value_length field to correspond with the changed value.

   Here is an example of the C5.conf file:
3. Edit the C5.conf file and delete all the policies except the one you want to change. The policy definition starts with Selection Name.

4. Change the field values you think are incorrect. Be sure to change the field length to correspond with the new field value.

5. Load the file into the APM database using the following command:

   C5Maint -l

See the C5Maint man page for more information.

Using the APMSetStatus Command

To change the policy status, follow this procedure:

1. Stop the C5d daemon using the following command:
   `/usr/0V/bin/ovstop C5d`

2. Enter the following command:

   APMSetStatus policy_name newStatus

Replace policy_name with the name of the policy whose status you are changing. Replace newStatus with one of the following values:

**Distributed**

   Changing to this status causes the failed nodes list to be deleted from the policy.

**NeverDistributed**

   Changing to this status causes the failed nodes and distributed nodes lists to be deleted from the policy.

**PartiallyDistributed**

   Changing to this status does not affect the node lists.

**PartiallyDeleted**

   Changing to this status does not affect the node lists.
PendingDeleteDistribute
Changing to this status does not affect the node lists.

PendingModifyDistribute
Changing to this status requires that a modified version of the policy already exists in the database. You can confirm this by using the `ovobjprint mod_definitionName` command. Changing to this status causes failed and distributed nodes lists to be deleted from the policy.

Deleted
Changing to this status deletes the policy and any modified versions of the policy from the database, regardless of distribution status or existing policies in MLMs and SIAs. No distribution of the delete is done. Not changing the status to any status except PendingModifyDistribute causes any modified versions of the definition existing in the database to be deleted.

Note that the APMSetStatus command will not cause the executing environment to be updated in accordance with your policy changes. That is, all the agents and MLMs which are running versions of the definitions you may be changing will not be updated. This is a database update only. You must make sure that the MLMs and SIAs reflect the changes being made.

See the APMSetStatus man page for more information.

Copying an APM Policy from One System to Another

To copy an APM policy from one system to another, follow these steps:

1. Stop the C5d daemon using the following command:
   `/usr/OV/bin/ovstop C5d`
2. Enter the following command:
   `/usr/OV/bin/C5Maint -d`

   The `/usr/OV/conf/C5.conf` file is created with all your APM policy definitions in it. See "The C5.conf File" on page 217 for information about the format of the C5.conf file.
3. Edit the C5.conf file and delete all the system-specific information, such as failed and distributed node lists and invalid SmartSets. Change the distribution status to NeverDistributed. That is, change the field_value field corresponding to the dfStatus field to NeverDistributed and change the field_value_length field to 16.
4. Load the file into the APM database on the system to which you want to copy the policy using the following command:
   `C5Maint -l`

   See the C5Maint man page for more information.

Completing the APM Policy Dialog Boxes

This section explains how to fill in the fields on the APM policy dialog boxes.
If you are familiar with Tivoli NetView MLM, the dialogs will be familiar to you. They are very similar to the dialog boxes in the Tivoli NetView MLM Configuration Application, with the added ability of assigning SmartSets to the policies. APM policy names and SmartSet names must be compliant with Tivoli NetView MLM requirements.

**Threshold and Data Collection Policy Dialog Box**

To define a threshold policy, select **Threshold/Data Collection** using the Policy Type button on the Agent Policy Manager Configuration main menu.

The following dialog box is displayed as shown in Figure 43.

![Figure 43. Threshold and Data Collection Dialog Box](image-url)
The fields in this dialog box have the following meanings:

### Table 18. Threshold Policy Dialog Box Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Any unique character string.</td>
<td><strong>Purpose:</strong> Specifies the name of the particular threshold entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Description:</strong> The value in the Name field is used as a label and an instance ID, identifying each field for this threshold, and is appended to the corresponding object ID. The instance IDs for each field in the same entry are the ASCII values for each letter of the name. For example, the value <code>APP_TIME</code> causes the instance ID for this entry to be <code>65.80.80.95.84.73.77.69</code>, where each letter is an ASCII integer representation of <code>APP_TIME</code>.</td>
</tr>
<tr>
<td>State</td>
<td>Valid values are: disabled enabledThresholdOnly enabledStoreOnly enabledThresholdStore</td>
<td><strong>Purpose:</strong> Determines whether the threshold entry is active and the thresholding actions that are being performed for this entry.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Description:</strong> Their values and their meanings are described in the following list:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the value is <code>disabled</code>, the entry is valid, but the MLM will not check the value of the specified MIB variable against this threshold.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the value is <code>enabledThresholdOnly</code>, the MLM is checking the retrieved MIB variable values against this threshold, then discarding the data.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the value is <code>enabledStoreOnly</code>, the MLM is storing the retrieved MIB variable values in a file on the MLM host during the polling process; the values are not checked against the threshold.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the value is <code>enabledThresholdStore</code>, the MLM is checking the retrieved MIB variable values against this threshold and storing the values in a file on the MLM host.</td>
</tr>
<tr>
<td>Description</td>
<td>Any valid character string.</td>
<td><strong>Purpose:</strong> The general purpose of the threshold being performed.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Description:</strong> Completing this field is not mandatory, but it is recommended. Describe the purpose and function of the threshold.</td>
</tr>
<tr>
<td>Poll Time</td>
<td>Valid values include an integer followed by one of these letters: d = days h = hours m = minutes s = seconds</td>
<td><strong>Purpose:</strong> Indicates the time period that should elapse before the next threshold polling operation is invoked.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If the letter is omitted, the default is minutes. For example, the value <code>1h10m</code> means to start this threshold polling operation 1 hour and 10 minutes following the last polling operation.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Values</td>
<td>Purpose and Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| MIB Object      | Any MIB variable in numeric, dot-notation format, followed by an instance ID in numeric or nonnumeric format. For multiple instances, a wildcard is valid. | **Purpose:** Indicates the MIB variables on which thresholding is to be performed.  
Do not prepend an alias or node address to the beginning of a MIB object. Because you are assigning the threshold to a collection, it is not necessary to specify a remote node.  
**Description:** You can specify the MIB variable in the following ways:  
- Specify one unique instance to be checked  
- Specify all instances of a variable  
For example, to perform threshold checking for the specific instance interface 1 of the following MIB variable:  
mib-2.interfaces.ifTable.ifEntry.ifInErrors  
the value would be: .1.3.6.1.2.1.2.2.1.14.1.  
To perform threshold checking for all instances of the previous MIB variable, the value is .1.3.6.1.2.1.2.2.1.14.*.  
You can use the alphanumeric string to indicate the instance to be checked. For example, to perform threshold checking for the MIB variable sm6kSystemFileSystemPercentUsed for the /usr directory instance, the value is .1.3.6.1.4.1.2.6.12.2.5.2.1.4./usr.0.  
By default, thresholds for Counter type variables are computed by calculating the change per second in the sampled values and checking these delta values against the threshold or rearm value.  
For Gauge and Integer type variables, the actual variable value is checked against the threshold or rearm value. To override this default behavior, add the keyword **delta** or **value** in front of the condition, separating the keyword and condition by a space. |
| Threshold        | Valid condition values are:  
- `=`  
- `<`  
- `<=`  
- `>`  
- `>=`  
- `&`  
- `|`  
- `changes`  
- `doesNotChange`  
- `exists`  
- `doesNotExist`  
- `!<condition>` | **Purpose:** Indicates the condition used when checking retrieved MIB values against the threshold or rearm value.  
**Description:**  
- `=`, `<`, `<=`, `>`, and `>=` mean the retrieved value being checked against the threshold or rearm value must be either equal to, less than (or equal to), greater than (or equal to) the value, respectively.  
- Prepend `!` to a condition to indicate what the value must **not** be.  
- `&` and `I` cause the retrieved values and threshold or rearm value to be combined with a logical AND or OR statement, respectively. A non-zero (0) result meets the condition; the trap or command operation is performed.  
- `changes` and `doesNotChange` cause watchdog operations to be performed. If a retrieved value changes or does not change, respectively, between consecutive polls, the condition is met. Threshold or rearm values are not used.  
- `exists` and `doesNotExist` check if the MIB variable exists. To check for an instance of the Tivoli NetView subagent (trapgend daemon) within the MLM Process table, use `exists` and `.1.2.6.1.4.1.2.6.12.2.7.2.1.2.trapgend.*` as the variable.  
By default, thresholds for Counter type variables are computed by calculating the change per second in the sampled values and checking these delta values against the threshold or rearm value.  
For Gauge and Integer type variables, the actual variable value is checked against the threshold or rearm value. To override this default behavior, add the keyword **delta** or **value** in front of the condition, separating the keyword and condition by a space. |
### Table 18. Threshold Policy Dialog Box Fields (continued)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value and Rearm Value</td>
<td>Depending on the MIB variable, the value can be a number, a floating point number, or a character string.</td>
<td>Specifies the threshold or rearm value for a MIB variable against which retrieved values are checked. This value must match the type of MIB variable that is being checked. If the MIB variable is numeric, use a number and a numeric check is done. If the MIB variable is a display string, and both the retrieved variable and this value can be converted to a floating point number, a floating point check is done. Otherwise, a string check is done. For example, if the MIB variable that is checked is a Counter type, the value should be an unsigned integer.</td>
</tr>
<tr>
<td>Specific or ReArm Specific</td>
<td>Any valid SNMP enterprise-specific trap number.</td>
<td>This field is on the Threshold Actions and Rearm Actions dialog boxes. It specifies the SNMP enterprise-specific trap which is to be sent when the threshold or rearm condition is met. Use the value 1 for arm and the value 2 for rearm. If this value is 0 (zero), no trap is sent.</td>
</tr>
<tr>
<td>Enterprise or ReArm Enterprise</td>
<td>The valid SNMP enterprise ID, in dot notation, for the specified trap.</td>
<td>This field is on the Threshold Actions and Rearm Actions dialog boxes. It specifies the SNMP enterprise ID for the enterprise-specific trap which is to be sent when the threshold or rearm condition is met. Use the .1.3.6.1.4.1.2.6.12.5.1 for the enterprise ID. If this field is blank, the Threshold Table enterprise ID is used.</td>
</tr>
<tr>
<td>Trap Description or ReArm Trap Description</td>
<td>Any valid character string. The description can use a set of defined environment variables.</td>
<td>This field is on the Threshold Actions and Rearm Actions dialog boxes. It indicates the general reason for sending the trap. It is the first variable in the trap that is sent to the operator.</td>
</tr>
<tr>
<td>Command To Execute or ReArm Command To Execute</td>
<td>Any valid command can be specified. The command can use a set of defined environment variables.</td>
<td>This field is on the Threshold Actions and Rearm Actions dialog boxes. It specifies a command to be executed by the MLM when the threshold or rearm condition is met. The command is executed using the environment in which the MLM is running. If this field is blank, no command will be executed. See “Matched, Armed, and Disarmed Command Environment Variables” on page 259 for information about environment variables that can be used in scripts.</td>
</tr>
</tbody>
</table>

### File Monitor Policy Dialog Box

To define a file monitor policy click **File Monitor** using the Policy Type button on the Agent Policy Manager Configuration main menu. The following dialog box is displayed as shown in [Figure 44 on page 224](#).
The fields in this dialog box have the following meanings:

**Table 19. File Monitor Policy Dialog Box Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Any unique character string that is a valid Tivoli NetView MLM name. No spaces are permitted in the string.</td>
<td><strong>Purpose</strong>: Specifies the name to be used as a label and instance ID identifying each field for this command entry. The name is appended to the corresponding object ID. The instance IDs for each field in the same entry are the ASCII values for each letter of the name. For example, a name of RCMON causes the instance ID for this entry to be 82.67.77.79.78, which is an ASCII representation of the name RCMON.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Values</td>
<td>Purpose and Description</td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>---------------------------------------------</td>
<td>----------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>State</td>
<td>Valid values are: disabled enabled enabledFromBegin</td>
<td><strong>Purpose</strong>: Specifies the current state of the file monitor table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Description</strong>: The values mean:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the value is <strong>disabled</strong>, file monitoring is turned off.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the value is <strong>enabled</strong>, file monitoring is turned on. Actual monitoring starts</td>
</tr>
<tr>
<td></td>
<td></td>
<td>when the activation time you specified is reached. Tivoli NetView MLM begins searching</td>
</tr>
<tr>
<td></td>
<td></td>
<td>at the end of the file, unless it determines that the file has wrapped or has</td>
</tr>
<tr>
<td></td>
<td></td>
<td>decreased in size; if so, Tivoli NetView MLM starts searching at the top of the file.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>In this way, Tivoli NetView MLM uses only the newest file information.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>When the string you specified is found in the file, a trap is issued, and any action</td>
</tr>
<tr>
<td></td>
<td></td>
<td>you specified in the Command to Execute field is run.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the value is <strong>enabledFromBegin</strong>, file monitoring is active. Monitoring starts</td>
</tr>
<tr>
<td></td>
<td></td>
<td>at the top of the file. You might specify this value if the file to be monitored is</td>
</tr>
<tr>
<td></td>
<td></td>
<td>actually being generated each polling interval by a command that you specified to be</td>
</tr>
<tr>
<td></td>
<td></td>
<td>run before each search, or if you always want the file to be searched from top to</td>
</tr>
<tr>
<td></td>
<td></td>
<td>bottom each polling interval. Note that this state requires more CPU processing than</td>
</tr>
<tr>
<td></td>
<td></td>
<td>the <strong>enabled</strong> state.</td>
</tr>
<tr>
<td>Description</td>
<td>Any valid character string.</td>
<td><strong>Purpose</strong>: States the general purpose of the file being monitored.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Description</strong>: Describes the purpose of the monitored file and explains the action</td>
</tr>
<tr>
<td></td>
<td></td>
<td>that is taken when a search string or some other change to the file is detected.</td>
</tr>
<tr>
<td>Poll Time</td>
<td>Valid values include an integer followed by</td>
<td><strong>Purpose</strong>: Specifies the poll interval.</td>
</tr>
<tr>
<td></td>
<td>one of these letters: d = days h = hours m =</td>
<td><strong>Description</strong>: This variable specifies how often SIA should check the monitored file</td>
</tr>
<tr>
<td></td>
<td>minutes s = seconds</td>
<td>for the specified string.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If the letter is omitted, the default is minutes. Multiple units can be specified in</td>
</tr>
<tr>
<td></td>
<td></td>
<td>this field. For example, you can specify a polling interval of 1h45m (1 hour and 45</td>
</tr>
<tr>
<td></td>
<td></td>
<td>minutes). If you do not specify a polling interval, the default is 10 seconds.</td>
</tr>
<tr>
<td>File to Monitor</td>
<td>Any valid character string.</td>
<td><strong>Purpose</strong>: Specifies the path name where the monitored file resides.</td>
</tr>
<tr>
<td>(Full Path Name)</td>
<td></td>
<td><strong>Description</strong>: Specify the name of the file to be monitored, including the fully</td>
</tr>
<tr>
<td></td>
<td></td>
<td>qualified path where the file resides. The file must be stored on the same node as the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SIA, and a <strong>root</strong> user must have read permission for the file. Monitoring of the file</td>
</tr>
<tr>
<td></td>
<td></td>
<td>differs, depending on whether or not the file exists.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the file already exists, monitoring begins with the last byte position in the file</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(the most recent records).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the file does not exist, when you enable file monitoring, Tivoli NetView MLM waits</td>
</tr>
<tr>
<td></td>
<td></td>
<td>until the file is created and begins searching for the string at the beginning of the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>file.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Values</td>
<td>Purpose and Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Monitor Type</strong></td>
<td>Values are:</td>
<td><strong>Purpose</strong>: Specify the type of monitoring to be performed.</td>
</tr>
<tr>
<td></td>
<td>• string</td>
<td><strong>Description</strong>: This field specifies the type of monitoring to be performed on the specified string or condition.</td>
</tr>
<tr>
<td></td>
<td>• dataChange</td>
<td>• If the value is <strong>string</strong>, the SIA watches the specified file for the string specified in the String to Monitor field. When the string appears, the specified actions are done.</td>
</tr>
<tr>
<td></td>
<td>• statusChange</td>
<td>• If the value is <strong>dataChange</strong>, the SIA watches the file for any change to the contents of the file, such as added or deleted characters. If any data change occurs, the specified actions are done.</td>
</tr>
<tr>
<td></td>
<td>• strDataStatus</td>
<td>• If the value is <strong>statusChange</strong>, the SIA watches the file for any change to the status of the file, including the file owner, file group, and file permissions. The file mode, user ID, and group ID that will be used for comparison are shown in the File Mode, User ID, and Group ID fields on the File Monitor window. The actual file mode, user ID, and group ID can be changed by modifying these fields in the File Monitor window. If the file mode, file owner, or file group changes, the specified actions are done.</td>
</tr>
<tr>
<td></td>
<td>• notExist</td>
<td>• If the value is <strong>strDataStatus</strong>, the SIA watches for any of the previous three types of changes (data changes, the appearance of a string, or file status changes). If any of these changes occur, the specified actions are done.</td>
</tr>
<tr>
<td></td>
<td>• exist</td>
<td>• If the value is <strong>notExist</strong>, the SIA does the specified actions if the monitored file disappears (such as if it is erased).</td>
</tr>
<tr>
<td></td>
<td>• all</td>
<td>• If the value is <strong>exist</strong>, the SIA does the specified actions if the monitored file appears.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the value is <strong>all</strong>, the SIA performs all types of monitoring (basically any change at all to the file).</td>
</tr>
<tr>
<td><strong>Traps</strong></td>
<td>Valid values are:</td>
<td><strong>Purpose</strong>: Specifies the types of traps that you want to have forwarded by the file monitoring function.</td>
</tr>
<tr>
<td></td>
<td>• send</td>
<td><strong>Description</strong>: The values mean:</td>
</tr>
<tr>
<td></td>
<td>• noSend</td>
<td>• If the value is <strong>send</strong>, the file monitoring function sends all types of traps. The trap destination is specified in the snmp.conf file.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the value is <strong>noSend</strong>, traps are not sent.</td>
</tr>
<tr>
<td><strong>String to</strong></td>
<td>Any valid character string.</td>
<td><strong>Purpose</strong>: Specifies the string that will be monitored.</td>
</tr>
<tr>
<td><strong>Monitor</strong></td>
<td></td>
<td><strong>Description</strong>: Specifies the file string that SIA should search for in the file. The string or pattern can be any limited regular expression (RE) in the style of the UNIX ed or egrep commands.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Note that if you use anchor symbols ( ` or $) in the expression, the string is only found if the line containing the anchor symbols is terminated with a new line (\n) character.</td>
</tr>
<tr>
<td><strong>Case Sensitive</strong></td>
<td>Valid values are:</td>
<td><strong>Purpose</strong>: Specifies whether the search is case sensitive.</td>
</tr>
<tr>
<td></td>
<td>• case – Perform a case-sensitive search</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• ignoreCase – Ignore case on search</td>
<td></td>
</tr>
</tbody>
</table>
Table 19. File Monitor Policy Dialog Box Fields (continued)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activation Schedule</td>
<td>Any valid character string</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Purpose</strong>: specifies the time and days of the week when the file monitor condition is to be activated and deactivated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Activation: Use this field to delay the onset of file monitoring. Specify time in the format HH:MM, where HH is an integer in the range 0–23 and MM is an integer in the range 0–59. If this field is not set, the time is set to 00:00.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Deactivation Time: Use this field to set the time when file monitoring will be deactivated. Specify time in the format HH:MM, where HH is an integer in the range 0–23 and MM is an integer in the range 0–59. If this field is not set, the time is set to 00:00.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Select the days of the week when you want file monitoring to be active.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Select the days of the week when you want file monitoring to be deactivated.</td>
</tr>
<tr>
<td>Automated Actions</td>
<td>Any valid character string</td>
<td><strong>Purpose</strong>: The command to be run before monitoring begins, and the command to run if the string is found.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Description</strong>: Click this button to set up commands to be run before file monitoring and during monitoring when a condition is found.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Command to Execute Before Monitor: Indicates the command to be run before searches for the specified string or other change to the file are performed. This MIB variable can be used to generate, modify, or translate the file prior to the search.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Command to Execute if Monitor Type Condition is Met: Indicates the command to be run automatically when a certain string of text or some other kind of change occurs in the monitored file. For example, you could run a command to expand the size of a file system if you are monitoring for a message that the file system is full. Indicates the command to be run before searches for the specified string or other change to the file are performed.</td>
</tr>
<tr>
<td>SmartSet Assignments</td>
<td></td>
<td>Use this field to assign SmartSets to this policy. Click <strong>Assign</strong> to open a dialog box for selecting SmartSets.</td>
</tr>
</tbody>
</table>

**Command Policy Dialog Box**

To define a command policy, click **Command** using the Policy Type button on the Agent Policy Manager **Configuration** main menu. The following dialog box is displayed as shown in **Figure 45 on page 228**.
Figure 45. Command Policy Dialog Box

The fields in this dialog box have the following meanings:
### Table 20. Command Policy Dialog Box Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Any unique ASCII character string that is a valid Tivoli NetView MLM name. Spaces are not valid in the character string.</td>
<td><strong>Purpose:</strong> Specifies the name of a particular command in the table. <strong>Description:</strong> The value in the Name field is used as a label and an instance ID, identifying each field for this command, and is appended to the corresponding object ID. The instance ID for each field in the same entry is the ASCII values for each letter of the name. For example, the value APP_TIME causes the instance ID for this entry to be 65.80.80.95.84.73.77.69, where each letter is an ASCII integer representation of APP_TIME. Use short names if possible.</td>
</tr>
</tbody>
</table>
| State      | Valid values are: enabled, disabled, invalid | **Purpose:** Identifies the current availability of a command. **Description:** The values mean:  
- If the value is enabled, the command is available for execution.  
- If the value is disabled, the command is valid, but it is not currently available for execution.  
  The default value is disabled.  
  If you click Delete on the APM Configuration main menu, the State field is changed to invalid, which causes the selected entry to be deleted. |
| Description| Any valid text string. | **Purpose:** States the general purpose of the command. **Description:** The description character string can be as detailed as necessary. This field is for informational purposes. This field is not mandatory, but is recommended. |
| Get Command| | **Purpose:** Specifies the command that is to be executed in the Korn shell, the kernel memory of the operating system, or the shared memory segment of a user application for an SNMP get request. **Description:** An SNMP get request must be issued to the defined Output Result Index (set in the Result Type field). This index is used for conversion to the output types display string, integer, counter, and gauge. The results are returned to the MIB variable that is specified by the defined Output Result Index (set in the Result Type field) and propagated to other result types. Therefore, if the index is set to displaystring, an SNMP get request to the DisplaystringResult causes the get command expression to execute and return a displaystring. Commands use preset environment variables in the command string.  
  To use shared or kernel memory, specify the keyword SHARED_MEMORY: or KERNEL_MEMORY: as the first 14 characters of the command.  
  For more information about using the Get Command field, refer to the Tivoli NetView MLM User's Guide. |
### Table 20. Command Policy Dialog Box Fields (continued)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Command</td>
<td></td>
<td><strong>Purpose:</strong> Specifies the command to be executed on an SNMP set request or a set request to a shared memory segment.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Description:</strong> This is a way to specify a value for a command that is to be run in the Korn shell. The value of the set is placed in the environment variable <code>SM6K_COMMAND_SET_VALUE</code>, which can be used on the command.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>An SNMP set request must be issued to the defined Output Result Index (set in the Result Type field). This index is used for conversion to the output types displaystring, integer, counter, and gauge. Therefore, if the index is set to <code>displaystring</code>, an SNMP set request to the DisplaystringResult (the Result field) causes the Set command expression to execute using the <code>SM6K_COMMAND_SET_VALUE</code>. The output of the command is placed in the appropriate MIB variable defined by the Output Result Index (set in the Result Type field). Commands can use preset environment variables in the command string. If the command execution results in a nonzero return code, the set request fails and a <code>BAD VALUE</code> message is returned.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To use a set request in shared memory, specify the keyword <code>SHARED_MEMORY:</code> as the first 14 characters of the command. This notifies the SIA that a set shared memory operation is requested.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For more information about using the Set Command field, refer to the <em>Tivoli NetView MLM User’s Guide</em>.</td>
</tr>
<tr>
<td>Time Out</td>
<td>Any valid integer, from 0 to 4294967295, representing a time period.</td>
<td><strong>Purpose:</strong> The length of time, in seconds, for the get or set command, or memory operation to execute.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Description:</strong> The time out value is critical. It allows the SIA to run the command, or it allows the command and memory operation to be cancelled if the command does not complete. The default value is 3 seconds.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If you specify the value 0, the command or memory operation is executed with no time out; no output or result is returned.</td>
</tr>
<tr>
<td>Time To Live</td>
<td>Any valid integer, from 0 to 4294967295, representing a time period.</td>
<td><strong>Purpose:</strong> Specifies the length of time, in seconds, before the command is executed again in response to an SNMP query.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Description:</strong> Shows the amount of time in seconds that must pass before another command is executed in response to an SNMP get request. If another get request is issued and the time to live interval has not passed, the command specified on the get request is not executed. Use this variable to allow time for multiple get requests on the output from a previous command.</td>
</tr>
</tbody>
</table>
### Table 20. Command Policy Dialog Box Fields (continued)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Result Type</strong></td>
<td>Valid values are: displaystring</td>
<td><strong>Purpose:</strong> Indicates the type of output request, or Output Result Index, to be used for operation results.</td>
</tr>
<tr>
<td></td>
<td>integer</td>
<td><strong>Description:</strong> To execute the defined SNMP get or set command expression, you must issue the get or set request to an Output Result Index.</td>
</tr>
<tr>
<td></td>
<td>counter</td>
<td>Based on the Result Type value, the results for the command are saved to a predefined corresponding MIB variable. If the result is an integer, it may be propagated to the other predefined MIB variables. For example, if the Result Type value is integer, when the command is run, the command output is converted to an integer, saved in the IntegerResult MIB variable, and propagated to all other variables based on IntegerResult.</td>
</tr>
<tr>
<td></td>
<td>gauge</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Row Index</strong> and Column Index</th>
<th>Any valid integer in the range of 0 to 4294967295.</th>
<th><strong>Purpose:</strong> Allows command output parsing for individual rows and columns.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td><strong>Description:</strong> Parsing for rows is based on new lines; parsing for columns is based on white spaces. Use these values together to control the output that is returned. A Row Index value of 0 (zero) returns all rows. A Column Index value of 0 (zero) returns all columns. Used together, the following results are possible:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If both the Row and Column Index values are 0 (zero), all output is returned.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the Row Index value is not 0 and the Column Index value is 0 (zero), the specified row is returned.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If the Row Index value is 0 (zero) and the Column Index value is not 0, the specified column is returned.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• If neither the Row Index value nor Column Index value are 0 (zero), only the row and column intersection field is returned.</td>
</tr>
</tbody>
</table>

For example, if the Row Index value is 0 and the Column Index value is 0, the **ps -ef** command could return the following results:
```
cag 19765 1 0 2:55:25 - 0:00 xterm
ja 29752 43830 0 1:24:51 pts/25 0:00 bm/ksh
```
In this example, if the Row Index value was 2 and the Column Index was 2, the output from running the **ps -ef** command would be 29752.

---

**Filter Policy Dialog Box**

To define a filter policy, select **Filter** using the Policy Type button on the Agent Policy Manager Configuration main menu. The following dialog box is displayed as shown in Figure 46 on page 232.
The fields in this dialog box have the following meanings:

Figure 46. Filter Policy Dialog Box
Table 21. Filter Policy Dialog Box Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Any unique character string.</td>
<td><strong>Purpose:</strong> Specifies the name of the particular filter rule.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Description:</strong> A value in the Name field is used as a label and an instance ID, identifying each filter rule field, and is appended to the corresponding object ID. The instance IDs for the fields in the same filter rule are the ASCII values for each letter of the name. For example, for the filter rule name Test, the instance ID is 84.101.115.116.</td>
</tr>
<tr>
<td>State</td>
<td>Valid values are:</td>
<td><strong>Purpose:</strong> Specifies the current availability of the filter rule.</td>
</tr>
<tr>
<td></td>
<td>Disabled</td>
<td><strong>Description:</strong> • Disabled means that the filter rule is dormant and is not used when traps are received.</td>
</tr>
<tr>
<td></td>
<td>Enabled</td>
<td>• Enabled means that the filter rule may be used in the filter evaluation process when a trap is received. Whether or not the filter rule will actually be used at a specific time is determined by the values in the Activation/Deactivation fields. See the Activation Schedule field to determine whether a rule is currently being used. When you click <strong>Delete</strong> the Agent Policy Manager Configuration main menu, the State field is changed to invalid, which causes the selected entry to be deleted.</td>
</tr>
<tr>
<td></td>
<td>Invalid</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Any valid character string.</td>
<td><strong>Purpose:</strong> Specifies the general purpose of the filter rule.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Description:</strong> This field is not mandatory, but is recommended. This field should describe the purpose and actions of the filter.</td>
</tr>
<tr>
<td>Enterprise</td>
<td>Valid values are:</td>
<td><strong>Purpose:</strong> The enterprise ID is compared to the enterprise ID in the received trap. The IDs must match.</td>
</tr>
<tr>
<td></td>
<td>• Enterprise ID in dot notation (fully qualified)</td>
<td><strong>Description:</strong> You can specify a single fully-qualified enterprise ID in this field, or you can indicate multiple enterprises in one expression by using a wildcard to specify a partial enterprise ID. The asterisk must be the last character in the string. You can also indicate an alias name, defined in the Alias Table, that represents an enterprise ID.</td>
</tr>
<tr>
<td></td>
<td>• Enterprise ID in dot notation (with wildcard value)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Alias name (from the Alias Table)</td>
<td></td>
</tr>
<tr>
<td>Field Name</td>
<td>Values</td>
<td>Purpose and Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------------------------------------------------------</td>
<td>-------------------------</td>
</tr>
<tr>
<td>Agent Address</td>
<td>Valid values are:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Alias name (from the Alias Table)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Host name</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• IP address expression</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Purpose:</strong> Specifies the set of agents, or nodes, that may have</td>
<td></td>
</tr>
<tr>
<td></td>
<td>generated the received trap.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Description:</strong> The Agent Address expression can be a single address</td>
<td></td>
</tr>
<tr>
<td></td>
<td>value, or a list of multiple address values separated by commas.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>An IP address expression should be in the format of 4 dot-separated</td>
<td></td>
</tr>
<tr>
<td></td>
<td>address subexpressions. Each subexpression may be one of the</td>
<td></td>
</tr>
<tr>
<td></td>
<td>following:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• A decimal number in the range 0–255.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• An inclusive range of 0–255 separated by a dash (for example, 0–127 or [0–127]).</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• An asterisk.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Each address subexpression represents 1-byte of the agent address</td>
<td></td>
</tr>
<tr>
<td></td>
<td>for which the range of values 0–255. An asterisk represents any byte</td>
<td></td>
</tr>
<tr>
<td></td>
<td>in the range of 0–255, which can be interpreted as [0–255].</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For example, a filter rule with the IP address expression 9.[67-69].5.* matches the received-trap agent address 9.68.5.181.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you leave this field blank for a filter rule, all agent addresses in</td>
<td></td>
</tr>
<tr>
<td></td>
<td>received traps will be considered a match.</td>
<td></td>
</tr>
<tr>
<td>Generic Expression</td>
<td>Valid values are:</td>
<td><strong>Purpose:</strong> Specifies the generic trap numbers that will be considered a match for this filter rule.</td>
</tr>
<tr>
<td></td>
<td>0 (Cold Start)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1 (Warm Start)</td>
<td><strong>Description:</strong> You can enter a single or multiple generic trap numbers separated by commas. This field does not support pattern matching characters (wildcards) or number ranges.</td>
</tr>
<tr>
<td></td>
<td>2 (Link Down)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3 (Link Up)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4 (Authentication Failure)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5 (EGP Neighbor Loss)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6 (Enterprise Specific)</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Description:</strong> The Agent Address expression can be a single address</td>
<td></td>
</tr>
<tr>
<td></td>
<td>value, or a list of multiple address values separated by commas.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>An IP address expression should be in the format of 4 dot-separated</td>
<td></td>
</tr>
<tr>
<td></td>
<td>address subexpressions. Each subexpression may be one of the</td>
<td></td>
</tr>
<tr>
<td></td>
<td>following:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• A decimal number in the range 0–255.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• An inclusive range of 0–255 separated by a dash (for example, 0–127 or [0–127]).</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• An asterisk.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Each address subexpression represents 1-byte of the agent address</td>
<td></td>
</tr>
<tr>
<td></td>
<td>for which the range of values 0–255. An asterisk represents any byte</td>
<td></td>
</tr>
<tr>
<td></td>
<td>in the range of 0–255, which can be interpreted as [0–255].</td>
<td></td>
</tr>
<tr>
<td></td>
<td>For example, a filter rule with the IP address expression 9.[67-69].5.* matches the received-trap agent address 9.68.5.181.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you leave this field blank for a filter rule, all agent addresses in</td>
<td></td>
</tr>
<tr>
<td></td>
<td>received traps will be considered a match.</td>
<td></td>
</tr>
<tr>
<td>Field Name</td>
<td>Values</td>
<td>Purpose and Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------------------------</td>
<td>----------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Specific Expression</td>
<td>Any valid enterprise-specific trap number.</td>
<td><strong>Purpose</strong>: Specifies the enterprise-specific trap numbers that will be considered a match for this filter rule.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Description</strong>: You can enter a single specific trap number or multiple specific trap numbers separated by commas. You can specify a range of specific trap numbers separated by a dash, for example, 2–5. Also, you can associate an enterprise ID with a group or range of enterprise-specific trap numbers. This association enables you to pair enterprise-specific traps from one vendor with those of another vendor in the same filter rule.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To associate an enterprise ID with a group of trap numbers, enclose the trap numbers in parentheses and brackets, as shown in the following example:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>.1.3.6.1.4.1.2.6.12( [21-23]), .1.3.6.1.4.1.2.6.12( 21, 23), 25</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If there is an enterprise ID associated with the specific-trap number, the number is matched with that enterprise ID. If there is no enterprise ID associated with the specific-trap number, the number is matched with the default enterprise ID.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>This field is checked only if the number 6 is entered in the Generic Expression field. If this field is left blank and the number 6 is specified in the Generic Expression field, all enterprise-specific trap numbers contained in received traps are considered matches.</td>
</tr>
<tr>
<td>Variable Expression</td>
<td>Any valid MIB expression</td>
<td><strong>Purpose</strong>: Specifies the MIB variable expressions in the traps that will be considered a match for this filter rule.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Description</strong>: MIB variables are shown in an expression that is set off with relational operators (&lt;, &gt;, &lt;=, &gt;=, ==, !=) and logical operators (&amp;&amp;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The format of the expression is &quot;RE:expression, where expression is a regular expression in the style of an ed or egrep command. The MIB variable must be prepended with a dollar sign ($). Valid MIB variables include all environment variables for the trap. Environment variables are shown in Table 27 on page 260.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Here is an example of a valid MIB variable expression: ( $SM6K_TRAP_ENTERPRISE == .1.3.6.1.4.1.2.6.12) &amp;&amp; ( $SM6K_TRAP_SPECIFIC_NUM &lt;= 10 )</td>
</tr>
<tr>
<td>Field Name</td>
<td>Values</td>
<td>Purpose and Description</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Action</td>
<td>Valid values are: sendTraps</td>
<td><strong>Purpose</strong>: Determines whether traps matching the filter rule are forwarded to the Tivoli NetView hosts that are specified in the Trap Destination field. If the Trap Destination field for this filter rule is blank, the trap destination is obtained from the Trap Destination Table.</td>
</tr>
</tbody>
</table>
|             | blockTraps throttleTraps       | **Description**:  
|             |                                | • *SendTraps* indicates that matching traps will be forwarded to the specified top-level managers.  
|             |                                | • *BlockTraps* indicates that matching traps are not forwarded. However, the MLM will forward the trap if it matches another filter rule that specifies *sendTraps*.  
|             |                                | • *ThrottleTraps* indicates that the MLM may or may not send the matching trap, depending on the configured throttle criteria. This criteria is specified in the Throttle Settings dialog box. |
| Throttle Type| Valid values are: sendFirstN    | **Purpose**: This field is on the Throttle Settings dialog box. It specifies whether the throttle function will forward the first of \( N \) matching traps to the top-level manager or the traps received after \( N \).                                           |
|             | sendAfterN                     | **Description**:  
|             |                                | • *SendFirstN* means to forward the first \( N \) traps and block those traps received after \( N \).  
|             |                                | • *SendAfterN* means to block the first \( N \) traps and forward those traps received after \( N \).  
<p>|             |                                | The value for ( N ) is defined in the Arm Count field.                                                                                                      |
| Arm Count   | Any integer ( \geq 1 )       | <strong>Purpose</strong>: This field is on the Throttle Settings dialog box. It indicates the number of matching traps to receive before the throttle is armed.                                                                        |
|             |                                | <strong>Description</strong>: Based on the selected Throttle Type, an armed throttle either allows the first ( N ) matching traps to be forwarded and blocks subsequent matching traps (sendFirstN), or blocks the first ( N ) matching traps and forwards subsequent matching traps (sendAfterN). |
| Armed       | Any valid command can be      | <strong>Purpose</strong>: This field is on the Throttle Settings dialog box. It specifies a command to be executed by the MLM when the number of received matching traps matches the value for the Arm Count, and the throttle is armed. |
| Command     | specified. The command invokes a set of defined environment variables. | <strong>Description</strong>: The command is executed using the environment in which the MLM is running. For a descriptive list of valid environment variables that can be used by the command, see <em>Matched, Armed, and Disarmed Command Environment Variables</em> on page 259. |</p>
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose and Description</th>
</tr>
</thead>
</table>
| **Disarm Timer** | Valid values include an integer followed by one of these letters: d = days h = hours m = minutes s = seconds | **Purpose**: This field is on the Throttle Settings dialog box. It specifies the amount of time to pass, starting with the receipt of the first matching trap, before resetting or disarming the throttle. If the throttle is disarmed at the end of the time period, the **disarmed** command is executed.  
**Description**: If the letter is omitted, the default is minutes. For example, the value 1h10m means to start this threshold polling operation 1 hour and 10 minutes following the last polling operation.  
The Disarm Timer and the Disarm Count methods for disarming a throttle can be used together.  
With a Disarm Count of greater than 0 (zero), it is possible for the throttle to be disarmed (by reaching the Disarm Count value) before the Disarm Timer period has elapsed.  
Using the Disarm Timer method, the timer is checked when a trap matching the filter rule is received. If the amount of time specified has elapsed since the throttle was started, the throttle is disarmed. If the amount of time specified has not elapsed since the throttle was started, the throttle action continues as defined.  
The timer is started when the throttle is started (the first matching trap is received), not when the throttle is armed (the Arm Count criteria is met and the **armed** command is executed). This means that it is possible for the throttle not to be armed before the time period has elapsed. Consequently, the time period elapses, but the throttle is only reset, not disarmed. Therefore, the **disarmed** command is not executed. |
| **Disarm Count** | Any valid integer value. | **Purpose**: This field is on the Throttle Settings dialog box. It indicates the number of matching traps that must be received to cause the throttle to be reset, or disarmed.  
**Description**: This is one method of disarming a throttle. Using this method, a count of matching traps is incremented when a trap matching the filter rule is received. When the number equals the defined Disarm Count, the throttle is disarmed. The counter starts over the next time the throttle is armed.  
This value is a delta, referring to the number of matching traps received after the throttle has been armed. This value is ignored if set to 0 (zero).  
Another method is to use the Disarm Count with the Disarm Timer. With a value set for Disarm Timer and a Disarm Count greater than 0 (zero), it is possible for the throttle to be disarmed (by the Disarm Timer period elapsing) before the Disarm Count value is reached. |
| **Disarmed Command** | Any valid command can be specified. The command can invoke a set of defined environment variables. | **Purpose**: This field is on the Throttle Settings dialog box. It specifies a command to be executed by the MLM when the currently armed throttle is disarmed. The throttle can be disarmed when the number of received matching traps matches the Disarm Count value, or when the specified Disarm Timer period has elapsed.  
**Description**: The command is executed using the environment in which the MLM is running. For a descriptive list of valid environment variables that can be used by the command, see [Matched, Armed] and Disarmed Command Environment Variables" on page 253. |
### Table 21. Filter Policy Dialog Box Fields (continued)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activation Schedule</td>
<td>Any valid character string for activation time</td>
<td><strong>Activation:</strong> Use this field to delay the onset of filtering. Specify time in the format HH:MM, the hour (HH) can be an integer in the range of 0–23. The minute (MM) can be an integer 0–59. If this field is not set, the time is set to 00:00. Select the days of the week when you want filtering to be active.</td>
</tr>
</tbody>
</table>
| Trap Destinations           | Host+service or host+protocol name/port, where host can be one of the following:  
- IP addresses  
- Alias names (from the Alias Table)  
- Host names | **Purpose:** This field is on the Automated Actions dialog box. It specifies the nodes to which a matching trap will be sent.  
**Description:** The values in this field will be used instead of the entries in the Trap Destination Table. If this field is left blank, the addresses defined in the Trap Destination Table are used. If duplicate addresses are specified for the destination of a trap, only one instance of the trap is sent to the address.  
You can use one of the following formats to specify the port number and protocol for trap delivery (UDP or TCP):  
- Host+port number/protocol name  
- Host+protocol name/port number  
- Host+service  
  The service must be the name of a service listed in the /etc/services file that corresponds to a port number/protocol name.  
If you do not define the protocol name or port number, the default value is used. The default protocol name is udp: the default port number is 162. |
| Matched Command             | Any valid command can be specified. The command can use a set of defined environment variables. | **Purpose:** This field is on the Automated Actions dialog box. It specifies a command to be executed by the MLM when a received trap matches the filter rule.  
**Description:** The command is executed using the environment variables in which the MLM is running. For a descriptive list of valid environment variables, see “Matched, Armed, and Disarmed Command Environment Variables” on page 259. |

### Trap Destination Policy Dialog Box

To define a trap destination policy, click **Trap Destination** using the Policy Type button on the Agent Policy Manager Configuration main menu. The following dialog box is displayed as shown in Figure 47 on page 239.
The fields in this dialog box have the following meanings:

**Table 22. Trap Destination Policy Dialog Box Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Any unique character string.</td>
<td><strong>Purpose</strong>: Specifies the name of the particular trap destination entry. <strong>Description</strong>: The value in the Name field is used as a label and an instance ID, identifying each field for this entry, and is appended to the corresponding object ID. The instance ID for the fields in the same entry are the ASCII values for each letter of the name. For example, for the entry name Test, the instance ID is 84.101.115.116.</td>
</tr>
</tbody>
</table>

*Figure 47. Trap Destination Policy Dialog Box*
Table 22: Trap Destination Policy Dialog Box Fields (continued)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>Valid values are: disabled, enabled, invalid</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Purpose:</strong> Indicates the current state of the trap destination entry.</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Description:</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Disabled means that the trap destination is not used as a destination for received traps.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Enabled means that the entry is used as a default destination for the traps that are received by the MLM.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>When you click <strong>Delete</strong>, the State field is changed to <strong>invalid</strong>, which causes the selected entry to be deleted.</td>
<td></td>
</tr>
<tr>
<td>Destination</td>
<td><em>Host+service or host+protocol name/portnumber</em>, where <em>host</em> can be one of the following:</td>
<td><strong>Purpose:</strong> Specifies the nodes to which the MLM will forward traps.</td>
</tr>
<tr>
<td></td>
<td>• IP addresses</td>
<td><strong>Description:</strong> Multiple nodes may be specified in a list, with each name separated by a comma or white space.</td>
</tr>
<tr>
<td></td>
<td>• Alias names (from the Alias Table)</td>
<td>You can specify the port number to which traps will be delivered and the protocol to be used in trap delivery (UDP or TCP) by using one of the following formats:</td>
</tr>
<tr>
<td></td>
<td>• Host names</td>
<td>• Host+port number/protocol name</td>
</tr>
<tr>
<td></td>
<td><strong>Description:</strong></td>
<td>• Host+protocol name/port number</td>
</tr>
<tr>
<td></td>
<td>The mask value may be set using values in decimal, hex (0x##), or octet (0###) notation.</td>
<td><strong>Purpose:</strong> Provides a filtering mechanism, preventing certain traps from being sent to the defined Destination.</td>
</tr>
<tr>
<td></td>
<td><strong>Description:</strong></td>
<td><strong>Description:</strong> The value is a numeric representation of a bit mask. Bits corresponding to the generic trap numbers 0–6 are used in the mask to indicate which traps will be forwarded to the defined nodes. The bits are read from left to right, with the leftmost bit representing the generic trap number 0, the next bit representing the generic trap number 1, and so on. The generic trap numbers are:</td>
</tr>
<tr>
<td></td>
<td>The mask value may be set using values in decimal, hex (0x##), or octet (0###) notation.</td>
<td><strong>Description:</strong> The value is a numeric representation of a bit mask. Bits corresponding to the generic trap numbers 0–6 are used in the mask to indicate which traps will be forwarded to the defined nodes. The bits are read from left to right, with the leftmost bit representing the generic trap number 0, the next bit representing the generic trap number 1, and so on. The generic trap numbers are:</td>
</tr>
<tr>
<td></td>
<td><strong>Purpose:</strong></td>
<td>• 0 = Cold Start</td>
</tr>
<tr>
<td></td>
<td><strong>Description:</strong></td>
<td>• 1 = Warm Start</td>
</tr>
<tr>
<td></td>
<td>The mask value may be set using values in decimal, hex (0x##), or octet (0###) notation.</td>
<td>• 2 = Link Down</td>
</tr>
<tr>
<td></td>
<td>Provides a filtering mechanism, preventing certain traps from being sent to the defined Destination.</td>
<td>• 3 = Link Up</td>
</tr>
<tr>
<td></td>
<td>The mask value may be set using values in decimal, hex (0x##), or octet (0###) notation.</td>
<td>• 4 = Authentication Failure</td>
</tr>
<tr>
<td></td>
<td>Provides a filtering mechanism, preventing certain traps from being sent to the defined Destination.</td>
<td>• 5 = EGP Neighbor Loss</td>
</tr>
<tr>
<td></td>
<td>Provides a filtering mechanism, preventing certain traps from being sent to the defined Destination.</td>
<td>• 6 = Enterprise Specific</td>
</tr>
<tr>
<td></td>
<td>To send all traps received from the filtering mechanism to the defined nodes, set the bit mask to 254 or 0xFE. To send only Link Down (2) and Link Up (3) traps to the defined nodes, set the corresponding bits to “on” (0011 0000). To set the bit mask, enter the decimal value representation 48 or the hexadecimal value 0x30.</td>
<td>To send all traps received from the filtering mechanism to the defined nodes, set the bit mask to 254 or 0xFE. To send only Link Down (2) and Link Up (3) traps to the defined nodes, set the corresponding bits to “on” (0011 0000). To set the bit mask, enter the decimal value representation 48 or the hexadecimal value 0x30.</td>
</tr>
</tbody>
</table>
Table 22. Trap Destination Policy Dialog Box Fields (continued)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activation</td>
<td>Any valid character string for activation time.</td>
<td><strong>Purpose</strong>: Specifies the time and days of the week when the trap destination policy is to be activated and deactivated.</td>
</tr>
<tr>
<td>Schedule</td>
<td></td>
<td><strong>Activation</strong>: Use this field to delay the onset of sending traps. Specify time in the format <code>HH:MM</code>, where <code>HH</code> is an integer in the range of 0–23 and <code>MM</code> is an integer in the range of 0–59. If this field is not set, the time is set to 00:00.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Select the days of the week when you want the trap destination policy to be active.</strong></td>
</tr>
<tr>
<td>Deactivation Time</td>
<td></td>
<td><strong>Deactivation Time</strong>: Use this field to set the time when the trap destination policy will be deactivated. Specify time in the format <code>HH:MM</code>, where <code>HH</code> is an integer in the range of 0–23 and <code>MM</code> is an integer in the range of 0–59. If this field is not set, the time is set to 00:00.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Select the days of the week when you want the trap destination policy to be deactivated.</strong></td>
</tr>
</tbody>
</table>

**Analysis Policy Dialog Box**

To define an analysis policy, select **Analysis** using the Policy Type button on the Agent Policy Manager Configuration main menu. The following dialog box is displayed as shown in Figure 48 on page 242.
The fields in this dialog box have the following meanings:

Table 23. Analysis Policy Dialog Box Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose/Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Any unique ASCII character string.</td>
<td><strong>Purpose</strong>: Specifies the name of a particular analysis entry. <strong>Description</strong>: The value in the Name field is used as a label and an instance ID, identifying each field for this entry, and is appended to the corresponding object ID. The instance IDs for each field in the same entry are the ASCII values for each letter of the name. For example, a value of APP_TIME causes the instance ID for this entry to be 65.80.80.95.84.73.77.69, where each letter is an ASCII integer representation of APP_TIME.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Values</td>
<td>Purpose/Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| State           | Valid values are: Disabled, Enabled, Invalid                           | Purpose: Identifies the current activity state of the analysis entry.  
**Description:**  
- **Disabled** means that the entry is valid, but it is not currently available for use.  
- **Enabled** means that the entry is available for use, but the results are not stored.  
When you click **Delete**, the State field is changed to **invalid**, which causes the selected entry to be deleted. |
| Description     | Any valid character string                                             | Purpose: Specifies the general purpose of the analysis operation.  
**Description:** The description character string can be as detailed as necessary. This field is for informational purposes. This field is not mandatory, but is recommended. |
| Poll Time       | Valid values include an integer followed by one of these letters: d = days, h = hours, m = minutes, s = seconds, u = milliseconds  
The minimum is 1 minute. | Purpose: Indicates the time period that should elapse before the next polling operation is invoked.  
**Description:** The polling is done at the defined intervals and the retrieved values are cached for later use in evaluating expressions. Thus, no SNMP get request is necessary while an expression is being processed.  
The default Poll Time value is **1m**. If the letter is omitted, the default is minutes. For example, the value **1h10m** means to start this polling operation 1 hour and 10 minutes following the last polling operation. |
| MIB Variable Expression | Can be comprised of valid MIB object IDs, operators, and functions. | Purpose: Indicates the mathematical expression to be evaluated on the retrieved MIB variable values.  
**Description:** In MLM expressions, you can specify MIB variables from local and remote nodes. To use remote nodes, use the remote node alias name, host name, or IP address and a colon (:) as a prefix to the MIB object ID. You can also specify MIB variables from multiple instances by appending a wildcard character (*) to the end of the MIB object ID.  
In your expression, use operators from a predefined set or the provided built-in functions. See [Operators for MIB Variable Expressions](#) and [Functions for MIB Variable Expressions](#) for descriptive lists of the available operators and functions.  
Click **Select** to access the Tivoli NetView MIB browser, which can help you obtain object IDs and other information about the MIB variables on which you want to perform analysis.  
When MIB variables return multiple instance values, all resolved values are averaged before they are used in the expression. |
Table 23. Analysis Policy Dialog Box Fields (continued)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose/Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Result Type</td>
<td>Valid values are:</td>
<td><strong>Purpose:</strong> Indicates the type of output request to be used for operation results.</td>
</tr>
<tr>
<td></td>
<td>Integer</td>
<td><strong>Description:</strong> Based on the Result Type value, the results for the command are</td>
</tr>
<tr>
<td></td>
<td>Counter</td>
<td>saved to a predefined corresponding MIB variable. The results may be propagated to</td>
</tr>
<tr>
<td></td>
<td>Gauge</td>
<td>the other predefined MIB variables. For example, if the Result Type value is</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>integer</code>, when the expression is evaluated, the output is converted to an integer,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>saved in the IntegerResult MIB variable, and propagated to all other variables</td>
</tr>
<tr>
<td></td>
<td></td>
<td>based on IntegerResult.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Results for <code>integer</code> are signed mathematical values. Results for <code>counter</code> and</td>
</tr>
<tr>
<td></td>
<td></td>
<td><code>gauge</code> are unsigned mathematical values. The results for <code>counter</code> wrap if the</td>
</tr>
<tr>
<td></td>
<td></td>
<td>value is over or under; the results for <code>gauge</code> latch at 0 (zero), if under and 4294967295 if over.</td>
</tr>
</tbody>
</table>

Alias Policy Dialog Box

To define an alias policy, click **Alias** using the Policy Type button on the Agent Policy Manager Configuration main menu. The following dialog box is displayed as shown in Figure 49 on page 245.
The fields in this dialog box have the following meanings:

**Table 24. Alias Policy Dialog Box Fields**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose/Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Any character string expression.</td>
<td><strong>Purpose</strong>: Defines the alias name for the objects in the List field.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Description</strong>: This alias can be used in other policies, such as threshold, analysis, or trap destination to perform functions on multiple remote nodes.</td>
</tr>
</tbody>
</table>
Table 24. Alias Policy Dialog Box Fields (continued)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose/Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>Valid values are:</td>
<td><strong>Purpose:</strong> Indicates whether the alias is currently available for use.</td>
</tr>
<tr>
<td></td>
<td>Enabled</td>
<td><strong>Description:</strong></td>
</tr>
<tr>
<td></td>
<td>Disabled</td>
<td>• Enabled means that the alias entry is valid and available for use.</td>
</tr>
<tr>
<td></td>
<td>Invalid</td>
<td>• Disabled means that the alias entry is valid, but is dormant and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>is not currently being used.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>When you select <strong>Delete</strong>, the State field is changed to invalid, which</td>
</tr>
<tr>
<td></td>
<td></td>
<td>causes the selected entry to be deleted.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>List:</strong></td>
</tr>
<tr>
<td></td>
<td>Valid values are:</td>
<td><strong>Purpose:</strong> Indicates the nodes represented by the alias.</td>
</tr>
<tr>
<td></td>
<td>Alias names</td>
<td><strong>Description:</strong></td>
</tr>
<tr>
<td></td>
<td>Host names</td>
<td>• Separate multiple values by a comma or space.</td>
</tr>
<tr>
<td></td>
<td>IP addresses</td>
<td>• You can group alias names under other alias names.</td>
</tr>
<tr>
<td></td>
<td>Enterprise IDs</td>
<td>• Also, nodes can be grouped under multiple alias names to meet your needs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• You may want to give an alias name to enterprise IDs for use in the Filter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Table, enabling filtering by enterprise ID.</td>
</tr>
</tbody>
</table>

Administration Policy Dialog Box

To define an administration policy, select Administration using the Policy Type button on the Agent Policy Manager Configuration main menu. The following dialog box is displayed as shown in Figure 50 on page 247.
The fields in this dialog box have the following meanings:

Table 25. Administration Policy Dialog Box Fields

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose and Description</th>
</tr>
</thead>
</table>
| Name       | Any unique ASCII character string. | **Purpose:** Specifies the name of a particular entry in the table.  
**Description:** The value in the Name field is used as a label and an instance ID, identifying each field for this entry, and is appended to the corresponding object ID. The instance IDs for each field in the same entry are the ASCII values for each letter of the name. For example, a value of APP_TIME causes the instance ID for this entry to be 65.80.80.95.84.73.77.69, where each letter is an ASCII integer representation of APP_TIME.
Table 25. Administration Policy Dialog Box Fields (continued)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Values</th>
<th>Purpose and Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>Valid values are: Valid</td>
<td>Purpose: Identifies whether the entry is current.</td>
</tr>
<tr>
<td></td>
<td>Outdated Valid</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Invalid</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Purpose: Identifies whether the entry is current.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Description:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Valid means that the information in this entry is authentic and can be used.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Outdated means that the information in this entry is obsolete and should not be used or should be used with this knowledge.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>When you select Delete, the State field is changed to invalid, which causes the selected entry to be deleted.</td>
<td></td>
</tr>
<tr>
<td>Description</td>
<td>Any valid character string.</td>
<td>Purpose: Indicates the general purpose of the information in the entry.</td>
</tr>
<tr>
<td></td>
<td>Description:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The description character string can be as detailed as necessary. This field is for informational purposes. This field is not mandatory, but is recommended.</td>
<td></td>
</tr>
<tr>
<td>Value</td>
<td>Any valid ASCII string.</td>
<td>Purpose: Provides administrative information.</td>
</tr>
<tr>
<td>Owner</td>
<td>Any valid ASCII string.</td>
<td>Purpose: Indicates the owner of this piece of information.</td>
</tr>
</tbody>
</table>

Diagnosing Problems Using the Problem Determination Assistance Facility

APM provides a tool that can aid you in determining what the conditions were that triggered a threshold or file monitor trap or just display data about an APM policy. To see the Problem Determination Assistance dialog box, do the following:

1. Double-click APM Monitors.
   A map with the SmartSets used by APM is displayed.
2. Double-click a SmartSet icon to see the objects in the SmartSet.
3. Double-click down to the node-level view, which looks similar to Figure 51 on page 249.
If a threshold or file monitor condition was met, one of the icons is red. If an icon is blue, then either the distribution of the policy to the node failed or the session between the node and its managing MLM went down.

4. Double-click the icon.

Note that you can display the Problem Determination Assistance dialog box even if all the icons are green.

A Problem Determination Assistance dialog box, similar to Figure 52 on page 250, is displayed.
From this menu, you can see a summary of the policy settings, start some applications, perhaps to help you diagnose the problem that triggered the threshold or file monitor condition, and reset the color of file monitor icons. You can also create your own scripts to maintain log files, erase files, and do automated recovery. Commands run on the Tivoli NetView host.

Resetting the Color of a File Monitor Icon

You can reset the color of a file monitor icon from the Problem Determination Assistance dialog box by clicking **Reset** in the middle of the dialog box. The color of the icon from which you started the Problem Determination Assistance dialog box is
changed to green, and the change in status is propagated up to the root map. If there are other file monitor icons you want to reset, you must start the Problem Determination Assistance dialog box from each icon to reset the color.

Starting Applications from the Problem Determination Assistance Dialog Box

The following diagnosis applications are provided on the Problem Determination Assistance dialog box:

- **Display submap** starts a submap showing the node-level view for the affected node. This view has two icons.
  - The node interface card, as on other Tivoli NetView maps.
  - One or more executable icons indicate thresholds and file monitor conditions that have been created.

- **Dynamic events for this node** opens a dynamic workspace showing all events for the node.

- **MIB Browser** starts the xnmbrowser application for this node.

- **Historic Graph Data** starts the xnmgrapher application and graphs the trap data that was stored for file monitors and thresholds against the node. The data is stored in the /usr/OV/databases/C5 directory.

By clicking on the **Add** button, you can add the following applications to the Problem Determination Assistance dialog box:

- **Historic events for this node** opens a static workspace showing all events for the node.

- **Print** starts the Tivoli NetView Print Tool application.

- **Mail** starts the mail application.

Select the application you want to start on the Problem Determination Assistance dialog box and then click **Start**.

Adding an Application to the Problem Determination Assistance Dialog Box

To add your own application to the Problem Determination Assistance dialog box, follow this procedure:

1. Click **Add** on the Problem Determination Assistance dialog box (when no applications are selected).

The Actions List panel is displayed as shown in Figure 53 on page 252.
2. Click **Define/Edit**.
   The Define/Edit dialog box is displayed as shown in Figure 54.

3. Enter the name for your application in the Description field and the full path name of the command or shell script that is to be executed in the Command Field and click **OK**. For example, to add an application named Display Warning...
Pop, which executes the ovxecho command, enter Display Warning Pop Up in the Description field and the following commands in the Command field:

```
export DISPLAY=host1:0;/usr/OV/bin/ovxecho 'I have a problem'
```

Use a semicolon (;) to separate multiple commands as shown in Figure 54 on page 252.

The name of your application, Display Warning Pop Up, is added to the Application Actions section on the Problem Determination Assistance dialog box.

4. Select Display Warning Pop Up on the Problem Determination Assistance dialog box and select Start to execute your application.

---

**Example of Defining and Distributing a Policy**

To help you understand how APM interacts with the SmartSet Facility, Tivoli NetView, and Tivoli NetView MLM, read this step-by-step scenario of how an administrator would define a SmartSet and set up a file monitoring condition.

1. A network administrator at a university has been having problems with someone logging into a file server in the network and tampering with files to which only root has access. On the assumption that the troublemaker is guessing at root passwords, and thus making several faulty login attempts each time a machine is broken into, the administrator decides to monitor the /etc/security/failedlogin file for failed attempts. He will use the file monitoring capability provided by the Tivoli NetView MLM SIA, which is installed on each of the file servers. Since the /etc/security/failedlogin file is in binary, the administrator will use the `fwtemp` accounting command provided on the operating system to translate the file to ASCII before file monitoring begins.

2. First, the administrator defines a SmartSet called fileservers that includes the important nodes, starts the SmartSet Editor from the Tivoli NetView Tools menu and clicks Add to add a new SmartSet. On the Add SmartSet dialog, the administrator defines a SmartSet called fileservers, using the Node List definition type to specify the IP addresses of the operating system file servers. Click OK.

3. Next, the administrator selects APM Configuration from the Tools pull-down menu and clicks Add/Copy to add a file monitoring condition.

4. On the File Monitor dialog box as shown in Figure 55 on page 254, the administrator defines the following entries:
The file monitor entry Badlogins will check the /tmp/failedlog.ascii file for the string root every five minutes.

The administrator clicks on **Automated Actions**. He defines the following entries in the Automated Actions dialog box as shown in Figure 56 on page 255.

---

**Figure 55. Example of a File Monitoring Condition**

The file monitor entry Badlogins will check the /tmp/failedlog.ascii file for the string root every five minutes.

The administrator clicks on **Automated Actions**. He defines the following entries in the Automated Actions dialog box as shown in Figure 56 on page 255.
The command `/usr/sbin/acct/fwtmp < /etc/security/failedlogin > /tmp/failedlog.ascii` uses the fwtmp utility to convert the binary file to ASCII and put the translated information into a temporary file called failedlog.ascii.

After this command is executed, file monitoring begins. Every five minutes, the file is tested for the string `root`. If the string is found, a special file monitor trap is sent to Tivoli NetView.

By default, the SIA would mark the file at the location where it stopped monitoring and would resume monitoring at that point in the file. In this particular installation, that default has been changed to reset monitoring to the beginning of the file between polling intervals. To prevent the SIA from generating another trap in the next polling interval for the string that was already found in the file, the administrator specifies the command `grep -v root /tmp/failedlog.ascii | /usr/sbin/acct/fwtmp -ic > /etc/security/failedlogin` as the command to be executed after the string is found. This command takes all contents of the failedlog.ascii file except the failed root login and writes the contents back into the failedlogin file. During the next polling cycle, when the binary file is again converted into ASCII, it will not have the records that caused the previous file monitor trap to be sent.

5. To specify SmartSets to which this file monitoring policy will be applied, the administrator clicks Assign on the File Monitoring dialog box. On the SmartSet Assignments dialog that is displayed, the administrator selects Fileservers and then clicks Assign. Fileservers is moved to the Assigned SmartSets list, and the administrator clicks OK to make the assignment. (Starting the SmartSet Editor and defining the SmartSet from this dialog if it was not already defined, is an option.)

6. On the File Monitor dialog box, the administrator clicks Apply to apply the setting. A message is displayed in the Messages area that indicates the policy was saved successfully.

7. Now the administrator needs to distribute the policy to the machines that make up the fileservers collection. From the APM main dialog, he selects the badlogins policy and clicks Distribute.

8. On the Distribute Definitions dialog that is displayed, click Start to distribute the collection. Messages indicate when the set to each node is complete. (If one or more of the sets had failed, a message indicating why it failed it displayed. The policy status will be partially distributed.

9. The administrator goes back to the Tivoli NetView Control Desk and selects Create..Dynamic Workspace to start a dynamic workspace. On the dialog box that is displayed, he selects Filter Activation in the Filter section of the dialog. On the dialog box that is displayed, select File List to get a list of filters. From the list, he selects C5filters and clicks OK. In the Available Filters in File
section of the dialog box, he selects APMConsole and Activate. He then clicks Close and OK to open the workspace.

10. Later that afternoon, the SmartSets icon on the root map turns yellow. The administrator quickly checks the dynamic workspace set up earlier and sees that file monitor specific trap 21 (String Found) arrived. The perpetrator has been caught in the process of breaking into one of the file servers.

11. Some time later (after locating the guilty party in the university computing center), the administrator double-clicks down through the APM Monitors submaps to the node-level view map for the affected node. Double-clicks the executable icon to display the Problem Determination Assistance dialog box. Resets the status on the node so that the color returns to green.

APM Reference

The following sections have more detailed information about the APM and how it interacts with Tivoli NetView MLM and the network.

Configuring Community Names

The APM application uses SNMP community names to control the access the Tivoli NetView manager has to managed objects in the network. Community names are passed back and forth in SNMP get and set requests and in traps. The community name is similar to a password in that it determines whether an entity can gain access to information or perform an action.

On SNMP-based network devices, SNMP community names are defined in a file called snmpd.conf. The SNMP daemon checks this file when it receives a request for information.

Tivoli NetView and its associated applications, such as Tivoli NetView MLM, use another file called ovsnmp.conf to associate objects with community names. Tivoli NetView and Tivoli NetView MLM check this file when they send requests for information.

In summary:

- When a node sends a request, it uses the ovsnmp.conf file to determine what community name should be sent in the request. The community name associated with the target node is included in the request.
- When a node receives a request, it uses the snmpd.conf file to validate the community name sent by the requester. The community name associated with the requesting node is compared with the name included in the request.

The community name sent by the requester node and the community name expected by the receiving node must match for the request to be executed as requested. If they do not match, the receiving node will use a community name of public by default.

Mismatched community names cause authentication errors. These errors prevent the distribution of an APM file monitor or threshold policy from being successful.

You should not edit the ovsnmp.conf file directly. Select SNMP Configuration... from the Options pull-down menu to set the community name to be sent to the target node.
If you are defining a threshold condition in your network, your Tivoli NetView manager station must have SNMP set access to the nodes running MLMs. For this reason, it might be easier to use a single community name in your network. The Tivoli NetView [security feature] can provide the application security that you require.


**Community Name Examples**

Here are two scenarios of how entries in the ovsnmp.conf and snmpd.conf files change, depending on which objects you are monitoring. For simplicity, these examples show a single machine running each agent. In a real working network, community names are be set up on a much wider basis, such as across a subnet or even across the whole network.

### First Example

Tivoli NetView, an MLM, and an SIA are on a machine called `nvmgr`. You want to set a thresholding condition against that MLM and monitor a log file using the SIA file monitoring function.

- The ovsnmp.conf file is edited using the SNMP Configuration dialog from the Tivoli NetView Options menu:
  ```
  # MLM entry (LOOPBACK)
  127.0.0.1:system:*:::::system:
  # SIAentry
  nvmgr:siaset:*:::::siaset:
  ```

- The snmpd.conf file entries are entered as follows:
  ```
  # MLM name
  community system 127.0.0.1 255.255.255.255 readWrite
  # SIA name
  community siaset 127.0.0.1 255.255.255.255 readWrite
  ```

### Second Example

An MLM and an SIA are on a machine called `workstation`. From the `nvmgr` node, you want to set a thresholding condition against that MLM and monitor a log file using the SIA file monitoring function.

- The ovsnmp.conf file is edited using the SNMP Configuration dialog from the Tivoli NetView Options menu:
  ```
  # MLM entry
  workstation:mlmset:*:::::mlmset:
  # SIAentry
  workstation:siaset:*:::::siaset:
  ```

- The snmpd.conf file entries are entered as follows:
  ```
  # MLM name
  community system 127.0.0.1 255.255.255.255 readWrite
  # SIA name
  community siaset 127.0.0.1 255.255.255.255 readWrite
  ```

See the *Tivoli NetView for UNIX Mid-Level Manager User’s Guide* for extensive scenarios showing how you would configure community names depending on where you have the Tivoli NetView MLM installed.
APM Aliases

You can use APM to customize an Alias Table that is used for several MLM functions.

Managing Aliases

If the administrator configures the MLMs in the network to take over local discovery and status monitoring duties from the Tivoli NetView manager, the MLM assigns aliases for groups of nodes to facilitate management of the nodes. APM also uses the Alias Table to keep track of groups of nodes. It sets aliases for SmartSets that have thresholds set against them, and for the thresholds themselves.

After you define a threshold and distribute it to a SmartSet, an alias is defined for this SmartSet. Aliases for SmartSets have names in the format {NV<ip-address>_ssname}, where ip-address is the IP address of the Tivoli NetView manager, and ssname is the name of the SmartSet. For example, a SmartSet called FileServers assigned to the MLM by the Tivoli NetView manager on address 9.67.102.16 would be assigned an alias of {NV9.67.102.16_Fileservers}.

Similarly, there is an alias for each threshold you define. Thresholds have aliases in the format {NV<ip-address>_thresholdname}, where ip-address is the IP address of the Tivoli NetView manager, and thresholdname is the name of the threshold. Do not update these aliases with the Tivoli NetView MLM Configuration Interface.

The alias policies are APM’s mechanism for managing changes to SmartSets and thresholds. SmartSets are dynamically updated to reflect changes in a network. When a SmartSet is changed, the change is made to the alias policy as well. You do not have to redistribute a threshold policy if you use the SmartSet Editor to change the members of a SmartSet; the APM automatically redistributes, based on the changes to the alias.

APM MLM Domains

For thresholding, APM defines domains for the MLMs in the network. Basically, it determines where the MLMs are and divides the managed objects as equitably as possible.

Initially, MLM domains are defined based on subnet IDs. Nodes that are in the same subnetwork as an MLM are assigned to that MLM. Nodes that do not have an MLM in their subnet are assigned to a default domain. There is no overlap in these initial policies; each node is managed by one MLM only. If there are several MLMs in a subnet, then the first one found manages the nodes in the subnet, and the other MLMs only manage themselves.

To see these domains from the Tivoli NetView root map, double-click the MLM Managers icon. Also, if you start the SmartSet Editor, you will see the MLM domains (as well as the mlmDomain_Default domain) listed as already existing SmartSets.

APM chooses the MLM that will serve as the default domain depending on how your network is set up:

1. If there is an MLM on the Tivoli NetView manager, it will be automatically assigned responsibility for the default domain.
2. If the MLM is not installed on the Tivoli NetView manager, you must assign the default domain SmartSet to one of the MLMs in your network using the SmartSet Editor.
If there is only one MLM in your network, that MLM is assigned all nodes, and a default domain might not be created.

**Rearranging MLM Domains Manually**

You might want to alter the distribution of nodes to MLMs to facilitate your management of the network. For example, you might want an MLM to manage all the routers in the network, or to manage all objects in a physical location. To do this, start the SmartSet Editor and select the domain you want to change, then edit the list of nodes. Because the MLM domain is just a SmartSet, you can use the same rule logic that you use for defining SmartSets to set thresholds against. You do not need to reassign any nodes that were in the old domain but are not in the new domain; APM takes care of assigning new domain responsibilities for the nodes. APM puts the nodes in the default domain or the domain for the MLM for the subnet.

**Note:** When you remove subnet responsibility from an MLM, you should assign the responsibility to another MLM in the subnet. If you do not, the entire subnet will be placed in the default domain.

The netmon daemon and APM can share the mlmDomain collections for work distribution if you configure the netmon daemon to do so. See the `netmon` man page for more information.

If you have MLMs that are offloading discovery and status monitoring from Tivoli NetView and are very busy as a result, you might want to lighten their workload by moving nodes out of their APM domains.

If you add a new MLM to your network, APM recognizes it as an MLM and assigns a domain SmartSet, but the SmartSet contains only the MLM. If you edit this SmartSet and add nodes to it, APM takes care of moving node responsibilities accordingly.

**Automatic MLM Backup Domains**

If an MLM disappears from the network, APM automatically redistributes that MLM’s workload. A new SmartSet is created called delDomain__mlmname, where `mlmname` is the name of the MLM. The nodes for which the MLM was responsible are redistributed to other MLMs in the network. They are assigned according to their subnets.

If the MLM later reappears in the network, the domain is re-created and the nodes are reassigned to the original MLM.

**Matched, Armed, and Disarmed Command Environment Variables**

The commands entered in the Matched Command, Armed Command, and Disarmed Command fields on the Threshold Policy and Filter Policy dialog boxes can use the environment variables that contain information from the received trap.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SM6K_HOSTNAME</td>
<td>Contains the local host name of the node on which the MLM is running.</td>
</tr>
<tr>
<td>SM6K_DOMAIN_NAME</td>
<td>Contains the local domain name, or fully qualified host name, on which the MLM is running.</td>
</tr>
</tbody>
</table>
Table 26. MLM Environment Variables (continued)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SM6K_HOST_ADDRESS</td>
<td>Contains the IP address of the host on which the MLM is running.</td>
</tr>
<tr>
<td>SM6K_INSTANCE_NAME</td>
<td>Contains the name of the filter rule that matched the trap.</td>
</tr>
<tr>
<td>SM6K_INSTANCE_ID</td>
<td>Contains the instance ID of the filter rule that matched the trap.</td>
</tr>
<tr>
<td>SM6K_EXECUTION_REASON</td>
<td>Indicates why the command is being executed. The possible values are Filter_Matched, Filter_Armed, or Filter_Disarmed:</td>
</tr>
<tr>
<td></td>
<td>• <strong>Filter_Matched</strong> means the command is being entered in the Matched Command field on the Filter Table window. The command will be executed by the MLM when a received trap matches the filter rule.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Filter_Armed</strong> means the command is being entered in the Armed Command field on the Throttle window. The command will be executed by the MLM when the number of received matching traps matches the value for the Arm Count, and the throttle is armed.</td>
</tr>
<tr>
<td></td>
<td>• <strong>Filter_Disarmed</strong> means the command is being entered in the Disarmed Command field on the Throttle window. The command will be executed by the MLM when the currently armed throttle is disarmed. The throttle may be disarmed, because the number of received matching traps matches the value for the Disarm Count or the time period specified in the Disarm Timer has elapsed.</td>
</tr>
</tbody>
</table>

Table 27 shows the environment variables that are set to reflect values from the received trap:

Table 27. Environment Variables Reflecting Information Received in Traps

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SM6K_TRAP_ENTERPRISE</td>
<td>Contains the enterprise ID contained in the trap.</td>
</tr>
<tr>
<td>SM6K_TRAP_AGENT_ADDRESS</td>
<td>Contains the IP address of the agent sending the trap.</td>
</tr>
<tr>
<td>SM6K_TRAP_GENERIC_NUM</td>
<td>Contains the generic trap number of the trap.</td>
</tr>
<tr>
<td>SM6K_TRAP_SPECIFIC_NUM</td>
<td>Contains the specific trap number of the trap.</td>
</tr>
<tr>
<td>SM6K_TRAP_TIME_TICKS</td>
<td>Contains the time ticks since the agent generating the trap was started.</td>
</tr>
<tr>
<td>SM6K_NUM_TRAP_VARS</td>
<td>Contains the number of MIB variables contained in the trap.</td>
</tr>
</tbody>
</table>

The following environment variables correspond to the MIB variables contained in the trap. These environment variables are set for each variable in the trap. To differentiate between these environment variables for the different MIB variables,
the positional number is appended to the environment variable. MIB variable numbering ranges from 1 to \( n \), where \( n \) corresponds to the number of MIB variables contained in the trap.

The environment variables that correspond to the MIB variables in the received trap are listed in Table 28.

### Table 28. Environment Variables Reflecting MIB Variables from Received Traps

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SM6K_TRAP_VAR_OBJ_ID#</td>
<td>Contains the object ID for MIB variable number # in the variable bindings.</td>
</tr>
<tr>
<td>SM6K_TRAP_VAR_TYPE#</td>
<td>Indicates the type of the MIB variable number #. The type is one of the following:</td>
</tr>
<tr>
<td></td>
<td>• INTEGER</td>
</tr>
<tr>
<td></td>
<td>• Counter</td>
</tr>
<tr>
<td></td>
<td>• Gauge</td>
</tr>
<tr>
<td></td>
<td>• TimeTicks</td>
</tr>
<tr>
<td></td>
<td>• IpAddress</td>
</tr>
<tr>
<td></td>
<td>• OCTET_STRING</td>
</tr>
<tr>
<td></td>
<td>• Opaque</td>
</tr>
<tr>
<td></td>
<td>• OBJECT_IDENTIFIER</td>
</tr>
<tr>
<td></td>
<td>• NULL</td>
</tr>
<tr>
<td></td>
<td>• Unknown</td>
</tr>
<tr>
<td>SM6K_TRAP_VAR_VALUE#</td>
<td>Contains the value of the MIB object. For the MIB types INTEGER, Counter, Gauge, and TimeTicks the value is a decimal integer. For the MIB type IpAddress, the value is stored in dot notation. For the MIB types OCTET_STRING and Opaque, the value is stored as an ASCII string unless it contains unprintable characters. If it contains unprintable characters, it is stored as a hexadecimal string. For MIB type OBJECT_IDENTIFIER, the value is stored using dot notation. For MIB type NULL, the value is set to NULL.</td>
</tr>
</tbody>
</table>

The environment variables described in the following table are valid for Armed and Disarmed commands. These environment variables correspond to specific filter throttle MIB variables, as shown in Table 29.

### Table 29. Environment Variables Reflecting Filter Throttle MIB Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SM6K_THROTTLE_ARM_COUNT</td>
<td>Contains the value of the ThrottleArmTrapCount MIB variable (the Arm Count value).</td>
</tr>
<tr>
<td>SM6K_THROTTLE_DISARM_COUNT</td>
<td>Contains the value of the ThrottleDisarmTrapCount MIB variable (the Disarm Count value).</td>
</tr>
<tr>
<td>SM6K_EXECUTION_REASON</td>
<td>Set to Filter_Armed when the throttle is armed, or set to Filter_Disarmed when the throttle is disarmed.</td>
</tr>
<tr>
<td>SM6K_THROTTLE_DISARM_TIMER</td>
<td>Contains the string value of the ThrottleDisarmTimer MIB variable (the Disarm Timer value).</td>
</tr>
</tbody>
</table>

### Operators for MIB Variable Expressions

The MIB variable expression that you specify in an analysis policy can be assigned the following operators. The expressions are evaluated in order of precedence and associativity of operators:
( )  Grouping
−  Unary minus
*  Multiplication
/  Division
%  Remainder
+  Addition
−  Subtraction
<<  Bitwise left shift
>>  Bitwise right shift
&  Bitwise AND
^  Bitwise exclusive OR
|  Bitwise inclusive OR

Functions for MIB Variable Expressions

The MIB variable expression that you specify in an analysis policy can use the following functions:

Notes:
1. An MLM MIB variable can have an IP address, host name, or alias appended to it and can resolve to multiple values.
2. All wildcard MIB variables are averaged before being used.

@sum(MIB variable,MIB variable,...)
   Returns the sum of all values.

@avg(MIB variable,MIB variable,...)
   Returns the average of all values.

@min(MIB variable,MIB variable,...)
   Returns the minimum of the values.

@max(MIB variable,MIB variable,...)
   Returns the maximum of the values.

@diff(MIB variable expression,MIB variable expression...)
   Returns the positive difference of the two expressions. This function is useful for counters that are always positive.

@abs(MIB variable expression...)
   Returns the absolute value of the expression.

@delta(MIB variable expression)
   Uses the delta value calculated by subtracting the last returned MIB variable value from the current MIB variable value. This function can be used to force delta calculations on integer, gauge, and displaystring MIB variables.

@value(MIB variable expression)
   Uses the current MIB variable value. This function can be used to force the use of actual values for counter MIB variables.

@rand()
   Returns a random number between (0) and MAX_INT.

@numInst(regular expression)
   Returns the number of instances found for the expression. This function enables you to monitor a table for a change in the number of instances and to monitor the process table for same-name processes to see whether any is removed or added.
Appendix A. Tivoli NetView Internal Traps

Use this reference to understand the traps that are generated by Tivoli NetView.

Terms and Conventions

The following terms and conventions are used:

- Words in italics are explanatory variable names. They are replaced by the value of the variables at run time when the event occurs.
- The term event is used interchangeably with the term trap.
- The term event log generally implies either the trapd.log file or the event card application display.
- Time stamp values are displayed in epoch time. Epoch time is the number of seconds elapsed since an epoch, which in UNIX is January 1, 1970.
- The term objid is an OVw database object ID.
- Link Level Address, (LLA) is the same as physical address which is the physical address of the interface cards. The LLA is typically displayed as a 6 byte, (or 12-digit hexadecimal) number.
- When a demand poll is used as a method of correction, it should be pointed out that a poll is, by default, performed once-a-day for each node. Performing a demand poll is a method of effecting immediate action.

Note: A demand poll can be performed from a separate window by the command `nmdemandpoll nodename` where node name is the node name as it exists in the topology database (generally, the fully qualified DNS name for the node).

Internal Tivoli NetView Traps

The only trap that affects the operation of Tivoli NetView is the SNMP_EV (58916871) trap, which can be generated to facilitate Configurable Status.

All Tivoli NetView traps are generated with the enterprise ID .1.3.6.1.4.1.2.6.3.1 and the generic number of 6, which means enterprise-specific. The specific number for each trap is listed in Table 30 on page 263. A Tivoli NetView trap is generated with 5 variable bindings. The OID for each varbind is also listed, though it has relatively no importance in the generated trap.

Varbind 1

MIB OID: .1.3.6.1.4.1.2.6.3.1.1.2.0

This variable is the source ID. It is an integer value that corresponds to the internal component of the Tivoli NetView that generated the event. Following are the separate Source ID values and their corresponding components. Each source ID value has a unique letter (identified in parenthesis) that is used for identification in the trapd.log file. For instance, the Node Up event text will be preceded by an N, indicating that the source for the trap was the netmon application.

(A)gent (n)netmon related
(a)pplication (O)SI_SuperAgent
(D)data Collector (M)ap/ovtopmd
(d)emo/LoadHosts (S)ecurity Agent
(E)vent Application (s)spappld
(I)PMAP_SuperAgent (T)rapd
Varbind 2
MIB OID: .1.3.6.1.4.1.2.6.3.1.1.3.0

This variable is a string value identifying the host name to which this trap applies. If there is no applicable node, <none> is displayed. As in our previous varbind example, if the Node Up event is generated, then varbind-2 will contain the host name of the node that was detected as operational.

Varbind 3
MIB OID: .1.3.6.1.4.1.2.6.3.1.1.4.0

This variable is a string value containing a description of the event that was generated. For the Node Up event, varbind-3 would contain the text Node Up. This varbind is most important because the 3rd varbind is the text that gets displayed in the event log. This is most evident by looking at the Event Configuration screen of the product. For each Tivoli NetView, the corresponding event log information contains $3. This indicates that the 3rd varbind’s contents are to be displayed in the event log. The exact text for each trap is documented in Table 30 on page 265.

Varbind 4
MIB OID: .1.3.6.1.4.1.2.6.3.1.1.5.0

This variable is a string value containing internal data that is particular to the trap. This field generally contains data such as Time stamp values, object ID values for node and interface objects, IP addresses, and so forth. This information generally has no meaning to the user. The exact value for each trap is documented in the tables.

Varbind 5
MIB OID: .1.3.6.1.4.1.2.6.3.1.1.6.0

This variable contains a string value containing the database name. This name must be openview.

Trap list

The traps are listed in numerical order. For each trap listed, the following information is provided:
- Specific trap number
- Trap name
- Trap description
- Description field (descr). This contains the contents of varbind-3.
- Data field. This contains the contents of varbind-4.
- Condition. This indicates the condition that causes the trap to be generated.

Traps generated by Tivoli NetView are described as follows.
### Table 30. Tivoli NetView Internal Traps

<table>
<thead>
<tr>
<th>Number, Name, and Description</th>
<th>Fields and Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>50462720</strong> WARN_EV Warnings</td>
<td><strong>Descri field</strong>&lt;br&gt;It can be one of the following strings. The number here corresponds with the number under the Condition field.&lt;br&gt;1. WARNING Attempted addition of object with no object ID, <em>objectName</em>&lt;br&gt;2. WARNING Attempted addition of existing object <em>objectId</em>, returning error&lt;br&gt;3. WARNING could not allocate object ID for segment <em>segName</em>&lt;br&gt;4. WARNING Unset field values with object id <em>oid</em> failed: OvwError = <em>errorString</em>.&lt;br&gt;5. WARNING invalid SNMPTrap packet from agent <em>addr</em> source <em>sourceId</em> pid <em>pid</em>&lt;br&gt;<strong>Data field</strong>&lt;br&gt;NULL&lt;br&gt;<strong>Condition</strong>&lt;br&gt;Following are the conditions:&lt;br&gt;1. Attempt was made to add an object with null object ID to the topology database.&lt;br&gt;2. Attempt was made to add an already existing object to the topology database.&lt;br&gt;3. Creation of object ID for segment fails.&lt;br&gt;4. Unsetting of field values in the topology database fails during cleanup.&lt;br&gt;5. Error parsing an SNMP trap from an agent.</td>
</tr>
<tr>
<td><strong>50790400</strong> NM_EV Node Marginal</td>
<td><strong>Descri field</strong>&lt;br&gt;Node Marginal&lt;br&gt;<strong>Data field</strong>&lt;br&gt;Contains the following:&lt;br&gt;• Time stamp&lt;br&gt;• objid of the node object&lt;br&gt;<strong>Condition</strong>&lt;br&gt;Node status change to a marginal state.</td>
</tr>
<tr>
<td><strong>50790401</strong> SN_EV Segment Normal</td>
<td><strong>Descri field</strong>&lt;br&gt;Segment <em>segName</em> Up.&lt;br&gt;<strong>Data field</strong>&lt;br&gt;Contains the following:&lt;br&gt;• Time stamp&lt;br&gt;• objid of the segment object&lt;br&gt;<strong>Condition</strong>&lt;br&gt;Segment status changes to normal.</td>
</tr>
<tr>
<td><strong>50790402</strong> SM_EV Segment Marginal</td>
<td><strong>Descri field</strong>&lt;br&gt;Segment <em>segName</em> Marginal&lt;br&gt;<strong>Data field</strong>&lt;br&gt;Contains the following:&lt;br&gt;• Time stamp&lt;br&gt;• objid of the segment object&lt;br&gt;<strong>Condition</strong>&lt;br&gt;Segment status changes to marginal.</td>
</tr>
<tr>
<td>Number, Name, and Description</td>
<td>Fields and Condition</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------------</td>
</tr>
</tbody>
</table>
| 50790403 NETN_EV Network Normal | **Descr field**  
Network *netName* Up  
**Data field**  
Contains the following:  
• Time stamp  
• objid of the network object  
**Condition**  
Network status changes to normal. |
| 50790404 NETM_EV Network Marginal | **Descr field**  
Network *netName* Marginal  
**Data field**  
Contains the following:  
• Time stamp  
• objid of the network object  
**Condition**  
Network status changes to marginal. |
| 50790405 SA_EV Segment Added | **Descr field**  
Segment *segName* Added  
**Data field**  
Contains the following:  
• Time stamp  
• objid of the segment object  
**Condition**  
A segment was added. |
| 50790406 SD_EV Segment Deleted | **Descr field**  
Segment *segName* Deleted  
**Data Field**  
Contains the following:  
• Time stamp  
• objid of the segment object  
**Condition**  
A segment was deleted. |
| 50790407 NETA_EV Network Added | **Descr field**  
Network *netName* Added.  
**Data field**  
Contains the following:  
• Time stamp  
• objid of the network object  
**Condition**  
A network was added. |
<table>
<thead>
<tr>
<th>Number, Name, and Description</th>
<th>Fields and Condition</th>
</tr>
</thead>
</table>
| **50790408**<br>NETD_EV<br>Network Deleted | **Descr field**<br>Network *netName* Deleted.  
**Data field**  
Contains the following:  
• Time stamp  
• objid of the network object  
**Condition**  
A network was deleted. |
| **50790411**<br>CPP_EVChange<br>Polling Period | **Descr field**<br>Polling Intervals changed  
**Data field**  
NULL  
**Condition**  
Polling intervals were changed via the Topology/Status Polling Intervals window. This trap informs netmon that it needs to re-read the polling intervals file. |
| **50790412**<br>FP_EV<br>Forced Poll Event | **Descr field**<br>Demand polling on node *nodename*  
**Data field**  
Contains the following:  
• *pipename*<br>The pipe opened for communication with netmon  
• *debugflag*<br>Reserved for future use  
**Condition**  
When demand-poll is initiated by nmdemandpoll. The application communicates to netmon by sending this event. |
| **50790416**<br>MNET_EV<br>Manage Network | **Descr field**<br>Network *netname* Managed  
**Data field**  
Contains the following:  
• Time stamp  
• objid of the network object  
**Condition**  
Event generated upon managing a network. |
| **50790417**<br>UNET_EV<br>Unmanage Network | **Descr field**<br>Network *netname* Unmanaged  
**Data field**  
Contains the following:  
• Time stamp  
• objid of the network  
**Condition**  
Event generated upon unmanaging a network. |
<table>
<thead>
<tr>
<th>Number, Name, and Description</th>
<th>Fields and Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>50790418 MN_EV Manage Node</td>
<td>Descr field</td>
</tr>
<tr>
<td></td>
<td>Node Managed</td>
</tr>
<tr>
<td></td>
<td>Data field</td>
</tr>
<tr>
<td></td>
<td>Contains the following:</td>
</tr>
<tr>
<td></td>
<td>• Time stamp</td>
</tr>
<tr>
<td></td>
<td>• objid of the node</td>
</tr>
<tr>
<td>Condition</td>
<td>Event generated upon managing a node.</td>
</tr>
<tr>
<td>50790419 UN_EV Unmanage Node</td>
<td>Descr field</td>
</tr>
<tr>
<td></td>
<td>Node Unmanaged</td>
</tr>
<tr>
<td></td>
<td>Data field</td>
</tr>
<tr>
<td></td>
<td>Contains the following:</td>
</tr>
<tr>
<td></td>
<td>• Time stamp</td>
</tr>
<tr>
<td></td>
<td>• objid of the node</td>
</tr>
<tr>
<td>Condition</td>
<td>Event generated upon unmanaging a node.</td>
</tr>
<tr>
<td>50790420 MSEG_EV Manage Segment</td>
<td>Descr field</td>
</tr>
<tr>
<td></td>
<td>Segment segname Managed</td>
</tr>
<tr>
<td></td>
<td>Data field</td>
</tr>
<tr>
<td></td>
<td>Contains the following:</td>
</tr>
<tr>
<td></td>
<td>• Time stamp</td>
</tr>
<tr>
<td></td>
<td>• objid of the segment</td>
</tr>
<tr>
<td>Condition</td>
<td>Event generated upon managing a segment.</td>
</tr>
<tr>
<td>50790421 USEG_EV Unmanage Segment</td>
<td>Descr field</td>
</tr>
<tr>
<td></td>
<td>Segment segname Unmanaged</td>
</tr>
<tr>
<td></td>
<td>Data field</td>
</tr>
<tr>
<td></td>
<td>Contains the following:</td>
</tr>
<tr>
<td></td>
<td>• Time stamp</td>
</tr>
<tr>
<td></td>
<td>• objid of the segment</td>
</tr>
<tr>
<td>Condition</td>
<td>Event generated upon unmanaging a segment.</td>
</tr>
<tr>
<td>50790423 NMTM_EV Netmon Change Trace Mask</td>
<td>Descr field</td>
</tr>
<tr>
<td></td>
<td>Netmon Trace mask changed.</td>
</tr>
<tr>
<td></td>
<td>Data field</td>
</tr>
<tr>
<td></td>
<td>tracemask</td>
</tr>
<tr>
<td>Condition</td>
<td>If the -M option is used to change netmon's tracemask, then this trap will be generated to inform the netmon daemon of the new tracemask value.</td>
</tr>
<tr>
<td>Number, Name, and Description</td>
<td>Fields and Condition</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><strong>50790427</strong></td>
<td></td>
</tr>
<tr>
<td>CIS_EVChange</td>
<td>Descr field</td>
</tr>
<tr>
<td>Interface Segment</td>
<td>One of the following fields:</td>
</tr>
<tr>
<td></td>
<td>* Interface <code>iflabel</code> no longer connected to segment.</td>
</tr>
<tr>
<td></td>
<td>* Interface <code>iflabel</code> transferred to segment <code>segName</code></td>
</tr>
<tr>
<td></td>
<td>* Interface <code>iflabel</code> Transferred to segment <code>segld</code></td>
</tr>
<tr>
<td></td>
<td>Data field</td>
</tr>
<tr>
<td></td>
<td>Contains the following:</td>
</tr>
<tr>
<td></td>
<td>* IP address</td>
</tr>
<tr>
<td></td>
<td>* Time stamp</td>
</tr>
<tr>
<td></td>
<td>* Node objid</td>
</tr>
<tr>
<td></td>
<td>* Interface objid</td>
</tr>
<tr>
<td></td>
<td>* Segment objid</td>
</tr>
<tr>
<td></td>
<td>Condition</td>
</tr>
<tr>
<td></td>
<td>A change of interface event.</td>
</tr>
<tr>
<td><strong>50790438</strong></td>
<td></td>
</tr>
<tr>
<td>FMTCHG</td>
<td>Descr field</td>
</tr>
<tr>
<td>trapd.conf file format change</td>
<td><code>progname</code> changed format file <code>/usr/OV/conf/C/trapd.conf</code>.</td>
</tr>
<tr>
<td></td>
<td>Data field</td>
</tr>
<tr>
<td></td>
<td>NULL</td>
</tr>
<tr>
<td></td>
<td>Condition</td>
</tr>
<tr>
<td></td>
<td>Generated when contents of the trapd.conf are changed either by the graphical interface application, xnmtrap, or the command line interface, addtrap. This informs any interested applications that they need to re-read the trapd.conf file.</td>
</tr>
<tr>
<td><strong>50790439</strong></td>
<td></td>
</tr>
<tr>
<td>MIBCHGASN.1</td>
<td>Descr field</td>
</tr>
<tr>
<td>mib definition file format changed</td>
<td>New MIB library file.</td>
</tr>
<tr>
<td></td>
<td>Data field</td>
</tr>
<tr>
<td></td>
<td>NULL</td>
</tr>
<tr>
<td></td>
<td>Condition</td>
</tr>
<tr>
<td></td>
<td>Generated when a MIB is loaded with the MIB Compiler/Loader. This informs any interested applications that a new MIB binary file is available.</td>
</tr>
<tr>
<td><strong>50790440</strong></td>
<td></td>
</tr>
<tr>
<td>COLCHGASNMP</td>
<td>Descr field</td>
</tr>
<tr>
<td>data collector file format changed</td>
<td>SNMP data collector started.</td>
</tr>
<tr>
<td></td>
<td>Data field</td>
</tr>
<tr>
<td></td>
<td><code>xnmcollect</code> changed format file <code>/usr/OV/conf/snmpCol.conf</code></td>
</tr>
<tr>
<td></td>
<td>Condition</td>
</tr>
<tr>
<td></td>
<td>Generated when the SNMP Data collector configuration file is updated. This informs the snmpCollect daemon that it needs to re-read its configuration file.</td>
</tr>
<tr>
<td><strong>50790441</strong></td>
<td></td>
</tr>
<tr>
<td>MI_EV</td>
<td>Descr field</td>
</tr>
<tr>
<td>Manage Interface</td>
<td>Interface <code>iflabel</code> managed.</td>
</tr>
<tr>
<td></td>
<td>Data field</td>
</tr>
<tr>
<td></td>
<td>Contains the following:</td>
</tr>
<tr>
<td></td>
<td>* IP address</td>
</tr>
<tr>
<td></td>
<td>* Time stamp</td>
</tr>
<tr>
<td></td>
<td>* Node and interface objids</td>
</tr>
<tr>
<td></td>
<td>Condition</td>
</tr>
<tr>
<td></td>
<td>Generated upon managing an interface.</td>
</tr>
<tr>
<td>Number, Name, and Description</td>
<td>Fields and Condition</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>50790442 UI_EV</td>
<td><strong>Descri field</strong> Interface <em>iflabel</em> unmanaged.</td>
</tr>
</tbody>
</table>
| Unmanage Interface           | **Data field** Contains the following:  
|                              | • IP address  
|                              | • Time stamp  
|                              | • Node and interface objids |
|                              | **Condition** Generated upon unmanaging an interface. |
| 50790443 NETFC_EV            | **Descri field** Network *netName* has new flags: *netFlags*. |
| Network Flags changed        | **Data field** Contains the following:  
|                              | • Time stamp  
|                              | • Network object ID  
|                              | • New network flags  
|                              | • Old network flags |
|                              | **Condition** Event noting change of internal flags for network. Flags are:  
|                              | REMOVED USER_ADDED LAYOUT_OFF  
|                              | USE_XY SERIAL_NETWORK |
| 50790444 SEGFC_EV            | **Descri field** Flags for Segment *segName* changed to *segNewFlags* |
| Segment Flags changed        | **Data field** Contains the following:  
|                              | • Time stamp  
|                              | • Segment objid  
|                              | • New segment flags  
|                              | • Old segment flags |
|                              | **Condition** Event noting change of internal flags for segment. Flags are:  
|                              | REMOVED USER_ADDED LAYOUT_OFF  
|                              | USE_XY BUS_SEG STAR_SEG  
<p>|                              | TOKEN_RING FDDI_RING SERIAL_SEG |</p>
<table>
<thead>
<tr>
<th>Number, Name, and Description</th>
<th>Fields and Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>50790445 NFC_EV</td>
<td>Descr field</td>
</tr>
<tr>
<td></td>
<td>Node Flags changed to nodeFlags</td>
</tr>
<tr>
<td>Data field</td>
<td>Contains the following:</td>
</tr>
<tr>
<td></td>
<td>• Time stamp</td>
</tr>
<tr>
<td></td>
<td>• Node object ID</td>
</tr>
<tr>
<td></td>
<td>• NewNodeFlags</td>
</tr>
<tr>
<td></td>
<td>• OldNodeFlags</td>
</tr>
<tr>
<td>Condition</td>
<td>Event noting change of internal flags for node. Flags are:</td>
</tr>
<tr>
<td></td>
<td>REMOVED USER_ADDED LAYOUT_OFF</td>
</tr>
<tr>
<td></td>
<td>USE_XY CONNECTOR GATEWAY</td>
</tr>
<tr>
<td></td>
<td>STAR_HUB IS_SMART_CONN IS_BRIDGE</td>
</tr>
<tr>
<td>50790446 IFC_EV</td>
<td>Descr field</td>
</tr>
<tr>
<td></td>
<td>Flags for Interface iLabel changed to newIfFlags</td>
</tr>
<tr>
<td>Data field</td>
<td>Contains the following:</td>
</tr>
<tr>
<td></td>
<td>• IP address</td>
</tr>
<tr>
<td></td>
<td>• Time stamp</td>
</tr>
<tr>
<td></td>
<td>• Node objid</td>
</tr>
<tr>
<td></td>
<td>• Interface objid</td>
</tr>
<tr>
<td></td>
<td>• NewIfFlags</td>
</tr>
<tr>
<td></td>
<td>• OldIfFlags</td>
</tr>
<tr>
<td>Condition</td>
<td>Event noting change of internal flags for interface. Flags are:</td>
</tr>
<tr>
<td></td>
<td>REMOVED USER_ADDED</td>
</tr>
<tr>
<td></td>
<td>LAYOUT_OFF USE_XY</td>
</tr>
<tr>
<td></td>
<td>CONNECTED_TO_NET</td>
</tr>
<tr>
<td></td>
<td>CONNECTED_TO_SEG</td>
</tr>
<tr>
<td></td>
<td>SEGMENT_HUB</td>
</tr>
<tr>
<td></td>
<td>NOT_CONNECTED</td>
</tr>
<tr>
<td>58720256 CPUL_EV</td>
<td>Descr field</td>
</tr>
<tr>
<td></td>
<td>CPU Load</td>
</tr>
<tr>
<td>Data field</td>
<td>Value for the CPU usage index.</td>
</tr>
<tr>
<td>Condition</td>
<td>If the CPU Load usage index exceeds a specified threshold, then this event gets generated. This is obsolete functionality. The SNMP Data Collector now performs all thresholding.</td>
</tr>
</tbody>
</table>
### Table 30. Tivoli NetView Internal Traps (continued)

<table>
<thead>
<tr>
<th>Number, Name, and Description</th>
<th>Fields and Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>58720257</strong></td>
<td>Descr field</td>
</tr>
<tr>
<td>DSPU_EV</td>
<td>Disk space percentage used <em>string</em></td>
</tr>
<tr>
<td>Disk Space Percentage Used</td>
<td>Data field</td>
</tr>
<tr>
<td></td>
<td>Disk usage</td>
</tr>
<tr>
<td><strong>Condition</strong></td>
<td>If the disk space usage exceeds a specified threshold, then this event gets generated. This is obsolete functionality. The SNMP Data Collector now performs all thresholding.</td>
</tr>
<tr>
<td><strong>58720258</strong></td>
<td>Descr field</td>
</tr>
<tr>
<td>IPD_EV</td>
<td>Interface percent Deferred <em>interfaceAddr</em></td>
</tr>
<tr>
<td>Interface Percent Deferred</td>
<td>Data field</td>
</tr>
<tr>
<td></td>
<td>New value.</td>
</tr>
<tr>
<td><strong>Condition</strong></td>
<td>If the percent packets deferred exceeds a specified threshold, then this event gets generated. This is obsolete functionality. The SNMP Data Collector now performs all thresholding.</td>
</tr>
<tr>
<td><strong>58720259</strong></td>
<td>Descr field</td>
</tr>
<tr>
<td>IPC_EV</td>
<td>Interface Percent Collisions <em>interfaceInetAddr</em></td>
</tr>
<tr>
<td>Interface Percent Collisions</td>
<td>Data field</td>
</tr>
<tr>
<td></td>
<td>New value.</td>
</tr>
<tr>
<td><strong>Condition</strong></td>
<td>If the percent collision packets exceeds a specified threshold, then this event gets generated. This is obsolete functionality. The SNMP Data Collector now performs all thresholding.</td>
</tr>
<tr>
<td><strong>58720260</strong></td>
<td>Descr field</td>
</tr>
<tr>
<td>ICE_EV</td>
<td>Interface CRC Error <em>interfaceInetAddr</em></td>
</tr>
<tr>
<td>Interface CRC Errors</td>
<td>Data field</td>
</tr>
<tr>
<td></td>
<td>New value.</td>
</tr>
<tr>
<td><strong>Condition</strong></td>
<td>If the number of CRC errors exceeds a specified threshold, then this event gets generated. This is obsolete functionality. The SNMP Data Collector now performs all thresholding.</td>
</tr>
<tr>
<td><strong>58720261</strong></td>
<td>Descr field</td>
</tr>
<tr>
<td>IPIE_EV</td>
<td>Interface Percent Input Error <em>interfaceInetAddr</em></td>
</tr>
<tr>
<td>Interface Percent Input Errors</td>
<td>Data field</td>
</tr>
<tr>
<td></td>
<td>New Value</td>
</tr>
<tr>
<td><strong>Condition</strong></td>
<td>If the percent input errors exceeds a specified threshold, then this event gets generated. This is obsolete functionality. The SNMP Data Collector now performs all thresholding.</td>
</tr>
<tr>
<td>Number, Name, and Description</td>
<td>Fields and Condition</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><strong>58720262</strong>&lt;br&gt; IPOE_EV&lt;br&gt; Interface Percent Output Errors</td>
<td><strong>Descr field:</strong> Interface Percent Output Errors interfaceNetAddr&lt;br&gt; <strong>Data field</strong> New value&lt;br&gt; <strong>Condition</strong> If the percent output errors exceeds a specified threshold, then this event gets generated. This is obsolete functionality. The SNMP Data Collector now performs all thresholding.</td>
</tr>
<tr>
<td><strong>58720263</strong>&lt;br&gt; DCOL_EV&lt;br&gt; Data Collector Detected Threshold</td>
<td><strong>Descr field</strong> mibAlias instance threshold exceeded (&gt;thresholdValue): snmpValue&lt;br&gt; <strong>Data field</strong> mibName&lt;br&gt; <strong>Condition</strong> The SNMP Data Collector will generate this event when it detects that a threshold MIB value has been exceeded.</td>
</tr>
<tr>
<td><strong>58720264</strong>&lt;br&gt; DCRA_EV&lt;br&gt; Data Collector re-arm event</td>
<td><strong>Descr field</strong> mibAlias instance threshold rearmed (&lt;=resetValue): snmpValue. Sampled high of highValue at highTime&lt;br&gt; <strong>Data field</strong> mibName&lt;br&gt; <strong>Condition</strong> The SNMP Data Collector will generate this event when it detects that a thresholded MIB variable has descended below the rearm value. The MIB variable is then rearmed for the threshold event, DCOL_EV.</td>
</tr>
<tr>
<td><strong>58785792</strong>&lt;br&gt; IADD_EV&lt;br&gt; Interface Added</td>
<td><strong>Descr field</strong> It can be one of the following strings:&lt;br&gt; • Interface iLabel Added&lt;br&gt; • Connection Added to segmentName&lt;br&gt; • Connection Added&lt;br&gt; <strong>Data field</strong> Contains the following:&lt;br&gt; • IP address&lt;br&gt; • Time stamp&lt;br&gt; • Node and interface objids&lt;br&gt; <strong>Condition</strong> netmon has detected a new interface -- from a successful ping.</td>
</tr>
<tr>
<td>Number, Name, and Description</td>
<td>Fields and Condition</td>
</tr>
<tr>
<td>------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><strong>58785793</strong> IDEL_EV Interface Deleted</td>
<td><strong>Descr field</strong>&lt;br&gt;It can be one of the following strings:&lt;br&gt;• Interface iLabel Deleted&lt;br&gt;• Connection Deleted to segmentName&lt;br&gt;• Connection Deleted&lt;br&gt;<strong>Data field</strong>&lt;br&gt;Contains the following:&lt;br&gt;• IP address&lt;br&gt;• Time stamp&lt;br&gt;• Node and interface objids&lt;br&gt;<strong>Condition</strong>&lt;br&gt;The interface is to be deleted, either from user interaction, or from netmon.</td>
</tr>
<tr>
<td><strong>58785794</strong> NADD_EV Node Added</td>
<td><strong>Descr field</strong>&lt;br&gt;Node Added&lt;br&gt;<strong>Data field</strong>&lt;br&gt;Time stamp and node objid&lt;br&gt;<strong>Condition</strong>&lt;br&gt;The node is added to the topology database.</td>
</tr>
<tr>
<td><strong>58785795</strong> NDEL_EV Node Deleted</td>
<td><strong>Descr field</strong>&lt;br&gt;Node Deleted&lt;br&gt;<strong>Data field</strong>&lt;br&gt;Contains the following:&lt;br&gt;• Time stamp&lt;br&gt;• Node objid&lt;br&gt;<strong>Condition</strong>&lt;br&gt;The node is removed from the topology database.</td>
</tr>
<tr>
<td><strong>58785796</strong> HSRPADD_EV Hot Standby Router Protocol (HSRP) interface added</td>
<td><strong>Descr field</strong>&lt;br&gt;HSPR interface added&lt;br&gt;<strong>Data field</strong>&lt;br&gt;Contains the following:&lt;br&gt;• Time stamp&lt;br&gt;• Node objid&lt;br&gt;<strong>Condition</strong>&lt;br&gt;An HSRP IP address is discovered by netmon.</td>
</tr>
<tr>
<td><strong>58785797</strong> HSRPDEL_EV Hot Standby Router Protocol (HSRP) interface deleted</td>
<td><strong>Descr field</strong>&lt;br&gt;HSPR interface deleted&lt;br&gt;<strong>Data field</strong>&lt;br&gt;Contains the following:&lt;br&gt;• Time stamp&lt;br&gt;• Node objid&lt;br&gt;<strong>Condition</strong>&lt;br&gt;An HSRP IP address is taken over by another router or is no longer being used.</td>
</tr>
<tr>
<td>Number, Name, and Description</td>
<td>Fields and Condition</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------------</td>
</tr>
</tbody>
</table>
| 58851329 ERR_EV Non-Fatal errors | Most of these are internal debugging messages. The "Agent in distress" messages have been seen with misbehaving agents. **Descr field** It can be one of the following strings. The number here corresponds with the number under the Condition field:  
1. ERROR Internal Error - object of bad type objectType detected in addObj  
2. ERROR Internal Error - object of bad type objectType detected in remove  
3. ERROR Internal Error - object of bad type objectType detected in delete  
4. ERROR Could not create directory directory: errorNumber  
5. ERROR Could not create main database directory: errorNumber  
6. ERROR Could not allocate object ID for interface ifName  
7. ERROR Could not allocate object ID for net netName  
8. ERROR Could not allocate Topo Object ID for Node nodeName  
9. ERROR Unable to format time in formatDBtime, return value = retVal  
10. ERROR Could not look up field id for field ovwNselectionName  
11. ERROR Could not get unique object name for baseName, ovw_error = ErrorString |
| 58851329 ERR_EV Non-Fatal errors (continued) | **Descr field (continued)** It can be one of the following strings. The number here corresponds with the number under the Condition field:  
12. ERROR Could not allocate nor lookup object ID for objectName  
13. ERROR Internal Error: too many fields for fieldBindList!  
14. ERROR SetFieldValues returned returnedValue for obj objectId, OVwError = errorNumber: errorString.  
15. ERROR Too many fields to unset, skipping extra  
16. ERROR Could not look up field id for name ovwFIPAddress  
17. ERROR Too many values for ListFieldValue  
18. ERROR Could not look up field id for name field ovwFIPNetworkName  
19. ERROR Could not look up field id for name field ovwFIPHostName  
20. ERROR delNetworkCmd: Error removing net netName from topology: topoErrString  
21. ERROR delNetworkCmd: Error removing seg segName from topology: topoErrString  
22. ERROR Demand poll: cannot stat pipe pipename  
23. ERROR Demand poll: pipename not a pipe |
| 58851329 ERR_EV Non-Fatal errors (continued) | **Descr field (continued)** It can be one of the following strings. The number here corresponds with the number under the Condition field:  
24. ERROR Pipe open failure pipename (errno = errorNumber)  
25. ERROR Pipe fdopen failure pipename (errno = errorNumber)  
26. ERROR netmon stopped -- dumping list  
27. ERROR Fatal error in placeOfError logged in trace file - exiting  
28. ERROR netmon: stopping in timeout_snmps  
29. ERROR Agent in distress: spinning in ipRouteTable on nodeName: Aborting request!  
30. ERROR Agent in distress: spinning in ifTable on nodeName: Aborting request!  
31. ERROR Agent in distress: spinning in ipAddrTable on nodeName: Aborting request! |
## Table 30. Tivoli NetView Internal Traps (continued)

<table>
<thead>
<tr>
<th>Number, Name, and Description</th>
<th>Fields and Condition</th>
</tr>
</thead>
</table>
| **58851329** | **Data field**  
NULL  

**Condition**  
Following are the conditions:  
1. The program enters in default case in the function addObjToTopo().  
2. The program enters in default case in the function removeObjFromTopo().  
3. The program enters in default case in the function deleteObjFromTopo().  
4. Creation of directory fails.  
6. Creation of object ID for an interface fails.  
7. Creation of object ID for a network fails.  
8. Creation of object ID for a node fails.  
9. Conversion of time format between system and INGRES/SQL formats failed.  
10. Failed to get the field id from selection name.  
11. Failed to get unique object name from selection name ID and base name.  
12. Either failed to create object id or failed to get an object ID for an already existing object.  

| **58851329 (continued)** | **Data field**  
NULL  

**Condition (continued)**  
Following are the conditions:  
13. An attempt is made to increase the number of fields in the bind list when it is already full.  
14. Unable to set the value of the fields in the bind list.  
15. The number of maximum topology specific fields which an object can have is MAX_UNSET_FIDS. During cleanup if more than 50 fields are found this event is sent.  
16. Failed to convert a field name to field ID.  
17. When an attempt is made to add another value to a field which is of list type and already the field has maximum number of permissible values.  
18. Failed to get field ID from field name.  
19. Failed to get field ID from field name.  
20. Failed to remove network from topology.  
21. Failed to remove segment from topology.  
22. ‘Stat’ call fails on pipe.  
23. A pipe descriptor was expected but it is not a pipe.  
24. Failed to create a pipe.  

<table>
<thead>
<tr>
<th>Number, Name, and Description</th>
<th>Fields and Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>58851329 (continued)</strong></td>
<td>Data field</td>
</tr>
<tr>
<td></td>
<td>NULL</td>
</tr>
<tr>
<td><strong>Condition (continued)</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Following are the conditions:</td>
</tr>
<tr>
<td></td>
<td>25. Failed to open a pipe (fdopen call failed).</td>
</tr>
<tr>
<td></td>
<td>26. Core dump has occurred while running netmon.</td>
</tr>
<tr>
<td></td>
<td>27. Fatal error has occurred.</td>
</tr>
<tr>
<td></td>
<td>28. The SNMP wait queue has nodes to process, yet the node contains no state information.</td>
</tr>
<tr>
<td></td>
<td>29. The specified node is spinning in its ipRouteTable. Aborting requests.</td>
</tr>
<tr>
<td></td>
<td>30. The specified node is spinning in its ifTable. Aborting requests to the node.</td>
</tr>
<tr>
<td></td>
<td>31. The specified node is spinning in its ipAddrTable. Aborting requests.</td>
</tr>
<tr>
<td><strong>58851330</strong></td>
<td>Descr field</td>
</tr>
<tr>
<td>FERR_EV</td>
<td>It can be one of the following strings. The number here corresponds with the number under the Condition field.</td>
</tr>
<tr>
<td>Fatal Errors</td>
<td>1. FATAL ERROR Out of memory -- exiting.</td>
</tr>
<tr>
<td></td>
<td>2. FATAL ERROR Could not map field fieldName into OVwFieldId</td>
</tr>
<tr>
<td></td>
<td>3. FATAL ERROR&gt;&quot;Node/Iface or memory allocation error in where&quot; - exiting</td>
</tr>
<tr>
<td></td>
<td>4. Netmon probably died: ungracefully disconnected from trapd</td>
</tr>
<tr>
<td></td>
<td>5. snmpCollect probably died: ungracefully disconnected from trapd</td>
</tr>
<tr>
<td></td>
<td>6. topmd probably died: ungracefully disconnected from trapd</td>
</tr>
<tr>
<td></td>
<td>7. applicationName reached maximum number of outstanding events, disconnecting from trapd.</td>
</tr>
<tr>
<td></td>
<td>8. Application reached maximum number of outstanding events, disconnecting from trapd.</td>
</tr>
<tr>
<td><strong>Data field</strong></td>
<td>NULL</td>
</tr>
<tr>
<td><strong>Condition</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Following are the conditions:</td>
</tr>
<tr>
<td></td>
<td>1. When allocation of dynamic memory fails.</td>
</tr>
<tr>
<td></td>
<td>2. Failed to convert a list of field names into corresponding field IDs.</td>
</tr>
<tr>
<td></td>
<td>3. System ran out of memory.</td>
</tr>
<tr>
<td></td>
<td>4. trapd has detected that the connection to netmon has closed, but netmon did not send a close_event.</td>
</tr>
<tr>
<td></td>
<td>5. trapd has detected that the connection to snmpCollect has closed, but snmpCollect did not send a close_event.</td>
</tr>
<tr>
<td><strong>58851330 (continued)</strong></td>
<td>Condition (continued)</td>
</tr>
<tr>
<td>FERR_EV</td>
<td>Following are additional conditions:</td>
</tr>
<tr>
<td>Fatal Errors</td>
<td>6. trapd has detected that the connection to ovtopmd has closed, but ovtopmd did not send a close_event.</td>
</tr>
<tr>
<td></td>
<td>7. trapd is closing the connection to applicationName because trapd has reached the maximum number of outstanding events that it will queue up.</td>
</tr>
<tr>
<td></td>
<td>8. trapd is closing the connection to the application, because trapd has queue up the maximum number of events for the application.</td>
</tr>
<tr>
<td>Number, Name, and Description</td>
<td>Fields and Condition</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><strong>58916864</strong>&lt;br&gt;NUP_EV&lt;br&gt;Node Up</td>
<td><strong>Descr field</strong>&lt;br&gt;Node Up&lt;br&gt;<strong>Data field</strong>&lt;br&gt;Contains the following:&lt;br&gt;• Time stamp&lt;br&gt;• Object ID&lt;br&gt;<strong>Condition</strong>&lt;br&gt;All detected interfaces for the node are up.</td>
</tr>
<tr>
<td><strong>58916865</strong>&lt;br&gt;NDWN_EV&lt;br&gt;Node Down</td>
<td><strong>Descr field</strong>&lt;br&gt;Node Down&lt;br&gt;<strong>Data field</strong>&lt;br&gt;Contains the following:&lt;br&gt;• Time stamp&lt;br&gt;• Object ID&lt;br&gt;<strong>Condition</strong>&lt;br&gt;All detected interfaces for the node are down.</td>
</tr>
<tr>
<td><strong>58916866</strong>&lt;br&gt;IUP_EV&lt;br&gt;Interface Up</td>
<td><strong>Descr field</strong>&lt;br&gt;Interface <code>interfaceLabel</code> up.&lt;br&gt;<strong>Data field</strong>&lt;br&gt;IP address, the time stamp of the event, the node and interface ids.&lt;br&gt;<strong>Condition</strong>&lt;br&gt;An interface that was previously down has responded to a ping.</td>
</tr>
<tr>
<td><strong>58916867</strong>&lt;br&gt;IDWN_EV&lt;br&gt;Interface Down</td>
<td><strong>Descr field</strong>&lt;br&gt;Interface <code>interfaceLabel</code> Down.&lt;br&gt;<strong>Data field</strong>&lt;br&gt;IP address, the time stamp of the event, the node and interface ids.&lt;br&gt;<strong>Condition</strong>&lt;br&gt;An interface that was previously up is not responding to a ping.</td>
</tr>
<tr>
<td><strong>58916868</strong>&lt;br&gt;SC_EV&lt;br&gt;Segment Critical</td>
<td><strong>Descr field</strong>&lt;br&gt;Segment <code>segmentId</code> Down&lt;br&gt;<strong>Data field</strong>&lt;br&gt;Contains the following:&lt;br&gt;• Time stamp&lt;br&gt;• Segment object ID&lt;br&gt;<strong>Condition</strong>&lt;br&gt;All nodes within a segment are critical.</td>
</tr>
<tr>
<td><strong>58916869</strong>&lt;br&gt;NC_EV&lt;br&gt;Network Critical</td>
<td><strong>Descr field</strong>&lt;br&gt;Network <code>netName</code> Down&lt;br&gt;<strong>Data field</strong>&lt;br&gt;Time stamp of the event and network objid&lt;br&gt;<strong>Condition</strong>&lt;br&gt;All segments for the network are critical.</td>
</tr>
<tr>
<td>Number</td>
<td>Name</td>
</tr>
<tr>
<td>------------</td>
<td>------------</td>
</tr>
<tr>
<td>58916871</td>
<td>SNMP_EV</td>
</tr>
<tr>
<td>58916872</td>
<td>SUGUP_EV</td>
</tr>
<tr>
<td>58916873</td>
<td>SUGDN_EV</td>
</tr>
<tr>
<td>58916964</td>
<td>NV6KUP_EV</td>
</tr>
<tr>
<td>58916965</td>
<td>NV6KDN_EV</td>
</tr>
<tr>
<td>Number, Name, and Description</td>
<td>Fields and Condition</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td><strong>58982400</strong> LLAC_EV</td>
<td><strong>Descr field</strong></td>
</tr>
<tr>
<td>Link Level Address Changed</td>
<td>Link Address For <code>interfaceLabel</code> Changed to <code>physicalAddress</code></td>
</tr>
<tr>
<td></td>
<td><strong>Data field</strong></td>
</tr>
<tr>
<td></td>
<td>IP address, the time stamp of the event, node and interface objids</td>
</tr>
<tr>
<td></td>
<td><strong>Condition</strong></td>
</tr>
<tr>
<td></td>
<td>The product has detected that <code>interfaceLabel</code> has a new physical Address.</td>
</tr>
</tbody>
</table>

| **58982401** MLLA_EV         | **Descr field**      |
| Mismatch of Link Level Address| `nodeName1` reported different Link Address than obtained from `nodeName2` by SNMP |
|                               | **Data field**       |
|                               | `nodeName` of the node that reported different address |
|                               | **Condition**        |
|                               | Tivoli NetView has discovered a discrepancy in the LLA being reported by 2 different nodes. `nodeName1`’s `ifTable` is being interrogated for LLA vs. the information in `nodeName2`’s ARP cache. This can be diagnosed by using `rnetstat -I nodeName1` to determine the interfaces and LLA for `nodeName1`, and then issue `rnetstat -A nodeName2 | grep nodeName1` to determine the `nodeName2` ARP entry for `nodeName1`. |
|                               | It is possible that the information stored in the topology database for `nodeName1` is out of date. If so, performing a demand poll on `nodeName1` should resolve the situation. The command: `ovtopodump -L | grep nodeName1` could also be performed to check the LLA that Tivoli NetView currently has associated with the interfaces for `nodeName1`. |
|                               | Some vendors make wide assumptions about documenting information in the ARP cache, thus there are many reasons why a mismatch may occur, but the network configuration is still valid. Netmon supports a flag to disable this check completely. To disable, select the option `ignore` for Ring bit-swapping storage flag through the Tivoli desktop Set Options for netmon daemon job. |

<p>| <strong>58982402</strong> ULLA_EV         | <strong>Descr field</strong>      |
| Undetermined Link Level Address| One of the following description fields is possible with this trap: |
|                               | • If there are two nodes that have reported different LLA address then the message displayed is &quot;<code>nodeName</code> reports a different Link Address for this node from that reported by <code>secondNodeName</code>&quot; |
|                               | • If the same node reports a different LLA address then what is reported earlier then the message displayed is &quot;<code>nodeName</code> reports a different Link Address for this node than it reported earlier&quot;. |
|                               | <strong>Data field</strong>       |
|                               | <code>nodeName</code>, <code>secondNodeName</code> |
|                               | <code>nodeName</code>           |
|                               | The name of the node which has reported an LLA address which is different from the one which this (reporting) node got earlier either from the same node (<code>nodeName = secondNodeName</code>) or from a different node (<code>nodeName != secondNodeName</code>) |
|                               | <code>secondNodeName</code>    |
|                               | Name of the node which had earlier reported LLA address. |</p>
<table>
<thead>
<tr>
<th>Number, Name, and Description</th>
<th>Fields and Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>58982402</strong> ULLA_EV</td>
<td><strong>Condition</strong> NodeName gives LLA address for interface which is different than we obtained in the past from the IP Address. If this message keeps repeating, it indicates that the LLA for interface keeps flopping; this usually means there is more than one interface with the same IP address. If it happens once or twice and settles down, it usually means that a LAN card has changed. There might be valid network reasons why the ULLA_EV is being displayed. The same Tivoli desktop option described above to disable the MLLA_EV would disable the ULLA_EV.</td>
</tr>
<tr>
<td><strong>58982403</strong> OIC_EV</td>
<td><strong>Descr field</strong> Object Identifier changed to newObjectId <strong>Data field</strong> Time stamp, node object ID, new sysObject ID <strong>Condition</strong> A new sysObjectId was detected</td>
</tr>
<tr>
<td><strong>58982404</strong> SDC_EV</td>
<td><strong>Descr field</strong> System Description Changed to sysDescr <strong>Data field</strong> Contains the following: • Time stamp • Node object ID • New sysDescr <strong>Condition</strong> A new sysDescr was detected</td>
</tr>
<tr>
<td><strong>58982405</strong> SNC_EV</td>
<td><strong>Descr field</strong> System Name changed <strong>Data field</strong> Contains the following: • Time stamp • Node object ID • New sysName <strong>Condition</strong> A new sysName was detected</td>
</tr>
<tr>
<td><strong>58982406</strong> SMC_EV</td>
<td><strong>Descr field</strong> Network Mask for interfaceLabel changed to inetAddr <strong>Data field</strong> IP address, the time stamp of the event, node id, interface id and interface type. <strong>Condition</strong> A new subnet mask for an interface was detected</td>
</tr>
<tr>
<td>Number, Name, and Description</td>
<td>Fields and Condition</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>----------------------</td>
</tr>
</tbody>
</table>
| **58982407**  
FSC\_EV  
Forwarding Status Change | **Descr field**  
It can be one of the following fields:  
- Forwarding status changed -- Now a Gateway  
- Forwarding status changed -- Now a host  
- Forwarding status changed -- Now Unknown  
**Data field**  
Contains the following:  
- Time stamp  
- Node object ID  
- 1 if a gateway and 2 if a host  
**Condition**  
A change was detected in a node’s ipForwarding status. |
| **58982408**  
FTH\_EV  
Forwarding to a host | **Descr field**  
Incorrect Routing to node *nodeName*  
**Data field**  
nnodeName  
**Condition**  
The node for the event contains a route in its routing table to nodeName, yet our topology database entry for nodeName indicates that its ipForwarding status is not a gateway. This could be corrected via a demand poll to nodeName which would update its ipForwarding status in the database. This event could also be generated if nodeName is not allowing SNMP communication to the manager. In this case, the manager cannot determine the ipForwarding status for nodeName. |
| **58982410**  
SCC\_EV  
System Contact Change | **Descr field**  
System Contact Changed to *contactName*  
**Data field**  
Contains the following:  
- Time stamp  
- Node object ID  
- New system contact  
**Condition**  
Detected a new sysContact for the node. |
| **58982411**  
SLC\_EV  
System Location Change | **Descr field**  
System Location Changed to *nodeLocation*  
**Data field**  
Contains the following:  
- Time stamp  
- Node object ID  
- New system location  
**Condition**  
Detected a new sysLocation for the node. |
<table>
<thead>
<tr>
<th>Number, Name, and Description</th>
<th>Fields and Condition</th>
</tr>
</thead>
</table>
| 58982412 ITC_EV Interface Type Change | Descr field: Interface Type for **interfaceLabel** changed to **typeName**  
Data field: IP address, time stamp, node id, interface ids and interface type.  
Condition: Detected a new interface type for an interface. |
| 58982413 IDC_EV Interface Description Change | Descr field: Interface Descriptor for **interfaceLabel** changed to **interfaceName**  
Data field: IP address, time stamp, node id, interface ids and interface description.  
Condition: Detected a new interface description for an interface. |
| 58982414 BSM_EV Bad Subnet Mask | Descr field: Inconsistent subnet mask **subnetMask** on interface **inetAddr**  
Data field: Bad interface address and bad subnet mask  
Condition: The subnet mask obtained for an IP address does not match the assumed subnet mask for the subnet in which that interface resides. |
| 59047936 AA_EV Application Alert | Descr field: **alertapp**: classStr: **alertMsg**  
**alertApp**  
The application name which has sent the alert  
**classStr**  
Can take one of the following values:  
INFORMATION WARNING ERROR DISASTER  
**alertMsg**  
The Alert message sent by the application  
Data field: NULL  
Condition: When an application makes use of the OVwAlertMsg API to send a message. |
| 59179056 APUP_EV Application Up event | Descr field: **applicationName** connected to trapd  
Data field: NULL  
Condition: When an application gets connected to trapd via its internal socket API. |
<table>
<thead>
<tr>
<th>Number, Name, and Description</th>
<th>Fields and Condition</th>
</tr>
</thead>
</table>
| 59179057 APDN_EV Application Down Event | **Descr field**  
applicationName disconnecting from trapd  
**Data field**  
NULL  
**Condition**  
When an application disconnects from trapd, this event is generated. |
| 59179068 TATM_EV Tralerld Change Tracemask Event | **Descr field**  
Tralerld trace mask change request  
**Data field**  
tracemask  
**Condition**  
The tralerld daemon uses this for on-the-fly changing of its tracemask. |
| 59179070 NMCR_EV Netmon Change Retry Count Event | **Descr field**  
Netmon retry count changed  
**Data field**  
retry count value  
**Condition**  
Generated when the netmon option to change the retry count is issued. This event is obsolete with the retries allowed via the ovsnmp.conf file. |
| 59179072 NVCOLD_UP nvcold Initialization Event | **Descr field**  
nvcold initialization complete  
**Data field**  
NULL  
**Condition**  
Generated when the nvcold daemon completes its initialization phase. |
| 70000030 NVOT001 Vertex Created | **Descr field**  
Can be one of the following values:  
• namebinding(int)  
• protocol(int)  
• name(string)  
• detailsld(string)  
**Condition**  
Creation of a vertex in gtmd. |
| 70000031 NVOT002 Vertex Deleted | **Descr field**  
Can be one of the following values:  
• namebinding(int)  
• protocol(int)  
• name(string)  
**Condition**  
Deletion of a vertex in gtmd. |
<table>
<thead>
<tr>
<th>Number, Name, and Description</th>
<th>Fields and Condition</th>
</tr>
</thead>
</table>
| **70000032** NVOT003 Vertex Status Changed | **Descr field** Can be one of the following values:  
  - namebinding(int)  
  - protocol(int)  
  - name(string)  
  **Condition** Status change of a vertex in gtmd. |
| **70000033** NVOT004 Graph Created | **Descr field** Can be one of the following values:  
  - namebinding(int)  
  - protocol(int)  
  - name(string)  
  - detailsld(string)  
  **Condition** Creation of a graph or box in gtmd. |
| **70000034** NVOT005 Graph Deleted | **Descr field** Can be one of the following values:  
  - namebinding(int)  
  - protocol(int)  
  - name(string)  
  **Condition** Deletion of a graph or box in gtmd. |
| **70000035** NVOT006 Arc Created | **Descr field** Can be one of the following values:  
  - namebinding(int)  
  - Aprotocol(int)  
  - Aname(string)  
  - Zprotocol(int/str)  
  - Zname(string)  
  - arcindexld(int)  
  - detailsld(string)  
  **Condition** Creation of an arc in gtmd. |
| **70000036** NVOT007 Arc Deleted | **Descr field** Can be one of the following values:  
  - namebinding(int)  
  - Aprotocol(int)  
  - Aname(string)  
  - Zprotocol(int/str)  
  - Zname(string)  
  - arcindexld(int)  
  **Condition** Deletion of an arc in gtmd. |
Table 30. Tivoli NetView Internal Traps (continued)

<table>
<thead>
<tr>
<th>Number, Name, and Description</th>
<th>Fields and Condition</th>
</tr>
</thead>
</table>
| 70000037 NVOT008 Arc Status Changed | **Descr field**

  - Can be one of the following values:
    - `namebinding(int)`
    - `Aprotocol(int)`
    - `Aname(string)`
    - `Zprotocol(int/str)`
    - `Zname(string)`
    - `arcindexid(int)`

**Condition**

Status change of an arc in gtmd.
Appendix B. Additional Copyright and License Information

The product described in this document also contains software downloaded from several web servers. Permission to download and use such software is conditioned upon inclusion of the following notices.

gd 1.2 © Copyright 1994, 1995, Quest Protein Database Center, Cold Spring Harbor Labs. Permission granted to copy and distribute this work provided that this notice remains intact. Credit for the library must be given to the Quest Protein Database Center, Cold Spring Harbor Labs, in all derived works. This does not affect your ownership of the derived work itself, and the intent is to assure proper credit for Quest, not to interfere with your use of gd. If you have questions, ask. (“Derived works” includes all programs that utilize the library. Credit must be given in user-visible documentation.)

gd 1.2 was written by Thomas Boutell and is currently distributed by boutell.com, Inc.

If you wish to release modifications to gd, please clear them first by sending email to boutell@boutell.com; if this is not done, any modified version of the gd library must be clearly labeled as such.

The Quest Protein Database Center is funded under Grant P41-RR02188 by the National Institutes of Health.

Written by Thomas Boutell, 2/94–8/95.

The GIF compression code is based on that found in the pbmplus utilities, which in turn is based on GIFENCOD by David Rowley. See the notice below:

Based on GIFENCOD by David Rowley. A Lemple-Ziv compression based on “compress”.

Modified by Marcel Wijkstra.

Copyright © 1989 by Jef Poskanzer.

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appears in all copies and that both that copyright notice and this permission appear in all supporting documentation. This software is provided “as is” without express or implied warranty.

The Graphics Interchange Format © is the Copyright property of CompuServe Incorporated. GIF (sm) is a Service Mark property of CompuServe Incorporated.

The GIF decompression is based on that found in the pbmplus utilities, which in turn is based on GIFDECOD by David Koblas. See the notice below:


Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above
copyright notice appears in all copies and that both that copyright notice and this
permission appear in all supporting documentation. This software is provided “as is”
without express or implied warranty.

GIFtrans v1.12

Convert any GIF file into a GIF89a. Allows for setting the transparent or background
color, changing colors, adding or removing comments. Also code to analyze GIF
contents.

Copyright © 24.2.94 by Andreas Ley

Permission to use, copy, modify, and distribute this software for any purpose and
without fee is hereby granted, provided that the above copyright notice appears in
all copies. This software is provided “as is” without express or implied warranties.
This glossary defines technical terms used in the documentation for Tivoli products and includes selected terms and definitions from:

- The *American National Standard Dictionary for Information Systems*, ANSI X3.172-1990, copyright 1990 by the American National Standards Institute (ANSI). Copies may be purchased from the American National Standards Institute, 11 West 42nd Street, New York, New York 10036. Definitions are identified by the symbol (A) after the definition.
- The ANSI/EIA Standard—440-A, *Fiber Optic Terminology*. Copies may be purchased from the Electronic Industries Association, 2001 Pennsylvania Avenue, N.W., Washington, DC 20006. Definitions are identified by the symbol (E) after the definition.
- The *Information Technology Vocabulary* developed by Subcommittee 1, Joint Technical Committee 1, of the International Organization for Standardization and the International Electrotechnical Commission (ISO/IEC JTC1/SC1). Definitions of published parts of this vocabulary are identified by the symbol (I) after the definition; definitions taken from draft international standards, committee drafts, and working papers being developed by ISO/IEC JTC1/SC1 are identified by the symbol (T) after the definition, indicating that final agreement has not yet been reached among the participating National Bodies of SC1.
- Internet Request for Comments: 1208, *Glossary of Networking Terms*
- Internet Request for Comments: 1392, *Internet Users’ Glossary*

The following cross-references are used in this glossary:

**Contrast with:**
This refers the reader to a term that has an opposed or substantively different meaning.

**See:**
This refers the reader to (a) a related term, (b) a term that is the expanded form of an abbreviation or acronym, or (c) a synonym or more preferred term.

**Obsolete term for:**
This indicates that the term should not be used and refers the reader to the preferred term.

**A**

**ABAP/4.** See *Advanced Business Application Programming/I*.

**absolute path.** A path that begins with the root directory. The absolute path may also be known as the “full pathname.” Contrast with *relative path*.

**abstract model.** In Tivoli Global Enterprise Manager, the business description files that logically describe a particular business system.

**abstract syntax notation 1 (ASN.1).** The Open Systems Interconnection (OSI) method for abstract syntax specified in the following standards:

**accelerator.** (1) In a user interface, a key or combination of keys that invokes an application-defined function. (2) In the AIX® Toolkit, a keyboard alternative to a mouse button action; for example, holding the <Shift> and <M> keys on the keyboard can be made to post a menu in the same way that a mouse button action does. Accelerators typically provide increased input speed and greater convenience.

**access control.** In computer security, the process of ensuring that the resources of a computer system can be accessed only by authorized users in authorized ways.

**access control list.** (1) In computer security, a collection of all access rights for one object. (2) In computer security, a list associated with an object that identifies all the subjects that can access the object and their access rights; for example, a list associated with a file that identifies users who can access the file and identifies their access rights to that file.

**ACF.** See *Adapter Configuration Facility*.

**action.** (1) An operation on a managed object, the semantics of which are defined as part of the managed object class definition. (2) In the AIX® operating system,
a defined task that an application performs. An action modifies the properties of an object or manipulates the object in some way.

**Action Message Retention Facility (AMRF).** An OS/390 facility that, when active, retains all action messages except those specified by the installation.

**adapter.** (1) A part that electrically or physically connects a device to a computer or to another device. (2) Software that enables different software components or products to interact with one another. (3) See event adapter.

**Adapter Configuration Facility (ACF).** In the Tivoli Enterprise Console®, a graphical user interface that enables a Tivoli administrator to easily configure and customize event adapters.

**ADE.** See Tivoli Application Development Environment.

**ADF.** See application description file.

**Administrative Domain.** A collection of hosts and routers, and the interconnecting networks, managed by a single administrative authority.

**administrator.** See Tivoli administrator.

**administrator collection.** In a Tivoli environment, the collection for administrator objects that is generated by Tivoli Enterprise™ software. This container is represented by the Administrator icon on the Tivoli desktop; opening the icon provides access to information about each Tivoli administrator.

**admin role.** See authorization role.

**Advanced Business Application Programming/4 (ABAP/4).** A fourth-generation programming language in which SAP R/3 application software is written.

**AEF.** See Tivoli Application Extension Facility.

**agent.** (1) In systems management, a user that, for a particular interaction, has assumed an agent role. (2) An entity that represents one or more managed objects by (a) emitting notifications regarding the objects and (b) handling requests from managers for management operations to modify or query the objects. (3) A system that assumes an agent role.

**Agent Policy Manager (APM).** In Tivoli NetView, a function that controls Mid-Level Manager (MLM) configurations in a network from a single, central location.

**agent role.** In systems management, a role assumed by a user in which the user is capable of performing management operations on managed objects and of emitting notifications on behalf of managed objects.

**aggregate object.** In the NetView Graphic Monitor Facility, an object that represents a collection of real objects.

**AIXwindows Toolkit.** An object-oriented collection of C language data structures and subroutines that supplement the Enhanced X-Windows Toolkit and simplify the creation of interactive client application interfaces.

**alarm.** A signal, either audible or visual, at a device such as a display station or printer that is used to notify the user that a condition requiring the user's attention exists.

**alarm level.** In Tivoli Distributed Monitoring, the state of a monitor when a specified threshold has been reached. A Tivoli administrator can set thresholds for each alarm level and have Tivoli Distributed Monitoring trigger a different response (an action and an event) for each level. There can also be several responses for each alarm level.

**alert.** (1) A message sent to a management services focal point in a network to identify a problem or an impending problem. (2) In SNA management services (SNA/MS), a high priority event that warrants immediate attention.

**alias name.** A name that is defined in one network to represent a logical unit name in another interconnected network. The alias name does not have to be the same as the real name; if these names are not the same, translation is required.

**alias name translation facility.** In Tivoli NetView for OS/390, a function for converting logical unit names, logon mode table names, and class-of-service names used in one network into equivalent names to be used in another network.

**allomorphism.** The ability of an instance of a class to be managed as an instance of one or more different but compatible managed object classes.

**AMP.** See application management package.

**AMRF.** See Action Message Retention Facility.

**AMS.** See Application Management Specification.

**AOF.** See application object file.

**AON.** See Automated Operations Network.

**APAR.** See authorized program analysis report.

**API.** See application programming interface.

**APM.** See Agent Policy Manager.

**application.** A collection of software components used to perform specific types of user-oriented work on a computer.
**application description file (ADF).** In the context of the Application Management Specification (AMS), a readable, ASCII text file that contains information for managing an application. Application description files are based on the Management Information Format (MIF). Application description files include component description files, global description files, and business description files (business system description files, business system component description files, business system mapping description files, and business subsystem description files).

**application management package (AMP).** In a Tivoli environment, a compressed file that contains the application description files and other necessary files for managing an application. These include one global description file, one or more component description files, task scripts, and executable programs. The application management package can also include the application object file or the source files for the application itself.

**Application Management Specification (AMS).** A specification that presents a standard for managing applications. The Application Management Specification was developed in collaboration with the Tivoli Partners and Tivoli customers to address the problems associated with multitiered applications.

**application object file (AOF).** In a Tivoli environment, an ASCII text file that contains the names of the global description file and the component description files, which together describe the management characteristics of an application. The Tivoli Module Designer and the Tivoli Module Builder can import an application object file that was created by the obsolete Tivoli Developer Kit.

**application plane.** In Tivoli NetView, the submap layer on which symbols of objects that are managed by at least one network or systems management application program are displayed. Symbols on the application plane are displayed without shading, which makes them appear directly against the background plane. See used plane.

**application programming interface (API).** A software interface that enables applications to communicate with each other. An API is the set of programming language constructs or statements that can be coded in an application program to obtain the specific functions and services provided by an underlying operating system or service program.

**application registration file (ARF).** A file created to integrate an application program into Tivoli NetView by defining (a) the application program’s position in the menu structure for Tivoli NetView, (b) where help information is found, (c) the number and types of parameters allowed, (d) the command used to start the application program, and (e) other characteristics of the application program.

**Application Response Measurement (ARM).** An application programming interface that was developed by a group of leading technology vendors, including Tivoli Systems Inc., and that can be used to monitor the availability and performance of business transactions within and across diverse applications and systems. The monitoring is done from the perspective of the applications; therefore, it reflects the units of work that are important from the perspective of the business. For example, using ARM, a business could instrument an application to discover:

- Whether the application is hung
- The level of response time that the application is experiencing
- Where the bottlenecks are occurring during the execution of the application
- Who is using the application and how much they are using it
- How to tune the system environment to run the application more efficiently
- What the application is doing during the reported response time
- Where in the system environment a transaction is spending its time

**APPNTAM.** See SNA topology manager.

**APPN® Topology and Accounting Manager (APPNTAM).** See SNA topology manager.

**ARF.** See application registration file.

**ARM.** (1) See Application Response Measurement. (2) See automatic restart managed.

**ARM agent.** An agent that monitors software that is instrumented using the Application Response Measurement (ARM). The ARM agent is shipped as part of Tivoli Distributed Monitoring.

**ASN.1.** See abstract syntax notation 1.

**ASYNC.** See asynchronous.

**asynchronous (ASYNC).** (1) Pertaining to two or more processes that do not depend upon the occurrence of specific events such as common timing signals. (T) (2) Without regular time relationship; unexpected or unpredictable with respect to the execution of program instructions.

**asynchronous monitor.** In Tivoli Distributed Monitoring, a monitor that receives data in an unsolicited event and interprets the data immediately. Contrast with synchronous monitor.

**attribute.** A characteristic that identifies and describes a managed object. The characteristic can be determined, and possibly changed, through operations on the managed object.
authentication. (1) In computer security, verification of the identity of a user or the user’s eligibility to access an object. (2) In computer security, verification that a message has not been altered or corrupted. (3) In computer security, a process used to verify the user of an information system or protected resources.

authorization. (1) In computer security, the right granted to a user to communicate with or make use of a computer system. (T) (2) An access right. (3) The process of granting a user either complete or restricted access to an object, resource, or function.

authorization role. In a Tivoli environment, a role assigned to Tivoli administrators to enable them to perform their assigned systems management tasks. A role may be granted over the entire Tivoli Management Region or over a specific set of resources, such as those contained in a policy region. Examples of authorization roles include: super, senior, admin, and user.

authorized operator. In Tivoli NetView for OS/390, an operator who has been authorized to receive undeliverable messages and lost terminal messages. See authorized receiver.

authorized program analysis report (APAR). A report of a problem caused by a suspected defect in a current unaltered release of a program.

authorized receiver. In Tivoli NetView for OS/390, an authorized operator who receives the unsolicited and authorized-receiver messages that are not assigned to a specific operator.

Automated Operations Network (AON). In Tivoli NetView for OS/390, the component that handles automated resource monitoring, recovery, and tracking.

automated response. In a Tivoli environment, a predefined response to particular events that is automatically executed by a Tivoli application. For example, if the Tivoli Enterprise Console detects that a process has terminated early, it can automatically restart the process without the intervention of the Tivoli administrator.

automatic reactivation. In Tivoli NetView for OS/390, the activation of a node from the inactive state without any action by the network operator.

automatic restart manager (ARM). An OS/390 recovery function that can automatically restart batch jobs and started tasks after they or the system on which they are running terminate unexpectedly.

AutoPack Control Center. A Tivoli Software Distribution tool that is installed on a Windows-based PC and enables a Tivoli administrator to create an AutoPack file. The AutoPack Control Center produces the AutoPack file by (a) taking snapshots of the PC’s drive and system configuration before and after the installation of an application on the PC and (b) capturing the differences between these snapshots and the distribution instructions in the AutoPack file.

AutoPack file. In Tivoli Software Distribution, an installable image that is used to distribute “shrinkwrapped” applications to multiple PC targets. The file contains a description of PC software application files and directories, information on how to distribute these files and directories, and any system configuration changes needed by the application. A Tivoli administrator must associate an AutoPack file with an AutoPack profile.


autotask. (1) In Tivoli NetView for OS/390, an unattended operator station task that does not require a terminal or a logged-on user. Autotasks can run independently of VTAM® and are typically used for automated console operations. (2) Contrast with logged-on operator.

availability management. The Tivoli management discipline that addresses the gathering, collecting, and routing of information regarding the operational status of an organization’s network computing system and enables the appropriate corrective action. See deployment management, operations and administration, and security management.

B

backend. In the AIX operating system, the program that sends output to a particular device.

background plane. In Tivoli NetView, the lowest submap layer. The background plane provides the background against which symbols are displayed. A background picture can be placed in the background plane to provide a context for viewing symbols. See application plane and user plane.

background process. (1) A process that does not require operator intervention but can be run by the computer while the workstation is used to do other work. (2) In the AIX operating system, a mode of program execution in which the shell does not wait for program completion before prompting the user for another command. (3) Contrast with foreground process.

background task. A task that is running even though the user is not currently interacting with it. Contrast with foreground task.

bandwidth. A measure of the capacity of a communication transport medium (such as a TV cable) to convey data.
BARC program. Obsolete term for configuration program. "BARC" is an acronym for "before, after, removal, and commit."

BAROC. See Basic Recorder of Objects in C

base module. In a Tivoli environment, a management module that describes the basic management characteristics of a particular application or business system to the Tivoli management software. Unlike Tivoli GEM modules and Tivoli Plus modules, base modules are developed without the use of a template.

bash. Bourne-again shell. A portable, command-line interface and script interpreter that is compatible with the UNIX Bourne and Korn shells and includes some features of the UNIX C shell.

Basic Input/Output System (BIOS). Code that controls basic hardware operations, such as interactions with diskette drives, hard disk drives, and the keyboard.

Basic Object Adapter (BOA). Software that provides CORBA-compliant services for object implementations.

Basic Recorder of Objects in C (BAROC). In the event server of the Tivoli Enterprise Console, the internal representation of the defined event classes.

basic sequential access method (BSAM). In the NetView Performance Monitor (NPM), the method by which all PIUs collected for selected LUs can be logged into a sequential data set as they pass through VTAM.

BSAM. See basic sequential access method.

BSDF. See business system description file.

BSSDF. See business subsystem description file.

BSSDF. See business subsystem description file.

buffer. (1) A routine or storage used to compensate for a difference in rate of flow of data, or time of occurrence of events, when transferring data from one device to another. (2) To allocate and schedule the use of buffers. (3) A portion of storage used to hold input or output data temporarily.

bulletin board. In the Tivoli environment, the primary mechanism by which the Tivoli Management Framework and Tivoli applications communicate with Tivoli administrators. The bulletin board is represented as an icon on the Tivoli desktop through which the administrators can access notices. Tivoli applications use the bulletin board as an audit trail for important operations that the administrators perform.

business component. An application or other system resource that can be managed by systems management software.

business description file (BDF). In a Tivoli environment, a generic name for any of these application description files: business system description file (BSDF), business system component description file (BCDF), and business system mapping description file (BMDF), and business subsystem description file (BSSDF).

business subsystem description file (BSSDF). In the context of the Application Management Specification (AMS), an optional application description file that enables the logical grouping of business components in a business system. In this file, a Tivoli administrator can specify tasks and monitors that are common to the subsystem. The business subsystem description file references the applicable business system description file and one or more business system component description files.
defines the logical representation of a business component. When defining a business system, at least one business system component description file is required for each business system definition.

**business system description file (BSDF).** In the context of the Application Management Specification (AMS), the highest-level application description file that identifies the components of a business system, including monitors, tasks, and connections. A Tivoli administrator can also define icon and help files at this level (for creating a business system icon and help information).

**business system mapping description file (BMDF).** In the context of the Application Management Specification (AMS), the application description file that maps a real business component (which is defined in a component description file) to a logical business component (which is represented in a business system component description file). Each business system mapping description file references a business system component description file.

C

**CADAM.** Computer-Aided Design and Manufacturing. The use of computers in the design and manufacture of products such as cars, airplanes, ships, and computers.

**call.** (1) The action of bringing a computer program, a routine, or a subroutine into effect, usually by specifying the entry conditions and jumping to an entry point. (2) In data communication, the actions necessary to make a connection between two stations on a switched line. (3) In communications, a conversation between two users. (4) To transfer control to a procedure, program, routine, or subroutine. (5) To attempt to contact a user, regardless of whether the attempt is successful.

**callback.** In the AIX operating system, a procedure that is called if and when certain specified conditions are met.

**canonical.** In computer science, pertaining to an expression that conforms to a specific set of rules.

**CATIA.** Computer-Graphics Aided Three-Dimensional Interactive Application.

**CC.** See change control

**CCMS.** (1) See Computing Center Management System (2) See Configuration Change Management System

**CDF.** See component description file

**CDNM session.** See cross-domain network managed session

**CDS.** See control data set

**central site control facility (CSCF).** In Tivoli NetView for OS/390, NetView for VM, and NetView for VSE, a function that allows a network operator to execute the test facilities of the IBM 3172 Nways® Interconnect Controller and the IBM 3174 Establishment Controller remotely from the NetView console.

**change control (CC).** The use of change management commands for the installation or removal of software or data.

**change control administrator.** A person responsible for software distribution and change control activities.

**change control client.** A workstation that (a) receives software and data files from its change control server and (b) installs and removes software and data files as instructed by its change control server.

**change control domain.** A change control server and its change control clients.

**change control server.** A workstation that controls and tracks the distribution of software and data files to other workstations.

**change control single node.** A workstation that controls, tracks, installs, and removes software and data files for itself. A CC single node can also prepare software for distribution. Contrast with change control client and change control server.

**change control.** The process of planning (for example, scheduling) and controlling (for example, distributing, installing, and tracking) software changes over a network. This is sometimes known as “software management.”

**check box.** A square box with associated text that represents a choice. When a user selects the choice, the check box is filled to indicate that the choice is selected. The user can clear the check box by selecting the choice again, thereby deselecting the choice.

**checkpoint.** (1) Information about the status of a program’s execution or the status of a data transfer that is recorded to enable the program or the data transfer to be restarted if it is ever interrupted. (2) The time at which such information is recorded. (3) To record such information.

**child process.** In the UNIX operating system, a process, started by a parent process, that shares the resources of the parent process. See fork.

**child resource.** In the NetView Graphic Monitor Facility, a resource that is directly subordinate to another resource (the parent) in a hierarchy.

**CICS®.** See Customer Information Control System.

**class.** (1) In object-oriented design or programming, a model or template that can be instantiated to create
objects with a common definition and therefore, common properties, operations, and behavior. An object is an instance of a class. (2) In the AIX operating system, pertaining to the I/O characteristics of a device. System devices are classified as block or character devices.

**CLI.** See command line interface.

**client.** A computer system or process that requests a service of another computer system or process that is typically referred to as a server. Multiple clients may share access to a common server.

**client daemon.** An AIX process that performs the client’s operations.

**client/server.** In communications, the model of interaction in distributed data processing in which a program at one site sends a request to a program at another site and awaits a response. The requesting program is called a client; the answering program is called a server.

**client workstation.** In the NetView Graphic Monitor Facility, a workstation that depends on a server workstation to provide it with views and status information. A client workstation receives status information from the server workstation over an LU 6.2 session.

**cloning.** (1) In a Tivoli environment, an operation that enables a Tivoli administrator to replicate profiles. This capability simplifies the task of creating multiple profiles with similar properties. See prototype profile. (2) In a Tivoli environment, a function of Tivoli NetView for OS/390 that enables a system programmer to replicate NetView definitions across the systems comprising a sysplex, thus simplifying the task of creating multiple NetView definitions with similar properties.

**CNM.** See communication network management.

**CNM application program.** A VTAM application program that issues and receives formatted management services request units for physical units. Tivoli NetView for OS/390 is an example of a CNM application program.

**CNM processor.** In Tivoli NetView for OS/390, a program that manages one of the functions of a communication system. A CNM processor is executed under control of Tivoli NetView for OS/390.

**collaborative management.** A cooperative relationship between Internet commerce partners and Internet service providers (ISPs) to ensure the successful completion of business transactions.

**collection.** In a Tivoli environment, a container that groups objects on a Tivoli desktop, thus providing the Tivoli administrator with a single view of related resources. Either the Tivoli Management Framework or a Tivoli administrator can create a collection. The contents of a collection are referred to as its members. Examples of collections include the administrator collection and the generic collection; the administrator collection is an example of a collection generated by the Tivoli Management Framework.

**collection point block (CPB).** In the NetView Performance Monitor (NPM), a control block used to coordinate the collection of network and session data.

**combined alert.** In Tivoli NetView for OS/390, an alert that includes elements of a non-generic and a generic alert in one network management vector transport (NMVT).

**command.** (1) A request from a terminal for the performance of an operation or the execution of a particular program. (2) In Tivoli NetView for OS/390, a sequence of characters that is submitted to cause an action. A command contains a verb and an object.

**command authorization.** The process of authorizing a network operator to use various commands. See NetView command authorization table, Resource Access Control Facility, scope of command authorization, and System Authorization Facility.

**command facility.** In Tivoli NetView for OS/390, the component that is a base for command processors that can monitor, control, automate, and improve the operation of a network.

**command indicator.** In the NetView Graphic Monitor Facility, a numeric identifier that is assigned to a network resource by its controlling resource manager to indicate the command support characteristics for the resource.

**command interpreter.** In the AIX operating system, a program that sends instructions to the kernel.

**command line interface (CLI).** A type of computer interface in which the input command is a string of text characters. Contrast with graphical user interface.

**command list.** In Tivoli NetView for OS/390, a list of commands and statements designed to perform a specific function for the user. Command lists can be written in REXX or in the NetView command list language.

**command procedure.** In Tivoli NetView for OS/390, a command list, a command processor written in a high-level language (HLL), or a NetView pipeline.

**command processor.** In Tivoli NetView for OS/390, a module designed to perform a specific function for the user. Users can write command processors in assembler language or in a high-level language (HLL); command processors are invoked as commands.
command profile editor (CPE). In Tivoli Global Enterprise Manager and Tivoli NetView for OS/390, a function of the topology console that enables Tivoli administrators who have the proper administrative authority to control the content, order, and capabilities of pop-up menus for individual operators or groups of operators.

commit operation. In Tivoli Software Distribution, an operation performed by a configuration program on target managed nodes after a file package distribution. This function enables a Tivoli administrator to distribute a file package to multiple targets and to make the distributed information available on all targets at the same time.

Common Object Request Broker Architecture (CORBA). A specification produced by the Object Management Group (OMG) that presents standards for various types of object request brokers (such as client-resident ORBs, server-based ORBs, system-based ORBs, and library-based ORBs). Implementation of CORBA standards enables object request brokers from different software vendors to interoperate.

Common Programming Interface for Communications (CPI-C). An evolving application programming interface (API), embracing functions to meet the growing demands from different application environments and to achieve openness as an industry standard for communications programming. CPI-C provides access to interprogram services such as (a) sending and receiving data, (b) synchronizing processing between programs, and (c) notifying a partner of errors in the communication.

communication network management (CNM). The process of designing, installing, operating, and managing distribution of information and control among users of communication systems.

communications infrastructure. In the AIX operating system, a framework of communication that consists of a postmaster, an object registration service, a startup file, communication protocols, and application programming interfaces.

Communications Server. An IBM licensed program that supports (a) the development and use of application programs across two or more connected systems or workstations, (b) multiple concurrent connections that use a wide range of protocols, and (c) several application programming interfaces (APIs) that may be called concurrently and that are designed for client/server and distributed application programs. Communications Server includes the necessary interfaces for network management and is available on several operating systems (such as AIX, OS/2®, Warp, OS/390, and Windows NT®).

community. In the Simple Network Management Protocol (SNMP), an administrative relationship between entities.

community name. In the Simple Network Management Protocol (SNMP), a string of octets identifying a community.

component description file (CDF). In the context of the Application Management Specification (AMS), an application description file that contains information about a specific component in a management-ready application. Each management-ready application can contain multiple components, each of which is represented by one component description file.

Computing Center Management System (CCMS). The SAP interface for monitoring a SAP R/3 system.

configuration. (1) The manner in which the hardware and software of an information processing system are organized and interconnected. (T) (2) The devices and programs that make up a system, subsystem, or network.

Configuration Application. See MLM Configuration Application.

Configuration Change Management System (CCMS). In a Tivoli environment, a distributed, hierarchical database in which configuration data is stored for use by systems management applications in effecting configuration changes on groups of systems.

configuration file. A file that specifies the characteristics of a system device or network.

configuration management. The control of information necessary to identify both physical and logical information system resources and their relationship to one another.

configuration program. In Tivoli Software Distribution, a feature that enables a Tivoli administrator to perform operations (a) before or after file package distributions, (b) before or after file package removal, (c) during a file package commit operation, or (d) after an error stops a distribution or removal operation.

configuration repository. In a Tivoli environment, the relational database that contains information that is collected or generated by Tivoli applications. Following are examples of the information that is stored in the configuration repository:

- Tivoli Enterprise Console stores information regarding events.
- Tivoli Inventory stores information regarding hardware, software, system configuration, and physical inventory.
- Tivoli Software Distribution stores information regarding file package operations.
connector class. In Tivoli NetView, an object class used for objects that connect different parts of the network and that route or switch traffic between these parts. This class includes gateways, repeaters (including multiport repeaters), and bridges. Contrast with network class.

cnsole event. In a Tivoli environment, an event sent to the Tivoli Enterprise Console.

container. A visual user-interface component that holds objects.

control data set (CDS). In the NetView Performance Monitor (NPM), a System Modification Program (SMP) data set used in the NPM installation process.

control desk. In Tivoli NetView, a component of the graphical user interface (GUI) that enables the network operator to group application program instances together.

control program. (1) A computer program designed to schedule and to supervise the execution of programs of a computer system. (I) (A) (2) The part of the AIX operating system that determines the order in which basic functions should be performed.

control statement. In Tivoli NetView for OS/390, a statement in a command list that controls the processing sequence of the command list or allows the command list to send messages to the operator and receive input from the operator.

CORBA. See Common Object Request Broker Architecture.

correlation activity. See event correlation.

CPB. See collection point block.

CPE. See command profile editor.

CPI-C. See Common Programming Interface for Communications.

critical resource. In the NetView Graphic Monitor Facility, a resource that is considered important to the operation of the network and therefore has a high aggregation priority.

cron table. In the AIX operating system, a table that is used to schedule application programs and processes. "Cron" is an abbreviation for "chronological."

cross-domain network manager session. A session between two network managers (for example, Tivoli NetView for OS/390) in separate domains.

cross-system coupling facility (XCF). A component of the MVS™ operating system that provides functions to support cooperation between authorized programs running within a sysplex.

CSCF. See central site control facility.

current directory. See working directory.

Customer Information Control System (CICS). An IBM licensed program that provides online transaction processing services and management for critical business applications. CICS runs on many IBM and non-IBM platforms (from the desktop to the mainframe) and is used in various types of networks that range in size from a few terminals to many thousands of terminals. The CICS application programming interface (API) enables programmers to port applications among the hardware and software platforms on which CICS is available. Each product in the CICS family can interface with the other products in the CICS family, thus enabling interproduct communication.

custom monitor. In Tivoli Distributed Monitoring, a monitor that is implemented as a script or program by the Tivoli administrator.

D

daemon. A program that runs unattended to perform a standard service. Some daemons are triggered automatically to perform their task; others operate periodically.

DASD conservation option. In Tivoli NetView for OS/390, an installation option that allows Tivoli NetView for OS/390 to be installed without the online help facility and hardware monitor data presentation panels.

database. (1) A collection of data with a given structure for accepting, storing, and providing, on demand, data for multiple users. (T) (2) A collection of interrelated data organized according to a database schema to serve one or more applications. (T) (3) A collection of data fundamental to a system. (A) (4) A collection of data fundamental to an enterprise. (A)

data model. (1) A logical view of the organization of data in a database. (T) (2) In a database, the user's logical view of the data in contrast to the physically stored data, or storage structure. (A) (3) A description of the organization of data in a manner that reflects the information structure of an enterprise. (A)

data modeling. A structured set of techniques for defining and recording business information requirements. It is a depiction of the user's view of the data needs of the organization in a consistent and rigorous fashion. The data model eventually serves as the basis for translation to computer system databases.

data services command processor (DSCP). In Tivoli NetView for OS/390, a component that structures a request for recording and retrieving data in the application program's database and for soliciting data from a device in the network.
data services manager (DSM).  In Tivoli NetView for OS/390, a function that provides VSAM services for data storage and retrieval.

data services request block (DSRB).  In Tivoli NetView for OS/390, the control block that contains information that a data services command processor (DSCP) needs to communicate with the data services task (DST).

data services task (DST).  In Tivoli NetView for OS/390, the subtask that gathers, records, and manages data in a VSAM file or a network device that contains network management information.

data type.  In Tivoli NetView for OS/390, one of the three elements, which also include display type and resource type, that are used to describe the organization of panels. Data types include alerts, events, and statistics.

dce-pipe-pull.  A Printing Systems Manager (PSM) document transfer method in which the client saves documents in a file and transfers the address of the file to the server. The file is later transferred to the server upon request from the server. This is an efficient transfer method for large jobs. Contrast with with-request.

default policy.  In a Tivoli environment, a set of resource property values that are assigned to a resource when the resource is created.

definition statement.  (1) In VTAM, the statement that describes an element of the network.  (2) In NCP, a type of instruction that defines a resource to the NCP.

defragmentation.  The process of running a software utility to rewrite fragmented data to contiguous sectors of a computer storage medium to improve access and retrieval time. Contrast with fragmentation.

demand poll.  In Tivoli NetView, a polling operation initiated by the user.

deployment management.  The Tivoli management discipline that addresses the automation of configuration and change management activities for the ever-evolving components of a network computing system. See availability management, operations and administration, and security management.

desktop.  See Tivoli desktop.

Desktop Management Interface (DMI).  A protocol-independent set of application programming interfaces (APIs) defined by the Desktop Management Task Force (DMTF). These interfaces give management application programs standardized access to information about hardware and software in a system.

Desktop Management Task Force (DMTF).  An alliance of computer vendors that was convened to define streamlined management of the diverse operating systems commonly found in an enterprise.

developer key.  In the context of SAP application software, a key that is provided by SAP for a developer’s use in creating or changing Advanced Business Application Programming (ABAP) objects.

DFSMdtp™.  A DFSMS/MVS® component and a base element of OS/390 that provides functions for storage management, data management, program management, device management, and distributed data access ("dtp" represents "data facility product").

DFSMSdss™.  A DFSMS/MVS component and a base element of OS/390 that is used in copying, moving, dumping, defragmenting, and restoring data sets and volumes ("dss" represents "data set services").

DFSMShsm™.  A DFSMS/MVS component and a base element of OS/390 that is used in backing up data, in recovering data, in managing storage space on volumes in the storage hierarchy, and in disaster recovery ("hsm" represents "hierarchical storage manager").

DFSM/MVS.  An IBM licensed program that provides storage, data, and device management functions in an MVS/ESA™ Version 5 or an OS/390 environment. DFSMS/MVS includes these components: DFSMSdtp, DFSMSdss, DFSMSHsm, and DFSMSrmm™. "DFMS" represents “Data Facility Storage Management Subsystem.”

DFSMrmm™.  A DFSMS/MVS component and base element of OS/390 that manages removable media ("rmm" represents “removable media manager”).

DHCP.  See Dynamic Host Configuration Protocol.

directory.  In a hierarchical file system, a container for files or other directories. See path.

discriminator.  An object that enables a system to select operations and event reports relating to other managed objects. See event forwarding discriminator.

display type.  In Tivoli NetView for OS/390, one of the three elements, which also include data type and resource type, that are used to describe the organization of panels. Display types include total, most recent, user action, and detail.

distributed computing.  See network computing.

Distributed Monitoring engine.  In a Tivoli environment, the client software that is installed on each managed node, gateway, and endpoint that is being monitored by Tivoli Distributed Monitoring. The Distributed Monitoring engine monitors resources, compares data from monitored resources against configured thresholds, and runs automated responses.

Distributed Monitoring proxy.  See endpoint.
**distribution program.** See configuration program.

**DMI.** See Desktop Management Interface.

**DMTF.** See Desktop Management Task Force.

**domain.** (1) That part of a computer network in which the data processing resources are under common control. (T) (2) See Administrative Domain and domain name.

**domain name.** In the Internet suite of protocols, a name of a host system. A domain name consists of a sequence of subnames separated by a delimiter character. For example, if the fully qualified domain name (FQDN) of a host system is ralvm7.vnet.ibm.com, each of the following is a domain name:
- ralvm7.vnet.ibm.com
- vnet.ibm.com
- ibm.com

**double recording.** In Tivoli NetView for OS/390, pertaining to the recording of certain individual events under two resource levels.

**downcall.** In a Tivoli environment, a method invocation from the TMR server or the gateway “down” to an endpoint. Contrast with upcall.

**drag and drop.** To directly manipulate an object by moving it and placing it somewhere else using a pointing device (such as a mouse).

**DSCP.** See data services command processor.

**DSM.** See data services manager.

**DSRB.** See data services request block.

**DST.** See data services task.

**Dynamic Host Configuration Protocol (DHCP).** A protocol defined by the Internet Engineering Task Force (IETF) that is used for dynamically assigning IP addresses to computers in a network.

**Dynamic Host Configuration Protocol (DHCP).**

**E**

**e-business.** Either (a) the transaction of business over an electronic medium such as the Internet or (b) any organization (for example, commercial, industrial, nonprofit, educational, or governmental) that transacts its business over an electronic medium such as the Internet. An e-business combines the resources of traditional information systems with the vast reach of an electronic medium such as the Internet (including the World Wide Web, intranets, and extranets); it connects critical business systems directly to critical business constituencies—customers, employees, and suppliers. The key to becoming an e-business is building a transaction-based Web site in which all core business processes (especially all processes that require a dynamic and interactive flow of information) are put online to improve service, cut costs, and sell products.

**ECB.** See event control block.

**e-commerce.** The subset of e-business that involves the exchange of money for goods or services purchased over an electronic medium such as the Internet.

**EDB.** See event forwarding discriminato.

**EIF.** See Tivoli Event Integration Facility.

**EMS.** See event management services.

**encapsulation.** (1) In object-oriented programming, the technique that is used to hide the inherent details of an object. This technique is also known as “information hiding.” (2) In object-oriented programming, a software technique in which data is packaged with corresponding procedures. In CORBA, the object is the mechanism for encapsulation.

**endpoint.** (1) In a Tivoli environment, a Tivoli client that is the ultimate recipient for any type of Tivoli operation. (2) In a Tivoli environment, a Tivoli service that runs on multiple operating systems and performs Tivoli operations on those systems, thereby enabling the Tivoli Management Framework to manage the systems as Tivoli clients.

**endpoint list.** In a Tivoli environment, a list of all endpoint clients in the Tivoli Management Region with their assigned gateways. See endpoint manager.

**endpoint manager.** In a Tivoli environment, a service that runs on the Tivoli server, assigns endpoint clients to gateways, and maintains the endpoint list.

**endpoint method.** In a Tivoli environment, a method that runs on an endpoint client as the result of a request from other managed resources in the Tivoli Management Region. Results of the method are forwarded first to the gateway, then to the calling managed resource.

**Enhanced X-Windows Toolkit.** (1) In the AIX operating system, a collection of basic functions for developing a variety of application environments. Toolkit functions manage Toolkit initialization, widgets, memory, events, geometry, input focus, selections, resources, translation of events, graphics contexts, pixmaps, and errors. (2) See AIXwindows Toolkit and X Window System.

**entity.** Any concrete or abstract thing of interest, including associations among things; for example, a person, object, event, or process that is of interest in the context under consideration, and about which data may be stored in a database. (T)

**entry point (EP).** (1) The address or label of the first instruction executed on entering a computer program,
routine, or subroutine. A computer program, routine, or subroutine may have a number of different entry points, each perhaps corresponding to a different function or purpose. (1) (A) (2) In SNA, a type 2.0, type 2.1, type 4, or type 5 node that provides distributed network management support. It sends network management data about itself and the resources it controls to a focal point for centralized processing, and it receives and executes focal-point initiated commands to manage and control its resources.

**EP.** See [entry point](#).

**error record template.** In the AIX operating system, a template that describes the error class, error type, error description, probable causes, recommended actions, and failure data for an error log entry.

**euro.** The monetary unit of the European Monetary Union (EMU) that will be introduced alongside national currencies on the first of January 1999. In May 1998, eleven countries were confirmed for EMU membership beginning the first of January 1999: Austria, Belgium, Finland, France, Germany, Ireland, Italy, Luxembourg, the Netherlands, Portugal, and Spain. On the first of January 2002, euro notes and coins (hard currency) will be put into circulation, and national currencies will be withdrawn, probably over a six-month period.

**EuroReady product.** A product is EuroReady if the product, when used in accordance with its associated documentation, is capable of correctly processing monetary data in the euro denomination, respecting the euro currency formatting conventions (including the euro sign). This assumes that all other products (for example, hardware, software, and firmware) that are used with this product are also EuroReady. IBM hardware products that are EuroReady may or may not have an engraved euro sign key on their keyboards.

**EuroReady solution.** A solution is EuroReady when the solution providers have done the following:

1. Analyzed the euro requirements, including the need to comply with relevant European Community (EC) rules
2. Included the appropriate function according to these requirements
3. Clearly demonstrated this by (a) detailing the euro-related requirements, (b) describing how these requirements will be implemented, and (c) declaring when the implementation will be generally available.

**event.** (1) An occurrence of significance to a task (such as the opening of a window or the completion of an asynchronous operation). (2) In the Tivoli environment, any significant change in the state of a system resource, network resource, or network application. An event can be generated for a problem, for the resolution of a problem, or for the successful completion of a task. Examples of events are: the normal starting and stopping of a process, the abnormal termination of a process, and the malfunctioning of a server. (3) See [event report](#).

**event adapter.** In a Tivoli environment, software that converts events into a format that the Tivoli Enterprise Console can use and forwards the events to the event server. Using the Tivoli Event Integration Facility, an organization can develop its own event adapters, tailored to its network environment and specific needs.

**event/automation service.** In Tivoli NetView for OS/390, a facility that translates alerts and messages into events for the Tivoli Enterprise Console and translates these events into NetView alerts. The event/automation service communicates with Tivoli NetView for OS/390 using the program-to-program interface (PPI), and it communicates with the Tivoli Enterprise Console using TCP/IP.

**event card.** In Tivoli NetView, a graphical representation, resembling a card, of the information contained in an event sent by an agent to a manager reflecting a change in the status of one of the agent’s managed nodes.

**event class.** In the Tivoli Enterprise Console, a classification for an event that indicates the type of information that the event adapter will send to the event server.

**event console.** In the Tivoli Enterprise Console, a graphical user interface (GUI) that enables system administrators to view and respond to dispatched events from the event server. The Tivoli Event Integration Facility does not directly use or affect event consoles.

**event control block (ECB).** A control block used to represent the status of an event.

**event correlation.** In the Tivoli Enterprise Console, the process of correlating separate events to a common cause. For example, the Tivoli Enterprise Console may receive several *NFS server not responding* events from several different applications, as well as a host down event for the NFS server. The Tivoli Enterprise Console can then correlate the various NFS server not responding events to their common cause, which is: the NFS server is “down.” See [ECB](#).

**event filter.** (1) In a Tivoli environment, software that determines which events are forwarded to a specified destination. Filtering events helps to reduce network traffic. Tivoli administrators configure the event filters. (2) In Tivoli NetView, a logical expression of criteria that determine which events are forwarded to the application program that registers the event filter with the event sieve agent. A filter is referred to as “simple” or “compound” depending on how it is handled by the filter editor.

**event forwarding discriminator (EFD).** A managed object that describes the criteria used to select which event reports are sent and to whom they are sent.
**event group.** In the Tivoli Enterprise Console, a set of events that meet certain criteria. Each event group is represented by an icon on the event console. Tivoli administrators can monitor event groups that are relevant to their specific areas of responsibility.

**event handler.** A collection and correlation point for events and messages.

**event management services (EMS).** In Tivoli NetView, a centralized method of generating, receiving, routing, and logging network events.

**event manager.** In the NetView Graphic Monitor Facility, the component of the host subsystem that receives alert and resolution major vectors from Tivoli NetView for OS/390, translates these major vectors into generic event records, and applies the event status to the resource defined in the Resource Object Data Manager (RODM) cache.

**event report.** The unsolicited report that an event has occurred. In an Open Systems Interconnection (OSI) context, when a managed object emits a notification, the agent uses one or more event forwarding discriminators (EFDs) to find the destinations to which the report is sent.

**event repository.** See configuration repository.

**event server.** In the Tivoli Enterprise Console, a central server that processes events. The event server creates an entry for each incoming event and evaluates the event against a rule base to determine whether it can respond to or modify the event automatically. The event server also updates the event consoles with the current event information. If the primary event server is not available, events can be sent to a secondary event server.

**event sieve.** In Tivoli NetView, an object that is managed by the "ovesmd" daemon, which is the event sieve agent. The event sieve agent stores information about the event sieve object in a database and reads that information when the agent is started. See event filter and event forwarding discriminator.

**event slot.** In a Tivoli environment, a discrete area (a field) of an event record that contains a specific type of information about an event.

**event specifier.** In the Tivoli Enterprise Console, a rules-language program construct that is used to look for events in the event cache. For example, it can look for duplicate events, an event that matches a user-specified attribute, or an event that occurs within a certain time period. An event specifier is used in building rules, and it dictates how the Tivoli Enterprise Console will handle an event that it receives.

**exception.** An abnormal condition such as an I/O error encountered in processing a data set or a file.

**exclusive set.** In Remote Operations Service (ROPS), an option that indicates whether only the commands in the command list can be processed by ROPS or none of the commands in the command list can be processed by ROPS.

**exclusive submap.** In Tivoli NetView, a submap that is created by an application program wanting the exclusive right to control what happens in the application plane of the submap. Contrast with shared submap.

**exec.** (1) In the AIX operating system, to overlay the current process with another executable program. (2) See fork.

**executable symbol.** In Tivoli NetView, a symbol defined such that double-clicking on it causes an application program to perform an action on a set of target objects. Contrast with explodable symbol.

**execution target.** In a Tivoli environment, a managed node on which a job or other activity is performed. For example, if an application is being installed on a particular server, that server is the execution target for the installation activity.

**explicit command.** In Tivoli NetView for OS/390, a command that is used to request the display of information that the user would otherwise obtain by navigating through a hierarchy of panels.

**explodable symbol.** In Tivoli NetView, a symbol defined such that double-clicking on it or dragging and dropping it displays the child submap of the parent object that the symbol represents. Contrast with executable symbol.

**export/import.** In Tivoli Software Distribution, a feature that enables a Tivoli administrator to save (export) a file package definition as a text file, to edit the keywords and lists in the definition, and to retrieve (import) the definition from the text file to set the properties for the file package.

**extended enterprise.** The customers, suppliers, distributors, and other business partners with whom a company conducts e-business.

**extranet.** A private, virtual network that uses access control and security features to restrict the usage of one or more intranets attached to the Internet to selected subscribers (such as personnel from a sponsoring company and its business partners).

**F**

**failover system.** In Tivoli Manager for R/3, a computer that serves as a transparent backup to a primary computer. The primary computer and the failover...
system share access to a common R/3 database, thereby enabling either machine to provide full database support.

**fanout.** In communication, the process of creating copies of a distribution to be delivered locally or to be sent through the network.

**field.** (1) An identifiable area in a window. Examples of fields are: an entry field, into which a user can type or place text, and a field of radio button choices, from which a user can select one choice. (2) The smallest identifiable part of a record. (3) In Tivoli NetView, the building block of which objects are composed. A field is characterized by a field name, a data type (integer, Boolean, character string, or enumerated value), and a set of flags that describe how the field is treated by Tivoli NetView. A field can contain data only when it is associated with an object.

**field registration file (FRF).** In Tivoli NetView, a file used to define fields for use in the object database.

**file name substitution.** In the AIX operating system, the process in which the shell substitutes an alphabetically sorted list of file names in the place of a pattern. The shell recognizes a pattern (as opposed to a set of flags that describe how the field is treated by Tivoli NetView). A field can contain data only when it is associated with an object.

**file name substitution.** In the AIX operating system, the process in which the shell substitutes an alphabetically sorted list of file names in the place of a pattern. The shell recognizes a pattern (as opposed to a file name) by the occurrence of a word (character string) pattern. The shell recognizes a pattern (as opposed to an alphabetically sorted list of file names in the place of a pattern). The shell substitutes an alphabetically sorted list of file names in the place of a pattern. The shell recognizes a pattern (as opposed to an alphabetically sorted list of file names in the place of a pattern). The shell substitutes an alphabetically sorted list of file names in the place of a pattern.

**foreground process.** (1) In the AIX operating system, a process that must run to completion before another command is issued to the shell. The foreground process is in the foreground process group, which is the group that receives the signals generated by a terminal. (2) Contrast with background process.

**foreground task.** The task with which the user is interacting. Contrast with background task.

**foreign host.** See remote host.

**fork.** In the UNIX operating system, to create and start a child process.

**fpblock.** See file package block.

**FQDN.** See fully qualified domain name.

**fragmentation.** An operating system’s process of dividing a file into smaller parts that are placed on different physical devices or storage media. Fragmentation occurs when files are stored on multiple devices or storage media, and each part of the file is stored on a different device or storage media. The process of dividing a file into smaller parts that are placed on different physical devices or storage media is called fragmentation. Fragmentation occurs when files are stored on multiple devices or storage media, and each part of the file is stored on a different device or storage media.

**full pathname.** See absolute path.

**fully qualified domain name (FQDN).** In the Internet suite of protocols, the name of a host system that includes all of the subnames of the domain name. An example of a fully qualified domain name is ralvm7.vnet.ibm.com. See host name.

**G**

**gadget.** In the AIXwindows Toolkit, a windowless graphical object that looks like its equivalent like-named widget but does not support the translations, actions, or pop-up widget children supplied by that widget.

**gateway.** (1) A functional unit that interconnects two computer networks with different network architectures. A gateway connects networks or systems of different architectures. A bridge interconnects networks or systems with the same or similar architectures. (2) A functional unit that connects two networks or subnetworks having different characteristics, such as different protocols or different policies concerning security or transmission priority. (3) The combination of machines and programs that provide address
translation, name translation, and system services control point (SSCP) rerouting between independent SNA networks to allow those networks to communicate. A gateway consists of one gateway NCP and at least one gateway VTAM. (4) In a Tivoli environment, software running on a managed node that provides all communication services between a group of endpoints and the rest of the Tivoli environment. This gateway includes the multiplexed distribution (MDist) function, enabling it to act as the fanout point for distributions to many endpoints. (5) See router.

gateway-capable host. A host node that has a defined NETID and SSCPNAME but does not perform gateway control functions, such as cross-network session initiation and termination.

gateway host. (1) A host node that contains a gateway system services control point (SSCP). See gateway-capable host. (2) In the AIX operating system, a host that connects independent networks. It has multiple interfaces, each with a different name and address.

gateway method. In a Tivoli environment, a method that runs on the gateway’s proxy managed node on behalf of the endpoint. Results of the method are forwarded to the calling managed resource.

gateway method. In a Tivoli environment, a method that runs on the gateway’s proxy managed node on behalf of the endpoint. Results of the method are forwarded to the calling managed resource.

GCS. See graphic communication server.

GDDM®. See Graphical Data Display Manager.

GDDM interface for X Window System (GDDMXD). A graphical interface that formats and displays characters, graphics, and images on workstation display devices that support the X Window System.

GDDM XD. See GDDM interface for X Window System.

GDF. See global description file.

GDS. See graphic data server.

GEM. See Tivoli Global Enterprise Manager.

general topology manager (GTM). In Tivoli NetView, the component that accepts information about resources that are accessed through protocols other than the Internet Protocol (IP), stores this information in a database, and displays it to the user.

generic alert. In SNA management services (SNA/MS), alert information that is encoded using a method in which code points provide an index into short units of stored text. The use of generic alerts prevents the receiver from having to recognize and understand each unique problem for which an alert is sent. Contrast with non-generic alert.

generic collection. In a Tivoli environment, a collection that contains objects representing resources of any type.

gid. See group ID.

gif. See graphical interchange format.

global description file (GDF). In the context of the Application Management Specification (AMS), an application description file that provides global information about an application such as the application name, the version identifier, and a free-form description of the application. Each version of a management-ready application is represented by one global description file.

GMFHS. See Graphic Monitor Facility host subsystem.

Graphical Data Display Manager (GDDM). In the NetView Performance Monitor (NPM), an IBM licensed program used in conjunction with the Presentation Graphics Feature (PGF) to generate online graphs in the NPM Graphic Subsystem.

graphical interchange format (GIF). A digital format that is used to compress and transfer graphical information over computer networks. For example, GIF is a common format for graphical information on the Internet.

graphical user interface (GUI). A type of computer interface consisting of a visual metaphor of a real-world scene, often of a desktop. Within that scene are icons, representing actual objects, that the user can access and manipulate with a pointing device. Contrast with command line interface.

graphic communication server (GCS). The part of the NetView Graphic Monitor Facility that manages LU 6.2 sessions used for data transport between (a) Tivoli NetView for OS/390 and the server workstation and (b) the server workstation and its client workstations.

graphic data server (GDS). The part of the NetView Graphic Monitor Facility that receives network management data from Tivoli NetView for OS/390, maintains this data (except for dynamically created view information), and correlates this data with views.

graphic monitor. The graphical user interface (GUI) component of the NetView Graphic Monitor Facility.

Graphic Monitor Facility host subsystem (GMFHS). In Tivoli NetView for OS/390, a component that manages updates to the configuration and status of resources displayed in NetView Graphic Monitor Facility (NGMF) views.

graphics context (GC, Gcontext). In the Enhanced X-Windows Toolkit, the storage area for various kinds of graphics output, such as foreground pixels, background pixels, line widths, and clipping regions. A graphics context can be used only with drawables that have the same root and the same depth as the graphics context.

graphics data file (GDF). A picture definition in a coded format that is used internally by the Graphical
Data Display Manager (GDDM) and, optionally, provides the user with a lower level program interface than the GDDM application programming interface (API).

group ID (GID). In the AIX operating system, a number that corresponds to a specific group name. The group ID can often be substituted in commands that take a group name as a value.

group profile. In Tivoli User Administration, a profile that a Tivoli administrator uses to define and modify information about a group of users.

GTM. See general topology manager.

GUI. See graphical user interface.

hardcoded. Pertaining to software instructions that are statically encoded and not intended to be altered.

hardcopy task (HCT). In Tivoli NetView for OS/390, the subtask that controls the passage of data between the NetView program and the hardcopy device.

hardware monitor. In Tivoli NetView for OS/390, the component that helps identify and solve problems related to physical network elements (as opposed to logical sessions, which are managed by the session monitor). Contrast with session monitor.

HCT. See hardcopy task.

heartbeat. In software products, a signal that one entity sends to another to convey that it is still active.

home submap. In Tivoli NetView, the first submap that appears when a map is opened. Each map has a home submap. When new maps are created, the home submap is the root submap.

hook. A location in a computer program where an instruction is inserted for invoking a particular function.

host. (1) A computer that is connected to a network (such as the Internet or an SNA network) and provides an access point to that network. Also, depending on the environment, the host may provide centralized control of the network. The host can be a client, a server, or both a client and a server simultaneously. (2) In a Tivoli environment, a computer that serves as a managed node for a profile distribution. (3) See host processor.

host name. In the Internet suite of protocols, the name given to a machine. Sometimes, "host name" is used to mean fully qualified domain name; other times, it is used to mean the most specific subname of a fully qualified domain name. For example, if ralvm7.vnet.ibm.com is the fully qualified domain name, either of the following may be considered the host name:

- ralvm7.vnet.ibm.com
- ralvm7

host namespace profile. In Tivoli Enterprise software, a profile that contains information about the list of hosts and their properties, such as host IP addresses and host aliases.

host processor. (1) A processor that controls all or part of a user application network. (2) In a network, the processing unit in which the data communication access method resides.

host transit time. In the NetView Performance Monitor (NPM), the average time (in seconds) that all transactions spend in the host. It includes both VTAM and application time. It is also reported as an average for the transactions originating at the logical unit for which data collection is occurring.

HTML. See Hypertext Markup Language.

HTTP. See Hypertext Transfer Protocol.

hub. In a network, a point at which circuits are either connected or switched. For example, in a star network, the hub is the central node; in a star/ring network, it is the location of wiring concentrators.

Hypertext Markup Language (HTML). A markup language that is specified by an SGML document type definition (DTD) and is understood by all Web servers.

Hypertext Transfer Protocol (HTTP). In the Internet suite of protocols, the protocol that is used to transfer and display hypertext documents.

IAB. See Internet Architecture Board.

ICMP. See Internet Control Message Protocol.

IDL. See Interface Definition Language.

IETF. See Internet Engineering Task Force.

immediate command. In Tivoli NetView for OS/390, a command (such as GO, RESET, or LOGOFF) that begins processing as soon as the operator enters it, possibly preempting other ongoing processing. All other commands are called "regular commands" and are processed by a "regular command processor." Regular commands can run concurrently with other regular commands and can be interrupted by immediate commands. Most commands and all command lists are regular commands.

IMS™. See Internet Management Specification.

indicator. In Tivoli Distributed Monitoring, an icon on the Tivoli desktop that graphically displays the status of a monitor that has been associated with it. The icon
resembles a thermometer, which the Tivoli administrator can read to determine the status of the monitor.

**indicator collection.** In a Tivoli environment, a single location from which a Tivoli administrator can determine the status of monitors in different profiles, as well as clear and reset alarmed states.

**instance.** In object-oriented programming, an object created by instantiating a class.

**instantiate.** In object-oriented programming, to represent a class abstraction with a concrete instance of the class.

**instrument.** In application or system software, to use monitoring functions to provide performance and other information to a management system.

**instrumentation.** In application or system software, either (a) monitoring functions that provide performance and other information to a management system or (b) the use of monitoring functions to provide performance and other information to a management system.

**intelligent agent.** Software that monitors conditions or actions on a network node and contains logic enabling it to respond to these conditions or actions.

**interactive chart utility (ICU).** A utility provided by the Graphical Data Display Manager (GDDM) to allow basic graphic handling capability and a menu-driven generation of different forms of graphs. ICU is a part of the presentation graphics feature.

**Interface Definition Language (IDL).** In CORBA, a declarative language that is used to describe object interfaces, without regard to object implementation.

**Internet Architecture Board (IAB).** The technical body that oversees (at a high level) the work of the Internet Engineering Task Force (IETF). The IAB approves the membership of the IETF.

**Internet Control Message Protocol (ICMP).** The protocol used to handle errors and control messages in the Internet Protocol (IP) layer. Reports of problems and incorrect datagram destinations are returned to the original datagram source.

**Internet Engineering Task Force (IETF).** The task force of the Internet Architecture Board (IAB) that is responsible for solving the short-term engineering needs of the Internet. The IETF consists of numerous working groups, each focused on a particular problem. Internet standards are typically developed or reviewed by individual working groups before they can become standards.

**Internet Management Specification (IMS).** A draft specification for an open standard for managing Internet resources and services.

**internet object.** In Tivoli NetView, a node or a network that can be accessed by the Internet Protocol (IP).

**Internet Protocol (IP).** In the Internet suite of protocols, a connectionless protocol that routes data through a network or interconnected networks. IP acts as an intermediary between the higher protocol layers and the physical network. However, this protocol does not provide error recovery and flow control and does not guarantee the reliability of the physical network.

**Internet service provider (ISP).** An organization that provides access to the Internet.

**Internetwork Packet Exchange (IPX).** The network protocol used to connect Novell's servers, or any workstation or router that implements IPX, with other workstations. Although similar to the Internet Protocol (IP), IPX uses different packet formats and terminology.

**interprocess communication (IPC).** The process by which programs communicate data to each other and synchronize their activities. Semaphores, signals, and internal message queues are common methods of interprocess communication.

**intranet.** A private network that integrates Internet standards and applications (such as Web browsers) with an organization's existing computer networking infrastructure.

**IP.** See [Internet Protocol](#).

**IPC.** See [interprocess communication](#).

**IPX.** See [Internetwork Packet Exchange](#).

**ISP.** See [Internet service provider](#).

**IT.** Information technology.

**J**

**Java.** An object-oriented programming language for portable interpretive code that supports interaction among remote objects. Java was developed and specified by Sun Microsystems, Incorporated.

**JavaBeans.** A platform-independent, software component technology for building reusable Java components called “beans.” Once built, these beans can be made available for use by other software engineers or can be used in Java applications. Also, using JavaBeans, software engineers can manipulate and assemble beans in a graphical drag-and-drop development environment.

**Java Database Connectivity (JDBC).** An application programming interface (API) that has the same characteristics as Open Database Connectivity (ODBC) but is specifically designed for use by Java database applications. Also, for databases that do not have a JDBC driver, JDBC includes a JDBC to ODBC bridge,
which is a mechanism for converting JDBC to ODBC; it presents the JDBC API to Java database applications and converts this to ODBC. JDBC was developed by Sun Microsystems, Inc. and various partners and vendors.

Java Management Application Programming Interface (JMAPI). A specification proposed by Sun Microsystems that defines a core set of application programming interfaces for developing tightly integrated system, network, and service management applications. The application programming interfaces could be used in diverse computing environments that encompass many operating systems, architectures, and network protocols.

JDBC. See Java Database Connectivity.

JMAPI. See Java Management Application Programming Interface.

job. (1) A unit of work defined by a user that is to be accomplished by a computer. Loosely, the term job is sometimes used to refer to a representation of a job. This representation may include a set of computer programs, files, and control statements to the operating system. (1) (2) A Printing Systems Manager (PSM) object that represents a request to print one or more documents in a single printing session. (3) In a Tivoli environment, a resource consisting of a task and its preconfigured execution parameters. Among other things, the execution parameters specify the set of hosts on which the job is to execute.

JPEG. A standard format for storing compressed true-color images. “JPEG” represents “Joint Photographic Experts Group,” which is the name of the committee that developed this standard format.

K

Kerberos. The security system of the Massachusetts Institute of Technology’s (MIT’s) Project Athena. It uses symmetric key cryptography to provide security services to users in a network.

Kerberos master machine. In Kerberos, the host machine on which the Kerberos database resides.

Kerberos master password. In Kerberos, the password required to change or access the Kerberos database.

Kerberos principal. In Kerberos, a service or user that is known to the Kerberos system. See principal name.

Kerberos realm. In Kerberos, a set of managed nodes that share the same Kerberos database.

key. In computer security, a sequence of symbols that is used with a cryptographic algorithm for encrypting or decrypting data. See private key and public key.

keyword. (1) In programming languages, a lexical unit that, in certain contexts, characterizes some language construct; for example, in some contexts, IF characterizes an if-statement. A keyword normally has the form of an identifier. (1) (2) One of the predefined words of an artificial language. (A) (3) A name or symbol that identifies a parameter. (4) The part of a command operand that consists of a specific character string (such as `DSNAME=`). (5) See keyword operand.

keyword operand. (1) An operand that consists of a keyword followed by one or more values (such as `DSNAME=HELLO`). (2) Contrast with positional operand. (3) See definition statement.

keyword parameter. A parameter that consists of a keyword followed by one or more values.

L

LAN. See local area network.

LAN Network Manager (LNM). An IBM licensed program that enables a user to manage and monitor LAN resources from a central workstation.

LCCM. See link connection component manager.

LCSM. See link connection subsystem manager.

link connection component manager (LCCM). The transaction program that manages the configuration of the link connection.

link connection subsystem manager (LCSM). The transaction program that manages the sequence of link connection components that belong to a link connection.

Link Problem Determination Aid (LPDA®). A series of procedures that are used to test the status of and to control DCEs, the communication line, and the remote device interface. These procedures, or a subset of them, are implemented by host programs (such as Tivoli NetView for OS/390 and VTAM), communication controller programs (such as NCP), and IBM LPDA DCEs. See LPDA-1 and LPDA-2.

LNM. See LAN Network Manager.

local area network (LAN). (1) A computer network located on a user’s premises within a limited geographical area. Communication within a local area network is not subject to external regulations; however, communication across the LAN boundary may be subject to some form of regulation. (T) (2) A network in which a set of devices are connected to one another for communication and that can be connected to a larger network.
local distribution. In a Tivoli environment, a distribution to target machines in the same Tivoli Management Region as the source machine.

local overrides. In a Tivoli environment, a feature of all profile-based Tivoli applications—except for Tivoli Software Distribution—that allows changes made at the endpoint profile to override those in a distributed profile.

local registration file (LRF). In Tivoli NetView, a file that provides information about an agent or daemon, such as the name, the location of the executable code, the names of processes dependent on the agent or daemon, and details about the objects that an agent manages.

local topology database. A database in an APPN or LEN node containing an entry for each transmission group (TG) having at least one end node for an endpoint. In an end node, the database has one entry for each TG connecting to the node. In a network node, the database has an entry for each TG connecting the network node to an end node. Each entry describes the current characteristics of the TG that it represents. A network node has both a local and a network topology database while an end node has only a local topology database.

lock. The means by which integrity of data is ensured by preventing more than one user from accessing or changing the same data or object at the same time.

logged-on operator. (1) In Tivoli NetView for OS/390, an operator station task that requires a terminal and a logged-on user. (2) Contrast with autotask.

LPDA. See Link Problem Determination Aid.

LPDA-1. The first version of the Link Problem Determination Aid (LPDA) command set. LPDA-1 is not compatible with LPDA-2.

LPDA-2. The second version of the Link Problem Determination Aid (LPDA) command set. LPDA-2 provides all of the functions of LPDA-1; it also supports commands such as the following:

- DCE configuration
- Dial
- Set transmit speed
- Commands to operate a contact that can control external devices.

LRF. See local registration file.

LUC session. Communication, using LU type 0 protocols, between the LUC tasks of two Tivoli NetView for OS/390 programs. This communication is similar to an LU 6.2 conversation.

LUC task. A Tivoli NetView for OS/390 task, denoted by the NetView domain ID concatenated with the literal “LUC” (for example, CNM01LUC), that serves as the endpoint of an LUC session.

LU group. (1) In the NetView Performance Monitor (NPM), a file containing a list of related or unrelated logical units. The LU group is used to help simplify data collection and analysis. (2) In Tivoli NetView for OS/390, a grouping of logical units according to some affinity, such as their link to the same VTAM generic resource or VTAM USERVAR.

LU 6.2 verb. A syntactical unit in the LU 6.2 application programming interface representing an operation.

M

macroinstruction. (1) An instruction in a source language that is to be replaced by a defined sequence of instructions in the same source language and that may also specify values for parameters in the replaced instructions. (T) (2) In assembler programming, an assembler language statement that causes the assembler to process a predefined set of statements called a macro definition. The statements normally produced from the macro definition replace the macroinstruction in the program.

managed node. (1) In Internet communications, a workstation, server, or router that contains a network management agent. In the Internet Protocol (IP), the managed node usually contains a Simple Network Management Protocol (SNMP) agent. (2) In a Tivoli environment, any managed resource on which the Tivoli Management Framework is installed.

managed object. (1) A component of a system that can be managed by a management application. (2) The systems management view of a resource that can be managed through the use of systems management protocols.

managed resource. In a Tivoli environment, any hardware or software entity (machine, service, system, or facility) that is represented by a database object and an icon on the Tivoli desktop. Managed resources must be a supported resource type in a policy region and are subject to a set of rules. Managed resources include, but are not limited to, managed nodes, task libraries, monitors, profiles, and bulletin boards.

management by subscription. In a Tivoli environment, the concept of managing network resources by creating sets of profiles and distributing the profiles (through profile managers) to physical entities (Tivoli resources), called subscribers.

Management Information Base (MIB). (1) A collection of objects that can be accessed by means of a network management protocol. (2) A definition for management information that specifies the information available from a host or gateway and the operations allowed. (3) In
OSI, the conceptual repository of management information within an open system. (4) See MIB module.

Management Information Format (MIF). The Desktop Management Interface (DMI) specification that defines the syntax for describing management information about the hardware and software components that can be installed on a computer system.

management module. In a Tivoli environment, a file that contains the management information and instrumentation for enabling a particular application or business system to be managed by Tivoli management software. This file may be in the form of a Tivoli install image or an application management package. Types of management modules include base modules, Tivoli GEM modules, and Tivoli Plus modules. See Tivoli Module Builder and Tivoli Module Designer.

management region. In Tivoli NetView, the set of managed objects on a particular map that defines the extent of the network that is being actively managed. The management region may vary across maps.

management services (MS). (1) One of the types of network services in control points (CPs) and physical units (PUs). Management services are the services provided to assist in the management of SNA networks, such as problem management, performance and accounting management, configuration management, and change management. (2) Services that assist in the management of systems and networks in areas such as problem management, performance management, business management, operations management, configuration management, and change management.

management services focal point (MSFP). For any given management services discipline (for example, problem determination or response time monitoring), the control point that is responsible for that type of network management data for a sphere of control. This responsibility may include collecting, storing, or displaying the data, or all of these. (For example, a problem determination focal point is a control point that collects, and that may store or display, problem determination data.)

manager. (1) In systems management, a user that, for a particular interaction, has assumed a manager role. (2) An entity that monitors or controls one or more managed objects by (a) receiving notifications regarding the objects and (b) requesting management operations to modify or query the objects. (3) A system that assumes a manager role.

manager role. In systems management, a role assumed by a user where the user is capable of issuing management operations and of receiving notifications.

man page. In UNIX systems, one page of online documentation. “Man page” is an abbreviation for “manual page.” Each UNIX command, utility, and library function has an associated man page that can be viewed by entering this command: man command name.

map. In Tivoli NetView, a database represented by a set of related submaps that provide a graphical and hierarchical presentation of a network and its systems.

mapper. In Tivoli NetView for OS/390, a function that records errors from resources attached to a communication controller or from certain channel-attached devices.

marshall. To copy data into a remote procedure call (RPC) packet. Stubs perform marshalling. Contrast with unmarshall.

MCSL. See Monitoring Collection Specification Language.

MDist. Multiplexed distribution. In a Tivoli environment, a service that enables efficient distribution of large amounts of data across complex networks.

menu bar. (1) The area near the top of a window, below the title bar and above the rest of the window, that contains choices that provide access to other menus. (2) In the AIX operating system, a rectangular area at the top of the client area of a window that contains the titles of the standard pull-down menus for that application.

message style. In Tivoli Distributed Monitoring, the amount and format of information presented by certain monitors.

method. (1) In object-oriented design or programming, the software that implements the behavior specified by an operation. (2) In Tivoli NetView for OS/390, a program that runs in the Resource Object Data Manager (RODM) address space and communicates with RODM using an application programming interface (API). Methods are usually small programs that perform specific tasks on data in the data cache.

MIB. See Management Information Base.

MIB application program. A systems management application program used to monitor network devices.

MIB module. In the Simple Network Management Protocol (SNMP), a collection of objects relating to a common management area. See MIB variable.

MIB object. See MIB variable.

MIB tree. In the Simple Network Management Protocol (SNMP), the structure of the Management Information Base (MIB).

MIB variable. In the Simple Network Management Protocol (SNMP), a specific instance of data defined in a MIB module.
MIB view. In the Simple Network Management Protocol (SNMP), the collection of managed objects, known to the agent, that is visible to a particular community.

MIB walking. In the Simple Network Management Protocol (SNMP), a technique of looking for Management Information Base (MIB) tree information when it is presented in a hierarchical format.

Mid-Level Manager (MLM). In Tivoli NetView, the component that performs certain systems and network management tasks (for example, polling, status monitoring, and node discovering) for a defined set of Simple Network Management Protocol (SNMP) devices in the network, thereby offloading these tasks from Tivoli NetView.

MIF. See Management Information Format.

MIPS. A measure of computer processing performance that is equal to one million instructions per second.

MLM. See Mid-Level Manager.

MLM Configuration Application. A Tivoli NetView feature that is used to configure the Mid-Level Manager (MLM).

MNPS. See multinode persistent session.

module. See management module.

monitor. (1) A device that observes and records selected activities within a data processing system for analysis. Possible uses are to indicate significant departure from the norm, or to determine levels of utilization of particular functional units. (2) Software or hardware that observes, supervises, controls, or verifies operations of a system. (3) Software that monitors specific applications or the systems on which the applications rely. Monitors typically monitor information such as available disk space or application errors and compare the information to defined thresholds. When thresholds are exceeded, either system or network administrators can be notified, or an automated response can be performed. (4) In the NetView Graphic Monitor Facility, to open a view that can receive status changes from Tivoli NetView for OS/390. Problem determination and correction can be performed directly from the view. Contrast with browse.

monitoring collection. In Tivoli Distributed Monitoring, a collection of predefined monitors. Several monitoring collections are packaged with Tivoli Distributed Monitoring, but Tivoli administrators can use custom-developed and third-party monitoring collections as well. See Custom monitors.

Monitoring Collection Specification Language (MCSL). A proprietary programming language that is owned by Tivoli Systems Inc. and is used to define monitoring collections for Tivoli Distributed Monitoring.

MPM. See MultiPlatform Manager.

MS. See management services.

MSFP. See management services focal point.

multinode persistent session (MNPS). An LU-LU session that is retained after the failure of VTAM, the operating system, or the hardware.

MultiPlatform Manager (MPM). An application programming interface (API) that was developed by a group of leading technology vendors, including Tivoli Systems Inc., and that enables disparate management systems to interoperate with each other. Tivoli LAN Access and Tivoli IT Director support this API, which means that Tivoli Enterprise software or Tivoli IT Director can provide IT managers with unifying, centralized control over disconnected management resources.

Multiple Virtual Storage/Operator Communication Control Facility (MVS/OCCF). A facility that intercepts messages from the MVS supervisor. Tivoli NetView for OS/390 and MVS/OCCF help a network operator control multiple MVS systems from a central site.

multiplexed distribution (MDist). See MDist.

MultiSystem Manager. In Tivoli NetView for OS/390, the component that manages non-SNA resources, such as those in IP networks, NetWare networks, LAN Network Manager networks, and LAN NetView Management Utilities networks.

multitiered application. An application that is deployed on more than one physical machine. A client/server application is a common multitiered application in which there are two tiers: the client tier (for example, the presentation and the graphical user interface) and the server tier (for example, the service and the database).

MVS/OCCF. See Multiple Virtual Storage/Operator Communication Control Facility.

MVS system symbol. In a sysplex where a customer runs a copy of a given program (such as CICS or Tivoli NetView for OS/390) on more than one MVS image, a symbol that the customer can use to write generic JCL for use by each instance of the given program. An MVS system symbol behaves like a program variable that the sysplex resolves at execution time with the value that is appropriate to the MVS image on which the program instance is running.

name registry. In a Tivoli environment, a name service consisting of a two-dimensional table that maps...
resource names to resource identifiers and corresponding information within a Tivoli Management Region.

name translation. In SNA network interconnection, the conversion of logical unit names, logon mode table names, and class-of-service names used in one network to equivalent names for use in another network. This function can be provided through Tivoli NetView for OS/390 and invoked by a gateway system services control point (SSCP) when necessary. See alias name.

NAT. See network address translation

navigate. In the NetView Graphic Monitor Facility, to move between levels in the view hierarchy.

navigation tree. In Tivoli NetView, a component of the graphical user interface (GUI) that displays a hierarchy of open submaps illustrating the parent-child relationship. The navigation tree enables the network operator to determine which submaps are currently open and to close, restore, or raise the windows that contain submaps.

NCCF. In Tivoli NetView for OS/390, a command that starts the NetView command facility. Also, the use of the abbreviation “NCCF” indicates that various panels and functions are part of the command facility.

nested file package. In Tivoli Software Distribution, a file package that is added as an entry to another file package.

NetBIOS. (1) Network Basic Input/Output System. A standard interface to networks, IBM personal computers (PCs), and compatible PCs, that is used on LANs to provide message, print-server, and file-server functions. Application programs that use NetBIOS do not need to handle the details of LAN data link control (DLC) protocols. (2) See Basic Input/Output System.


NetView AutoBridge. In Tivoli Service Desk for OS/390, an application interface to Tivoli NetView for OS/390 that works with the NetView Bridge Adapter to update the Tivoli Service Desk for OS/390 database and to automate network monitoring. The NetView AutoBridge receives data from NetView alerts, messages, and other applications and uses this data to build and perform Tivoli Service Desk for OS/390 transactions.

NetView Bridge. In Tivoli NetView for OS/390, a set of application programming interfaces (APIs) that enable Tivoli NetView for OS/390 to interact with various types of databases in the OS/390 environment.

NetView Bridge Adapter. In Tivoli Service Desk for OS/390, a feature that provides a connection between the NetView Bridge and the Tivoli Service Desk for OS/390 database. The NetView Bridge Adapter enables the Tivoli Service Desk for OS/390 to act as a NetView database server and works with the NetView AutoBridge or other NetView applications to access problem records logged in the Tivoli Service Desk for OS/390 database.

NetView command authorization table. In Tivoli NetView for OS/390, a set of entries that define an operator’s authorization for accessing commands and (depending on the level of granularity that an enterprise chooses) command keywords and keyword values.

NetView command list language. In Tivoli NetView for OS/390, an interpretive language that is unique to the NetView program and that is used to write NetView command lists in environments where REXX is not supported.

NetView Graphic Monitor Facility (NGMF). In Tivoli NetView for OS/390, a function that provides the network operator with a graphical topological representation of a network and allows the operator to manage the network interactively.

NetView help desk. In Tivoli NetView for OS/390, an online information facility that guides the help desk operator through problem management procedures.

NetView Installation and Administration Facility/2 (NIAF/2). An OS/2-based tool that allows new users of Tivoli NetView for OS/390 or users migrating from a prior release to install, administer, and maintain Tivoli NetView for OS/390. NIAF/2 replaces the Interactive System Productivity Facility-based (ISPF-based) NetView Installation Facility.

NetView management console. See topology console.

NetView management console server. See topology server.

NetView-NetView task (NNT). In Tivoli NetView for OS/390, the task under which a cross-domain NetView operator session runs. See operator station task.

NetView Performance Monitor (NPM). An IBM licensed program that collects, monitors, analyzes, and displays data relevant to the performance of a VTAM telecommunication network. It runs as an online VTAM application program.

NetWare managed site. In a Tivoli environment, a resource that represents (a) a Novell NetWare server on which the Tivoli NetWare repeater (TNWR) is installed and (b) one or more clients. A NetWare managed site enables profiles to be distributed through the NetWare server to one or more specified client PCs using either TCP/IP or IPX.

network address translation (NAT). In a firewall, the conversion of secure IP addresses to external...
registered addresses. This enables communication with external networks but masks the IP addresses that are used inside the firewall.

**network class.** In Tivoli NetView, an object class used for symbols that represent compound objects that may contain objects such as hosts and network devices. Contrast with [connector class](#).

**network computing.** The use of a scalable distributed computing infrastructure that encompasses the key elements of today's networking technologies, such as systems and network management; the Internet and intranets; clients and servers; application programs; databases; transaction processing; and various operating systems and communication protocols.

**Network File System (NFS).** A protocol developed by Sun Microsystems, Incorporated, that allows any host in a network to mount another host’s file directories. Once mounted, the file directory appears to reside on the local host.

**network gateway accounting (NGA).** The NetView Performance Monitor (NPM) subsystem that receives traffic information from the gateway NCP for sessions that flow throughout a network.

**Network Information Center (NIC).** In Internet communications, local, regional, and national groups throughout the world who provide assistance, documentation, training, and other services to users.

**Network Information Services (NIS).** A set of UNIX network services (for example, a distributed service for retrieving information about the users, groups, network addresses, and gateways in a network) that resolve naming and addressing differences among computers in a network.

**network log.** A file that contains (a) messages, commands, and command procedures that have been processed by Tivoli NetView for OS/390 and (b) output resulting from commands, command procedures, and other activity occurring within Tivoli NetView for OS/390.

**network management gateway (NMG).** A gateway between Tivoli NetView for OS/390, which is the SNA network management system, and the network management function of one or more non-SNA networks.

**network management vector transport (NMVT).** A management services request/response unit (RU) that flows over an active session between physical unit management services and control point management services (SSCP-PU session).

**Network News Transfer Protocol (NNTP).** In the Internet suite of protocols, a protocol for the distribution, inquiry, retrieval, and posting of news articles that are stored in a central database.

**network session accounting (NSA).** The NetView Performance Monitor (NPM) subsystem that receives session accounting information from the NCP for sessions that flow throughout a network.

**network topology database.** The representation of the current connectivity between the network nodes within an APPN network. It includes (a) entries for all network nodes and the transmission groups interconnecting them and (b) entries for all virtual routing nodes to which network nodes are attached.

**NFS.** See [Network File System](#).

**NFS client.** A program or system that mounts remote file directories from another host called a Network File System (NFS) server.

**NFS server.** A program or system that allows authorized remote hosts called Network File System (NFS) clients to mount and access its local file directories.

**NGA.** See [network gateway accounting](#).

**NGMF.** See [NetView Graphic Monitor Facility](#).

**NIAF/2.** See [NetView Installation and Administration Facility/2](#).

**NIC.** See [Network Information Center](#).

**NIS.** See [Network Information Services](#).

**NLDM.** In Tivoli NetView for OS/390, a command that starts the session monitor. Also, the use of the abbreviation “NLDM” indicates that various panels and functions are part of the session monitor.

**NMG.** See [network management gateway](#).

**NMVT.** See [network management vector transport](#).

**NNT.** See [NetView-NetView task](#).

**NNTP.** See [Network News Transfer Protocol](#).

**non-generic alert.** In SNA management services (SNA/MS), alert information that is encoded such that it conveys to the receiver the set of screens that should be displayed for the network operator when the alert is received. The use of non-generic alerts requires that the receiver recognize and understand each unique problem for which an alert is sent. Contrast with [generic alert](#).

**NOS.** Network operating system.

**notice.** In a Tivoli environment, a message generated by a systems management operation that contains information about an event or the status of an application. Notices are stored in notice groups. See [bulletin board](#).
notice group. In a Tivoli environment, an application- or operation-specific container that stores and displays notices pertaining to specific Tivoli functions. The Tivoli bulletin board is comprised of notice groups. A Tivoli administrator can subscribe to one or more notice groups; the administrator’s bulletin board contains only the notices that reside in a notice group to which the administrator is subscribed.

notification. (1) An unscheduled, spontaneously generated report of an event that has occurred. (2) In systems management, information emitted by a managed object relating to an event that has occurred within the managed object, such as a threshold violation or a change in configuration status.

NPALU. In the NetView Performance Monitor (NPM), the virtual logical unit generated in an NCP with which the network subsystem communicates.

NPDA. In Tivoli NetView for OS/390, a command that starts the hardware monitor. Also, the use of the abbreviation “NPDA” indicates that various panels and functions are part of the hardware monitor.

NPM. See NetView Performance Monitor.

NSA. See network session accounting.

NT repeater. In a Tivoli environment, the first Windows NT machine on which the Tivoli Remote Execution Service is installed. Using fanout, the NT repeater distributes the Tivoli Remote Execution Service to all other NT clients during the client installation process.

null resource. In the NetView Graphic Monitor Facility, an object that is used only as an aid in formatting and drawing a view. A null resource always shows the status “unknown.”

object. (1) In object-oriented design or programming, a concrete realization of a class that consists of data and the operations associated with that data. (2) An item that a user can manipulate as a single unit to perform a task. An object can appear as text, an icon, or both. (3) In Tivoli NetView for OS/390, the part of a NetView command that follows the verb. The object describes where the action of the verb is to occur.

object dispatcher. See object request broker.

object identifier (OID). An administratively assigned data value of the type defined in abstract syntax notation 1 (ASN.1).

Object Management Group (OMG). A non-profit consortium whose purpose is to promote object-oriented technology and the standardization of that technology. The Object Management Group was formed to help reduce the complexity, lower the costs, and hasten the introduction of new software applications.

object path. In a Tivoli environment, an absolute or relative path to a Tivoli object, similar to paths in file systems.

object reference. In a Tivoli environment, the object identifier (OID) given to an object during its creation.

object registration service (ORS). In Tivoli NetView, a component that creates and maintains a global directory of object managers, their locations, and their protocols. The postmaster daemon uses this directory to route messages and provide location transparency for managers and agents.

object request broker (ORB). In object-oriented programming, software that serves as an intermediary by transparently enabling objects to exchange requests and responses. See Common Object Request Broker Architecture.

ODBC. See Open Database Connectivity.

OID. See object identifier.

OMG. See Object Management Group.

Open Database Connectivity (ODBC). A standard application programming interface (API) for accessing data in both relational and nonrelational database management systems. Using this API, database applications can access data stored in database management systems on a variety of computers even if each database management system uses a different data storage format and programming interface. ODBC is based on the call level interface (CLI) specification of the X/Open SQL Access Group and was developed by Digital Equipment Corporation (DEC), Lotus®, Microsoft®, and Sybase. Contrast with Java Database Connectivity.

operation. In object-oriented design or programming, a service that can be requested at the boundary of an object. Operations include modifying an object or disclosing information about an object.

operations and administration. The Tivoli management discipline that addresses the automation of activities that ensure the operational integrity and reliability of a network computing system. See availability management, deployment management, and security management.

operator. A person or a program that manages activities that are controlled by a specific computer program.

operator profile. In Tivoli NetView for OS/390, a specification of the resources and activities over which a network operator has control. The profile is stored in a file that is activated when the operator logs on.
operator station task (OST). In Tivoli NetView for OS/390, the task that establishes and maintains the online session with the network operator. There is one operator station task for each network operator who logs on to Tivoli NetView for OS/390. See NetView-NetView task.

ORB. See object request broker.

ORS. See object registration service.

oserv. The name of the object request broker used by the Tivoli environment. Oserv runs on the TMR server and each TMR client.

OST. See operator station task.

package definition file (PDF). In Tivoli IT Director, an ASCII text file that contains predefined workstation, sharing, and inventory property settings for a file package.

packet. In data communication, a sequence of binary digits, including data and control signals, that is transmitted and switched as a composite whole. The data, control signals, and, possibly, error control information are arranged in a specific format. (1)

parameter. (1) A variable that is given a constant value for a specified application and that may denote the application. (1) (2) In Common User Access (CUA®) architecture, a variable used in conjunction with a command to affect its result. (3) An item in a menu for which the user specifies a value or for which the system provides a value when the menu is interpreted. (4) Data passed to a program or procedure by a user or another program, namely as an operand in a language statement, as an item in a menu, or as a shared data structure. (5) In Tivoli NetView for OS/390, a part of a command's object. (6) See keyword and keyword parameter.

parent process. In the UNIX operating system, a process that creates other processes. See child process and fork.

parent resource. In the NetView Graphic Monitor Facility, a resource that has one or more child resources below it in a hierarchy.

PassTicket. In RACF® secured sign-on, a dynamically generated, random, one-time-use, password substitute that a workstation or other client can use to sign on to the host rather than sending a RACF password across the network.

PassTicket application key. In RACF secured sign-on, an encryption key that is used in the creation and evaluation of a PassTicket. The PassTicket application key is sometimes referred to as the “secured sign-on application key.”

path. (1) A list of one or more directory names and an object name (such as the name of a file) that are separated by an operating system-specific character, such as the slash (/) in UNIX operating systems, the backslash (\) in Windows® operating systems, and the semicolon (;) in OS/2 operating systems. The directory names detail the path to follow, in left-to-right order, to locate the object within the file system. This concept of path is also known as the “pathname.” (2) A list of directory names, usually separated by a colon (:), that are to be searched (in left-to-right order) to locate an object. This concept of path is also known as the “search path.” (3) See absolute path, directory, relative path, root directory, and working directory.

pathname. See path.

path test. A test provided by Tivoli NetView for OS/390 that enables a network operator to determine whether a path is available between two LUs that are currently in session.

pattern-matching character. A special character such as an asterisk (*) or a question mark (?) that can be used to represent one or more characters. Any character or set of characters can replace a pattern-matching character.

PC agent. In a Tivoli environment, software installed on a client PC that enables Tivoli operations to execute on the PC. See PC managed node.

PC managed node. In a Tivoli environment, an object that represents a client PC. The Tivoli Management Framework can communicate with the client PC only if the PC agent is installed on the PC. Client PCs are most often referred to as PC managed nodes.

PDF. (1) See package definition file (2) See Portable Document Format.

performance class. In Tivoli NetView for OS/390, a description of an objective or commitment of performance. It consists of a performance class name, boundary definitions, response time definition, response time ranges, and response time percentage objectives. Sessions may be assigned performance classes.

persistent LU-LU session. See persistent session.

persistent session. (1) In Tivoli NetView for OS/390, a network management session that remains active even though there is no activity on the session for a specified period of time. (2) An LU-LU session that...
VTAM retains after the failure of a VTAM application program. Following the application program's recovery, the application program restores or terminates the session. This session is sometimes referred to as a "single-node persistent session." See [multinode persistent session](#).

pipeline.  (1) A serial arrangement of processors or a serial arrangement of registers within a processor. Each processor or register performs part of a task and passes results to the next processor; several parts of different tasks can be performed at the same time. (2) To perform processes in series. (3) To start execution of an instruction sequence before the previous instruction sequence is completed to increase processing speed. (4) In Tivoli NetView for OS/390, a message processing procedure that consists of one or more programs known as stages.

pixel map.  (1) A three-dimensional array of bits. A pixel map can be thought of as a two-dimensional array of pixels, with each pixel being a value from zero to 2 to the power N -1, where N is the depth of the pixel map. (2) In the X Window System, a data type to which icons, originally created as bitmaps, are converted.

pixmap. See [pixel map](#).

platform. An ambiguous term that may refer to the hardware, the operating system, or a combination of the hardware and the operating system on which software programs run.

plex. A Printing Systems Manager (PSM) attribute used for defining the capability of a printer to support different placements of output images on a medium. For example, the plex attribute could specify whether the printer is to support simplex or tumble mode.

Plus module. See [Tivoli Plus module](#).

policy. In a Tivoli environment, a set of rules that are applied to managed resources. A specific rule in a policy is referred to as a "policy method."

policy region. In a Tivoli environment, a group of managed resources that share one or more common policies. Tivoli administrators use policy regions to model the management and organizational structure of a network computing environment. The administrators can group similar resources, define access to and control the resources, and associate rules for governing the resources. The policy region contains resource types and the list of resources to be managed. A policy region is represented on the Tivoli desktop by an icon that resembles a capitol building (dome icon). When a Tivoli Management Region (TMR) is created, a policy region with the same name is also created. In this case, the TMR has only one policy region. However, in most cases, a Tivoli administrator creates other policy regions and subregions to represent the organization of the TMR. A TMR addresses the physical connectivity of resources whereas a policy region addresses the logical organization of resources.

policy subregion. In a Tivoli environment, a policy region created or residing in another policy region. When a policy subregion is created, it initially uses the resource and policy properties of the parent policy region. The Tivoli administrator can later change or customize these properties to reflect the specific needs and differences of the subregion.

polling. (1) On a multipoint connection or a point-to-point connection, the process whereby data stations are invited, one at a time, to transmit. (2) Interrogation of devices for such purposes as to avoid contention, to determine operational status, or to determine readiness to send or receive data. (3) In network management, the process by which a manager interrogates one or more managed nodes at regular intervals.

populate. In a Tivoli environment, to fill a profile with information that is to be distributed to the subscribing managed resources.

port. To modify a computer program to enable it to run on a different platform.

Portable Document Format (PDF). A standard specified by Adobe Systems, Incorporated, for the electronic distribution of documents. PDF files are compact; can be distributed globally via e-mail, the Web, intranets, or CD-ROM; and can be viewed with the Acrobat Reader, which is software from Adobe Systems that can be downloaded at no cost from the Adobe Systems home page.

portmapper. A program that maps client programs to the port numbers of server programs. Portmapper is used with remote procedure call (RPC) programs.

positional operand. (1) An operand in a language statement that has a fixed position. (2) Contrast with [keyword operand](#). (3) See [definition statement](#).

postmaster. In Tivoli NetView, a process (daemon) that directs network management information between multiple application programs and agents running concurrently. The postmaster determines the route by using specified addresses or a routing table that is configured in the object registration service.

PPI. See [program-to-program interface](#).

presentation graphics feature (PGF). In the NetView Performance Monitor (NPM), a feature used in conjunction with the Graphical Data Display Manager (GDDM) to generate online graphs in the NPM graphic subsystem.
presentation services command processor (PSCP). In Tivoli NetView, a facility that processes requests from a user terminal and formats displays to be presented at the user terminal.

primary database. In Tivoli NetView for OS/390, the main database provided to the NetView user for recording error data. See secondary database.

primary POI task (PPT). In Tivoli NetView for OS/390, the subtask that processes all unsolicited messages that are received from the VTAM program operator interface (POI) and delivers them to the controlling operator or to the command processor. The PPT also processes (a) the initial command that is specified to execute when NetView is initialized and (b) timer request commands that are scheduled to execute under the PPT.

primary window. In OSF/Motif, the top-level window in an application program that can be minimized or represented by an icon. See submap window.

principal name. (1) In Kerberos, the name by which the Kerberos principal is identified. The principal name consists of three parts: a service or user name, an instance name, and a realm name. (2) In a Tivoli environment, an operating system user ID that is associated with a Tivoli administrator.

principal password. In Kerberos, the password that corresponds to the principal name. This password is used to authenticate services and users to each other.

print file document. A Printing Systems Manager (PSM) object that represents text or data to be printed by a job. Contrast with print resource document.

Printing Systems Manager (PSM). An IBM licensed program that applies print administration and management technology to a cross-platform, client/server print system. PSM provides a set of (a) printing functions for submitting and controlling print jobs and (b) systems management and operator functions to control print spoolers and print supervisors. PSM is based on the Palladium distributed print system.

print resource document. A Printing Systems Manager (PSM) object that represents a resource, such as graphics or fonts, used by a job to print a print file document.

Print Services Facility™ (PSF) for AIX. An IBM licensed printer driver program that produces printer commands from the data sent to it.

private key. In computer security, a key that is known only to its owner. Contrast with public key. See public key cryptography.

profile. In a Tivoli environment, a container for application-specific information about a particular type of resource. A Tivoli application specifies the template for its profiles; the template includes information about the resources that can be managed by that Tivoli application.

A profile is created in the context of a profile manager; the profile manager links a profile to the Tivoli resource (for example, a managed node) that uses the information contained in the profile. A profile does not have any direct subscribers.

profile manager. In a Tivoli environment, a container for profiles that links the profiles to a set of resources, called “subscribers.” A profile manager can contain (a) profiles of multiple types or (b) multiple profiles of the same type. Tivoli administrators use profile managers to organize and distribute profiles. A profile manager is created in the context of a policy region and is a managed resource in a policy region. See subscription list.

program-to-program interface (PPI). In Tivoli NetView for OS/390, a facility that allows user programs to send data buffers to or receive data buffers from other user programs. It also allows system and application programs to send alerts to the NetView hardware monitor.

prototype profile. In a Tivoli environment, a model profile from which a Tivoli administrator can create other profiles, often by cloning the prototype profile.

proxy endpoint. In Tivoli Distributed Monitoring, a representation for an entity (such as a network device or a host) that functions as a subscriber for Tivoli Distributed Monitoring profiles. A Tivoli administrator associates each proxy endpoint with a managed node; several proxy endpoints can be associated with a single managed node.

PSCP. See presentation services command processor.

PSF. Print Services Facility. See Print Services Facility for AIX.

PSM. See Printing Systems Manager.

public key. In computer security, a key that is made available to everyone. Contrast with private key. See public key cryptography.

public key cryptography. In computer security, cryptography in which public keys and private keys are used for encryption and decryption.

pull. A network operation that initiates an action by requesting the action from a resource. Contrast with push.

push. A network operation that sends information to resources. Contrast with pull.
query. In a Tivoli environment, a combination of statements that are used to search the configuration repository for systems that meet certain criteria.

query library. In a Tivoli environment, a facility that provides a way to create and manage Tivoli queries.

RACF. See Resource Access Control Facility.

RACF secured sign-on. In the Resource Access Control Facility (RACF), a function that enables workstations and other clients to sign on to the host and communicate in a secure way without having to send RACF passwords across the network. See PassTicket and PassTicket application key.

RDBMS. See relational database management system.

RDBMS Interface Module (RIM). In the Tivoli Management Framework, the module in the distributed object database that contains information about the installation of the relational database management system (RDBMS).

real object. In the NetView Graphic Monitor Facility, an object that represents an actual resource. See aggregate object.

real resource. (1) In VTAM, a resource identified by its real name and its real network identifier. (2) In the NetView Graphic Monitor Facility, an individual network resource represented by a real object.

recommended action. The procedures that Tivoli NetView for OS/390 recommends for determining and correcting the causes of network problems.

recording filter. In Tivoli NetView for OS/390, the function that determines which events, statistics, and alerts are stored in a database.

reference implementation. An implementation by which other implementations are judged for conformance to a standard or are tested for interoperability.

reference model. In the context of Tivoli software, the model configuration for a system or set of systems that is used to maintain consistent configurations in a distributed environment. In Tivoli Inventory, reference models are created in the configuration repository.

registered name. In a Tivoli environment, the name by which a particular resource is registered with the name registry when it is created.

registration file. See application registration file, field registration file, local registration file, and symbol registration file.

regular command. See immediate command.

relation. (1) In a relational database, a set of entity occurrences that have the same attributes. (T) (2) The comparison of two expressions to see if the value of one is equal to, less than, or greater than the value of the other. (3) In a relational database, a table that identifies entities and their attributes.

relational database. A database in which the data are organized and accessed according to relations. (T)

relational database management system (RDBMS). A collection of hardware and software that organizes and provides access to a relational database.

relative path. A path that begins with the working directory. Contrast with absolute path.

remote distribution. In a Tivoli environment, a distribution to target machines in a connected Tivoli Management Region.

remote host. Any host on a network except the host at which a particular operator is working.

Remote Operations Service (ROPS). In Communications Server, an application program on a client workstation that processes commands that are issued by Tivoli NetView for OS/390 through the Service Point Application (SPA) Router, thus enabling Tivoli NetView for OS/390 to manage distributed networks and application programs.

remote procedure call (RPC). (1) A facility that a client uses to request the execution of a procedure call from a server. This facility includes a library of procedures and an external data representation. (2) A client request to a service provider located in another node.

repeater. (1) A node of a local area network; a device that regenerates signals in order to extend the range of transmission between data stations or to interconnect two branches. (T) (2) See repeater site.

repeater range. In a Tivoli environment, the Tivoli clients that receive data from the repeater site.

repeater site. In a Tivoli Management Region, a managed node that is configured with the MDist feature. A repeater site receives a single copy of data and distributes it to the next tier of clients.

requester. See client.

Request for Comments (RFC). In Internet communications, the document series that describes a
part of the Internet suite of protocols and related experiments. All Internet standards are documented as RFCs.

**resource.** (1) Any facility of a computing system or operating system required by a job or task, and including main storage, input/output devices, the processing unit, data sets, and control or processing programs. (2) In Tivoli NetView for OS/390, any hardware or software that provides function to the network. (3) See managed resource.

**Resource Access Control Facility (RACF).** An IBM licensed program that provides for access control by identifying and verifying the users of the system, by authorizing access to protected resources, by logging the detected unauthorized attempts to enter the system, and by logging the detected accesses to protected resources.

**resource label.** In the NetView Graphic Monitor Facility, the textual information that identifies a particular aggregate or real resource. The resource label is displayed next to the resource symbol and cannot be changed by the network operator.

**resource level.** In Tivoli NetView for OS/390, the hierarchical position of a device (and the software contained within it) in a data processing system. For example, a first-level resource could be the communication controller, and the second-level resource could be the line connected to it.

**resource manager.** In Tivoli NetView for OS/390, an application program that manages specific network resources. Each resource manager is assigned a unique range of command indicators that specify the command support characteristics for the resources that it manages. The resource manager provides information to the NetView Graphic Monitor Facility (NGMF).

**Resource Object Data Manager (RODM).** In Tivoli NetView for OS/390, a component that operates as a cache manager and that supports automation applications. RODM provides an in-memory cache for maintaining real-time data in an address space that is accessible by multiple applications.

**resource resolution table (RRT).** In NetView Performance Monitor (NPM), a table that contains the names of the network resources for which data is to be collected. The NPM RRT corresponds with an NCP and is built by NPMGEN from an NCP Stage I and an NCP RRT.

**resource status collector.** In Tivoli NetView for OS/390, a function that collects status information on monitored resources and forwards this information to the resource status manager.

**resource status manager.** The part of the NetView Graphic Monitor Facility that maintains a database of SNA resource status information and that forwards this information to all attached server workstations.

**resource symbol.** In the NetView Graphic Monitor Facility, a geometric shape (such as a line, square, or octagon) that represents a particular kind of resource and indicates whether that resource is one resource or a composite of a group of resources.

**resource type.** (1) In a Tivoli environment, one of the properties of a managed resource. Resource types are defined in the default policy for a policy region. (2) In Tivoli NetView for OS/390, one of the three elements, which also include data type and display type, that are used to describe the organization of panels. Resource types in one category include central processing unit, channel, control unit, and I/O device; and in another category, they include communication controller, adapter, link, cluster controller, and terminal.

**response level.** See alarm level.

**response time.** (1) The elapsed time between the end of an inquiry or demand on a computer system and the beginning of the response; for example, the length of time between an indication of the end of an inquiry and the display of the first character of the response at a user terminal. (2) For response time monitoring, the time from the activation of a transaction until a response is received, according to the response time definition coded in the performance class.

**response time monitor (RTM).** A feature available with certain hardware devices to allow measurement of response times, which may be collected and displayed by Tivoli NetView for OS/390.

**review file.** In the NetView Performance Monitor (NPM), a VSAM key-sequenced data set (KSDS) containing data collected and recorded as a result of a network start display command or start monitor command.

**RFC.** See Request for Comments.

**RIM.** See RDBMS Interface Module.

**RIM repository.** See configuration repository.

**RODM.** See Resource Object Data Manager.

**RODM-based view.** In the NetView Graphic Monitor Facility (NGMF), a view that is predefined or dynamically built based on definitions in RODM. Examples of a RODM-based view are network views, exception views, configuration views, and more-detail views.

**RODM resource.** In the context of NetView Graphic Monitor Facility (NGMF) views, an object created in RODM to represent a resource. These objects can be created by loader files, user applications, or by the SNA topology manager.
role. See authorization role.

root directory. The highest level directory in a hierarchical file system.

root user. In the UNIX operating system, a user who has superuser authority.

ROPS. See Remote Operations Service.

router. (1) A computer that determines the path of network traffic flow. The path selection is made from several paths based on information obtained from specific protocols, algorithms that attempt to identify the shortest or best path, and other criteria such as metrics or protocol-specific destination addresses. (2) An attaching device that connects two LAN segments, which use similar or different architectures, at the reference model network layer. (3) In OSI terminology, a function that determines a path by which an entity can be reached. (4) Contrast with bridge.

RPC. See remote procedure call.

RRT. See resource resolution table.

RS/6000®. A family of workstations and servers based on IBM's POWER architecture. They are primarily designed for running multiluser numerical computing applications that use the AIX operating system.

RTM. See response time monitor.

rule. In the Tivoli Enterprise Console, a set of one or more logical statements that enable the event server to recognize relationships among events (event correlation) and to execute automated responses accordingly.

rule base. In the Tivoli Enterprise Console, a set of rules and the event class definitions for which the rules are written. The Tivoli Enterprise Console uses the rule base in managing events. An organization can create many rule bases, with each rule base fulfilling a different set of needs for network computing management.

schema. The set of statements, expressed in a data definition language, that completely describe the structure of a database.

scope check. In Tivoli NetView for OS/390, the process of verifying that an operator is authorized to issue a particular command. Contrast with span check. See scope of command authorization.

scope of command authorization. In Tivoli NetView for OS/390, the level of access authority that a system programmer or system administrator grants to a network operator to use various commands. See scope check.

script. (1) A computer program that is interpreted. (2) See shell script.

script stub. A placeholder for a particular shell script. For example, the Tivoli Module Builder generates a script stub (using a skeleton file) if a developer does not provide the script for implementing a particular task or monitor when defining the task or monitor; the script stub then displays a message that the script executed successfully and displays any variables that were passed to the script.

search path. See path.

seat. A slang term that refers to the number of licensed users of a software product, which is the same as the number of installations of the product. For example, if there were 100 Lotus Notes™ seats, there would be 100 licensed users of Lotus Notes (or 100 installations of Lotus Notes).

secondary database. One of two databases provided by Tivoli NetView for OS/390 for recording data. It provides backup or a temporary storage alternative to the primary database. See primary database.

security group. In a Tivoli environment, a group of managed resources over which a Tivoli administrator is granted authority. Examples of a security group include a policy region and the administrator collection.

security management. The Tivoli management discipline that addresses the organization’s ability to control access to applications and data that are critical to its success. See availability management, deployment management and operations and administration.

seed file. In Tivoli NetView, a file that contains a list of nodes within an Administrative Domain, which the automatic discovery function uses to accelerate the generation of the network topology map.

segment. (1) A portion of a computer program that may be executed without the entire computer program being resident in main storage. (2) A group of display elements. (3) A section of cable between components or devices. A segment may consist of a single patch cable, several patch cables that are
connected, or a combination of building cable and patch cables that are connected. (4) In the Enhanced X-Window Toolkit, one or more lines that are drawn but not necessarily connected at the endpoints. (5) In LANs or WANs, a subset of nodes in a network or subnet that are connected by a common physical medium.

**senior role.** See [authorization role](#).

**server.** A functional unit that provides services to one or more clients over a network. Examples include a file server, a print server, and a mail server.

**server workstation.** In the NetView Graphic Monitor Facility, a workstation with the graphic data server. This workstation uses the graphic monitor and the view administrator for administrative functions. The server workstation sends status information to client workstations over an LU 6.2 session.

**Service Level Reporter (SLR).** A licensed program that generates management reports from data sets such as System Management Facility (SMF) files.

**service point (SP).** An entry point that supports applications providing network management for resources that are not under its direct control as an entry point. Each resource is either under the direct control of another entry point or not under the direct control of any entry point. A service point accessing these resources is not required to use SNA sessions (unlike a focal point).

**Service Point Application Router.** In Communications Server, software that receives commands issued from Tivoli NetView for OS/390 and sends these commands to an application program, called the Remote Operations Service (ROPS), to be processed on a client workstation.

**service point command facility command service (SPCS).** In Tivoli NetView for OS/390, an extension of the command facility that allows the host processor to communicate with a service point by using the communication network management (CNM) interface.

**session data.** Session awareness data, session trace data, and session response time data that Tivoli NetView for OS/390 collects.

**session manager.** A product, such as NetView Access Services, that allows a user at a terminal to log on to multiple applications concurrently.

**session monitor.** In Tivoli NetView for OS/390, the component that collects and correlates session-related data and provides online access to this information. Contrast with [hardware monitor](#).

**session setup failure notification (SSFN).** In Tivoli NetView for OS/390, session awareness data that is provided when there is a failure. It identifies the system services control point (SSCP) that detects the error, the SSCP partners that are involved, and the names of the session partners affected.

**session statistics file.** In the NetView Performance Monitor (NPM), an online VSAM key-sequenced data set (KSDS) used for storing session data.

**session trace.** In Tivoli NetView for OS/390, the function that collects session trace data for sessions involving specified resource types or involving a specific resource.

**session trace data.** Data, relating to sessions, that is collected by Tivoli NetView for OS/390 whenever a session trace is started and that consists of session activation parameters, VTAM path information unit (PIU) data, and NCP data.

**severity level.** In the Tivoli Enterprise Console, a classification for an event that indicates its degree of severity. Severity levels can be modified by a user or a Tivoli Enterprise Console rule. The predefined severity levels, in order of descending severity, include: fatal, critical, warning, minor, harmless, and unknown.

**shared application program.** In Tivoli NetView, an application program that serves multiple action requests; however, only one instance of the application program can run in a given graphical user interface (GUI).

**shared submap.** In Tivoli NetView, a submap on which multiple application programs manage objects on the application plane. Shared submaps allow application programs to cooperatively contribute information to the same submap. Contrast with [exclusive submap](#).

**shell.** A software interface between a user and the operating system of a computer. Shell programs interpret commands and user interactions on devices such as keyboards, pointing devices, and touch-sensitive screens and communicate them to the operating system. Shells simplify user interactions by eliminating the user’s concern with operating system requirements. A computer may have several layers of shells for various levels of user interaction.

**shell procedure.** See [shell script](#).

**shell prompt.** In the UNIX operating system, the character string on the command line indicating that the system can accept a command (typically the $ character).

**shell script.** In the UNIX operating system, a series of commands, combined in a file, that carry out a particular
function when the file is run or when the file is specified as a value to the SH command.

**show cause.** The reason code in the record maintenance statistics (RECMS) that indicates to VTAM or to Tivoli NetView for OS/390 the threshold that was exceeded and whether the threshold has been dynamically altered.

**SIA.** System Information Agent. See Tivoli Distributed Monitoring the product that replaces the System Information Agent.

**signal.** In computer software, a message that is sent to a process to change its behavior based on the value sent to it.

**signature.** In computer software, the name of an operation and its parameters.

**simple connection.** In Tivoli NetView, the representation of connectivity as seen from one endpoint of a connection.

**Simple Network Management Protocol (SNMP).** In the Internet suite of protocols, a network management protocol that is used to monitor routers and attached networks. SNMP is an application layer protocol. Information on devices managed is defined and stored in the application’s Management Information Base (MIB).

**singular filter.** A filter that identifies a host, subnet, or all hosts with a single expression.

**skeleton file.** A program template that the Tivoli Module Builder uses to generate any text-based file, including scripts, Java or C source files, build files, and help text. A skeleton file includes substitution variables that are replaced at run time. The values for these variables originate from user-defined variables or values specified in a component description file (CDF) or a global description file (GDF) file.

**SLR.** See Service Level Reporter.

**SMF.** See System Management Facility.

**SMIT.** See System Management Interface Tool.

**SMS.** See Storage Management Subsystem.

**snapshot.** In Tivoli NetView, a copy of a map that reflects the topology and status of the map’s nodes and links at a given moment in time.

**SNATM.** See SNA topology manager.

**SNA topology manager (SNATM).** In Tivoli NetView for OS/390, a component that dynamically collects status and topology data into the Resource Object Data Manager (RODM) for display by the NetView Graphic Monitor Facility (NGMF). SNATM includes the function formerly provided by the APPN Topology and Accounting Manager (APPNTAM) feature of NetView for MVS V2R4.

**SNMP.** See Simple Network Management Protocol.

**Software Installer for OS/2.** An OS/2-based tool that is used to install workstation functions such as the NetView Graphic Monitor Facility.

**software management.** See change management.

**source host.** In Tivoli Software Distribution, the managed node on which the files and directories referenced in a file package reside.

**span.** In Tivoli NetView for OS/390, a user-defined group of network resources within a single domain. Spans provide a level of security by allowing the system administrator to define (a) the resources to which an operator can issue commands, (b) the views of resources that an operator can display, and (c) the resources in a view that an operator is allowed to see (an operator may not be authorized to see all the resources in a particular view). See span check.

**span check.** In Tivoli NetView for OS/390, the process of verifying that an operator is authorized to perform actions on a network resource, a NetView Graphic Monitor Facility (NGMF) view, or a resource within a view. Contrast with scope check.

**SPA Router.** See Service Point Application Router.

**SPCF.** See service point command facility.

**SQL.** A programming language that is used to define and manipulate data in a relational database.

**SRF.** See symbol registration file.

**SSFN.** See session setup failure notification.

**SSI.** See subsystem interface.

**stage.** In Tivoli NetView for OS/390, a program that processes messages in a NetView pipeline. Stages send messages to each other serially.

**statistics record.** In Tivoli NetView for OS/390, a resource-generated database record that contains various statistics about a resource.

**status monitor.** In Tivoli NetView for OS/390, a component that collects and summarizes information on the status of resources defined in a VTAM domain.

**Storage Management Subsystem (SMS).** A DFSMS/MVS facility that is used to automate and centralize the management of storage. Using SMS, a storage administrator describes data allocation characteristics, performance and availability goals, backup and retention requirements, and storage requirements to the system.
Structured Query Language. See SQL.

subagent. In the Simple Network Management Protocol (SNMP), something that provides an extension to the utility provided by the SNMP agent.

submap. In Tivoli NetView, a particular view of some aspect of a network that displays symbols representing objects. The application program that creates a submap determines what part of the network the submap displays.

submap pane. The area of a submap window in which the submap is displayed.

submap stack. In Tivoli NetView, a component of the graphical user interface shown on the left side of each submap window. The submap stack represents the navigational path used to reach the particular submap, and it can be used to select a previously viewed submap.

submap window. In Tivoli NetView, the graphical component that contains a menu bar, a submap viewing area, a status line, and a button box. A user can display multiple submap windows of an open map and an open snapshot at any given time. See primary window.

subnet. Any group of nodes that have a set of common characteristics, such as the same network ID.

subscriber. In a Tivoli environment, a Tivoli client, a profile manager, or any endpoint type (for example, a PC managed node or a proxy endpoint) that is subscribed to a profile manager. Although profiles are distributed to a subscriber, the subscriber may or may not be the final destination of the profile distribution.

subscription. In a Tivoli environment, the process of identifying the subscribers to which profiles will be distributed.

subscription list. In a Tivoli environment, a list that identifies the subscribers to a profile manager. Including a profile manager on a subscription list (in effect, a list within a list) is a way of subscribing several resources simultaneously rather than adding each one individually. In Tivoli Plus modules, a profile manager functions as a subscription list.

subsystem interface (SSI). The MVS interface by which routines (IBM-, vendor-, or installation-written) request services of, or pass information to, subsystems. The SSI is used by Tivoli NetView for OS/390 to receive system messages and enter system commands (when used with extended MCS consoles, it is used to receive commands, not messages), and to communicate with other instances of Tivoli NetView for OS/390.

super role. See authorization role.

superuser authority. In the UNIX operating system, the unrestricted authority to access and modify any part of the operating system, usually associated with the user who manages the system. See root user.

suppression character. In Tivoli NetView for OS/390, a user-defined character that is coded at the beginning of a command list statement or a command to prevent the statement or command from appearing on the operator’s terminal screen or in the network log.

symbol. In Tivoli NetView, a picture or an icon on a submap that represents an object (a network resource or an application). Each symbol belongs to a class, represented by the symbol's shape, and to a subclass, represented by the design within the shape. The symbol reflects characteristics of the object it represents, such as its status; it also has characteristics of its own, such as behavior.

symbol registration file (SRF). In Tivoli NetView, a file used to define symbol classes and subclasses.

class. In Tivoli Distributed Monitoring, a particular view of some aspect of a network that displays symbols representing objects. The application program that creates a submap determines what part of the network the submap displays.

system configuration. A process that specifies the devices and programs that form a particular data processing system.

System Authorization Facility (SAF). An interface defined by MVS that enables programs to use system authorization services in order to protect access to resources such as data sets and MVS commands. The IBM Resource Access Control Facility (RACF) is a product that uses the SAF interface.

sysplex. A set of MVS or OS/390 systems communicating and cooperating with each other through certain multisystem hardware components and software services to process customer workloads. This term is derived from “system complex.”

System Management Interface Tool (SMIT). An interface tool of the AIX operating system for installing, maintaining, configuring, and diagnosing tasks.

System Management Facility (SMF). A standard feature of OS/390 that collects and records a variety of system and job-related information.

Systems Interconnection (OSI) resources and their status across all layers of the OSI architecture.
configuration, identifying faults, securing access, accounting for resource usage, and analyzing performance.

T

TACF. See Tivoli Access Control Facility.

TACF database. In Tivoli Security Management, a database that contains the customized rules that the authorization daemon for the Tivoli Access Control Facility (TACF) uses to allow or to deny resource accesses in the UNIX environment.

TACF lookaside database. In Tivoli Security Management, a database that provides ID-to-name resolution, thereby enabling the Tivoli Access Control Facility (TACF) to convert UNIX IDs (user IDs, group IDs, IP addresses, and port numbers) to names at run time.

TAF. See terminal access facility.

TAP. See Telocator Alphanumeric Protocol.

target. See endpoint and execution target.

target host. See endpoint.

task. (1) In a multiprogramming or multiprocessing environment, one or more sequences of instructions treated by a control program as an element of work to be accomplished by a computer. (2) In a Tivoli environment, the definition of an action that must be routinely performed on various managed nodes throughout the network. A task defines the executables to be run when the task is executed, the authorization role required to execute the task, and the user or group name under which the task will execute.

task endpoint. See endpoint.

task library. In a Tivoli environment, a container in which a Tivoli administrator can create and store tasks and jobs.

Task Library Language (TLL). In a Tivoli environment, a programming language used to define a task library. The TLL definition can be used to copy a task library from one installation to another. The TLL also allows the arguments for each task to be described such that graphical user interface (GUI) tools can interpret them and present an interface for operators who want to create the tasks.


Telocator Alphanumeric Protocol (TAP). An industry standard protocol for the input of paging requests.

terminal access facility (TAF). In Tivoli NetView for OS/390, a facility that allows a network operator to control a number of subsystems. In a full-screen or operator control session, operators can control any combination of such subsystems simultaneously.

threshold. In software products, a value that defines a limit for a monitored condition. The monitored condition, the significance of the limit, and the particular software product’s response when the monitored condition reaches the specified threshold vary widely according to product.

throttle. In Tivoli NetView, a condition defined in the filter table and used to regulate the flow of traps.

time to live (TTL). A technique used by best-effort delivery protocols to inhibit endlessly looping packets. The packet is discarded if the TTL counter reaches 0.

Tivoli Access Control Facility (TACF). In Tivoli Security Management, an object-oriented security system that runs on UNIX-based operating systems and provides security functions that are not available on UNIX (such as an access rule database, an audit log, and administration tools). TACF is invoked immediately after the operating system has completed its initialization, and it places hooks in system services that should be protected, thereby enabling control to be passed to TACF before the services are performed.

Tivoli administrator. In a Tivoli environment, a system administrator who has been authorized to perform systems management tasks and manage policy regions in one or more networks. Each Tivoli administrator is represented by an icon on the Tivoli desktop.

Tivoli Application Development Environment. A Tivoli toolkit that contains the complete application programming interface (API) for the Tivoli Management Framework. This toolkit enables customers and Tivoli Partners to develop their own applications for the Tivoli environment.

Tivoli Application Extension Facility. A Tivoli toolkit that enables customers to extend the capabilities of Tivoli applications. For example, customers can add fields to a dialog, create custom attributes and methods for application resources, or create custom icons and bitmaps.

Tivoli client. A client of a Tivoli server. See TMR client and TMR server.

Tivoli Cross-Site®. The integrated suite of Tivoli products for managing an e-commerce environment to ensure that Web resources are secure and available and to enable applications and information to be distributed and maintained across the extended enterprise.

Tivoli Decision Support. A Tivoli product that consolidates, transforms, and presents IT data in many different views, enabling an enterprise to gain insight
into patterns and relationships among the data and to make critical business decisions based on this data.

**Tivoli desktop.** In a Tivoli environment, the desktop that system administrators use to manage their network computing environment.

**Tivoli Developer Kit.** See Tivoli Module Designer.

**Tivoli Distributed Monitoring.** A Tivoli product that monitors system resources, initiates any necessary corrective actions, and informs system administrators of potential problems. Tivoli Distributed Monitoring consists of a group of monitors that are installed on each managed node that is to be monitored. It resolves some events on its own and may send others to the Tivoli Enterprise Console.

**Tivoli Enterprise Console.** A Tivoli product that collects, processes, and automatically initiates corrective actions for system, application, network, and database events; it is the central control point for events from all sources. The Tivoli Enterprise Console provides a centralized, global view of the network computing environment; it uses distributed event monitors to collect information, a central event server to process information, and distributed event consoles to present information to system administrators.

**Tivoli Enterprise software.** The integrated suite of Tivoli products for systems management in a large organization. These products enable system administrators to manage their network computing enterprise according to the disciplines of availability management, deployment management, operations and administration, security management, and service-level management. This suite includes Tivoli Global Enterprise Manager, Tivoli NetView for OS/390, and Tivoli Decision Support.

**Tivoli environment.** The Tivoli applications, based upon the Tivoli Management Framework, that are installed at a specific customer location and that address network computing management issues across many platforms. In a Tivoli environment, a system administrator can distribute software, manage user configurations, change access privileges, automate operations, monitor resources, and schedule jobs.

**Tivoli Event Integration Facility.** A Tivoli toolkit that provides a simple application programming interface (API) to enable customers and Tivoli Partners to develop new event adapters that can forward events to the Tivoli Enterprise Console. A customer can also translate events from third-party or in-house applications.

**Tivoli GEM.** See Tivoli Global Enterprise Manager.

**Tivoli GEM module.** In a Tivoli environment, a management module that enables a particular application or business system to be managed by the Tivoli Global Enterprise Manager (Tivoli GEM).

**Tivoli Global Enterprise Manager (Tivoli GEM).** A Tivoli product that allows system administrators to graphically monitor, control, and configure applications residing in distributed and host (S/390®) environments and to use the concept of business systems management to organize related components, thereby providing a business perspective for management decisions. Tivoli Global Enterprise Manager gives information technology staff a logical view of the computing environment; this view shows, at a glance, the status of the multiple applications that comprise the enterprise’s business system, including application components, the relationships among and between components, and the flow of data between the applications. By providing this view from a business perspective, Tivoli Global Enterprise Manager enables system administrators to quickly make determinations about the business impact of any component failure. Addressing technology problems from the business perspective greatly improves the effectiveness of system administrators and provides a higher level of service to users.

**Tivoli install image.** In a Tivoli environment, a file that resides on a CD or in a file system and contains a Tivoli product to be installed. A Tivoli install image can be used to install the Tivoli Management Framework or to install an application onto the Framework for the first time. A single CD often includes both a Tivoli install image and a Tivoli upgrade image, and it may include Tivoli install images for more than one application. Contrast with Tivoli upgrade image.

**Tivoli Inventory.** A Tivoli product that enables system administrators to gather hardware and software information for a network computing environment. It scans the managed resources and stores inventory information in the configuration repository.

**Tivoli IT Director.** A Tivoli product for systems management in a small or medium organization. It is not sold directly by Tivoli Systems Inc. but rather through a Tivoli authorized reseller.

**Tivoli LAN Access.** A Tivoli product that enables system administrators to extend existing LAN management tools by integrating them with the Tivoli suite of products.

**Tivoli management agent.** In the Tivoli environment, an agent that securely performs administrative operations.

**Tivoli Management Framework.** The base software that is required to run the applications in the Tivoli product suite. This software infrastructure enables the integration of systems management applications from Tivoli Systems Inc. and the Tivoli Partners. The Tivoli Management Framework includes the following:

- Object request broker (oserv)
- Distributed object database
Basic administration functions
Basic application services
Basic desktop services such as the graphical user interface

In a Tivoli environment, the Tivoli Management Framework is installed on every client and server; however, the TMR server is the only server that holds the full object database.

**Tivoli management gateway.** In the Tivoli environment, a system that enables bidirectional communication with Tivoli Management Agents.

**Tivoli Management Region (TMR).** In a Tivoli environment, a Tivoli server and the set of clients that it serves. An organization can have more than one TMR. A TMR addresses the physical connectivity of resources whereas a policy region addresses the logical organization of resources.

**Tivoli management software.** The overall descriptor for software from Tivoli Systems Inc., which includes Tivoli Enterprise software (for systems management in a large organization), Tivoli IT Director (for systems management in a small or medium organization), and Tivoli Cross-Site (for the management of e-commerce systems). Tivoli management software enables organizations to centrally manage their computing resources (including the critical applications that drive business performance and profits) in a simple and straightforward manner.

**Tivoli Manager.** Tivoli management software that manages specific vendor systems, networks, applications, or databases.

**Tivoli Module Builder (TMB).** A Tivoli product that enables developers to create a special type of file, called a management module, for managing an application or business system with Tivoli management software. Management modules include Tivoli GEM modules and Tivoli Plus modules. The Tivoli Module Builder provides tools (such as the Tivoli Module Designer) and templates for describing the management characteristics of an application or business system and for building this information (together with the scripts, programs, and files that are required to implement the management function) into a Tivoli install image or an application management package. The Tivoli Module Builder uses file types defined in the Application Management Specification (AMS).

**Tivoli Module Designer (TMD).** A Tivoli tool that enables developers to describe the management characteristics of an application or business system and that generates the application description files and application management packages that the Tivoli management software uses to manage applications and business systems. The Tivoli Module Designer replaces the Tivoli Developer Kit.

**Tivoli NetView.** A Tivoli product that enables distributed network management across multiple operating systems and protocols. Unlike Tivoli NetView for OS/390, Tivoli NetView does not provide centralized management from an OS/390 host.

**Tivoli NetView for OS/390.** A Tivoli product that enables centralized systems and network management from an OS/390 environment. Through its MultiSystem Manager component, Tivoli NetView for OS/390 enables management of distributed resources, such as Internet Protocol (IP) resources, NetWare resources, asynchronous transfer mode (ATM) resources, and others. Contrast with Tivoli NetView.

**Tivoli NetWare repeater (TNWR).** In a Tivoli environment, a server application that is installed on a Novell NetWare server and that maintains a list of available clients for the server. The Tivoli NetWare repeater works with the NetWare managed site to perform profile distribution.

**Tivoli Partner Association.** A partnership program that is led by Tivoli Systems Inc. for business, industry, and product partners. The Tivoli Partner Association provides programs and benefits for business partners (including systems integrators, outsourcers, and resellers) to sell Tivoli Enterprise and IT Director products. Industry and product partners collaborate with Tivoli Systems Inc. in creating hardware and software products that are Tivoli Ready.

**Tivoli Plus module.** In a Tivoli environment, a management module that has been certified by the Tivoli Partner Association and that enables a specific vendor application to be managed by Tivoli management software. To be certified by the Tivoli Partner Association, the Tivoli Plus module must include certain features such as enablement for the Tivoli Global Enterprise Manager (Tivoli GEM).

**Tivoli Ready.** Pertaining to a product that has passed rigorous product certification testing by Tivoli Systems Inc. to ensure that the product delivers turnkey (or "out-of-the-box") integration with Tivoli management software. A product that has passed this certification testing carries the Tivoli Ready logo.

**Tivoli Remote Control.** A Tivoli product that enables a Tivoli administrator to control mouse and keyboard operations on an NT managed node or a PC managed node.

**Tivoli Remote Execution Service.** A service that enables a Tivoli environment to perform remote operations on machines. These operations include: remotely installing clients, connecting Tivoli Management Regions (TMRs), and starting oserv from a remote machine.

**Tivoli Security Management.** Tivoli Enterprise software that enables the consistent definition,
implementation, and enforcement of security policy in a network computing environment.

Tivoli server. The server that holds or references the complete set of Tivoli software, including the full object database. See Tivoli client, TMR client, and TMR server.

Tivoli Service Desk for OS/390. A Tivoli product that is an integrated set of tools, services, and interfaces for automating and customizing an organization's IT service and support operation in an OS/390 environment. It provides a structure that supports the gathering, organizing, locating, and reporting of information related to problem, change, and asset management.

Tivoli Software Distribution. A Tivoli product that automates software distribution to clients and servers in a network computing environment. An organization can use this product to install and update applications and software in a coordinated, consistent manner across a network. Tivoli Software Distribution creates file packages and distributes them to predefined subscribers.

Tivoli upgrade image. In a Tivoli environment, a file that resides on a CD or in a file system and contains updates for a Tivoli product. A Tivoli upgrade image contains only the files that have changed since the previous product release, with the scripts and commands that are needed for installing the new files and configuring the database. Contrast with Tivoli install image.

Tivoli User Administration. A Tivoli product that provides a graphical user interface (GUI) for centralized management of user and group accounts. It offers efficient, automated management of user and system configuration parameters, secure delegation of administrative tasks, and centralized control of all user and group accounts in a network computing environment.

Tivoli UserLink. A Tivoli product that provides IP address synchronization between a PC agent and its associated PC managed node using the Dynamic Host Configuration Protocol (DHCP). Tivoli UserLink also enables a PC user to pull a file package to a Windows, Windows 95, or Windows NT workstation.

TLL. See Task Library Language.

TMB. See Tivoli Module Builder.

TMD. See Tivoli Module Designer.

TME 10. See Tivoli Enterprise software.

TMR. See Tivoli Management Region.

TMR client. In a Tivoli environment, any computer—except the TMR server—on which the Tivoli Management Framework is installed. The oserv daemon runs on the TMR client, and the TMR client maintains a local object database. See Tivoli client and Tivoli server.

TMR server. A Tivoli server for a specific Tivoli Management Region (TMR). See Tivoli client and TMR client.

TNWR. See Tivoli NetWare repeater.

toggle button. In the AIXwindows Toolkit and the Enhanced X-Windows Toolkit, a graphical object that simulates a toggle switch; it switches sequentially from one optional state to another.

tool palette. In Tivoli NetView, a component of the graphical user interface (GUI) that enables the network operator to open application program instances by using the mouse to drag and drop the icons that represent the application program.

topology. In communications, the physical or logical arrangement of nodes in a network, especially the relationships among nodes and the links between them.

topology console. In the Tivoli Global Enterprise Manager and Tivoli NetView for OS/390, a Java-based graphical user interface that displays business system information from the topology server. The topology console displays each component as a separate icon or shape and draws lines between icons to denote links. It then uses color to indicate the status of each component and of the business system as a whole. As the topology server receives configuration and status updates for the business system, it updates the topology console. Therefore, the topology console always displays the real-time configuration and status of the business system.

topology database. See local topology database and network topology database.

topology database update (TDU). A message about a new or changed link or node that is broadcast among APPN network nodes to maintain the network topology database, which is fully replicated in each network node. A TDU contains information that identifies the following:

- The sending node
- The node and link characteristics of various resources in the network
- The sequence number of the most recent update for each of the resources described.

topology server. In Tivoli Global Enterprise Manager and Tivoli NetView for OS/390, a server that interacts with instrumented applications in a business system and provides information for display on the topology console. The topology server receives heartbeat events from instrumented applications or components and determines the business system in which a component belongs. The topology server also queries instrumented applications for related applications and for the status of
its monitors. All of this information is used to create and maintain a view of each business system's configuration and availability on the topology console.

**trace.** A record of the execution of a computer program. It exhibits the sequences in which the instructions were executed. (A)

**transaction.** A specific set of input data that triggers execution of a specific process or job; a message destined for an application program.

**transit time.** See response time.

**Transmission Control Protocol/Internet Protocol (TCP/IP).** A set of communications protocols that support peer-to-peer connectivity functions for both local and wide area networks.

**trap.** In the Simple Network Management Protocol (SNMP), a message sent by a managed node (agent function) to a management station to report an exception condition.

**triggered response.** In a Tivoli environment, the action that is taken when a monitor reaches or exceeds a threshold.

**trouble ticket.** In Tivoli NetView, a record of a problem that has occurred. The trouble ticket becomes the formal vehicle to trace a problem from its occurrence to its resolution.

**TTL.** See time to live.

**tuple.** In a relational database, a part of a relation that uniquely describes an entity and its attribute. A tuple can be represented by one row of a relation table. (T)

**user login map.** In a Tivoli environment, a mapping that associates a single user login name with a user account on a specified operating system. User login maps enable Tivoli administrators to log in to the Tivoli environment or perform operations within the Tivoli environment with a single user login name, regardless of the system that they are currently using.

**user plane.** In Tivoli NetView, the submap layer on which symbols of objects that are not managed by an application program are displayed. Symbols on the user plane are displayed with a shadow, which makes them appear higher than symbols on the application plane. See background plane.

**user role.** See authorization role.

**using node.** The NCP in the host’s domain that reports a link error condition.

**validation.** The checking of data for correctness or for compliance with applicable standards, rules, and conventions. (A)

**validation policy.** In a Tivoli environment, policy that ensures that all resources in a policy region comply with the region's established policy. Validation policy prevents Tivoli administrators from creating or modifying resources that do not conform to the policy of the policy region in which the resources were created.

**variable.** (1) In programming languages, a language object that may take different values, one at a time. The values of a variable are usually restricted to a certain data type. (I) (2) A quantity that can assume any of a given set of values. (A) (3) A name used to represent a data item whose value can be changed while the program is running. (4) In the Simple Network Management Protocol (SNMP), a match of an object instance name with an associated value. (5) In the NetView command list language, a character string beginning with "&" that is coded in a command list and is assigned a value during execution of the command list.
verb. (1) In Tivoli NetView for OS/390, the first word of a NetView command that is delimited by a blank or a comma and that indicates what action is to be taken. (2) See LU 6.2 verb.

view administrator. The part of the NetView Graphic Monitor Facility that downloads the views created by the view preprocessor and that provides these views to the graphic data server.

viewing filter. In Tivoli NetView for OS/390, the function that allows a user to select the alert data to be displayed on a terminal. All other stored data is blocked.

view manager. In the NetView Graphic Monitor Facility, a facility that generates views according to Resource Object Data Manager (RODM) definitions and that provides status changes to the graphic data server.

view preprocessor. The part of the NetView Graphic Monitor Facility that creates unformatted views of SNA resources from the VTAM definition library (VTAMLST).

view preprocessor resource. An SNA subarea resource whose status is reported by the resource status manager and is stored in the graphic data server (GDS) databases when views containing the resource are downloaded.

Virtual Telecommunications Access Method (VTAM). An IBM licensed program that controls communication and the flow of data in an SNA network. It provides single-domain, multiple-domain, and interconnected network capability.

vital product data (VPD). Information that uniquely defines system, hardware, software, and microcode elements of a processing system.

VPD. See vital product data.

VTAM. See Virtual Telecommunications Access Method.

W

webmaster. The person who is ultimately responsible for managing and maintaining a particular Web site.

well-behaved application program. An application program that runs without disruption to the network.

widget. (1) In the AIX operating system, a graphic device that can receive input from the keyboard or mouse and can communicate with an application or with another widget by means of a callback. Every widget is a member of only one class and always has a window associated with it. (2) The fundamental data type of the Enhanced X-Windows Toolkit. (3) An object that provides a user-interface abstraction; for example, a Scrollbar widget. It is the combination of an Enhanced X-Windows window (or subwindow) and its associated semantics. A widget implements procedures through its widget class structure.

wildcard character. See pattern-matching character.

with-request. A Printing Systems Manager (PSM) document transfer method in which the client transfers documents directly to the server. This is the default transfer method. Contrast with dce-pipe-pull.

wizard. A dialog within an application that uses step-by-step instructions to guide a user through a specific task.

working directory. The directory that is currently in use by an operating system or application. If no path is specified, this is the directory to which data is written, from which data is deleted, or in which data is searched.

work space. (1) That portion of main storage that is used by a computer program for temporary storage of data. (2) In Tivoli NetView, a container for a set of event cards that meet certain criteria. See event filter.

wrap count. In Tivoli NetView for OS/390, the number of events that can be retained in the database for a specific resource or the number of alerts that are retained in the database.

X

XCF. See cross-system coupling facility.

X Window System. A software system, developed by the Massachusetts Institute of Technology, that enables the user of a display to concurrently use multiple application programs through different windows of the display. The application programs may execute on different computers.

Y

Year 2000 challenge. A term used especially by the computer industry to refer to the problems, challenges, and issues involved in preparing computer systems and applications for transition to, and operation in, the twenty-first century. For example, many computer systems and applications use two digits to represent the year (“97” rather than 1997). When these computer systems and applications encounter the digits “00” for the year 2000, they can misinterpret this to mean the year 1900 and can produce computing errors or fail to function. Although some systems and applications may not be affected until the eve of the new millennium (on 31 December 1999), many systems and applications that use future dates (such as expiration dates for credit cards) have already experienced Year 2000 problems. This problem could also affect such things as elevator...
controls; household appliances such as VCRs and programmable coffee makers; heating, cooling, and security systems; telephone calls; driver's licenses; automated teller machines and bank vaults; and airline flight schedules.

**Year 2000 ready.** A product is Year 2000 ready if the product, when used in accordance with its associated documentation, is capable of correctly processing, providing, and/or receiving date data within and between the twentieth and twenty-first centuries, provided that all products (for example, hardware, software, and firmware) used with the product properly exchange accurate date data with it.

**Y2K.** See Year 2000 challenge.

**Z**

**zombie process.** In the UNIX operating system, a process that has been terminated but has not been cleaned up by its parent process. The existence of a large number of zombie processes could indicate an errant network daemon or application. Zombie processes are sometimes called “lingering terminated processes.”

**zoom.** In a user interface, to progressively increase or decrease the size of a part of an image on a screen or in a window.

**Numerics**

**4700 Support Facility.** In Tivoli NetView for OS/390, a component that enables the monitoring and control of IBM 3600 and 4700 Finance Communication Systems. The 4700 Support Facility can record, analyze, and display performance and status data on IBM 3600 and 4700 Finance Communication Systems.
Index

Special Characters
.Xdefaults file 93

A
about this book xi
access levels to maps xiii
accessibility information 
acknowledging, unacknowledging objects
  global-based mode
    from a map 75
    from an event 75
    overview 75
    status as an object attribute 75
    using nvdimport 76
import/export utility 76
map-based mode 75
overview 74
action ruleset node 112
actionsvr daemon 13
activating, deactivating event filters 136
activating event correlation rulesets 127
addalert command 148
adding
  and changing background graphics 89, 90
  connections 64
  correlation ruleset node 125
  manager-container associations 171
  objects and symbols 61
  SmartSets 79
adptrap command 145
algorithms, layout
  automatic layout, setting 91
  changing 71
  description 54
  redo layout, setting 90
aliases, Agent Policy Manager
  description 258
  managing 258
APM (Agent Policy Manager)
  aliases
    description 258
    managing 258
  community names
    configuring 256
    examples 257
  configuration interface, starting
    210
  daemon
    starting from the command line 209
    starting through the Tivoli desktop 208
  definition
    an example 253
    changing 215
    creating 211
    creating, using an existing definition 211
    deleting 216
    distributing to remote nodes 213
  (Agent Policy Manager) (continued)
    definition (continued)
      distribution failures 214
      file monitoring, dialog box fields 223
      file monitoring, procedure 206
      how to define and distribute 205
      modifying 215
      successful distribution 213
      threshold, procedure 219
      threshold dialog fields, description 221
      viewing, active 212
      viewing, pending 216
      description 203
      distributing changes 204
      managing agents 203
      diagnosing problems 248
      distribution status indicators 211
      MLM domains
        description 258
        rearranging, automatically 259
        rearranging, manually 259
      requirements 203
      SmartSet icons
        APM monitors 204
        MLM managers 204
        propagating status 205
        SmartSets 204
      app-defaults
        changing 94
        event card or list behavior 108
        manager backup 174
      application defaults
        changing 94
        event card or list behavior 108
        manager backup 174
      application plane 55
      applications
        collmap 76
        defined 57
        ipmap 2, 57
        nvevents 3
        owv 1
        performance 181
        xnmapmon 6
        xnmbrowser 4
        xnmbrowser2 4
        xnmbuilder 5
        xnmcollect 5
        xnmfault 5
        xnmgraph 5
        xnxxloadmib 4
        xnxxloadmib2 4
        xnrnrnreport 5
        xnmsnmpconf 4
        xnmsnmpmon 6
        xnmrunreport 5
        xnmrunreport2 4
        xnmtrapsm 3
        xxmap 3
        ARFs to SRFs, converting 39
        ARP cache information 162
assigning map access levels 50
assigning submaps
  home 73
  parent 73
audit data, security
  creating reports 37
  defining categories 32
  description 23
authentication, network
  description 19
  login
    considerations 20
    description 20
user profiles
  creating 26
  deleting 29
  valid characters for names and passwords 28
  viewing 29
automatic layout 55, 91
automatic network discovery 151
  information retrieved from nodes 151
  turning off 152
  using seed files, defining management regions 153

B
background 65
  graphics 89
    adding, changing 89
    deleting 90
plane 55
processes
  actionsvr 13
  C5d 11
gtm 10
mgragentd 8
netmon 8
noniptopod 9
nvcold 9
nvcorrd 13
nvlockd 9
nvold 11
nvpagerd 13
nvsecd 14
nvsecltd 15
nvserverd 14
orvd 12
otmd 10
ovelmd 13
ovesmd 13
ovsnpmd 7
ovtopmd 9
ovwb 9
pmd 12
snmpCollect 14
spappid 15
tralertrd 15
trapd 13
trapgend 14
backup manager configuration 167
backup manager (continued)
  default actions, changing 174
  determining current backup sessions 174
  dialog box 169
  responding to
    manager down notification 173
    manager restored notification 173
  using a seed file to configure 168
backupd daemon 168
boolean logic 77
browsing MIBs 179
building MIB applications 185
C
c_arf2srf command 39
C5d daemon 11
  starting
    from the command line 209
    through the Tivoli desktop 208
cache, ARP 162
cache, ovwdb 155
changing
  Agent Policy Manager definition 215
  backup manager default actions 174
  configuration 166
    host name, local maps 48
    host name, NFS-mounted maps 47
  control desk startup 95
  event card or list behavior 108
  graphical interface defaults 94
  map permissions 50, 92
  root permission for security administration 25
  status source 143
characteristics
  manager-container 168
  subnet 52
check route ruleset node 113
checking application messages 99
checking configured interfaces 161
child submap
  creating with default settings 71
  creating with modified settings 71
  default settings 71
  steps for creating with default settings 71
  steps for creating with modified settings 72
Cisco routers, discovering 154
client/server environment
  managing maps 47
  maps, local 48
    deleting invalid maps 48
    removing maps 49
    updating host name 48
  maps, NFS mounted 47
    changing the server 47
    improving network performance 47
collecting MIB data 178
collmap application 76
colors, changing in graphical interface 94
combining event correlation rulesets 126
Index
correlating events (continued)
rulesets (continued)
saving 126
testing 127
counter MIB values, displaying 100, 198
creating an Agent Policy Manager definition
from an existing definition 211
overview 205
procedure 211
creating event filters 131
creating filters
compound 135
trap-to-alert 139
creating independent submaps 72
creating MIB variable expressions 192
creating SmartSets
adding 79
description 76
examples 85
listing objects 85
modifying 84
types 77
using pattern matching 78
creating submaps
a child, default settings 71
a child, modified settings 71
child, independent 72
cron table entries
activating and deactivating 141
creating 141
modifying 141
sorting 141
customer support, contacting xiii
customizing
.Xdefaults file 93
event card or list display 108
failing resource display 95
grapher 96
graphical interface 89
manager backup default actions 174
maps 49
menu bar and tool palette 92
MIB graph applications 96
objects and connections 61
submaps 164
symbol placement 61
traps 143
cutting
procedure 67
restrictions 66
rules 65
cutting, copying, pasting objects and symbols 65
daemons (continued)
mibiisa 17
netmon 8
noniptopod 9
nvcold 9
nvcorr 13
nvlockd 9
nvold 11
nvopagerd 13
nvsec 14
nvsec1df 15
nvserverd 14
orsd 12
otmd 10
ovelmd 13
ovesmd 13
ovspmd 7
ovtopmd 9
ovwdb 9
pmd 12
snmpCollect 14
snmpd 17
spappld 15
tralertd 15
trapd 13
trapgend 14
data collector, MIB 188
Databases
general topology 16
IP topology 16
map 16
description 16
information 46
object 16
object registration service 16
default compound status 44
default settings
child submap 71
graphical interface 94
defining a security policy
global settings 32
group profiles
adding 29
copying 31
creating 29
deleting 31
viewing permissions 31
overview 24
user profiles
creating 26
deleting 29
viewing 29
defining SmartSets
adding 79
description 76
examples 85
listing objects 85
modifying 84
types 77
defining symbol characteristics 42
deleting
Agent Policy Manager definition 216
correlation ruleset connection 126
correlation ruleset node 126
group profiles, security 24
invalid maps on a client 48
menu items 93
objects 65
symbols 65
user profiles, security 29
description, object, modifying 74
designating nodes as managers 170
determining manager backup sessions 174
diagnosing problems, Agent Policy Manager
definitions 248
directory, report 200
disability information xiii
Discovering Cisco Routers 152
discovering the network 151
Cisco routers 154
configuration information retrieved during 151
open topology 153
using openmon 153
ways to 163
discovery and database daemons 7
disk space, monitoring 183
display presentation 100
displaying, hiding grid 100
displaying, owxdb object database
MIB interface information 162
MIB system information 162
displaying multiple protocols 60
distributing
Agent Policy Manager definition
distribution failures 214
procedure 213
status indicators 211
successful distribution 213
security configuration 36
distribution status indicators, Agent Policy
Manager 211
dynamic increase, file system and paging space 195
dynamic workspace, creating 107

event log (continued)
changing (continued)
size 109, 110
viewing 109
events
card or list behavior, controlling 108
configuring 142
converting to alerts 147
correlating, rulesets 110
activating 127
adding a node 125
combining rulesets 126
deleting a connection 126
deleting a node 126
environment variables for trap data 123
event attribute values 122
examples 127
inserting a ruleset 126
ruleset editor 124
sample 124
saving 126
testing 127
defining 103
displaying warning window 146
filtering 136
filters
activating and deactivating 136
creating 133
customizing 96
designating nodes as managers 170
distributing
Agent Policy Manager definition
distribution failures 214
procedure 213
status indicators 211
successful distribution 213
security configuration 36
distribution status indicators, Agent Policy
Manager 211
dynamic increase, file system and paging space 195
dynamic workspace, creating 107
E
editing rules 61
enabling the Agent Policy Manager daemon 208
enterprise-specific MIBs
browsing 179
loading and unloading 177
environment variables, security 31
event adapter, controlling 106
event and trap processing daemons 11
event attributes ruleset node 114
event history 109
event log
changing
color of cards and text 94
number of events 94
presentation 94

Index 333
examples
  collecting MIB data 191
  event correlation rulesets 127
  internet submap 61
  partitioned internet submap 68
  SmartSets 85
  using the Agent Policy Manager 253

failing resource display, customizing 95
fault isolation, router 159
file monitoring Agent Policy Manager definition
  an example 253
  changing 215
  creating 211
  creating from an existing definition 211
  deleting 216
diagnosing problems 248
dialog box fields, description 223
distributing
  distribution failures 214
  procedure 213
  successful distribution 213
how to define and distribute 205
modifying 215
procedure 206
viewing
  active 212
  pending 216
filter editor 132
filtered command 109
filtering events 136
  activating 138
  deactivating 139
filters
  activating and deactivating 136
  creating 131
    compound 135
    simple 133
customizing for users 96
editor, accessing 131
for event display 136
types 131
fonts, changing 94
forcing off logged in users 35
foreground processes 1
forward ruleset node 114
forwarding events to the Tivoli Enterprise Console 104
default format 104
event adapter, controlling 106
general topology database 16
generating performance reports 199
gettrap command 149
GIF 89

global-based Acknowledged status (continued)
  overview 75
  status as an object attribute 75
  using nvdbimport 76
global settings, security 32
graphical user interface
  changing 89
  control desk startup, preventing 95
  customizing 89
defaults 95
graphics
  adding 89
deleting 90
Graphics Interchange Format 89
graphing MIB variable expressions 192
graphs
  adding lines 101, 199
  changing display presentation 100
  changing line configuration 98
  checking application messages 99
customizing 96
displaying, hiding grid 100
displaying MIB counter values 100, 198
entering numeric values 97
getting application statistics 99
paging, graphs 100
printing 198
scaling y-axis 100
setting time intervals 97
using context menus 101, 199
groups, security
  adding 29
  changing 29
  copying 31
  creating 29
deleting 31
description 22
pre-configured, description 22
viewing permissions 31
gtmd daemon 10

H
hide objects and symbols 65, 74
home submap, assigning 73
host connection daemons 15
host name
  updating
    local maps 48
    NFS-mounted maps 47

I
identification, network
  description 19
login
  considerations 20
description 20
user profiles
  creating 26
deleting 29
mapadmin command 47, 48
maps and submaps
  authorization 50
  client/server environment
    description 47
    local maps, considerations 48
    local maps, invalid, deleting 48
    local maps, removing a client 49
    local maps, updating host name 48
  NFS-mounted maps, changing server 47
  NFS-mounted maps, considerations 47
  NFS-mounted maps, improving network performance 47
customizing 49, 89
database information 46
  editing 63
  learning about 46
  map layout 51, 54
    defining 54
    using algorithms 54
    using automatic layout 55
    using new object holding area 55
metacommunication 55
open map 46
snapshots 51
menu bar
  customizing
    OVwRegDir environment variable 92
  security services 23
messages, sending 35
metacommunication submaps
  behavior 57
  characteristics 56
  defining 55
  description 55
migragentd daemon 8
MIB
  application, adding 186
  application builder 185
  applications, comparison 182
  browser 179
  CPU Performance 183
  data, collecting 191
  data collection 187
  disk space 183
  displaying description 180
  displaying interface information 162
  displaying system information 162
  Ethernet errors 183
  Ethernet packet types 183
  Ethernet performance 183
  Ethernet traffic 183
  graphing counter values 100, 163, 198
  interface traffic 183
  IP errors 99
  loading and unloading 177
  querying 181
  real-time performance data 182
  retrieving configuration information 162
  SNMP authentication failures 185
  SNMP errors 184
MIB (continued)
  SNMP network activity 184
  SNMP operations 184
  SNMP traffic 184
  TCP connections 184
  values, displaying 162
  variable expressions, creating and graphing 192
mib2trap command 145
mibExpr.conf file 192
mibiisa daemon 17
MLM domains, Agent Policy Manager
  description 258
  rearranging
    automatically 259
    manually 259
MLM managers map icon 204
modem configuration file 111
modifying
  object descriptions 74
  SmartSets 84
  subnetmap settings 73
  symbol labels, displaying 91
monitoring local filesystem and paging space
  for specific events 196
  starting
    with a command 196
    with dynamic increase 195
    without dynamic increase 193
  stopping 195
monitoring network configuration 160
monitoring SmartSet status
  starting 77
  stopping 77

N
  navigation tree, starting as icon 94
  Netmon daemon 8
NetView, sending alerts to 149
network, distributed
  managing maps 47
  maps, local 48
    deleting invalid maps 48
    removing maps 49
    updating host name 48
  maps, NFS mounted 47
    changing the server 47
    improving network performance 47
network configuration
  information retrieved 151
  listing IP addresses 160
  monitoring 160
network discovery
  automatic 151
  using openmon 153
  using seed files 153
network monitoring, continuous
  description 21
  shift-in and shift-out 21
network submap, objects you can add 62
network topology events 103
NFS-mounted maps
 changing the server 47
 considerations 47
 improving network performance 47
 node configuration events 103
 node submap, objects you can add 62
 nodes
 configuring as backup managers 170
 unmanaged, suppressing events from 108
 nodes, ruleset
 action 112
 check route 113
 event attributes 114
 forward 114
 override 115
 pager 115
 pass on match 115
 query database field 117
 query database SmartSet 116
 query global variable 119
 set database field 118
 set global variable 119
 set MIB variable 120
 threshold 121
 trap settings 122
 noniptopod daemon 9
 nv.carriers configuration file 111
 nv colder daemon 9
 nvcorrd daemon 13
 nvdformat utility 17
 nvdimport utility 17
 nvevents application 3
 nvlockd daemon 9
 nvoft daemon 11
 nvpaged daemon 11
 nvpager.config configuration file 112
 nvpagerd daemon 13
 nvpaging.protocols configuration file 112
 nsvd daemon 14
 nsvectl daemon 15
 nsvserverd daemon 14
 nsv status 6

O
 object
 acknowledging, unacknowledging
 global-based mode 75
 import/export utility 76
 map-based mode 75
 overview 74
 adding and deleting 61, 65
 adding to submaps
 internet 62
 network 62
 node 62
 root 62
 segment 62
 steps 63
 basics 41
 copying 67

object (continued)
 customizing 61
 cutting 65
 database 16
 generating and customizing reports 17
 deleting 65
 description 61
 displaying ovwdb database 41
 hidden, listing 65, 74
 hiding 65, 74
 learning about 41
 managing, unmanaging 74
 modifying description 74
 SmartSets
 adding 79
 defining 76
 examples 85
 listing objects 85
 managing and unmanaging 77
 modifying 84
 status monitoring, starting 77
 status monitoring, stopping 77
 types 77
 status
 description 44
 symbols that use 43
 object holding area 55
 object registration service database 16
 open topology networks, discovering 153
 openmon application 153
 orsd daemon 12
 otmd daemon 10
 oveld daemon 13
 override ruleset node 115
 ovesmd daemon 13
 ovspmd daemon 7
 ovstart 6
 ovstatus 6
 ovstop 6
 ovtopmd daemon 9
 ovw application 1
 ovwdb cache 155
 ovwdb daemon 9
 ovwdb object database 16
 OVwRegDir environment variable 92
 starting as an icon 94
 using security services 23
 ovxecho and ovxbeep 147

P
 pager ruleset node 115
 paging, graph 100
 paging space, monitoring 193
 paging utility
 configuration files
 modem 111
 nv.carriers 111
 nvpager.config 112
 nvpaging.protocols 112
 configuring 111

Index 337
paging utility (continued)
  event correlation rulesets 110
  nvpage command 111
  pager service through security services 23
parent
  objects, definition 51
  submaps, assigning 73
partitioned internet submap
  example 68
  steps for creating 68
partitioned segment submap, steps for creating 70
pass on match ruleset node 115
passwords
  description 21
  setting 26
  valid characters 28
pasting
  procedure 67
  restrictions 66
  rules 65
pattern matching 78
performance
  gathering information 177
  improving, ipmap 155
  reports, generating 199
permissions
  map 92
  root for security administration, changing 25
  user 50
planes
  application 55
  background 55
  user 55
polling
  reducing router 159
  setting intervals 163
  turning on and off 156
printing graphed data 147, 198
process management 6
processes
  background
    event and trap 11
    host connection 15
    ovspmd daemon 7
    process management 6
    topology and database 7
  foreground 1
propagate-at-threshold-value compoundstatus 45
propagate-most-critical compound status 45
propagate status to Agent Policy Manager map
  icons 205
protocol switching 60

Q
query database field ruleset node 117
query database SmartSet ruleset node 116
query global variable ruleset node 119
querying logged in users 35
quick zoom 53

R
read-write permission 92
redo layout 90
reducing router polling 159
remote manager status, determining 172
removing
  invalid maps on a client 48
  maps before deinstalling a client 49
report directory, contents 200
report writing 201
reports, generating and customizing on the object database
  nvdbimport utility 17
  nvdt format utility 17
reports, generating performance 199
requirements for using the Agent Policy Manager 203
resource display, failing 95
resources
  monitoring 193
  Xdefault 95
retrieving MIB configuration information 162
role of Solaris SEA agents 17
root permission, changing for security administration 25
root submap 52
  objects you can add 62
router faults isolation 159
router polling, reducing 159
routers, discovering Cisco 154
Routers, Discovering Cisco 152
routines
  ovwchgrp 50
  ovwchmod 50
  ovwchown 50
  ovwls 50
routing tables, viewing information 161
ruleset editor 124
rulesets, event correlation
  activating 127
  creating
    adding a node 125
    combining rulesets 126
    deleting a connection 126
    deleting a node 126
    environment variables for trap data 123
    event attribute values 122
    inserting a ruleset 126
    ruleset editor 124
description 110
examples 127
how events are processed 110
nodes
  action 112
  check route 113
  event attributes 114
  forward 114
  override 115
  pager 115
  pass on match 115
query database field 117
query database SmartSet 116
rulesets, event correlation (continued)
nodes (continued)
   query global variable 119
   set database field 118
   set global variable 119
   set MIB variable 120
   thresholds 121
   trap settings 122
   paging utility, configuring 111
   sample 124
   saving 126
   testing 127
runapp command 187

S

saving
   event correlation rulesets 126
   performance data 197
scaling, y-axis 100
scaling submaps 53
SEA agents, role 17
searching for events
   by criteria 109
   by filter 109
security
   ARFs to SRFs, converting 39
   audit data
      creating reports 37
      defining categories 32
      description 23
   configuration, distributing 36
   converting ARFs to SRFs 39
   description 19
   distributing configuration 36
   environment variables 31
features
   audit management 23
   consistent security controls 23
   continuous, auditable network management 21
   log in considerations 20
   log in process 20
   network access control 22
   network authentication and identification 19
   pager service 23
   password protection 21
   shift-in, shift-out operation 21
   tailored Tivoli NetView graphical interface 23
forced off logged in users 35

groups
   adding 29
   changing 29
   copying 31
   creating 29
   deleting 31
   description 22
   pre-configured, description 22
   viewing permissions 31
permissions, verifying for shell scripts 39
querying logged in users 35
security (continued)
   security administration dialog
      accessing 24
      description 25
      root permission, changing 25
      sending messages 35
security administration dialog
   accessing 31
   description 25
   root permission, changing 25
security policy, defining
   global settings 32
   group profiles
      adding 29
      copying 31
      creating 29
      deleting 31
      viewing permissions 31
overview 24
user profiles
   creating 26
   deleting 29
   viewing 29
seed file,
   configuring backup manager
      example 169
      format 168
      process 169
discovery, using 153
seed file backup 168
segment submap, objects you can add 62
selectfilter command 142
selection rule, adding and changing 186
sending alerts to NetView 149
sending SNMP traps 104
services, listing configuration 162
set database field ruleset node 118
set global variable ruleset node 119
set MIB variable ruleset node 120
setthresh command 193
setting
   compound status 45
   environment variables, security 31
   map permissions 92
   polling intervals 163
shadow, turning off 62
shell scripts, verifying security permission 39
shift-in and shift-out 21
shortcut keys, keyboard xiii
shpmon application
   configuring 195
   dynamic increase 195
   monitoring specific events 196
   starting from the command line 196
simple filters, creating 133
size of graphical interface windows, changing 94
SmartSet editor
   adding a SmartSet 79
   examples 85
   listing objects 85
   modifying a SmartSet 84

Index 339
SmartSet editor (continued)
using pattern matching 78
SmartSet facility 76, 77
SmartSet icons
APM monitors 204
MLM managers 204
SmartSets 204
SmartSets
maintaining 9
SmartSets, defining
adding 79
description 76
dispatch objects 85
managing and unmanaging 77
modifying 84
status monitoring
starting 77
stopping 77
types 77
SmartSets, Defining and Managing 76
SmartSets, Managing and Unmanaging 77
SmartSets, Types of 77
snapshots, taking 51
SNMP nodes, configuring 163
snmpCol.conf file 5
snmpColDump command 188
snmpCollect daemon 14
snmpdx daemon 17
Solaris SEA agents, role 17
spappld daemon 15
starting
Agent Policy Manager configuration interface 210
Agent Policy Manager daemon
from the command line 209
through the Tivoli desktop 208
event display application 106
graph application 197
startup configuration file 7
status polling switch check button 156
status scheme, compound 44
submaps
assigning a parent 73
automating partitioning of 70
basics 51
characteristics 52
creating an independent 53, 72
customizing 61
home 52, 73
internet 59
metaconnections 55
modifying settings 73
network 59
partitioning 68
presentation 53
root 52
support, contacting customer xiii
suppressing events from unmanaged nodes 108
symbols
basics 41
behavior 42
symbols (continued)
characteristics
behavior 42
label 42
location 42
status 42
type 42
variety 42
class 42
classifying creation time and buffer 155
cutting, copying, and pasting 65
deleting 65
displaying labels 91
executable 42
explodable 42
hiding 65, 74
label 42, 44
learning about 41
location 42
modifying labels 91
propagation 44
status 43
compound status source 44
object status source 44
symbol status source 44
type, classes and subclasses 42
variety 42
synchronization
limitations during 58
message 58
system performance
monitoring
local file system 193
paging space 193
starting with dynamic increase 195
starting without dynamic increase 195
T
tables
configuration information, nodes 151
customizing 173
default compound status scheme 44
map database information 46
network topology layout algorithms 54
symbol characteristics 42
symbols recognized, internet submap 62
testing event correlation rulesets 127
threshold Agent Policy Manager definition
changing 215
creating 211
creating from an existing definition 211
defining
dialog box fields, description 221
procedure 219
deleting 216
diagnosing problems 248
distributing
distribution failures 214
procedure 213
successful distribution 213
threshold Agent Policy Manager definition (continued) how to define and distribute 205 modifying 215 viewing active 212 pending 216 thresholds, setting 134 thresholds ruleset node 121 time, changing for status check 155 Tivoli Enterprise Console forwarding events to 104 event adapter, controlling 106 event format 104 Tivoli NetView changing colors 94 fonts 94 sizes 94 tool palette 92 topology discovery daemons 7 tralerld daemon 15 trap settings ruleset node 122 trap-to-alert conversion 139 trapd daemon 13 trapgend daemon 14 traps creating correlation rulesets 124 customizing conditions 143 scenario 143 verifying 145 filtering 131 internal for Tivoli NetView 263 Types of SmartSets 77

U
understanding MIBs browsing 179 loading and unloading 177 unloading MIBs 178 unmanaged nodes, suppressing events from 108 unmanaging manager-container associations 173 updating host name local maps 48 NFS-mounted maps 47 user plane 55 user profiles, security creating 26 deleting 29 valid characters for names and passwords 28 viewing 29 user profiles management dialog 27 using layout automatic 91 redo 90 using predefined MIB applications 182 using seed files backup manager 168

V
verifying security permission for shell scripts 39 vfy_access command 39 viewing Agent Policy Manager definitions active 212 pending 216 event log 109 group profiles, security 31 routing table information 161 security registration files 26 user profiles, security 29

W
windows graphical interface, changing the size, fonts, and colors 94 main, starting as icon 94 warning for events 146 workspace changing default 94 creating automatically 107 dynamic 107 filtering events 136 writing reports 201

X
X11 monochrome bitmap format (XBM) 89 XBM 89 xnmappmon application 6 xnmbrowser application 4 xnmbrowser2 application 4 xnmbuilder application 5 xnmcollect application 5 xnmfault application 5 xnmgraph application 5 xnmgraph application 5 xnmloadmib application 4 xnmloadmib2 application 4 xnmrunreport application 5 xnmrsnmpconf application 4 xnmtrap application 3 xxmlmap application defining 60 description 3 presenting information 60 protocol switching 60

Z
zoom, quick 53 zooming submaps 53