# NetVisualyzer™ User's Guide

Document Number 007-0812-040

Contributors
Written by Susan Ellis
Edited by Loraine McCormick
Production by Laura Cooper
Engineering contributions by Kevin Conry, Charuhas Ghatge, Ron Jacoby, Jenny Leung, Victor Mitnick, Paul Robins

© Copyright 1992, Silicon Graphics, Inc.— All Rights Reserved
This document contains proprietary and confidential information of Silicon Graphics, Inc. The contents of this document may not be disclosed to third parties, copied, or duplicated in any form, in whole or in part, without the prior written permission of Silicon Graphics, Inc.

Restricted Rights Legend
Use, duplication, or disclosure of the technical data contained in this document by the Government is subject to restrictions as set forth in subdivision (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS 52.227-7013 and/or in similar or successor clauses in the FAR, or in the DOD or NASA FAR Supplement. Unpublished rights reserved under the Copyright Laws of the United States. Contractor/manufacturer is Silicon Graphics, Inc., 2011 N. Shoreline Blvd., Mountain View, CA 94039-7311.

Silicon Graphics and IRIS are registered trademarks and IRIX, 4DDN, Data Station, EFast, NetVisualyzer, and WorkSpace are a trademarks of Silicon Graphics, Inc. AppleTalk and EtherTalk are trademarks of Apple Computer, Inc. Banyan and VINES are registered trademarks of Banyan Systems, Inc. Spectrum is a registered trademark of Cabletron Systems, Inc. DEC, DECnet and LAT are trademarks of Digital Equipment Corporation. NetBIOS is a trademark of International Business Machines Corp. Ethernet and XNS are registered trademarks of Xerox Corporation. NetWare and Novell are registered trademarks and IPX is a trademark of Novell, Inc. Sun is a registered trademark and NFS and ToolTalk are trademarks of Sun Microsystems, Inc. SPARCstation is a trademark of SPARC International, Inc. licensed exclusively to Sun Microsystems, Inc. Wingz is a trademark of Informix Software, Inc. X Window System is a trademark of Massachusetts Institute of Technology.

NetVisualyzer™ User's Guide
Document Number 007-0812-040

# Contents

# Figures

# Tables

# Introduction

This guide describes NetVisualyzer™ Release 2.0. NetVisualyzer is a set of network management tools that visually monitor your network and display its activities. NetVisualyzer graphically shows network traffic so you can easily see network configuration and traffic patterns and identify network problems.

This chapter provides information on:

- this guide's audience
- how to use this guide
- hardware and software requirements for using NetVisualyzer
- documentation conventions
- user interface terminology
- common user interface operations
- references
- product support

## Audience

This guide is written for network administrators and for developers of distributed applications that use a client/server model. It assumes that the network is already up and running.

To use this guide, you should be familiar with IRIX™ networking concepts and utilities and basic IRIX window system concepts. If you are not familiar with IRIX networking utilities and the Internet Protocol or need to set up a network, see the *IRIX Advanced Site and Server Administration Guide*. If you are using NetVisualyzer with another supported network product, you can consult the relevant documentation for that product. For instance, if you are using NFS™, see the *NFS Administration Guide*.

For information on the IRIX window system and WorkSpace™, see the *IRIS Essentials*.

## How to Use This Guide

This guide is organized as follows:

- This Introduction provides general information about the hardware and software requirements for using NetVisualyzer, a list of the conventions used in this guide, descriptions of user interface terms and operations, information about the tutorial and examples provided with NetVisualyzer, and product support information.
- Chapter 1, "Getting Started with NetVisualyzer," gives an overview of NetVisualyzer; describes how NetVisualyzer snoops on networks to collect the data presented by the tools; explains how to do the setup tasks that enable you to run NetVisualyzer tools; and describes how to use the *netvis*(1M) command to start NetVisualyzer tools, view the NetVisualyzer online tutorial, and view the NetVisualyzer manual pages.
- Chapter 2 through Chapter 9 explain the NetVisualyzer tools.
- Chapter 10, "Creating and Using Filters," and Chapter 11, "Using NetVisualyzer in a DECnet Environment," explain general tasks that apply to many of the tools.

- Appendix A through Appendix D provide reference information on error messages, tool authorization requirements, supported protocols, and configuration file formats.
- Appendix E, “Introduction to MIBs,” explains Management Information Bases (MIBs) to users who are unfamiliar with this concept and the terminology associated with them.
- Appendix F contains manual pages for all of the NetVisualyzer tools and the daemons that provide information to the tools.

Most users should begin using this guide by reading Chapter 1. It explains how snooping and SNMP agents are used to collect data for NetVisualyzer tools and how to perform setup tasks that are required for running NetVisualyzer tools.

Chapter 2 through Chapter 9 present one or more NetVisualyzer tools. Chapter 2, “NetFilters,” should be examined first to introduce you briefly to filters and the repository of filters provided for use with many NetVisualyzer tools. The remaining tools can be investigated in any order.

The remainder of this guide, Chapter 10 through Appendix F, contains specialized task information—creating filters, and running NetVisualyzer in a DECnet environment—and reference information—error messages, authorization information, supported protocols, configuration file formats, MIB terminology, and manual pages—that is available if you need it.

This guide assumes that all NetVisualyzer tools are available on the workstation that you are using. In cases where only NetVisualyzer Data Station™ software is installed (see the next section for more information), the only NetVisualyzer tools that are available are NetCollect, NetPack, and NetSnoop.

Whether you are a new or an experienced NetVisualyzer user, be sure to read the *NetVisualyzer Release Notes* for additional installation information and information about new features and changes in this release.

## Hardware and Software Requirements

NetVisualyzer is available for all Silicon Graphics® IRIS® workstations running IRIX Release 4.0.5 or later.

The NetVisualyzer software is two separate software options:

- NetVisualyzer Data Station Software
- NetVisualyzer Display Station Software

The Data Station software, which allows Data Stations to capture information from their respective networks, can be used on Silicon Graphics or Sun® Microsystems SPARCstation™ workstations. Silicon Graphics and Sun Microsystems workstations used as Data Stations do not require graphics monitors.

The Display Station software, which provides centralized graphical monitoring of the information collected by the Data Stations, requires an IRIS graphics workstation. An IRIS graphics workstation can serve as both a Display Station and a Data Station when both types of software are installed.

The Data Station software for Sun Microsystems workstation is supported on SPARCstation 1+, SPARCstation 2, SPARCstation IPC, SPARCstation IPX, and SPARCstation ELC. Contact the Silicon Graphics Technical Assistance Center for the current list of supported Sun workstations.

Display Stations and Data Stations should have at least 16Mb of memory and 400Mb of disk space. Additional disk space may be required on Data Stations that use NetCollect to collect network traffic data. NetCollect traffic data for one hour is about 120K.

If your workstation has a CMC ENP-10 Ethernet® board, the firmware version of the board must be Silicon Graphics Version 4 or later (give the command **`hinv`** to see the firmware version of the CMC board).

Software prerequisites for NetVisualyzer are described in the *NetVisualyzer Release Notes* and also in “Installing NetVisualyzer Software” in Chapter 1 and “Enabling SNMP Agents” in Chapter 1 of this guide.

## Documentation Conventions

As you read this guide, you will notice that special fonts are used for certain words.

`typewriter font`
: Indicates system output, such as responses to commands that you enter and the text of messages that appear in Warning and other informational windows. This font is also used for examples of the contents of files, filters and filter components, examples of network addresses, NetVisualyzer protocol names, the names of resources in the */usr/lib/X11/app-defaults* files for NetVisualyzer tools, *snoopd*(1M) services, Management Information Base (MIB) object names, and example workstation and network segment names and addresses.

**`typewriter bold`**
: Indicates text you must enter, such as command lines and filter expressions. Names of nonprinting keys on the keyboard, such as the **`<Enter>`** key, also appear in typewriter bold and are surrounded by angle brackets.

**bold**
: Designates literal options to commands.

*italic*s
: Indicates file names, command names, and manual page names. Lowercase italic words also represent variables—text strings that you must specify. References to other documents, button names, *inst*(1M) subsystem names, user IDs, and group names are also in *italics*.

## User Interface Terminology

Figures I-1 and I-2 show examples of NetVisualyzer windows, with the window terms used in this guide noted.

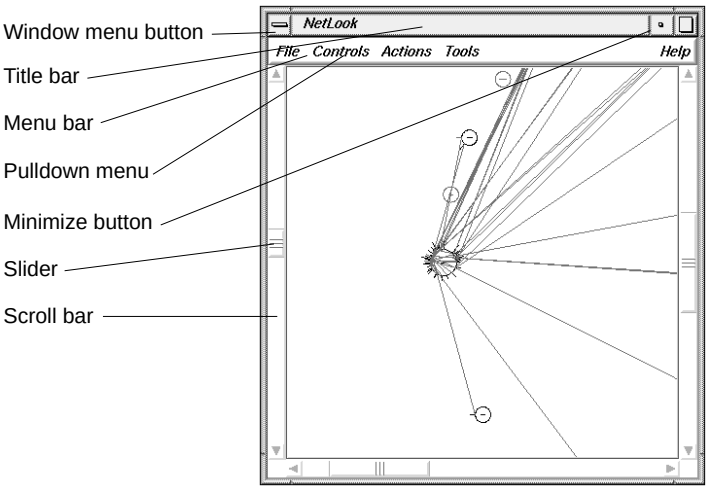



**Figure In-1** Window Terms

The mouse buttons have these functions:

| | |
|---|---|
| left | Perform most basic tasks: click buttons, select an entry field to type into, select menu choices, select items in a display, select text to modify, and so on. |
| middle | Reposition the viewing area in NetLook and re-position the graph in NetTop. |
| right | Access pop-up menus. Pop-up menus appear when you press the right mouse button in certain locations on the screen. |

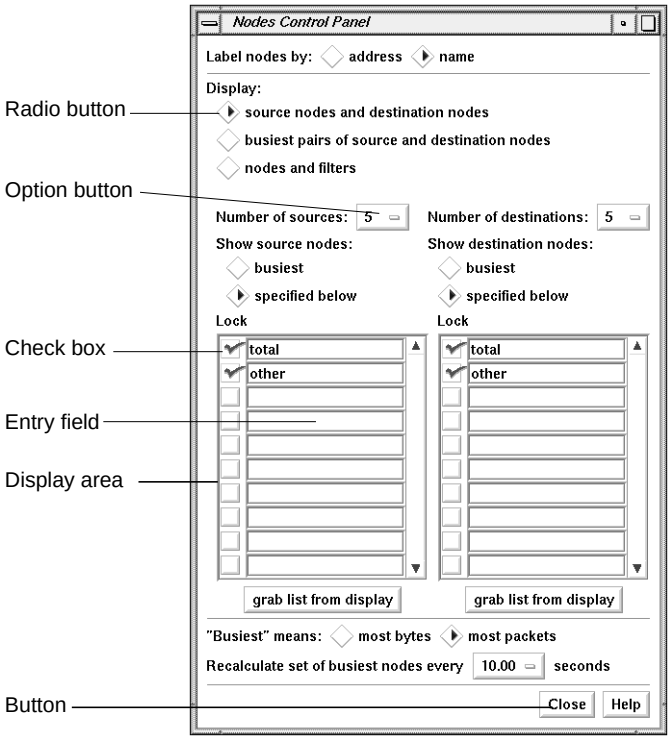



**Figure In-2** More Window Terms

This guide uses the following terms to describe the use of the mouse:

| | |
|---|---|
| press | Hold down a mouse button. |
| drag | Move the mouse while a mouse button is pressed. |
| click | Press a mouse button and immediately release it without moving the mouse. |
| double-click | Press and release a button twice in quick succession without moving the mouse. |
| triple-click | Press and release a button three times in quick succession without moving the mouse. |

| | |
|---|---|
| select | The term “select” is used in the following ways:<br>• Click the left mouse button on an item line to highlight it.<br>• Press the left mouse button in an entry field, drag the cursor across some or all of the text, and release the mouse button. The text becomes highlighted.<br>• Press the left mouse button on a menu title in a menu bar, move the cursor to a menu choice, and release the mouse button while a menu choice is highlighted.<br>• To select a traffic line, node, or network in the NetLook main window, double-click on it. |
| deselect | Click on a highlighted item to turn off the highlighting. |

## Common User Interface Operations

The graphical NetVisualyzer tools have a common look-and-feel for consistent operation and easy switching between tools. This guide assumes that you are familiar with using the mouse, working with windows, and using pulldown and rollover menus. These operations are described in the *IRIS Essentials*.

The sections below explain how to use additional components of the user interface that are common to several of the tools.

### Using Scroll Bars

You can use scroll bars (see Figure In-3) to change the area and scale of a viewing area and to display different lines or portions of lines in a display area. The size of the slider is proportional to the amount of the total that you are viewing. You operate scroll bars by pressing the left or middle mouse button when the cursor is in the scroll bar. There are several ways to operate the scroll bar:

- Press the left mouse button on the slider, drag the cursor to a new slider position, and release the button.

- Move the slider incrementally by clicking the triangles at each end of the scroll bar.
- Move the slider up or down by positioning the cursor in the trough above or below the slider and clicking the left mouse button.
- Move the slider to a specific position by positioning the cursor at that position and clicking the middle mouse button.

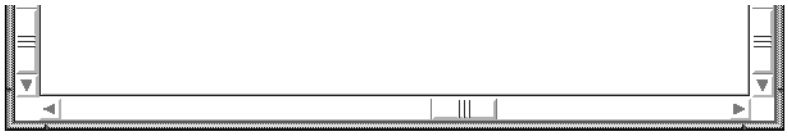


**Figure In-3** A Horizontal Scroll Bar

## Entering and Removing Text in a Field

Editing text in the entry fields (see Figure In-4) is the same as editing text in the entry fields of other applications:

- Position the text insertion point by moving the mouse to the entry field and clicking the left mouse button.
- Select (highlight) text by pressing the left mouse button at one end of the text that you want to select and dragging to the other end.
- Select a word including a space or punctuation-delimited characters by moving the cursor to the word and double-clicking the left mouse button.
- Select the entire contents of an entry field by moving the cursor over the entry field and triple-clicking the left mouse button.
- Delete selected (highlighted) text by pressing the **<Backspace>** key.
- Delete the character to the left of the insertion point by pressing the **<Backspace>** key.


**Figure In-4** An Entry Field

## Using Option Buttons

Option buttons (on the left in Figure In-5) let you select a numeric value from among a predefined set of choices. To use an option button, first press the option button with the left mouse button. A menu pops up (on the right in Figure In-5). Move the cursor to your selection and release the mouse button.

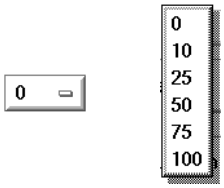



**Figure In-5** An Option Button and an Option Button Menu

## Using a File Prompter

File prompter windows (like the one in Figure In-6) are used to specify file names. You can choose a file name by double-clicking a name in the display area. You can also type the name into the File name entry field and press **<Enter>** or click the *Accept* button to complete your file name selection. You can change directories to the parent of the current directory by clicking the *Up* button, or return to the directory where you started the tool by clicking the *Original* button.

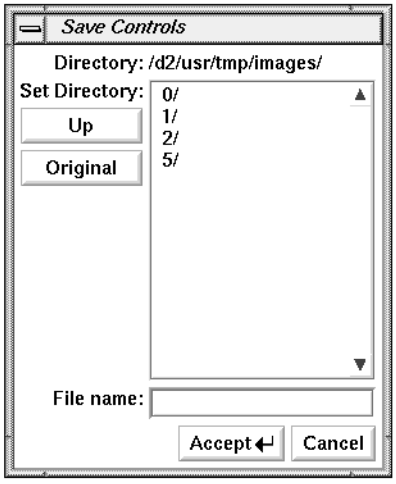



**Figure In-6** A File Prompter Window

## Using the Tools Menus

The menu bar of the main window of each graphical NetVisualyzer tool contains the Tools menu shown in Figure In-7. It contains the names of all of the graphical NetVisualyzer tools. You can use this menu to easily invoke other NetVisualyzer tools. When you select a choice from this menu, a Launch window appears. It contains an entry field with a command line for the tool you selected from the Tools menu. The command line includes the interface you are currently snooping on and the filter you are using, if applicable. Modify the command line if you need to and press the *Accept* button or press **<Enter>** to start the tool.


**Figure In-7** The Tools Menu

## Using Online Help

NetVisualyzer provides many online help files to help you as you learn to use the tools. You access these files from the Help menu in the menu bar of many NetVisualyzer windows (shown in Figure In-8 on top) and from the *Help* button that appears in some NetVisualyzer windows (on the bottom in Figure In-8).

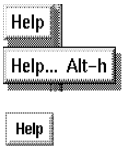



**Figure In-8** A Help Menu and a Help Button

When you choose "Help..." from a menu or click a *Help* button, a Showcase window appears and displays the first help card.

Some help files contain several cards. Page through these cards using the **<Page Up>** and **<Page Down>** keys in the cluster of six keys just to the right of the **<Backspace>** key or click the left mouse button on the arrows at the bottom of the pages. Make sure the cursor is in the Help window when you press these keys.

When you're finished reading a help file, you can close the Help window just as you close any other window, for instance, by double-clicking the Window menu button in the upper left corner of the window or by selecting "Quit" from the Window menu.

If you are using Analyzer, Browser, or NetFilters from an X terminal, the help cards are displayed as ASCII text rather than as Showcase files.

### Using Keyboard Accelerators

Keyboard accelerators (shortcuts) are available for many menu items and buttons in NetVisualyzer tools. The keyboard sequences are consistent across all of the tools so that once you learn the accelerator for Quit, you can use it with every tool. For example, **<Ctrl-s>** is the accelerator for Save Controls (user interface configuration) to the default file. Each tool that saves its user interface configuration understands this accelerator. In cases where a tool uses a button rather than a menu choice for a function that has a keyboard accelerator, you can still use the keyboard accelerator.

## References

The lists below contains general reference material that you may find useful as you use NetVisualyzer. A list of references for the protocols supported by NetVisualyzer is provided in "Protocol References" in Appendix C.

### Silicon Graphics Manuals

- *4DDN Network Management Guide and Man Pages*
- *FDDIVisualyzer User's Guide and Man Pages*
- *FDDIXPress Administration Guide*
- *IRIS Software Installation Guide*
- *IRIS Essentials*
- *IRIX Advanced Site and Server Administration Guide*
- *IRIX Network Programming Guide*
- *NFS Administration Guide*
- *NIS Administration Guide*

### Other Documents

- Kernighan, Brian W. and Ritchie, Dennis M., *The C Programming Language*, Prentice-Hall, Inc.
- Comer, Douglas E., *Internetworking with TCP/IP*, Second Edition, Volume I, Prentice-Hall, Inc.
- IEEE Standards Office, 345 East 47th Street, New York, NY 10017. Telephone (202) 705-7092

## Product Support

Silicon Graphics provides a comprehensive product support and maintenance program for its products. For further information, contact the Technical Assistance Center at 1-800-800-4SGI.

Chapter 1

# Getting Started with NetVisualyzer

*This chapter gives an overview of NetVisualyzer and the use of Data Stations and Display Stations to capture and display network traffic information. It also explains the set up tasks that you must do before using NetVisualyzer and how to use the netvis program to launch NetVisualyzer tools.*

*Chapter 1*

# Getting Started with NetVisualyzer

This chapter contains important information for NetVisualyzer users:

- "What Is NetVisualyzer?" presents a brief overview of the NetVisualyzer tools.
- "NetVisualyzer Display Stations and Data Stations" explains the terms Display Station and Data Station.
- "Snooping" explains how NetVisualyzer uses snooping to monitor and capture network traffic.
- "SNMP Agents" explains how NetVisualyzer uses SNMP agents to query and set Management Information Base (MIB) variables.
- "Event Logging" describes the event logging facility provided by NetVisualyzer tools.
- "Setting Up NetVisualyzer" describes what you need to do to install NetVisualyzer software, enable network snooping, authorize NetVisualyzer users, and set up network license management.
- "Using the netvis Directory View" explains how to use NetVisualyzer icons to start tools, view the online tutorial, and view manual pages.
- "NetVisualyzer Gifts and Resources" describes the sample scripts (4Dgifts) and resource files that are included with NetVisualyzer software.

## What Is NetVisualyzer?

NetVisualyzer is a set of software products that you install on workstations on your network to observe ("snoop on") network activity and capture packet data from the network. NetVisualyzer tools capture every packet of traffic on the network and graphically display packet data in a way that is easy to interpret.

With NetVisualyzer, you can answer questions such as:

- Which hosts are responsible for most of the traffic on my network right now?
- Which pairs of hosts have the most active conversations?
- Does my network topology make sense? How can I engineer my network segments to avoid routing as much as possible?
- How can I capture network traffic over time and analyze it later?
- What protocols predominate on the network at different times of the day?

NetVisualyzer provides network visualization tools that show real-time traffic between hosts and between network segments, a protocol analyzer, a traffic accounting package, and much more.

## NetVisualyzer Display Stations and Data Stations

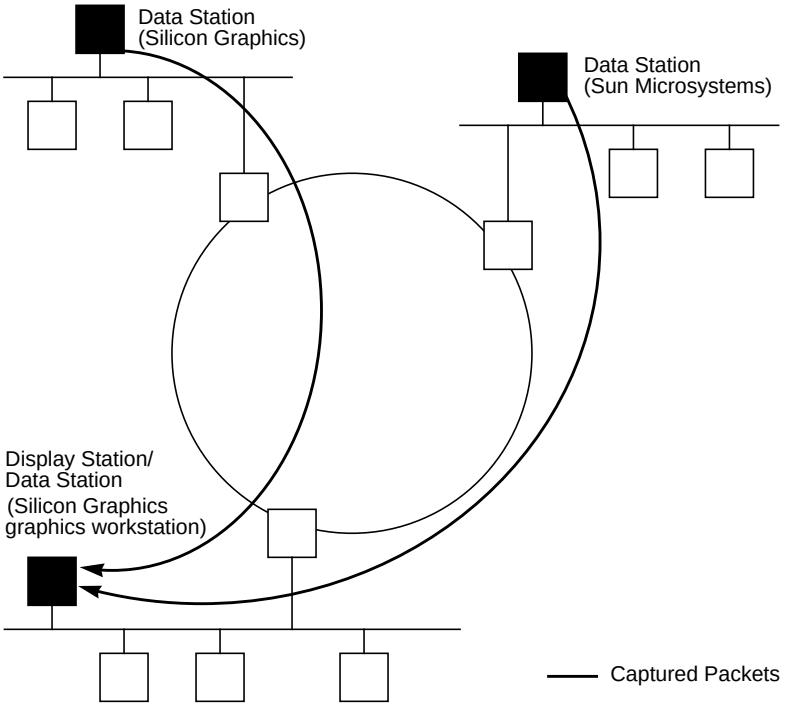
Workstations with NetVisualyzer software installed are called NetVisualyzer Data Stations and/or NetVisualyzer Display Stations.

A Data Station is a Silicon Graphics or Sun Microsystems workstation that has NetVisualyzer Data Station software installed.

A Data Station need not have graphics. Its purpose is to capture data for reports or for viewing and analyzing on Display Stations. For optimal network management and analysis, it's best to have a Data Station connected to each network segment.

Display Stations are Silicon Graphics workstations with graphics that have NetVisualyzer Display Station software installed. (Analyzer, Browser, and NetFilters can be run on X Window System™ terminals rather than on a graphics workstation.) Display Stations should also have Data Station software installed to enable them to collect data locally. Since they can access data collected by all Data Stations on your network, one or more Display Stations can be used to monitor your entire network at once. Figure 1-1 shows an example of a network with this configuration.

**Figure 1-1** Network Configuration for Monitoring an Entire Network




The NetVisualyzer tools include NetLook, NetGraph, Analyzer, NetSnoop, NetTop, NetCollect, NetPack, NetAccount, Browser, and NetFilters:

- Use NetLook to get an overall picture of the traffic patterns on the network. It displays your network's configuration, including its segments, hosts, routers, and traffic patterns.
- NetGraph displays real-time moving strip charts showing the amount of network traffic.

- Analyzer and NetSnoop capture and decode specific packets, enabling you to analyze packet protocol and data. Analyzer has a graphical user interface, while NetSnoop is a command-line utility.
- NetTop shows the volume of traffic between pairs of nodes and the volume of traffic to or from a node. Its 3-D graph shows up to one hundred node pairs at a time, giving you can get real-time information about network traffic at a glance.
- NetCollect, NetPack, and NetAccount capture network traffic over an extended period of time and generate statistical reports of that traffic.
- Browser displays MIB information for a node. You can browse the MIB structure, getting and setting values.
- NetFilters enables you to store and retrieve filters from archive files. This enables you to save frequently used filters and to quickly copy them for use with other NetVisualyzer tools.

**Note:** Most NetVisualyzer tools require that you have authorization on the Data Stations from which you are collecting network and node information. See "Authorizing NetVisualyzer Users for Snooping" and "Authorizing Browsing" in this chapter and Appendix B, "Authorization Reference," for details. ♦

## Snooping

NetVisualyzer tools monitor network traffic by continuously capturing all of the packet traffic on their network segment. They "see" packets from one node on the segment bound for another node on that segment, and they "see" packets that are traversing their segment while bound for nodes on other segments in the network. Each NetVisualyzer tool's view of the network is formed by the packets that flow past its network tap.

The monitoring of packet traffic is also referred to as "snooping." NetVisualyzer tools place the hardware interface into a promiscuous read mode and unobtrusively capture the information that permits each tool to provide different views of the network.

In a typical network, multiple network segments are joined by routers or non-transparent bridges. Segmented networks are used because they

manage the flow of packets among user communities and, therefore, conserve bandwidth and improve the availability of the internet. To monitor the entire network, NetVisualyzer tools must be able to snoop on each distinct network segment. Data Station software does this packet monitoring.

Data Station software is usually installed on one node on each segment of the network. The Data Stations can all be controlled from a single Display Station. Display Station tools provide the user interface and graphical output that present the network's behavior visually. Multiple Display Stations support multiple users, organizations, or locations.

Two types of snooping are used by NetVisualyzer tools. One type of snooping, RPC snooping, uses the Sun Remote Procedure Call protocol to communicate with a Display Station. The RPC snooping daemon, *snoopd*(1M), does snooping, packet decoding, and filtering. *snoopd*'s executable, *rpc.snoopd*, is started by the *inetd*(1M) daemon when a request for its service is received.

When a Display Station snoops on its own local interfaces, the other type of snooping, direct snooping, can be used. Direct snooping uses the IRIX kernel for snooping and bypasses *snoopd* completely. Direct snooping is available only to NetSnoop and Analyzer since these are the only tools that can request the return of full packets. Direct snooping provides the maximum performance at the sacrifice of the ability to connect to remote stations.

Each NetVisualyzer tool connects with *snoopd* and arranges for the services that it requires. *snoopd* returns either entire packets, portions of packets, or a distillation of the packets. Packets on the wire that are not of interest to you are not passed back by *snoopd*. Filter processing is performed on each Data Station, thus the packet traffic generated by NetVisualyzer is typically very small, less than a few percent of network bandwidth.

The packet-capturing services that *snoopd* provides to the NetVisualyzer tools are:

`netsnoop` service
: Analyzer and NetSnoop use this service. It returns entire packets or some specified number of bytes of each packet to the client.

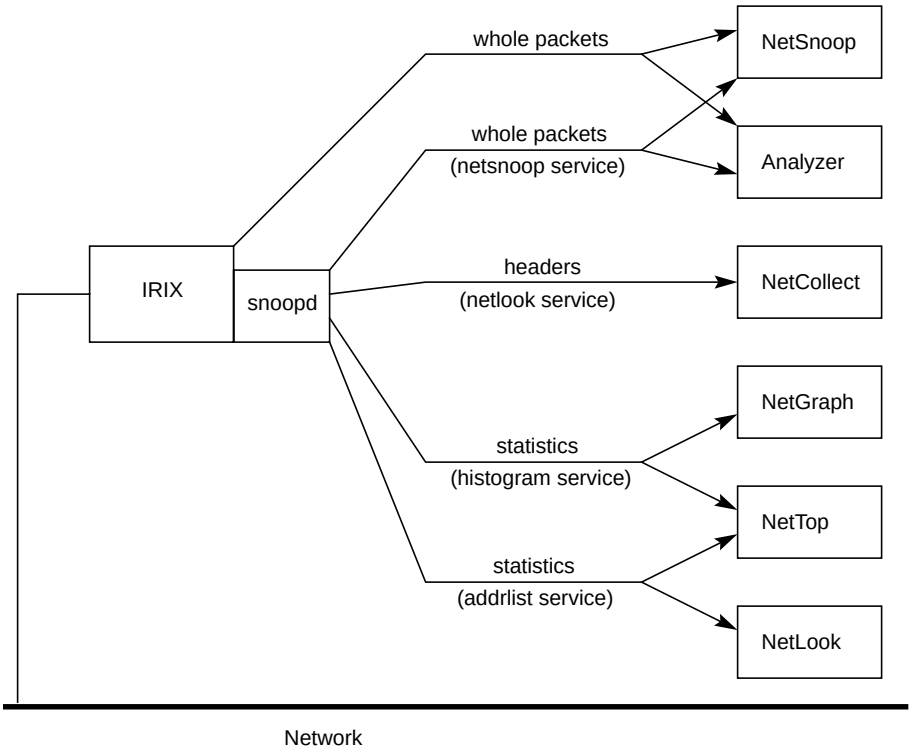
`netlook` service
: NetCollect uses this service. *snoopd* returns only the protocol headers from the packets.

`histogram` service
: NetGraph and NetTop use this service. *snoopd* returns only statistical data based on packet arrivals. No parts of the packets themselves are returned.

`addrlist` service
: NetLook and NetTop use this service. *snoopd* returns a list of source and destination pairs and the number of bytes and packets for each pair.

Figure 1-2 shows the relationship of the *snoopd* services to the NetVisualyzer tools.

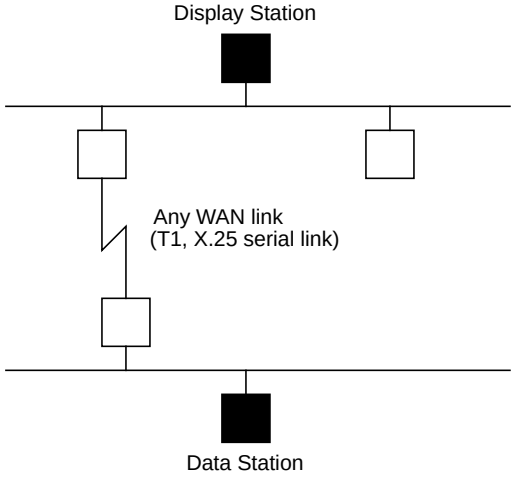
**Figure 1-2** Snooping and NetVisualyzer Tools



## Sending Data from a Data Station to the Display Station

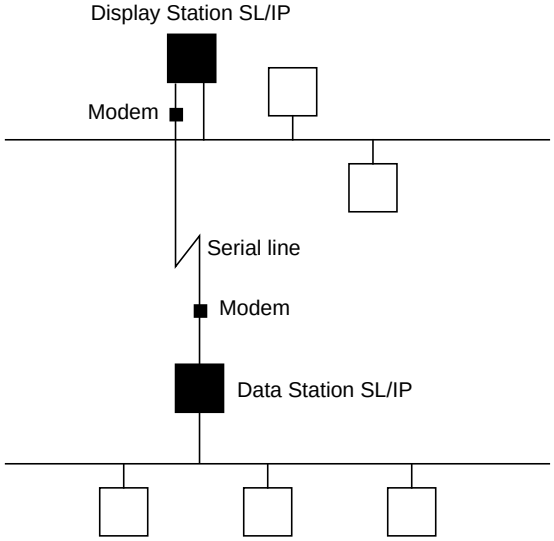
Distributed Data Stations send data to the central Display Station using the Internet Protocol (IP). Data Stations can forward the collected data to the Display Station through the local area network (LAN) medium through routers and bridges. In a wide-area network (WAN), Data Stations can forward data through any link, such as T1 or X.25, or any dial-up connection, such as serial line internet protocol (SL/IP), capable of passing IP packets (see Figure 1-3).

**Figure 1-3** Data Transmission over a WAN Link



In both LANs and WANs, a dial-up connection from the Data Station to the Display Station also provides an out-of-band means of monitoring network traffic, as shown in Figure 1-4.

**Figure 1-4** Data Transmission Using Dial-up Modems




## Address/Name Resolution

NetVisualyzer automatically resolves a node (host) name by using local files such as */etc/hosts* or by accessing the NIS and/or BIND name servers if they are specified by the `hostresorder` and `useyp` resources (you can use the **`–y`** command-line option instead of `useyp`). If you use a name server to resolve names, you must activate the appropriate daemon on the Display and Data Stations. NetVisualyzer tools display the names returned by the name server. They may be local names or full domain names.

The **`–y`** command line option (and the equivalent resource `useyp`) of Analyzer, NetAccount, NetGraph, NetLook, and NetTop controls whether or not NIS is used to resolve names from IP addresses. NIS is often used for more than just resolving names from IP addresses. NIS may have maps for mapping Ethernet address to name, service names to port number, RPC names to program number, and so on (give the command **`ypwhich -m`** to see

a list). The **–y** option enables or disables all of these maps. When the **–y** option isn't given, NIS is not used.

The `hostresorder` resource controls the order in which host lookup services are consulted when performing a host lookup based on name or IP address. Available services are NIS, BIND, and local files. Setting the `hostresorder` resource specifies the order in which these three services are tried. By default, the resource is set to `local`, which uses only the local files (and not NIS or BIND).

The **–y** option and `hostresorder` interact with each other in the following ways:

- If `hostresorder` is set to `local` and you give the **–y** option, NIS is used for several maps, but it is not consulted when doing a host lookup based on name or IP address because `hostresorder` says use only local files.
- If `hostresorder` is set to `local nis` and the **–y** option is not used, local files are tried first, and NIS is tried second. In this case, when a host lookup calls NIS for a query, NIS is off so it will just return with no match.
- If `hostresorder` is set to `local nis bind` and you give the **–y** option, a host lookup will first try local files, then NIS, then BIND. Since NIS is now on, a query generates a request to the NIS server, which returns appropriately.

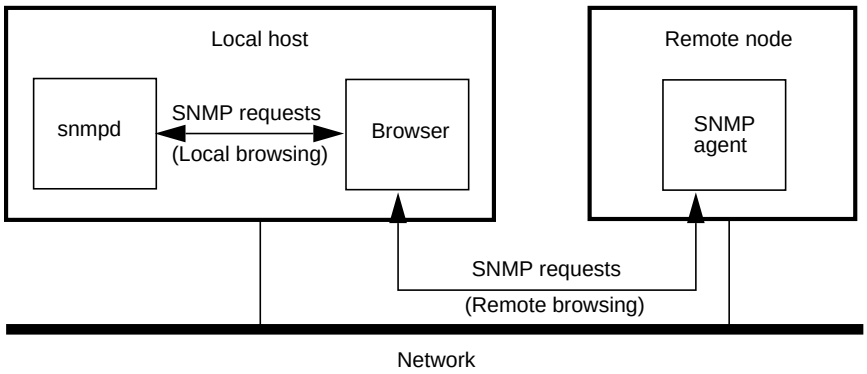
For information on NIS, see the *NIS Administration Guide*; for information on the BIND Name Server, see the *IRIX Advanced Site and Server Administration Guide*. For information on DECnet™ name resolution, turn to Chapter 11, "Using NetVisualyzer in a DECnet Environment."

## SNMP Agents

NetVisualyzer tools typically use snooping to obtain information. Browser, however, uses Simple Network Management Protocol (SNMP) agents to obtain information. The SNMP agent for Silicon Graphics workstations is *snmpd*(1M). Vendor-specific SNMP agents are used to obtain information about other types of nodes (see Figure 1-5). For more information on SNMP

see “SNMP Management Reference” in Appendix E. For information on enabling SNMP agents, see “Enabling SNMP Agents” in this chapter.

**Figure 1-5** SNMP Agents and Browser




## Event Logging

The NetVisualyzer tools Analyzer, Browser, NetCollect, NetGraph, NetLook, NetSnoop, and NetTop generate events if the ToolTalk™ message facility is installed and operating. These events are sent to the NetVisualyzer event server *nveventd*(1M). *nveventd* processes the events and places them in an event log. By default, the event log is the file */usr/lib/netvis/eventlog*.

NetVisualyzer tools automatically start *nveventd* if it is not already running when it is needed for event handling. Once started, it runs until it has no more clients and its idle timer has expired.

The events generated by the tools are listed below. Unless otherwise noted, they are generated by Analyzer, Browser, NetCollect, NetGraph, NetLook, NetSnoop, and NetTop.

`LOGGING BEGIN`
: *nveventd* starting time.

`NEW NET`
: NetLook detected traffic from a previously unknown network segment (NetLook only, configured off by default).

`NEW NODE`
: NetLook detected traffic from a previously unknown node (NetLook only, configured off by default).

`RATE HIGH ALARM`
: NetGraph detected an alarm high condition (NetGraph only).

`RATE HI CLEARED`
: NetGraph detected that an alarm high condition no longer is met (NetGraph only).

`RATE LO ALARM`
: NetGraph detected an alarm low condition (NetGraph only).

`RATE LO CLEARED`
: NetGraph detected that an alarm low condition no longer is met (NetGraph only).

`SHUTDOWN`
: The tool is exiting.

`START SNOOP`
: The tool is starting to snoop on an interface.

`STARTUP`
: The tool is starting up.

`STOP SNOOP`
: The tool is terminating snooping on an interface.

Event log entries include a timestamp, the event type, the name of the application that generated the event and its process id, the user's login name and user id, the alarm level for the event, the interface and filter being used for snooping, and other event specific information. Some sample entries are:

```
Sep 17 11:52:34 RATE LO ALARM NetGraph[11317] probins(3138) MINOR () total
1 packets/sec threshold: 10
Sep 17 11:52:36 RATE LO CLEARED NetGraph[11317] probins(3138) INFO () total
73 packets/sec threshold: 10
Sep 17 11:53:47 STARTUP NetGraph[11387] probins(3138) INFO ()
Sep 17 11:53:49 START SNOOP NetGraph[11387] probins(3138) INFO
bubba.wpd.sgi.com(192.26.75.178)
```

See the *nveventd*(1M) manual page for additional details.

## Setting Up NetVisualyzer

To install and prepare to use NetVisualyzer, you must perform these tasks:

1. Install NetVisualyzer Display Station software on workstations that you want to be Display Stations and NetVisualyzer Data Station software on workstations that you want to be Data Stations (see "Installing NetVisualyzer Software").
2. Enable snooping on Data Stations (see "Enabling Network Snooping").
3. Enable the SNMP agent on each node that you want to obtain information from using Browser (see "Enabling SNMP Agents").
4. Provide authorization for NetVisualyzer users in */usr/etc/rpc.snoopd.auth* on Data Stations (see "Authorizing NetVisualyzer Users for Snooping").
5. Provide authorization for NetVisualyzer users in */usr/etc/snmpd.auth* on Silicon Graphics workstations that you want to obtain information from using Browser (see "Authorizing Browsing").
6. Create */etc/passwd* entries for NetVisualyzer users on Data Stations (see "Creating Password File Entries").
7. Bring the */etc/hosts* file up to date on the Data and Display Stations if necessary (see "Updating /etc/hosts").
8. Configure event logging on Display Stations if the default event logging is not suitable (see "Configuring Event Logging").
9. Initialize NetVisualyzer network licensing on Display Stations (see "Setting Up Network Licensing").

These tasks are explained in the following sections. Most of these tasks need to be done just once—when you install NetVisualyzer software. You may need to perform the authorization tasks again at a later time to authorize additional NetVisualyzer users or reconfigure event logging.

The Display and Data Stations use TCP/IP as the standard networking protocol. If you are installing NetVisualyzer and setting up stations on a DECnet network, you need to do a few additional preparation tasks. For details, see Chapter 11, "Using NetVisualyzer in a DECnet Environment."

## Installing NetVisualyzer Software

Become superuser (*root*) and use *inst*(1M) to install NetVisualyzer software from IRIX.

NetVisualyzer software has been structured so that most files are installed in */usr/NetVis* and */usr/ToolTalk*. The exceptions are the configuration files */usr/etc/rpc.snoopd.auth* and */usr/lib/netvis/eventcfgrc*, and manual pages that are installed in their usual locations. When you install the *links* subsystems, symbolic links are created to link the files in */usr/NetVis* and */usr/ToolTalk* to familiar locations such as */usr/sbin*. This makes several configurations possible:

- To install software on your workstation, install the subsystems you want, such as *netvis_data.man.data*, *netvis_data.man.relnotes*, and *netvis_data.sw.data*, and also install the corresponding *links* subsystem (*netvis_data.sw.links* in this example).
- To use NetVisualyzer software that you have NFS-mounted from another workstation, install the non-*links* software on the remote workstation, NFS-mount it at */usr/NetVis*, and then install the *links* subsystem on your workstation.

No matter which configuration you choose, the *links* subsystem for each product you use must be installed on your workstation.

NetVisualyzer uses software that is provided in the subsystems *eoe2.sw.ipgate*, *eoe2.sw.netman*, and *eoe2.sw.tcp*. To check to see if these subsystems are installed on your workstation, give this command:

```
versions eoe2.sw.ipgate eoe2.sw.netman eoe2.sw.tcp
```

Each subsystem should be listed in the output. If not, use *inst*(1M) to install them from your system software tapes or CD.

Additional information on installing NetVisualyzer software is provided in the *NetVisualyzer Release Notes*. For more information on using the installation program *inst*(1M), see the *IRIS Software Installation Guide*.

For information on installing NetVisualyzer Data Station software on Sun Microsystems workstations, see the *NetVisualyzer Data Station Release Notes for Sun Systems*.

## Enabling Network Snooping

As Figure 1-2 shows, the *snoopd* daemon and IRIX listen to every packet of information on the network. For each Data Station on the network on which you want to snoop (including your own), you must turn on *snoopd*. To turn it on:

1. Find out if you are using network information service (NIS) by giving this command:

   **`/etc/chkconfig`**

   If the output you get contains `yp off`, you are not using NIS and you can skip to step 4.

2. Check for *snoopd* by giving this command:

   **`ypmatch 391000 rpc`**

   If you get this output:

   ```
   sgi_snoopd      391000  snoopd snoop
   ```

   your NIS master is a Silicon Graphics workstation and you can skip to step 4.

3. Since, you are using NIS, but your NIS master is not a Silicon Graphics workstation, you must add the following line to the file */etc/rpc* on your NIS master:

   ```
   sgi_snoopd      391000  snoopd snoop
   ```

4. Restart the *inetd*(1M) daemon by giving this command on your workstation:

   **`/etc/killall -HUP inetd`**

Restarting *inetd* after installing NetVisualyzer software causes the *snoopd* daemon to start when a NetVisualyzer tool sends a service request.

For more information, see *snoopd*(1M), *inetd*(1M), and *rpc*(4). For information on NIS, see the *NIS Administration Guide*.

## Enabling SNMP Agents

Browser must communicate with the SNMP agent on each node you wish to browse. SNMP agents and the procedures for enabling them are vendor-specific. The procedure for enabling the SNMP agent on Silicon Graphics workstations is described below. For other types of nodes, contact the system administrator for that node for help in enabling SNMP on that node.

The Silicon Graphics SNMP agent is *snmpd*(1M). The SNMP agent software is contained in the subsystem *eoe2.sw.netman*. To configure a workstation so that *snmpd* is started automatically when the system is rebooted, give this command as *root*:

```
chkconfig snmpd on
```

To check to see if the daemon is already running, give this command:

```
ps -e | grep snmpd
```

If there is no output from this command, *snmpd* is not running. Give this command as *root* to start *snmpd*:

```
snmpd
```

## Authorizing NetVisualyzer Users for Snooping

If you want to snoop on a Data Station, you must have authorization in the file */usr/etc/rpc.snoopd.auth* on that Data Station. You must be superuser (*root*) to read or write */usr/etc/rpc.snoopd.auth*. For security reasons, the owner and permissions of this file should not be changed.

A simple authorization line in */usr/etc/rpc.snoopd.auth* has the form:

`accept localhost:`*user*

This line authorizes the user *user* to use all *snoopd* services on this Data Station when he or she starts NetVisualyzer tools from this Data Station. *user* can be a login name, a numerical user id, or an asterisk (*), which stands for all users. To use `localhost` in */usr/etc/rpc.snoopd.auth*, `localhost` must be defined in */etc/hosts*.

Another simple authorization line is:

```
accept *
```

This line authorizes all users from all hosts.

By default, */usr/etc/rpc.snoopd.auth* contains this authorization line:

```
accept  localhost:root.sys
```

This line authorizes the user *root* with group *sys* to snoop locally on this Data Station.

"/usr/etc/rpc.snoopd.auth" in Appendix B contains additional information about the syntax of */usr/etc/rpc.snoopd.auth* and the types of authorizations and restrictions you can specify in this file.

## Authorizing Browsing

No special authorization other than a valid community string is required to browse on nodes other than Silicon Graphics workstations. (See "Browser Main Window" in Chapter 9 and "SNMP Management Reference" in Appendix E for more information about community strings.)

If you want to get and set MIB variables on a Silicon Graphics workstation using Browser, you must do several setup steps in addition to providing a valid community string while using Browser: confirm that the workstation you are browsing has SNMP agent software running, start it if necessary (see "Enabling SNMP Agents" in this chapter), and authorize your Display Station to browse on that workstation.

To be authorized to browse a Silicon Graphics workstation, the Display Station's host name must be specified in the file */usr/etc/snmpd.auth* on the workstation you are browsing. You must be superuser (*root*) to read or write */usr/etc/snmpd.auth*. For security reasons, the owner and permissions of this file should not be changed.

As an example, suppose that you want to browse a Silicon Graphics workstation named `tahoe`. Your workstation's name is `sequoia`. First, confirm that *snmpd* is running on `tahoe`:

```
rsh guest@tahoe 'ps -e | grep snmpd'
```

Assuming that it is running, log onto `tahoe` as superuser and add this line to */usr/etc/snmpd.auth*:

```
accept sequoia:*
```

This line authorizes anyone using your workstation to browse the workstation `tahoe` when they give any community string. These users can perform both `get` and `set` operations.

By default, */usr/etc/snmpd.auth* contains this authorization line:

```
accept  *:public/get
```

This line authorizes any user from any host who provides the community `public` to get variable values for this workstation.

See the *snmpd*(1M) manual page and the file for more information about the syntax used in this file.

## Creating Password File Entries

In most cases, if you want to access NetVisualyzer tools on a Data Station, you must have an entry in the file */etc/passwd* on that Data Station. If your login name does not appear in */usr/etc/rpc.snoopd.auth* (if the file includes the line `accept *`, for example) you do not need to have an entry in */etc/passwd*.

See *passwd*(4) and the *Personal System Administration Guide* for information on creating */etc/passwd* entries.

## Updating */etc/hosts*

NetVisualyzer tools use the information in */etc/hosts*, among other methods, to convert node names to IP addresses. It is important that the information in this file be accurate. Your system administrator can provide you with instructions on updating this file.

Your */etc/hosts* file must include a definition for `localhost` in order for you to use this name as a pseudonym for your workstation's name in */usr/etc/rpc.snoopd.auth* (see "Authorizing NetVisualyzer Users for Snooping" in this chapter).

## Configuring Event Logging

When ToolTalk software is installed and operating, NetVisualyzer tools automatically log events as described in "Event Logging" in this chapter. You can configure event logging so that it does not send certain types of events, so that it has non-default values for alarm events, and to specify the number, size, and name of event log files. If you do not install ToolTalk, all NetVisualyzer tools operate normally, but there will be no event logging.

Install ToolTalk, configure it, and start it by following these steps:

1. From the NetVisualyzer software distribution, install *ToolTalk.sw.runtime* and *ToolTalk.sw.links*. You can optionally install *ToolTalk.man.manpages*. If this installation is done at the same time as NetVisualyzer software installation, and you do not need to customize event logging, skip the remaining steps in this section. ToolTalk is started automatically in this case.
2. To customize event logging, edit the file */usr/lib/netvis/eventcfgrc*. See the comments in this file and the *nveventd*(1M) manual page for details on how to modify this file.
3. If ToolTalk was installed after NetVisualyzer, give these commands as *root* to start ToolTalk:

   ```
   cd /usr/lib/netvis
   /usr/sbin/tt_type_comp -d system nveventd_types
   killall -USR2 ttsession
   ```

4. If you modified */usr/lib/netvis/eventcfgrc*, give this command to stop *nveventd*:

   ```
   killall nveventd
   ```

   *nveventd* is restarted automatically using the new configuration file by NetVisualyzer tools.

## Setting Up Network Licensing

NetVisualyzer Display Station software is licensed to run on a single workstation only. Each Display Station is licensed to snoop on a limited number of Data Stations simultaneously. For example, say that your Display Station is licensed to snoop on five Data Stations. Using NetLook, you snoop on five different Data Stations. While you are snooping on NetLook, you can start NetTop, but you must use one of the Data Stations already used by NetLook because you are already snooping on five Data Stations. Snooping on two interfaces of a single Data Station counts as two Data Stations.

To initialize the license server for a Display Station, you must obtain a password string from Silicon Graphics. This password string licenses your Display Station to snoop on a specific number of Data Stations. Licensing information included with the NetVisualyzer Display Station software gives complete information for obtaining a password from Silicon Graphics. The general procedure is:

1. Start the NetVisualyzer license command, *nvlicense*(1M), by entering this command as *root*:

   **`nvlicense`**

   The window shown in Figure 1-6 appears.

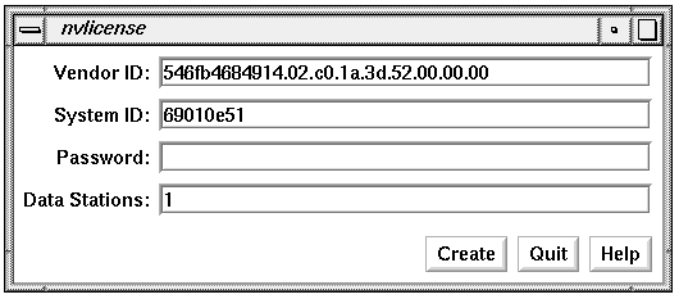



**Figure 1-6** *nvlicense* Window

2. Contact Silicon Graphic as instructed in the Network Licensing information sheet included with NetVisualyzer.

3. Report the string in the System ID entry field to Silicon Graphics, and confirm the number of Data Stations that your Display Station will be licensed to use.
4. Exit *nvlicense* by clicking the *Quit* button. After some period of time, a password will be generated and sent to you.
5. When you receive the password, start *nvlicense* again as *root*:

   ```
   nvlicense
   ```

6. In the Password entry field, enter the password string supplied by Silicon Graphics.
7. In the Data Stations entry field, enter the number of Data Stations from step 3. You must enter exactly the number of Data Stations that was used to create your password string.
8. Click the *Create* button to create the license.
9. Click the *Quit* button to exit *nvlicense*.

This procedure enables you to use all Display Station tools (Data Station tools are not licensed).

If you have been given a password that has an expiration date encoded in it, license expiration warning messages appear when you start some NetVisualyzer tools within 30 days of expiration. Contact Silicon Graphics to obtain a new password.

## Using the *netvis* Directory View

The *netvis*(1M) command provides a convenient way to start NetVisualyzer tools, to bring up the NetVisualyzer tutorial, and to view NetVisualyzer manual pages.


**Figure 1-7** *netvis* Icon

If you are using WorkSpace, the *netvis* icon, shown in Figure 1-7, is automatically displayed.

When you click the *netvis* icon, the *netvis* directory view window shown in Figure 1-8 appears.

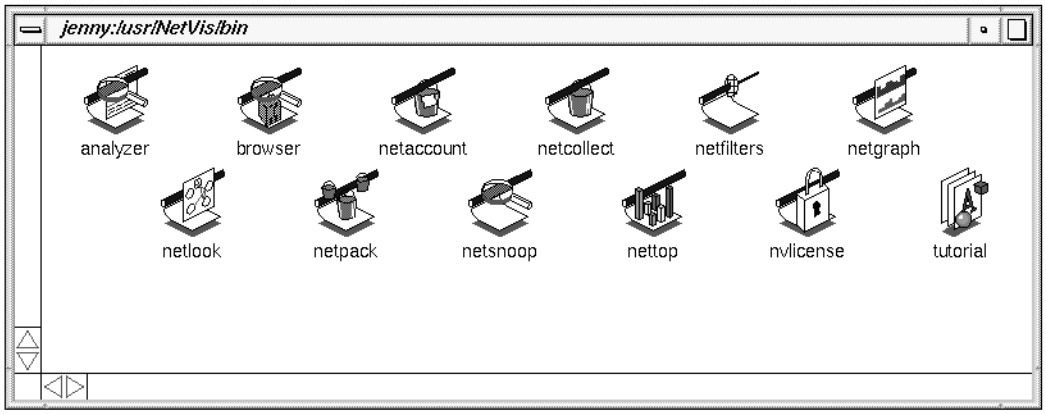



**Figure 1-8** *netvis* Directory View

If you are not using WorkSpace, give this command to display the *netvis* directory view in Figure 1-8:

```
netvis
```

**Note:** Under certain conditions, some tools require that you be superuser (root) to run them. When this is the case, you must start WorkSpace as root or give the netvis command as root when you want to invoke these tools from the netvis directory view. ♦

The following sections describe how to use the *netvis* directory view to start tools, bring up the tutorial, and view manual pages.

## Starting NetVisualyzer Tools

To start any NetVisualyzer tool without specifying command-line options, double-click the icon for that tool (or click once and select “Open” from the

WorkSpace pop-up menu). For example, to start NetLook, double-click the left mouse button on the *netlook* icon.

There are three methods for starting NetVisualyzer tools with options:

- double-clicking the tool icon with the **<Alt>** key pressed ("**<Alt>** Open")
- double-clicking a data file for the tool ("Open Data File")
- selecting one or more data files and dropping them on the tool icon ("Drop Data File")

Each of these methods is described in a following section. Not all of these methods work with each tool; Table 1-1 lists the methods, whether or not they can be used, and data files that are applicable to each tool.

**Table 1-1** Starting NetVisualyzer Tools with Arguments

| Tool | <Alt> Open? | Open Data File? | Drop Data File? |
|---|---|---|---|
| Analyzer | yes | yes, *packet_file* (from Analyzer or NetSnoop) | yes, *.analyzerrc* |
| Browser | yes | no | no |
| NetAccount | yes | yes, *traffic_file* (from NetCollect or NetPack) | yes, *traffic_file* (from NetCollect or NetPack) |
| NetCollect | yes | no | no |
| NetFilters | yes | yes, *filter_file* | yes, *filter_file* |
| NetGraph | yes | yes, *history_file* (from NetGraph) | yes, *.netgraphrc* |
| NetLook | yes | yes, *network.data* | yes, *.netlookrc* |
| NetPack | yes | no | yes, *traffic_files* (from NetCollect) |
| NetSnoop | yes | no | no |
| NetTop | yes | no | yes, *.nettoprc* |

### Using the “<Alt> Open” Method

When you use this method, a window appears in which you can type options and arguments. The steps are:

1. Select the icon that represents the command you want to execute by pressing and holding the **<Alt>** key and then double-clicking the left mouse button on the icon. A Launch Command window, such as the one shown in Figure 1-9, appears.
2. Complete the NetVisualyzer command (the command can include options, a filter expression, and other arguments).
3. Press the *Accept* button (or press **<Enter>**) to execute the command and start the tool.

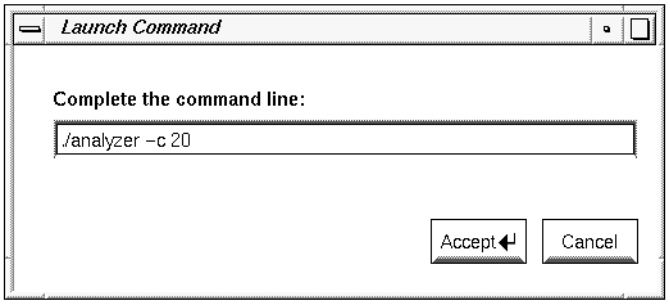



**Figure 1-9** Launch Command Window

### Using the “Open Data File” Method

Open a data file that you want to use as an argument to a NetVisualyzer tool. Simply double-click the file and it automatically becomes an argument to the tool. Table 1-1 lists the tools and data files that use this method.

### Using the “Drop Data File” Method

Select the data files that you want to use as arguments to a NetVisualyzer tool and drag them to the NetVisualyzer tool icon.

1. Click on a data file you want to use.
2. If you want to give additional data files as arguments, press and hold a **<Shift>** key while clicking on the additional data files.
3. Press the left mouse button on one of the data files.
4. Drag the cursor to the icon for the tool you want to use.
5. Release the left mouse button.

## Viewing the NetVisualyzer Online Tutorial

An online tutorial for NetVisualyzer is included in the subsystem *netvis_display.man.tutorial*. Designed for people who have not used NetVisualyzer before, it provides lessons on the basic operation of many of the NetVisualyzer tools.

To view the tutorial, double-click the *tutorial* icon in the *netvis* directory view.

## Viewing NetVisualyzer Manual Pages

You can view NetVisualyzer manual (man) pages easily using the *netvis* directory view. To display a NetVisualyzer man page:

1. Select the icon for the NetVisualyzer tool that interests you by clicking the left mouse button once on the icon. For example, if you want to see the Analyzer manual page, click the *analyzer* icon.
2. Display the Directory View pop-up menu by pressing and holding the right mouse button. Select “Manual page” from the menu.
3. A new window appears containing the manual page you requested. The top of the window shows the name of the page you are viewing. You can manipulate this window in the same way you manipulate any window; for example, you can close it by double-clicking the Window menu button.

## NetVisualyzer Gifts and Resources

The NetVisualyzer Display Station software product includes sample scripts that you may wish to use with NetVisualyzer tools. The directory */usr/people/4Dgifts/examples/netvis* contains these sample scripts. Refer to */usr/people/4Dgifts/examples/netvis/README* for a description of the contents of this directory.

Resource files for Analyzer, Browser, NetFilters, NetGraph, NetLook, and NetTop are included with Display Station software. They are installed in */usr/lib/X11/app-defaults*. The resources for each tool are listed in the tool's manual page in Appendix F, "NetVisualyzer Manual Pages."

Chapter 2

# NetFilters

*This chapter explains the use of NetFilters to store and retrieve filters. Filters are used with NetVisualyzer tools to capture only packets of interest. Filter syntax is explained in Chapter 10.*

*Chapter 2*

# NetFilters

This chapter explains how to use NetFilters to store and retrieve NetVisualyzer filters. NetFilters is a graphical tool that allows you to use the repository of filters provided with NetVisualyzer or create your own repository of useful filters. With NetFilters, you can easily extract a filter and use it in NetLook, Analyzer, NetGraph, or NetTop. The variable feature of NetFilters enables you to write "generic" filters and replace the variables with specific information such as host names when you use the filters. NetFilters can be used to manage several filter repositories.

This chapter explains how to:

- start NetFilters
- use the NetFilters main window to manage a repository of filters for use with NetVisualyzer tools
- create general filters and customize them when used
- copy filters from a NetFilters repository to other NetVisualyzer tools

In addition, a NetFilters example is provided. For complete information on NetFilters command line options and resources, see the *netfilters*(1M) manual page in Appendix F, "NetVisualyzer Manual Pages." For information about filter syntax and constructing filters, see Chapter 10, "Creating and Using Filters."

**Note:** To modify the filter repository that is included with NetVisualyzer, you must invoke NetFilters as root. You need not be root to use the standard filter repository without modifying it or to create and modify your own filter repositories. ♦

## Starting NetFilters

You can start NetFilters in several ways:

- Start NetFilters from the *netvis*(1M) directory view by double-clicking the *netfilters* icon.
- Give the NetFilters command from the shell by entering:

  **`netfilters`**

- Start NetFilters from other NetVisualyzer tools by using the *NetFilters* button provided near Filter entry fields. Put the insertion point in the entry field you want to fill in, and press the *NetFilters* button to start NetFilters.

Once you start NetFilters, the NetFilters main window appears. The default main window is shown in Figure 2-1.

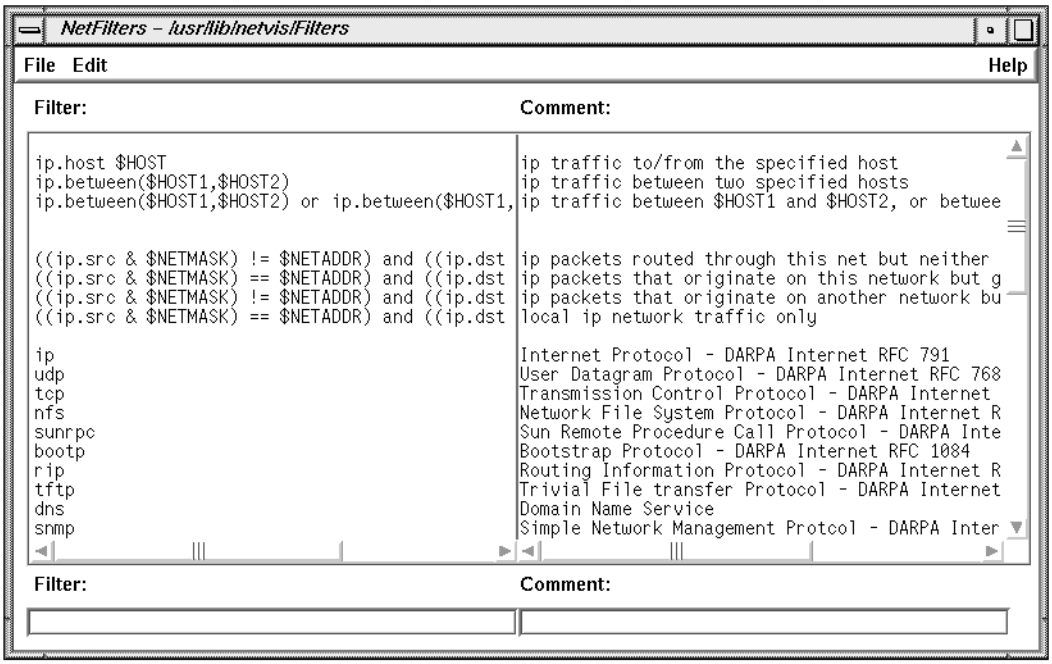



**Figure 2-1** NetFilters Main Window

By default, NetFilters looks first for a filter repository file called *Filters* in your home directory, then in */usr/lib/netvis*. You can override the default selection of a filter repository file by specifying the filter repository file as an argument to the **–f** command line option:

**`netfilters -f`** *repository*

Or, using the *netvis* directory view, you can press the left mouse button on the icon for the repository, drag the icon to the NetFilters icon, and release the mouse button.

## NetFilters Main Window

The main window contains a large Filter and Comment scrolling display area. The left side contains one line for each filter in the filter repository file. Next to each filter is a comment for the filter. Scroll bars on the right and beneath the filters and comments let you scroll to view additional filters, long filters, and long comments.

The Filter and Comment entry fields at the bottom of the window are used to enter and modify filter text and comments. In addition, you can perform these operations:

- To copy a filter and its comment from the Filter and Comment display area to the Filter and Comment entry fields, select the filter by clicking on it.
- To erase the contents of the entry fields when a filter in the display area is selected, click on the selected filter in the display area.
- To select a word in an entry field, double-click the word.
- To select an entire filter or comment in an entry field, triple-click on it.
- To use the Filter Variables window to substitute values for variables in the Filter entry field, see “Using Variables in Filters” in this chapter.

## NetFilters Edit Menu

To add a filter to the display area, remove it from the display area, or modify a filter in the display area, use "Add," "Remove," or "Replace" on the Edit menu:

| | |
|---|---|
| "Add" | Adds the text in the Filter and Comment entry fields to the list of filters in the display area. They are added at the end. |
| "Remove" | Removes the currently selected filter and comment from the display area. |
| "Replace" | Replaces the currently selected filter and comment with the filter text and comment in the entry fields. |

## Using Variables in Filters

Filters in the display area can include variables. Variables enable you to store a "generic" filter in the display area and fill in specific values when you use the filter. An example is:

```
ip.between($HOST1,$HOST2)
```

Variables are white-space or punctuation-delimited strings that begin with "$".

You can create filters with variables and add them to the display area, just like any other filter (see "NetFilters Main Window" in this chapter).

To use a filter with one or more variables, edit the filter in the Filter entry field to substitute values for the variables, or follow these steps:

1. Put the generic version of the filter in the display area if it isn't already there.
2. Select the generic filter you want to use by clicking on it in the filter display area.

3. Select “Specify variables...” from the Edit menu. The Filter Variables window shown in Figure 2-2 appears. It contains an entry field for each of the variables that appear in the filter in the display area.

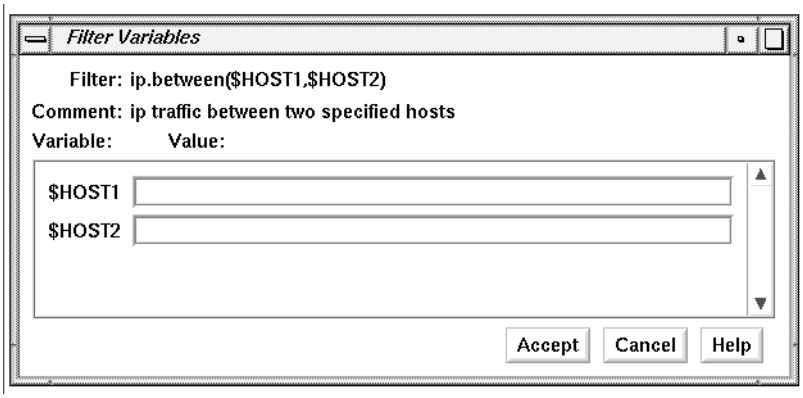



**Figure 2-2** Filter Variables Window

4. Enter the values of the variables in the entry fields in the Filter Variables window.
5. Click *Accept*. The Filter Variables window disappears. In the Filter entry field in the main window, the values you entered have been substituted.
6. You can now use the customized filter in another NetVisualyzer tool. See “Using NetFilters to Specify Filters for Other NetVisualyzer Tools” in this chapter for directions.

## NetFilters File Menu

The File menu provides choices that manage NetFilters repository files and quit NetFilters:

| | |
|---|---|
| "New" | Use "New" to begin the development of a new set of filters in the Filter and Comment display area. It erases all filters and comments from the Filter and Comment display area entry fields. If you have not saved the current display area, you are prompted to save it to a repository file with a dialog box before it is erased. |
| "Open" | Use "Open" to specify the name of the filter repository file you want to use. When you select "Open," a file prompter window appears. See "Using a File Prompter" in the Introduction for information on using this window. |
| "Save" | Clicking "Save" saves the current version of the Filter and Comment display area to the filter repository file whose name appears in the title of the NetFilters main window. |
| "Save As..." | When you click "Save As...", a file prompter window appears. Use this window to specify the name of the filter repository file where you want to save the current display area. See "Using a File Prompter" in the Introduction for information on using this window. |
| "Quit" | To quit NetFilters, select "Quit." If you have modified the filter and display area but not saved the changes to a filter repository file, a dialog box appears to prompt you to save it. |

## Using NetFilters to Specify Filters for Other NetVisualyzer Tools

You can easily copy a filter from the NetFilters display area or Filter entry field to an entry field in Analyzer, NetGraph, NetLook, or NetTop:

- To copy a filter from the list of filters in the display area when the insertion point is in the entry field where you want the copy, just double-click the filter you want to copy.
- To copy a filter from the list of filters in the display area when the insertion point is not in the entry field where you want the copy, click the filter in the display area to select it, move the cursor to the entry field where you want the copy, and click the middle mouse button.
- To copy a filter from the Filter entry field in NetFilters, triple-click on the filter (or drag the mouse over the entire filter to highlight it), move the cursor to the entry field where you want the copy, and click the middle mouse button.

## NetFilters Example

Suppose you are responsible for a network segment that is displaying much more traffic in NetLook than you would expect from the applications running on its nodes. You want to determine whether nodes on your network are responsible for the traffic you see.

First, eliminate common causes for this behavior such as `arp` storms using NetLook. To check for `arp` storms, first select “show the local hop” on the NetLook control panel, then put the insertion point in the Filter entry field in the Snoop control panel, and click the *NetFilters* button. When the NetFilters main window appears, double-click the filter `arp` in the display area. The filter is automatically copied to the Snoop control panel and NetLook automatically begins using that filter. If very little traffic is displayed after the filter takes effect, you’ve eliminated an `arp` storm as a possible cause of the problem.

Next, find the filter in the NetFilters display area that has the comment:

```
ip packets routed through this net but neither originate nor terminate on it
```

Click this filter to copy it to the Filter and Comment entry fields of the NetFilters main window. Select “Specify variables...” from the NetFilters Edit menu. In the Filter Variables window that appears, specify the netmask your network is using in the first entry field and the IP address of your network segment in the second entry field. Click the *Accept* button. The filter, with the variables replaced by your netmask and network segment IP address, appears in the Filter entry field of the NetFilters main window.

In the NetFilters main window, select “Add” from the Edit menu. The filter is copied from the Filter and Comment entry fields to the end of the filter list in the display area.

Double-click on the new filter. It is automatically placed in the Filter field of NetLook’s Snoop control panel, and NetLook automatically begins to use this filter. Now the only traffic you see on the network segment is traffic routed through the network segment.

Select “Analyzer” from the Tools menu of NetLook. The command line in the Tools Prompt dialog box that appears automatically contains the filter you are using in NetLook. Capture 200 packets with Analyzer. Inspect the sources and destinations of these packets in the Summary pane of Analyzer. This information tells you which nodes on which networks are routing traffic through your network. This information can help you detect routing problems, make a case for re-engineering your network to add segments and routers, and so on.

In NetFilters, use “Save As...” on the File menu to save the filters in the display area to a personal filter file for later use. Now you can pass the new filter you created to any NetVisualyzer tool to periodically look at the problem.

Chapter 3

# NetLook

*NetLook provides a graphical display of network configuration and traffic flow. This chapter explains how to NetLook and provides several NetLook examples.*

*Chapter 3*

# NetLook

This chapter describes NetLook, the window on your network that monitors network configuration and traffic flow. It provides a bird's-eye view of the network, which can reveal at a glance a connectivity problem, a security breach, or an effective way to reconfigure a network. Developers of distributed applications can use NetLook to see the pattern of traffic generated by an application.

NetLook's windows allow you to view network traffic and configuration, and to specify the type of network traffic you want to view. It displays complex network information in a simple-to-use and easy-to-understand fashion; you don't need to know packet-level details.

Using lines of varying colors to represent traffic volume between the communicating nodes, NetLook displays both the location and intensity of your network's traffic. A node can be any network device such as a workstation interface or router.

This chapter explains how to:

- start NetLook
- use the NetLook main window to display network configuration and traffic information
- use the NetLook control panels to specify the type of traffic you want to view and configure the display of traffic
- use choices on the Actions menu to perform operations on a single node or network segment
- use configuration files to retain network and NetLook configuration information for later use

In addition, a variety of NetLook examples are provided. For complete information on NetLook command-line options and resources, see the

*netlook*(1M) manual page in Appendix F, "NetVisualyzer Manual Pages." Additional information about NetLook configuration files is provided in Appendix D, "Configuration File Formats."

**Note:** You must have authorization to use NetLook. See "Authorizing NetVisualyzer Users for Snooping" in Chapter 1 and Appendix B, "Authorization Reference," for details. ♦

## Starting NetLook

To start NetLook, double-click the *netlook* icon in the *netvis* directory view or enter:

```
netlook
```

The first window you see is a NetLook Progress window indicating that NetLook is starting or that your NetVisualyzer license expires within 30 days. You may also see a NetLook Warning window with the message: `Could not open ~/network.data: No such file or directory.` This message lets you know that NetLook could not find network configuration information saved from a previous NetLook session. Click *Continue* or press **<Enter>** to make these windows disappear.

Figure 3-1 shows the appearance of the NetLook main window when NetLook cannot find saved network data and user interface configuration information. In this case, NetLook does not begin snooping and displaying network configuration and traffic information until you use the Snoop control panel to start snooping. Using the Snoop control panel is explained in “Snoop Control Panel” in this chapter.

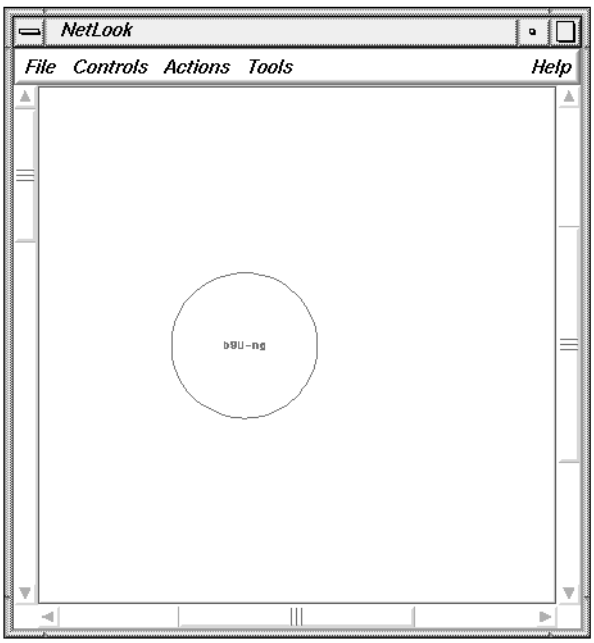



**Figure 3-1** NetLook Main Window at Startup (No Saved Configuration Information)

If NetLook finds one or both of the configuration files *~/network.data* (network data) and *~/.netlookrc* (user interface), it uses them to draw the network, set up the window configuration, and adjust the settings of the control panels. Figure 3-2 shows how the NetLook main window might appear.

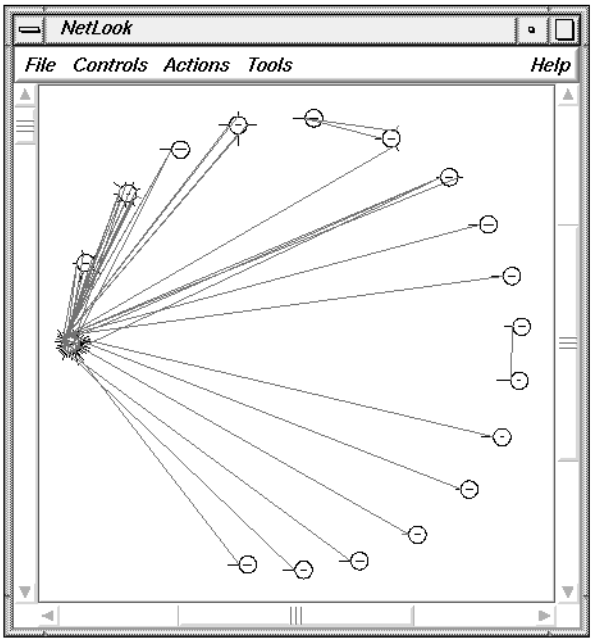



**Figure 3-2** NetLook Main Window at Startup (Saved Configuration Information)

Network data files, typically called *network.data*, describe the network configuration of network segments and nodes. NetLook uses this configuration to set up the NetLook main window. Each network segment defined in the *network.data* file is shown in the main window as a separate circle. The name of the network segment given in the *network.data* file is shown as the name of the network circle. Each interface defined in the *network.data* file is shown as a node on a network circle. For more information about network data configurations files, see "NetLook Network Data File" in Appendix D.

NetLook looks for *~/network.data* and *~/.netlookrc* by default; you can override these locations with the **–f** and **–u** command-line options or by specifying alternate files in a NetLook resources file; see the *netlook*(1M) manual page in Appendix F for more information.

If you start NetLook without a *network.data* file, NetLook must learn the network configuration. A saved network data file, however, tells NetLook about the networks you want to observe so that they are displayed at startup.

If NetLook reads a user interface file that was created while snooping was in progress, snooping starts automatically. See "NetLook User Interface Configuration File" in Appendix D for more information about user interface configuration files.

## NetLook Main Window

The NetLook main window shows a detailed view of the network segments known to NetLook. Each network segment is represented by a circle whose size is relative to the number of nodes on the network. Nodes (workstations, routers, bridges, and hubs) appear around the perimeter of each network circle. You can display a node by its name or address (see "NetNode Control Panel" in this chapter).

The ring of network segment circles is updated as new nodes and network segments are discovered. Comparing the network ring to a clock, new nodes are added just before 9 o'clock, and the circle is adjusted so that nodes are equidistant from each other.

Network traffic is displayed in the NetLook main window by color-coded straight lines that appear between nodes. The color at each end of a line indicates the amount of traffic generated at that node that is destined for the node at the other end of the line. You can display traffic between nodes using source and destination routing (lines are shown from source to destination) or physical routing (lines show the "local hops," the physical path traffic takes through gateways from source to destination). See "Traffic Control Panel" in this chapter for more information.

## The Use of Color in the NetLook Main Window

Network segments active with traffic are displayed in light blue (cyan), and inactive network segments are displayed in dark blue. Network segments that have not experienced any activity while NetLook is running are not known to NetLook and are not displayed unless they are included in a *network.data* network configuration file that has been read.

Most nodes appear in green. Routers (gateways) are cyan. Display Stations and Data Stations with snooping turned on are magenta. NIS masters are peach, and NIS slaves are white. A user-selected node appears in yellow, regardless of its function. Table 3-1 summarizes these NetLook colors. The colors are listed in order of decreasing precedence, meaning that if a node fits in two or more categories, its color is the color of the first category in the list.

**Table 3-1** NetLook Colors

| Color | Representation |
|---|---|
| Light blue circle | Networks active with traffic |
| Dark blue circle | Networks with no traffic |
| Yellow characters | Node that has been selected |
| Yellow characters | Node that has been adjusted |
| Magenta characters | Node running snoop process |
| Cyan characters | Gateway |
| Peach characters | NIS master |
| White characters | NIS slave |
| Green characters | Node |

Traffic between nodes is measured in packets or bytes, as configured in the Traffic control panel. The amount of traffic is represented by color-coded lines drawn between communicating nodes. The colors are updated every 5 seconds by default, based on activity during the previous 5 seconds.

The range of possible color values is a range of color map values. Workstations with at least 24 bitplanes use the color range 144 (dark purple)

to 151 (light green) by default. Workstations with 8 bitplanes use the color range 8 through 15. You can use *showmap*(6D) to see what colors these numbers represent (see the section "Show Color Map" in the *IRIS Utilities Guide* for more information).

Each color step represents a certain number of packets or bytes. You can choose the number of packets or bytes represented by each color. By default, each change (step) in color represents one packet per second in packet mode, or one kilobyte per second in byte mode.

The volume of traffic originating at a node determines the color at the end of a line connecting that node with another. Thus the colors at each end of a line tell you about the volume of traffic in each direction between those two nodes. The colors in the middle of a line are an interpolation of the color map colors in the range from one end point to the other.

For example, suppose the color map range 58 (dark green) to 63 (bright green) on a 24-bitplane workstation is being used and each color represents 10 packets/second. Figure 3-3 shows the relationship between colors and number of packets per second. NetLook is snooping on a network segment with nodes `A` and `B` and finds that for a 5-second interval, node `A` has sent an average of 5 packets per second to node `B`, and node `B` has sent 45 packets per second to node `A`. Since node `A` generates packets in the range of 0 to 10, color 58 is used at node `A`, and since node `B` generates packets in the range 40 to 50, color 62 is used at node `B`. Since the range from color 58 to color 62 is five colors, the first 20 percent of the line is color 58, the second 20 percent is color 59, and so on. Because of the automatic Gouraud shading done by many Silicon Graphics workstations, the color transitions may appear gradual rather than discreet.

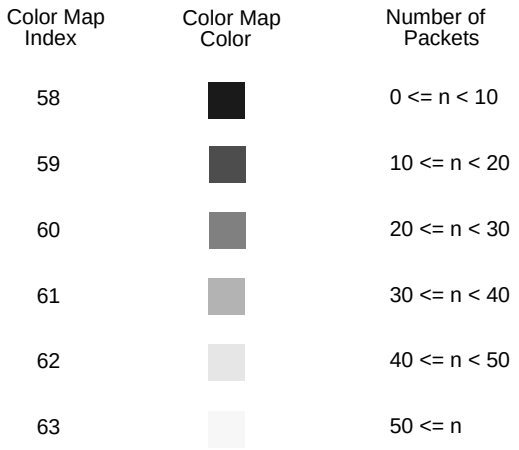



**Figure 3-3** Traffic Line Colors and the Color Map

Additional information about configuring traffic lines is available in "Traffic Control Panel" in this chapter.

## Adjusting the Viewing Area with the Scroll Bars and Mouse

You can change the portion of the network ring seen in the NetLook main window in both area and scale using the window's scroll bars. Movement is restricted to the ring of network segments known to NetLook.

Left scroll bar – zoom
: Use the left scroll bar to change the scale of the area displayed. By moving the bar down, you can zoom in on a node, individual network segment, or other area.

Bottom scroll bar – shift view left and right
: The bottom scroll bar moves the view of the map left or right. Moving the scroll bar to the left shifts the view to the left, making objects appear to move to the right across the window.

Right scroll bar – shift view up and down
: The right scroll bar moves the view up and down within the window.

In addition, you can use three other methods to change the view area:

- Drag the mouse. Press the middle mouse button in the window and drag the hand cursor until the view you desire is displayed; the view will follow the cursor. When you are satisfied with the view, release the middle mouse button.
- Drag out a new viewing area. Move the cursor to a corner of the new viewing area you want. Hold the **<Alt>** key, press the middle mouse button, and move the hand cursor to the opposite corner of the viewing area you want. Release the **<Alt>** key and the middle mouse button.
- Use the Map control panel. Additional methods of adjusting the viewing area and scale using the Map control panel are described in "Map Control Panel" in this chapter.

## Selecting Nodes and Network Segments in the Main Window

You can select a node or network segment by single-clicking the left mouse button on a node name for a node or in a network segment circle. The selected node, network, or network segment changes to yellow, regardless of its role in the network. If a dialog box is displayed when you select a choice on the Actions menu (except for "Find..."), the name of the selected node or network is automatically filled in for you.

## Rearranging Network Order

You can change the order in which nodes appear on a network ring. By default, NetLook arranges nodes in the order in which they are discovered. To rearrange nodes, move the cursor to the node you want to move, press the left mouse button, and drag the node to the position you want it to appear on the ring.

## NetLook Control Panels


**Figure 3-4**
Controls Menu

NetLook provides five control panels for use in controlling snooping and configuring the display of nodes and network segments in the main window. Figure 3-4 shows the Controls menu.

To bring up a control panel, select it from the Controls menu. The control panel is automatically placed at the location specified in a *.netlookrc* file if one is being used, or in a default location otherwise.

The control panels are discussed in the following sections.

### Snoop Control Panel

Use the Snoop control panel to start or stop monitoring network traffic on a Data Station you specify. Monitoring network traffic on a Data Station (snooping) enables you to display traffic within that Data Station's network segment. To get the maximum amount of information about traffic in your network, turn on snooping on one Data Station in each segment of your network.

Using the Snoop control panel, you can specify a filter if you want to restrict the packets shown by NetLook to a subset of interest to you. An example of the Snoop control panel with snooping turned on for one Data Station is shown in Figure 3-5.

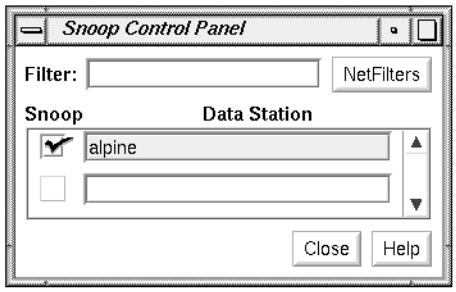



**Figure 3-5** Snoop Control Panel

You must have authorization to snoop on a Data Station. See "Authorizing NetVisualyzer Users for Snooping" in Chapter 1 and Appendix B, "Authorization Reference," for more information.

The sections below describe how to use the Snoop control panel to start and stop snooping and to specify a NetLook filter.

### Starting Snooping on a Data Station

To turn snooping on, enter the name or IP address of the Data Station you want to snoop on in a Data Station entry field if the Data Station isn't in the list, and click the left mouse button on the check box to check it (see Figure 3-5).

If a Data Station you snoop on has multiple interfaces, NetLook assumes that you want to snoop on the interface that matches the name (or IP address) you specified. To specify an interface, use its name as shown in the Address column of **`netstat -i`** output. For example, the output of **`netstat -i`** is:

```
Name Mtu   Network     Address           Ipkts Ierrs    Opkts Oerrs  Coll
ipg0 4352  wpd-fddi    squaw.wpd.sgi.c  873875    18   485607     1     0
ec0* 1500  b9U-ng      gate-squaw.wpd.       0     0     4807     0     0
lo0  32880 loopback    localhost         14656     0    14656     0     0
```

You want to snoop on the Ethernet interface, ec0, so use **`gate-squaw`** in the Data Station entry field. If you specify **`squaw`**, snooping is done on the FDDI interface, ipg0.

You should snoop on only one Data Station per network segment; snooping on other Data Stations on the same network segment will not provide NetLook with any additional information.

As it begins to collect data, NetLook draws traffic patterns and shows network transactions in the NetLook main window. Figure 3-6 shows an example of the NetLook main window after snooping has started. See "The Use of Color in the NetLook Main Window" and "Traffic Control Panel" in this chapter for more information about the colors used for the traffic lines.

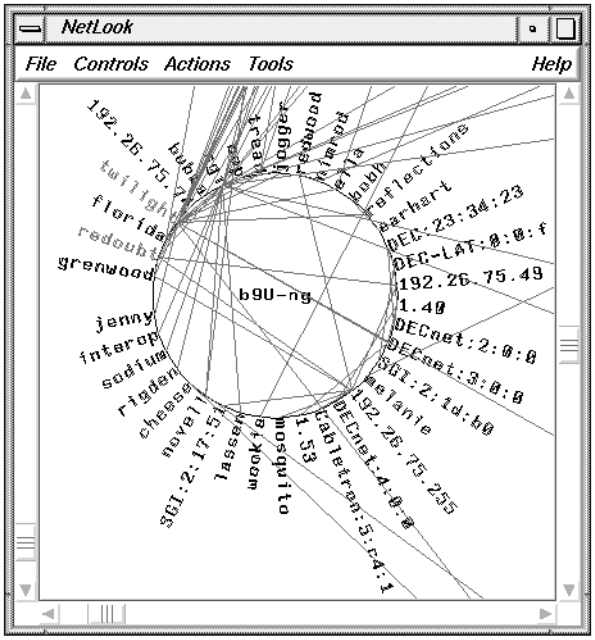



**Figure 3-6** NetLook Main Window While Snooping

To get information on the physical routing of packets (as opposed to endpoint-to-endpoint information), you must snoop on each network segment that packets travel through. (See "Seeing the Physical Path of Traffic Between Two Nodes" in this chapter for more information on observing the physical routing of packets.)

For more information on snooping remotely, see "Monitoring Traffic on Other Network Segments" in this chapter.

### Stopping Snooping on a Data Station

To stop snooping on a Data Station, click the "Snoop" check box to the left of the Data Station name or address. The check disappears and snooping on that Data Station is terminated.

### Specifying a Filter

To specify a filter to NetLook, enter the filter in the Filter entry field and press **<Enter>** to make it take effect. You can either type in the filter or use the *NetFilters* button to invoke NetFilters. When you select a filter in a NetFilters archive, it is automatically copied to the Filter entry field of the Snoop control panel. See Chapter 10, “Creating and Using Filters,” for information on constructing and using filters and Chapter 2, “NetFilters,” for more information about NetFilters.

## Map Control Panel

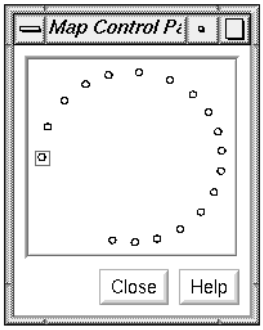



**Figure 3-7**
Map Control Panel

The Map control panel displays an overview of the network known to NetLook. It shows the entire network ring. (The NetLook main window may show just a portion of the network ring.) A yellow rectangle shows the current position of the view shown on the main window. Figure 3-7 shows the Map control panel for the NetLook main window as it appears in Figure 3-6.

You can change the position of the network currently shown in the main window by manipulating the yellow rectangle in the Map control panel. To do so, move the arrow cursor to another position in the window and press the middle mouse button. The arrow cursor changes to a hand. When you drag the hand cursor, the yellow rectangle follows, and the NetLook main window is updated to reflect the position of the yellow rectangle. Release the mouse button when you are satisfied with the location of the rectangle.

You can also resize the yellow rectangle to change the view and scale of the network shown in the main window. To do this, first press and hold the **<Alt>** key. Move the arrow cursor to one corner of the new yellow rectangle you want. Press the middle mouse button and drag the hand cursor to the opposite corner. Release the middle mouse button and the **<Alt>** key. The view in the NetLook main window changes to correspond to new size and location of the yellow rectangle.

## NetNode Control Panel

The NetNode control panel controls how network segments and nodes are labeled and some aspects of the display of network segments and nodes. Figure 3-8 shows the default NetNode control panel.

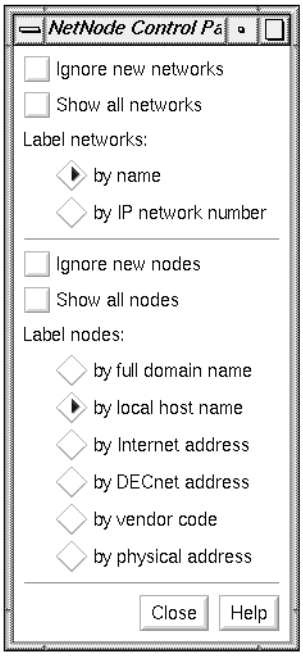



**Figure 3-8**
NetNode Control Panel

If NetLook receives a packet from a network segment it does not know, it either adds the network segment to the network map or ignores it. If the "Ignore new networks" check box shown in Figure 3-9 is not checked (the default), the new network segment is added; if the check box is checked, all new network segments are ignored. These new network segments are not written to the network data file *network.data*. To turn off the display of network segments already discovered, see "Hide Control Panel" in this chapter.


**Figure 3-9** Ignore New Networks Check Box

If the "Show all networks" check box shown in Figure 3-10 is checked, all network segments are displayed as though they are active (they appear in light blue). If the check box is not checked, only network segments with active connections appear in light blue. Inactive network segments appear in dark blue. By default, this check box is not checked.

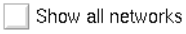



**Figure 3-10** Show All Networks Check Box

The Label networks radio buttons shown in Figure 3-11 control how network segments are labeled:

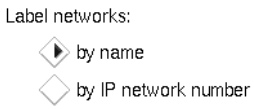



**Figure 3-11**
Label Networks Radio Buttons

by name — Label networks by the name found in the network database file (for example, */etc/networks*, a *network.data* file, the map that serves the networks, NIS, or DNS). The network database files used are controlled by the **–y** command-line option and the `useyp` and `hostresorder` resources (see "Address/Name Resolution" in Chapter 1 for more information). Label networks by name is the default. If no name can be found, IP network numbers are used.

by IP network number
: Label networks by IP address. If you are not using IP, networks are labeled by name even if this button is selected.

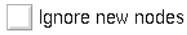



**Figure 3-12**
Ignore New Nodes Check Box

If the "Ignore new nodes" check box shown in Figure 3-12 is checked, NetLook ignores new nodes not previously known to it. New nodes become known to NetLook when there is traffic to or from them. Checking this check box lets you concentrate on the nodes already discovered. If it is checked, information about new nodes is not written to the network data file Side*network.data*.

If the "Ignore new nodes" check box is not checked, new nodes are displayed as they become known to NetLook. By default, this check box is not checked.

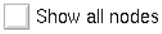



**Figure 3-13**
Show New Nodes Check Box

If the "Show all nodes" check box shown in Figure 3-13 is checked, all nodes in active networks known to NetLook are displayed, whether or not there is currently traffic to display at that node. If the check box is not checked, only nodes that meet the criteria of the Traffic control panel are shown. By default, this check box is not checked.

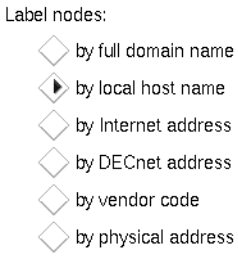



**Figure 3-14**
Label Nodes Radio Buttons

The Label nodes radio buttons shown in Figure 3-14 control how nodes are labeled

by full domain name
: Display nodes by domain name, if known, for example, `alpine.eng.sgi.com`.

by local host name
: Display nodes by node name, if known, for example, `alpine`. The default for labeling nodes is by local host name.

by Internet address
: Display nodes by IP address, for example, `192.26.61.143`.

by DECnet address
: Display nodes by DECnet node address, for example, `1.323`.

by vendor code
: Display the certified vendor code translated from the first 3 bytes of a physical address, for example, `SGI:2:29:d4`. This code is assigned by the IEEE Standards Office (see "References" in the Introduction for the IEEE address).

by physical address
: Display nodes by physical (Ethernet) address, for example, `8:0:69:2:29:d4`.

When a new node is discovered, its name or address is displayed as requested by the selected radio button. If the requested name or address cannot be found, the labeling specified by the next item in the radio button list is tried, and so on down the list (wrapping to the top of the list if necessary) until a name or address is found. Depending upon whether the **–y** option was given and the values of the `useyp` and `hostresorder` resources, NetLook uses */etc/hosts*, NIS, and BIND to search for names and IP addresses. See "Address/Name Resolution" in Chapter 1 for more information.

## Traffic Control Panel

The Traffic control panel determines how network traffic is displayed in the NetLook main window. Figure 3-15 shows the default Traffic control panel. The remainder of this section describes each section of this control panel.

The For traffic radio buttons shown in Figure 3-16 control whether endpoint routing or physical (gateway) routing of packets is shown:

show the source and destination
: Display the source and destination of traffic (endpoint routing). Traffic from point A to point B is shown as a straight line and does not show how the packet was physically routed. This is the default.

show the local hop
: Display the path of how the traffic was physically routed (the traffic "hops" or gateway routing). For this display to work from source to destination, you must be snooping on each network segment that the traffic passes through, and you must set up the NetLook *network.data* configuration file to show traffic on routers or gateways (see "Showing Gateway Nodes" in this chapter).

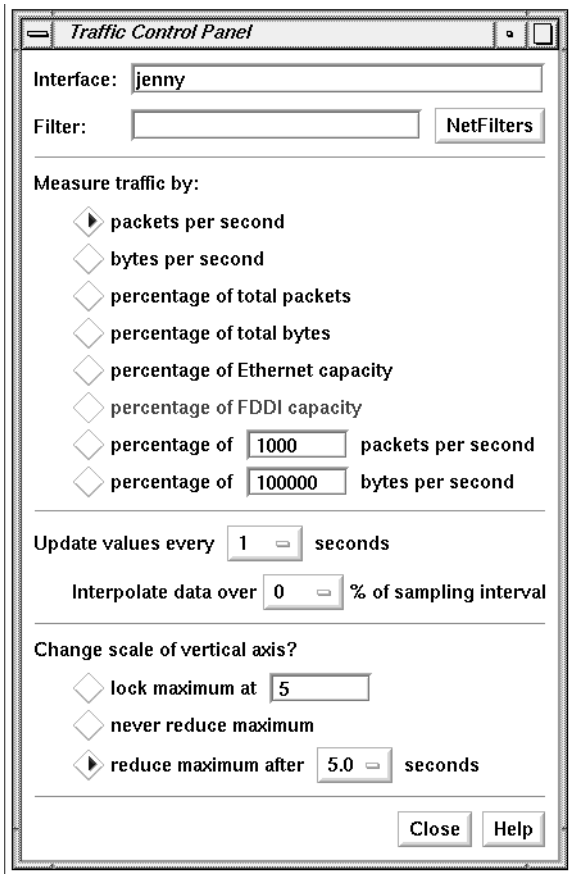



**Figure 3-15** Traffic Control Panel

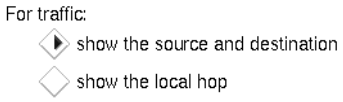



**Figure 3-16** For Traffic Radio Buttons

The Base scale of traffic radio buttons shown in Figure 3-17 enables you to specify whether NetLook calculates the color of a line based on the number of packets or the number of bytes passing between nodes:

| | |
|---|---|
| packets | Scale traffic based on packet count. This is the default. |
| bytes | Scale traffic based on byte count. |

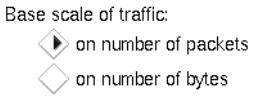



**Figure 3-17** Base Scale of Traffic Radio Buttons

Figure 3-18 shows the "Each color step represents" and "Traffic volume color map indices" portions of the Traffic control panel.

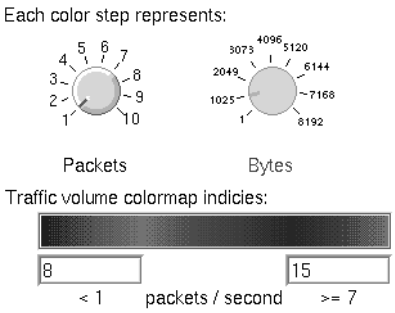



**Figure 3-18** Each Color Step Dials and Traffic Volume Entry Fields

The colors used for traffic lines are taken from the color map, each color step being one entry in the color map. (See *showmap*(6D) for information on displaying the color map.) The range of color map entries is specified by the two traffic-volume entry fields. The color map entries in this range are shown in the color bar. They are shown dithered, since the traffic lines in the NetLook main window are dithered.

The dials are used to specify how many packets or bytes (depending upon the scale of traffic) each color map entry represents. The line below the entry fields shows the number of packets or bytes per second that will be indicated by the end color map indices. In the example above, the end indices are 8 and

15, a range of 8 color map entries. Since each color step represents 1 packet per second, color map entry 8 represents 0 packets per second and color map entry 15 represents 7 or more packets per second.

To adjust a dial, click the left mouse button inside the dial circle at the number you want, or press the left mouse button inside the circle and move the cursor so that the marker spins to the number you want.

To change color map indices, replace the current number with the number you want and press **<Enter>**.

The default color step settings are 1 packet/second or 1024 bytes/second. For 8-bitplane workstations, default color map indices are 8 and 15; for workstations with more than 8 bitplanes, the default indices are 144 and 151.

See "The Use of Color in the NetLook Main Window" in this chapter for more information about traffic line colors.

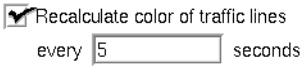



**Figure 3-19**
Recalculate Color Check Box and Entry Field

The "Recalculate color of traffic lines" check box and its entry field are shown in Figure 3-19. If this box is checked, NetLook periodically adjusts the color of the traffic displayed, based on the volume of traffic over the last recalculation interval. By default, this box is checked and the recalculation is done every 5 seconds. To change the value, edit the number and press **<Enter>**.

If the volume of traffic is greater than the previous volume, the color is rescaled to reflect the new, higher volume. If the volume is smaller than the previously displayed volume, the color is adjusted downward by only one color step. The result is that NetLook always shows high-volume traffic and smoothes intermittent drops in traffic.

You can use scaling to distinguish real-time traffic flow from long-term patterns. When set low (2 or 3 seconds), the traffic display is updated frequently to allow monitoring of real-time traffic. When set high (10 or 15 seconds), traffic patterns accumulate to show traffic on a long-term basis.

If this check box is not checked, the traffic lines are never rescaled down, only up, and the main window shows every connection's highest-ever volume.


**Figure 3-20** Delete Traffic Line Check Box and Entry Field

The "Delete traffic lines" check box and entry field are shown in Figure 3-20. This line controls the amount of time a traffic line between two nodes remains on the screen after the last packet is seen. If it is checked, traffic lines are removed after the time-out period specified in the entry field. By default, the check box is checked and the time-out period is set to 60 seconds.

To change the value, edit the number and press **<Enter>**. A short time-out period (for example, 15 seconds) monitors real-time traffic. A longer time-out period (for example, 180 seconds) shows traffic patterns on a long-term basis.

If the check box is not checked, traffic lines remain on the screen indefinitely. This can be used to monitor all connections to the network over a day or a weekend. Unexpected traffic from unexpected sources can be easily detected.

## Hide Control Panel

The Hide control panel is used to specify network segments and nodes that you don't want displayed. An example is shown in Figure 3-21. When you want to see these network segments or nodes, you can "unhide" them with this control panel.

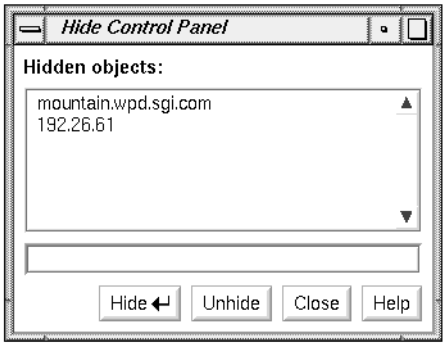



**Figure 3-21** Hide Control Panel

In the entry field on the Hide control panel, enter the name or address of the segment or node that you want to hide, and click the *Hide* button or press

**<Enter>**. The existing traffic lines and nodes are removed from the NetLook main window. The display of nodes and traffic lines resumes, but does not include the network segment or node you specified. The name or address is put in the Hidden objects list in the control panel.

To display a hidden network segment or node, click on its name or address in the Hidden objects list, then click the *Unhide* button. The network segment or node reappears in the NetLook main window right away, and it disappears from the Hidden objects list.

Hiding an object differs from using "Delete..." on the Actions menu in that NetLook "forgets" an object when you delete it (frees all memory associated with the object), but does not forget it when you hide it. When you hide it, you do not see the object again until you explicitly unhide it. When you delete an object, it is redisplayed only if NetLook discovers it again. When you hide an object, information about that object is written to *network.data*. When you delete an object and it is not re-discovered, that object is omitted from any *network.data* file that you save.

## NetLook Actions


**Figure 3-22** Actions Menu

The Actions menu provides you with a variety of choices that are described below. Figure 3-33 shows the basic Actions menu. If Spectrum® software is installed on your Display Station, an additional "Spectrum" choice appears.

Many of the choices on the Actions menu bring up a Prompt dialog box and ask you to supply the name or address of a network segment or node. To use a shortcut, select the network segment or node that you are going to supply to the dialog box before selecting from the Actions menu. The name or address of that network segment or node automatically appears in the entry field of the Prompt dialog box that appears.

### Information

When you choose "Information" from the Actions menu and no objects are currently selected, a window appears that displays the number of network segments and nodes that are currently known to NetLook. An example of this NetLook Information window is shown in Figure 3-23.

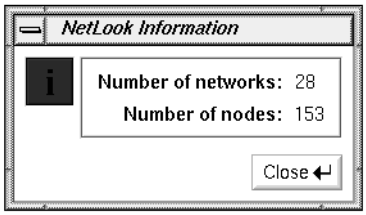



**Figure 3-23** General Information Window

If a node is selected when you choose “Information,” a window like the one shown in Figure 3-24 appears.

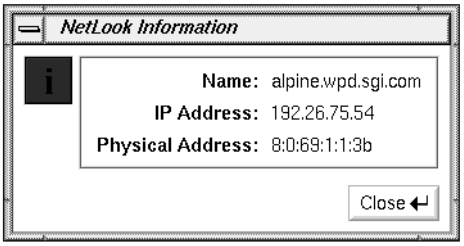



**Figure 3-24** Node Information Window

If a network segment is selected when you choose “Information,” a window like the one shown in Figure 3-25 appears.

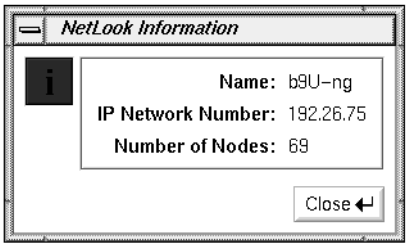



**Figure 3-25** Network Segment Information Window

If a traffic line is selected when you choose “Information,” a window like the one shown in Figure 3-26 appears. When NetLook is displaying a small

portion of the network and a traffic line goes off the screen, you can easily trace the connection by selecting the line and choosing “Information” from the Actions menu.

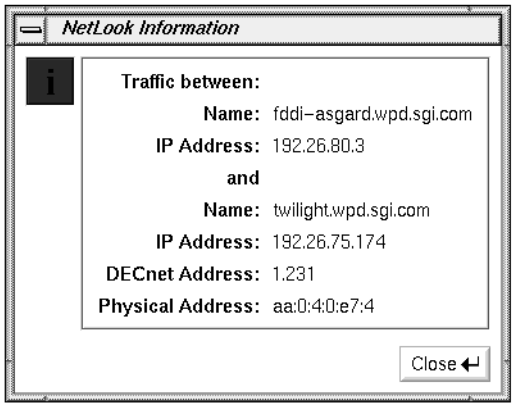



**Figure 3-26** Traffic Line Information Window

Like most NetLook menu choices, “Information” has a keyboard accelerator, **<Alt-i>**, that can be used instead of selecting “Information” from the Actions menu.

## Find

The “Find...” action places a particular node or network segment in the center of the NetLook main window. It is a convenient way to locate a particular node or network segment when the network ring in the main window is large and complex.

Click “Find...” on the Actions menu or use the keyboard accelerator **<Alt-f>** and the window shown in Figure 3-27 appears. Type the name or address of the node or network segment you wish to find. Press **<Enter>** or click the *OK* button. The node or network segment is placed in the center of the viewing window and displayed in yellow, regardless of its role in the network.

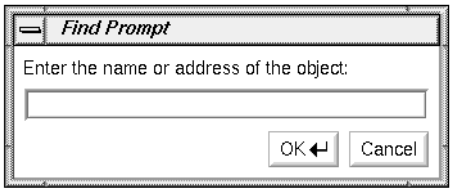



**Figure 3-27** Find Prompt Dialog Box

## Ping

The “Ping...” action sends a request for a response to a node. This interface to the *ping*(1M) command is an easy way to generate traffic that you can observe in the NetLook main window.

You can see the network traffic generated by “Ping...” in the NetLook main window only if ICMP protocol traffic is displayed. This is the default; make sure that any filter you have specified allows `icmp` traffic to be displayed. Also, in the Traffic control panel, if you choose the “show the source and destination” radio button, it is easier to recognize the traffic you have generated in the NetLook main window.

Select “Ping...” on the Actions menu or use the keyboard accelerator **`<Alt-p>`** and the window shown in Figure 3-28 appears. Type the name or address of the node you wish to ping into the entry field, and press **`<Enter>`** or click the *OK* button. The default *ping* command is `ping -R`. The command that is run is set with a resource that you can change.

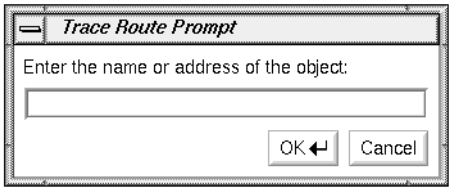



**Figure 3-28** Ping Prompt Dialog Box

A window appears and shows the response of the node (*ping* output). Figure 3-29 shows an example.

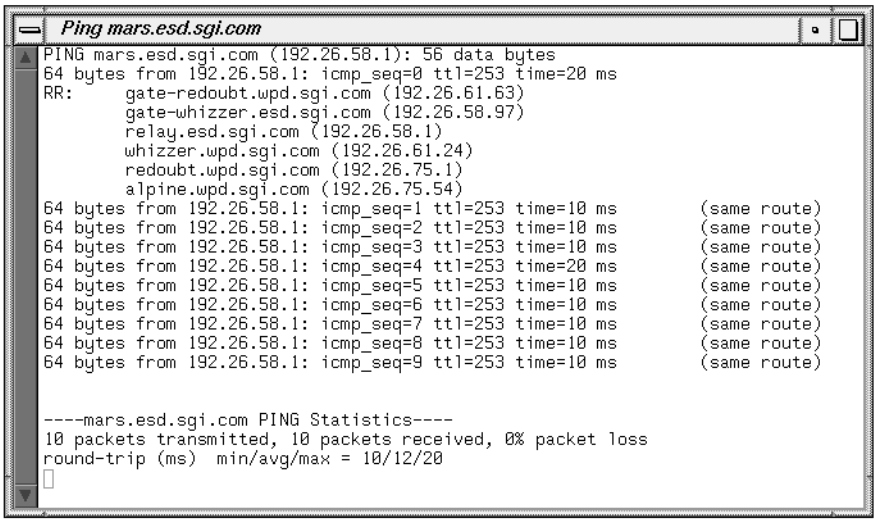



**Figure 3-29** Ping Output Window

The *ping* process continues until you stop it by moving the cursor into the Ping window and pressing **<Ctrl-c>**. To close the Ping window, choose "Quit" on the window's window menu or double-click the *Window menu* button in the upper left corner of the Ping output window.

For more information, see the *ping*(1M) manual page.

## Trace Route

The "Trace Route..." action traces the route taken by a packet. It displays a list of the gateways that a packet travels through to get to a node that you specify. "Trace Route..." is an interface for *traceroute*(1M). To use "Trace Route...", the subsystem *eoe2.sw.ipgate* must be installed, and you must have started NetLook as superuser (*root*).

When you select "Trace Route..." on the Actions menu or use the keyboard accelerator **<Alt-t>**, the window shown in Figure 3-30 appears; type the

name or address of the node to which you want to send a packet, and press **<Enter>** or click the *OK* button. `traceroute` is the default command, but it can be changed using a resource.

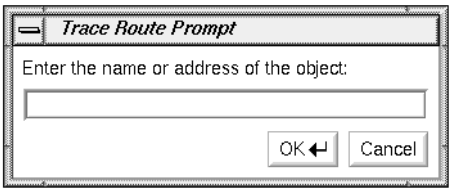



**Figure 3-30** Trace Prompt Dialog Box

A Trace Route output window appears and shows the response of the node (*traceroute* output). An example is shown in Figure 3-31.

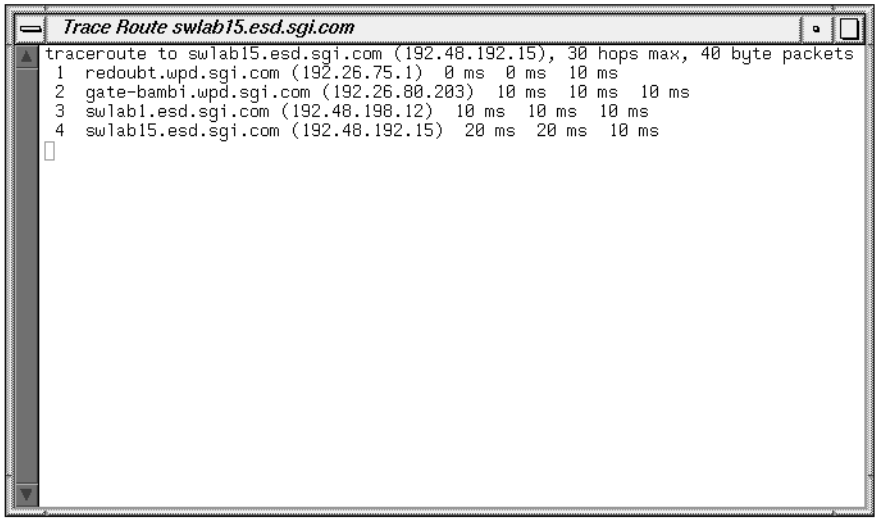



**Figure 3-31** Trace Route Output Window

To close the Trace Route window, choose “Quit” on the window menu or double-click the *Window menu* button in the upper left corner of the Trace Route window.

For more information, see the *traceroute*(1M) manual page in Appendix F.

## Home

When you choose “Home” from the Actions menu or press the **<Home>** key, the NetLook main window is redrawn to show the entire network.

## Delete

“Delete...” on the Actions menu enables you to delete a network segment or node from the NetLook main window. Thus you can display only the network segments and nodes you want to study. If NetLook rediscovers a network segment or node that you have deleted (because it detects traffic to or from that object), the network segment or node reappears.

When you choose “Delete...” from the Actions menu or use the keyboard accelerator **<Alt-d>**, the window shown in Figure 3-32 appears.

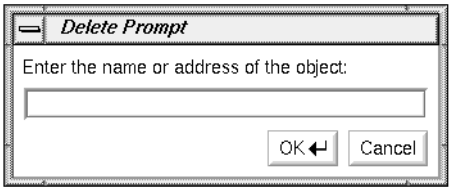



**Figure 3-32** Delete Prompt Dialog Box

Enter the name or address of the network segment or node you want to delete. If you select the object before choosing “Delete...”, the name or address is automatically placed in the entry field. Press **<Enter>** or click the *OK* button to delete the network segment or node.

If you save the configuration when you quit NetLook, network segments and nodes you have deleted will not appear in the main window the next time you start NetLook, because the deleted network segments and nodes were not saved.

If you have a large network and want to delete many nodes, it may be faster to edit the *network.data* file (see "Monitoring Selected Nodes" in this chapter for details).

"Delete..." differs from the Hide control panel in that NetLook forgets all information about a deleted network segment or node, but does not forget about a hidden network segment or node. Memory is freed when a network segment or node is forgotten, which can be useful if you have a large network.

"Ignore new networks" and "Ignore new nodes" in the NetNode control panel are similar to "Delete...". Use those check boxes to automatically delete network segments and nodes that are discovered in the future.

## Delete All

"Delete All" or the keyboard accelerator **<Alt-a>** clears all network segments and nodes from the display except your network segment and terminates all snooping. All network segments and nodes that were known to NetLook are forgotten, just as if you had used the "Delete..." action for each one. To restart snooping, use the Snoop control panel as described in "Snoop Control Panel" in this chapter. Once you restart snooping, network segments and nodes appear as they are discovered.

"Delete All" provides a convenient way to rid your display of network segments and nodes that no longer exist and to restart NetLook with a "clean slate."

## Spectrum

When you select "Spectrum" from the Actions menu, NetLook sends the selected network segment or node (if any) to Spectrum. If the Spectrum user interface, SpectroGRAPH, is running, it opens a window with the Spectrum view of that network segment or node. If SpectroGRAPH is not running or the selected object is unknown to Spectrum, nothing happens.

The menu choice "Spectrum" appears on the Actions menu only if Spectrum software is installed.

## NetLook File Menu


**Figure 3-33** File Menu

The choices on the File menu enable you to open NetLook network configuration files, save the current network data and user interface configuration to files, and quit. Figure 3-33 shows the File menu.

### Open

"Open..." opens a previously saved network data file.

Click "Open..." on the File menu and a file prompter window appears. Use the procedure in the section "Using a File Prompter" in the Introduction to specify the file name from which you want to read network configuration data. The file is then read, and all snooping stops. To restart snooping according to the new configuration, use the Snoop control panel as described in "Snoop Control Panel" in this chapter.

See "NetLook Network Data File" in Appendix D for more information about network data files.

### Save Networks

The "Save Networks" choice enables you to save network configuration information. This choice has a rollover menu with two choices, "Save" and "Save As...". These choices save network configuration information to the file *~/network.data* or to a file name of your choice, respectively. When you choose "Save As...", a file prompter window appears. Use the procedure in the section "Using a File Prompter" in the Introduction to specify the file name for the network configuration data.

By default, the file *~/network.data* is read when you start NetLook; you can specify a different network data file at startup with the **–f** option or read a network data file at any time using "Open..." on the File menu.

See "NetLook Network Data File" in Appendix D for more information.

## Save Controls

"Save Controls" enables you to save NetLook user interface configuration information. This choice has a rollover menu with two alternatives, "Save" and "Save As..." to save user interface configuration information to the file *~/.netlookrc* or to a file name of your choice, respectively. When you choose "Save As...", a file prompter window appears. Use the procedure in the section "Using a File Prompter" in the Introduction to specify the file name for the user interface configuration data.

By default, the file *~/.netlookrc* is read when you start NetLook; you can specify a different user interface configuration file at startup with the **–u** command-line option or the `NetLook*controlsFile` resource.

See "NetLook User Interface Configuration File" in Appendix D for more information.

## Quit

To exit NetLook, select "Quit" from the File menu. A NetLook Question window appears. To save the current network and user interface configuration in the files shown in the message and to quit NetLook, click the *Yes* button. To quit without saving configuration information, click the *No* button. If you want to write the information to other files or decide not to quit NetLook, click the *Cancel* button.

## NetLook Examples

This section provides a variety of examples and tips for using NetLook. The remaining sections in this chapter describe examples of using NetLook, editing the *network.data* file to provide additional information to NetLook, and customizing the display of network segments and nodes.

### Monitoring Protocols in a Multiprotocol Network

In some situations such as a multivendor, multiprotocol network, you may want to monitor only certain traffic types. To do this, use NetLook's protocol filters to isolate traffic by protocol.

For example, to optimize NFS client/server configuration in a multiprotocol network, examine only its NFS traffic. Just specify a filter of

```
nfs
```

and press **<Enter>** in the filter entry field of the Snoop control panel. This filtering process reduces the amount of information displayed for analysis and makes NFS traffic patterns easier to understand.

### Tuning Traffic Line Parameters

Because each network is unique, you may want to try different settings on the Traffic control panel to see which ones work best for you. To see greater differentiation in traffic, for example, recalculate the color of traffic lines every 5 seconds and color step to 1 packet/second or 1024 bytes/second. For less differentiation in traffic, set the number of packets or bytes that a color step represents to a higher number.

## Monitoring Traffic on Other Network Segments

Assume that a Display Station is connected to `net1`. As NetLook captures and displays network packets, traffic lines accumulate as shown in .

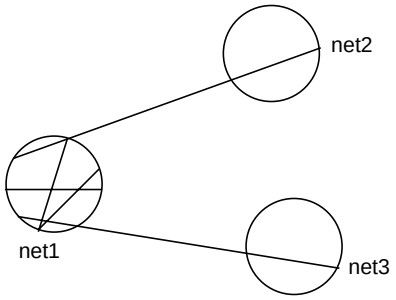



**Figure 3-34** NetLook Display with a Display Station on `net1`

The display shows only traffic packets that have passed through `net1`, where the Display Station is attached. In fact, `net1` is the only network segment with any internal traffic displayed. This, however, does not indicate that no internal traffic occurs in other network segments or that no traffic occurs between other network segments. In reality, it is likely that just as much traffic occurs within `net2` and `net3` while the Display Station is busily collecting traffic data in `net1`. NetLook cannot show the traffic within `net2` and `net3` because it is not snooping on those network segments. It can only capture and display packets internal to those network segments on which it is snooping.

To use NetLook to monitor additional network traffic from the central Display Station, install Data Station software on one workstation in every network segment. Then, from the Display Station, you can activate the remote data-collection mechanism by using the Snoop control panel to start snooping on each of these Data Stations. Doing so causes each Data Station to collect local traffic data in exactly the same way that the central Display Station collects traffic information in its own network segment. Each Data Station forwards the data to the central Display Station for simultaneous graphical display. The resulting NetLook main window will look like Figure 3-35.

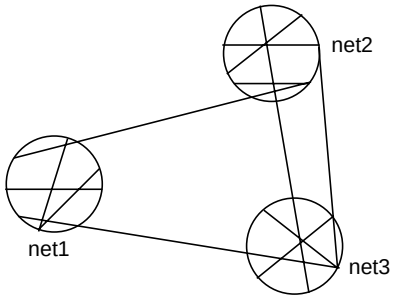



**Figure 3-35** NetLook Display with a Display Station on `net1` and Data Stations on `net2` and `net3`

NetLook's display of overall traffic distribution tells a great deal about whether the network's configuration is optimal. For example, when sources and destinations are shown (rather than local hops), if two nodes on two separate network segments consistently show an intense connection, move them to the same network segment. Leaving them on two different network segments exerts unnecessary load on the interconnecting router.

Also, if one network segment shows much more internal traffic than the others, you may be able to improve the network's overall response by moving some of that network segment's nodes to the other network segments and balancing the overall distribution of traffic loads.

## Understanding "Missing" Nodes

At times, NetLook's display of the network configuration may not be complete, even though it may appear to have stabilized. You may, for example, know that a particular node named `elm` is on another network segment and that `elm` is listed in */etc/hosts*; however, NetLook doesn't show it. Possible explanations include:

- `elm` is connected to the network but has not been communicating with other nodes while NetLook has been snooping.
- `elm` is no longer connected to the network. It may be powered down or disconnected from the network. To determine if `elm` is responding to the network, use "Ping..." on the Actions menu to ping node `elm`. Pinging a node causes 64-byte packets to be continuously sent to that

node until stopped with **<Ctrl-c>**. An unsuccessful ping with 100% packet loss means that `elm` is not receiving any of the packets and that it is no longer connected to the network.

- `elm` has been communicating with other nodes, but its traffic packets have never been routed to or through a network segment on which NetLook is snooping. NetLook captures packets only on network segments on which it is snooping. If `elm` communicates only with nodes in network segments on which there is no snooping, NetLook will not be able to capture and display the packets.
- `elm`'s protocol is not monitored. You may be using a filter that excludes the protocol that traffic to and from `elm` is using.

## Seeing the Physical Path of Traffic Between Two Nodes

With the NetNode control panel, you choose either "show the source and destination" (display source and destination routing) or "show the local hop" (display physical routing). This example explains the difference between these two different ways to display traffic and describes how to see the physical routing of packets that travel through gateways.

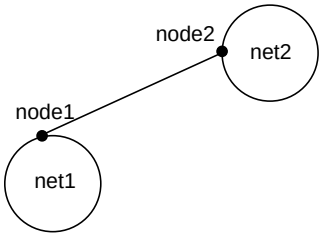



**Figure 3-36** Source and Destination Routing Display

Assume that you are snooping on two network segments, as shown in Figure 3-36, and your Display Station is in `net1`. The current traffic display setting is "show the source and destination." `node1` in `net1` copies a file to `node2` in `net2` using *rcp*(1). As shown in Figure 3-36, NetLook displays a connection between `node1` and `node2` to indicate that the two nodes are communicating with each other.

In source and destination routing display, the connection appears as a straight line regardless of how packets are physically routed. NetLook displays the logical connection based on a packet's source (`node1`) and destination (`node2`).

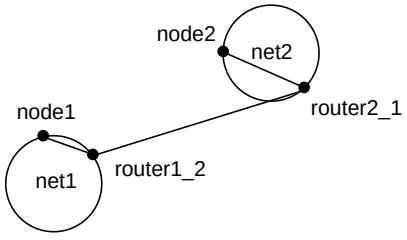



**Figure 3-37** Physical Routing Display with Gateway Nodes

To see the physical path that the remote copy traffic takes, do these things:

- Change the *network.data* file as described in "Showing Gateway Nodes" in this chapter so that NetLook can recognize that two or more interfaces are on single node. Do this for all nodes that could possibly serve as gateways between `net1` and `net2`.

- Verify that you are snooping on all network segments along all possible paths of traffic from `node1` to `node2`.
- Enter a filter that filters out all traffic except the traffic you want to see. For example, the filter `ip.between node1 node2` shows only traffic between `node1` and `node2`.
- On the Traffic control panel, select "show the local hop."

The remote copy between `node1` and `node2` now appears as shown in Figure 3-37. This display shows that the node `router` and its interfaces `router1_2` and `router2_1` are used as the physical path for packets between `node1` and `node2`.

Selecting "show the local hop" without editing the *network.data* file to specify which nodes are gateways is also useful. In this case, no end-to-end traffic is shown between nodes on different network segments. For example, Figure 3-38 shows the display for the *rcp* example above. You can see all of the traffic internal to each network segment.

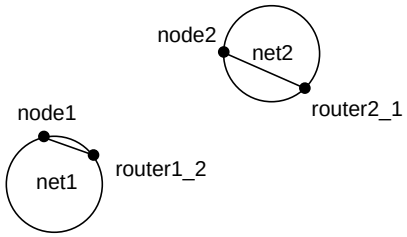



**Figure 3-38** Physical Routing Display without Gateway Nodes

## Using NetLook to Monitor Network Security Intrusions

Suppose you plan to leave for the weekend and want to monitor and record all nodes trying to access the node named `secret` during the weekend. Set up NetLook this way before you leave:

1. Capture only traffic going to or from `secret` by specifying a filter in the Snoop control panel. If `secret`'s address and name are mapped in */etc/ethers* (or NIS or BIND), use this form:

   **`host secret`**

   Or, you can use `secret`'s physical address instead:

   **`host 8:0:69:2:f:c1`**

2. Make sure that the "Delete traffic lines" check box in the Traffic control panel is not checked so that all collected data will appear on the screen indefinitely.
3. Click "Delete All" on the Actions menu to clear the NetLook main window.

4. Using the Snoop control panel, start snooping on a workstation on the same network segment as `secret`.

After the weekend, check the NetLook screen. No traffic lines indicate that no security breach occurred; no traffic packets have either entered or left `secret`. If there are traffic lines, a breach has occurred. The names of all nodes that communicated with `secret` during the weekend appear on the screen.

## Showing Gateway Nodes

A router or gateway is a node that has two or more interfaces, each to a different network. Routers forward packets between the networks to which they are connected. NetLook recognizes each of the interfaces, but does not know that they are on the same node; however, with your help, NetLook can display gateway or router traffic between the two interfaces.

To see this configuration represented in NetLook, you must edit the *network.data* file and create the association between the interfaces of the gateway.

For example, suppose two networks, `engineering-1` and `engineering-2`, have the gateway named `redoubt` between them. The *network.data* file saved from NetLook looks like this:

```
NetLook 1.10
Network engineering-1 {
        IPNet   192.26.75
        Segment engineering-1 {
                IPNet  192.26.75
                Node  {
                        Interface redoubt {
                                PhysAddr        8:0:69:2:4:45
                                IPAddr          192.26.75.1
                        }
                }
        }
}
Network engineering-2 {
        IPNet               192.26.61
        Segment engineering-2 {
                IPNet               192.26.61
```

```
                Node  {
                        Interface gate-redoubt {
                                PhysAddr
2:cf:1f:b0:0:16
                                IPAddr          192.26.61.1
                        }
                }
        }
}
```

The node `redoubt` has two node objects, and each node object describes one of `redoubt`'s interfaces. Create the association between the interfaces by editing the *network.data* file and changing the node objects for `redoubt`. Add the name `redoubt` to each node object definition, which makes both node statements describe the same node object.

The edited file looks like this:

```
NetLook 1.10
Network engineering-1 {
        IPNet   192.26.75
        Segment engineering-1 {
                IPNet   192.26.75
                Node redoubt {
                        Interface redoubt {
                                PhysAddr        8:0:69:2:4:45
                                IPAddr          192.26.75.1
                        }
                }
        }
}
Network engineering-2 {
        IPNet   192.26.61
        Segment engineering-2 {
                IPNet   192.26.61
                Node redoubt {
                        Interface gate-redoubt {
                                PhysAddr
2:cf:1f:b0:0:16
                                IPAddr          192.26.61.1
                        }
                }
        }
}
```

When you restart NetLook using physical (“local hop”) routing display and this *network.data* file, you will see a line that connects the two interfaces as traffic flows from one interface on `redoubt` to the other.

## Displaying Two Bridged Segments as Separate Segments

NetLook cannot determine if a network segment is made up of a single segment or multiple segments that are connected with a repeater or bridge. For example, suppose you have a network with two bridged segments and you want each segment to appear as a separate circle in the NetLook main window.

The original *network.data* file looks like this:

```
NetLook 1.10
Network engineering-1 {
        IPNet   192.26.75
        Segment engineering-1 {
                IPNet 192.26.75
                Node  {
                        Interface cheese {
                                PhysAddr        aa:0:4:0:e8:4
                                IPAddr          192.26.75.14
                                DNAddr          1.232
                        }
                }
                Node  {
                        Interface squaw {
                                PhysAddr        8:0:69:2:0:f9
                                IPAddr          192.26.75.11
                        }
                }
                Node  {
                        Interface kaibab {
                                PhysAddr        8:0:69:2:1:51
                                IPAddr          192.26.75.29
                        }
                }
                Node  {
                        Interface illyria {
                                PhysAddr        8:0:69:2:f:8c
                                IPAddr          192.26.75.12
                        }
```

```
                }
        }
}
```

Assume two nodes are on each segment, `cheese` and `kaibab` on a segment named `segment-1`, and `squaw` and `illyria` on a segment named `segment-2`. You must edit the *network.data* file to divide the network into its segments. First add another segment in the network `engineering-1` and then separate the nodes appropriately. Also add names for each of the segments.

After editing, the file looks like this:

```
NetLook 1.10
Network engineering-1 {
        IPNet   192.26.75
        Segment segment-1 {
                Node  {
                        Interface cheese {
                                PhysAddr        aa:0:4:0:e8:4
                                IPAddr          192.26.75.14
                                DNAddr          1.232
                        }
                }
                Node  {
                        Interface kaibab {
                                PhysAddr
8:0:69:2:1:51
                                IPAddr           192.26.75.29
                        }
                }
        }
        Segment segment-2 {
                Node  {
                        Interface squaw {
                                PhysAddr
8:0:69:2:0:f9
                                IPAddr           192.26.75.11
                        }
                }
                Node  {
                        Interface illyria {
                                PhysAddr
8:0:69:2:f:8c
```

```
                                    IPAddr          192.26.75.12
                            }
                    }
            }
}
```

The names `segment-1` and `segment-2` are added to the segment definitions. Nodes `cheese` and `kaibab` are listed under `segment-1` and nodes `squaw` and `illyria` are listed under `segment-2`. When you start NetLook using this file, two circles appear, each representing a segment. If the segments are connected by a "smart" bridge, however, you must have a Data Station on each segment and define the Data Station in the file in order to show traffic on each segment.

## Monitoring Selected Nodes

To monitor just a few nodes, comment out or remove the node description for those nodes that you do not want to appear on the network. To comment out a node, begin each line of the node description with a pound (#) character. Make sure each remaining node definition contains two opening and two closing braces, as shown in the previous examples. (See "Delete" and "Hide Control Panel" in this chapter for alternate ways to remove nodes from the network.)

Restart NetLook using the edited file (use the **–f** command-line option or use "Open" on the File menu). You must also stop nodes not previously known to NetLook from being added to the network. To do this, open the NetNode control panel and check the "Ignore new nodes" check box.

Chapter 4

# NetGraph

*NetGraph provides information about network traffic over time. This chapter explains how to configure NetGraph graphs, save them, and replay them. It also provides NetGraph examples.*

*Chapter 4*

# NetGraph

The previous chapter described how to use NetLook for an overall view of current network traffic. This chapter explains how to use NetGraph to monitor network traffic over a period of time. NetGraph's real-time strip charts delineate network usage and allow you to graph specific types of traffic such as the number of packets or the number of bytes per second. Filters can be used to restrict the graphs to the traffic of interest to you. An alarm mechanism warns when a graphed value exceeds a threshold you specify, and a history feature lets you play back previously recorded network data.

NetGraph's charts help you determine network overloading and traffic congestion. Use the information in the graphs when you want to expand or divide your network or determine where to place bridges, routers, and gateways on your network. For example, if NetLook displays intense traffic between two nodes, you can use NetGraph to graph how much traffic comes from or goes to each node, the percentage of total network traffic transmitted by each node, the times when traffic is the most intense, and so forth.

Developers of distributed applications can use NetGraph to look at the traffic rates between servers and clients and between multiple servers in applications that have them.

This chapter explains how to:

- start NetGraph
- use the NetGraph main window to display graphs of traffic volumes

- use the NetGraph control panels to specify the type of traffic you want to view and configure the display of traffic
- collect traffic information and play it back in NetGraph later

In addition, a variety of NetGraph examples are provided. For complete information on NetGraph command line options and resources, see the *netgraph*(1M) manual page in Appendix F, "NetVisualyzer Manual Pages." Additional information about the NetGraph configuration file is provided in Appendix D, "Configuration File Formats."

**Note:** You must have authorization to use NetGraph. See "Authorizing NetVisualyzer Users for Snooping" in Chapter 1 and Appendix B, "Authorization Reference," for details. ♦

## Starting NetGraph

To start NetGraph, double-click the *netgraph* icon in the *netvis* directory view or enter:

```
netgraph
```

The default NetGraph main window appears as shown in Figure 4-1.

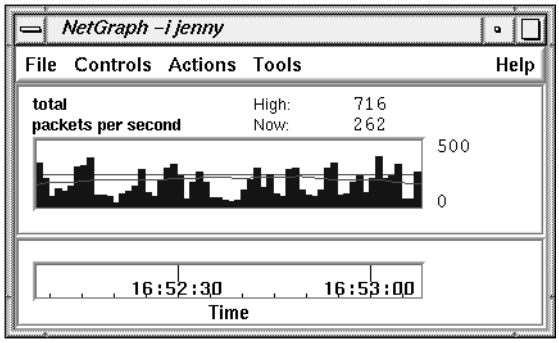



**Figure 4-1** Default NetGraph Main Window

When NetGraph starts, it uses a configuration file to determine the types of graphs to display. By default, NetGraph looks for *.netgraphrc* in your home

directory. If this file does not exist, NetGraph displays a Warning window and displays the default graph in the NetGraph main window.

You can specify a configuration file on the NetGraph command line with the **–u** option or in the NetGraph resources file with the `NetGraph*controlsFile` resource.

## NetGraph Main Window

In its default configuration, NetGraph shows a strip chart of total network traffic as measured in packets per second and displays a scrolling time legend below the graph that shows the time of day. You can add, delete, and edit graphs by using the NetGraph Edit menu.

For each protocol that NetGraph understands, it can produce a graph that shows one of the following:

- packets or bytes per second
- percentage of total packets or total bytes
- percentage of Ethernet or FDDI capacity
- percentage of a given number of packets or bytes per second

Figure 4-2 shows an example of a NetGraph main window with several graphs.

The NetGraph main window title bar includes "–i" and the interface it is snooping on. The interface is in the same format as the argument to the **–i** command line option.

Above each graph, two lines list the filter being used and how the traffic is being measured. For example, in Figure 4-2 the filter for the bottom graph is `nfs` and the traffic is measured in packets per second. Above the graph on the right side, the maximum (highest) value attained for the graph since it started and the current value of the graphed quantity are displayed.

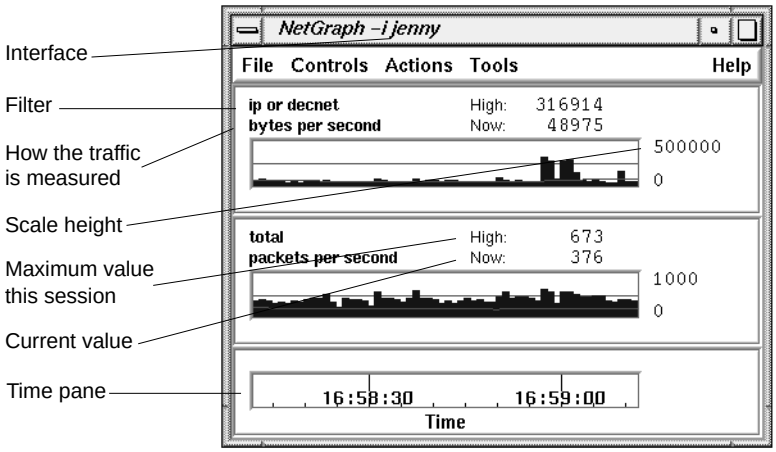



**Figure 4-2** Example NetGraph Main Window

Horizontal scale lines on the graphs help you visualize values. NetGraph's minimum-sized window shows just one scale line. However, if you increase the window size, up to 4 scale lines appear to delineate 5 vertical sections. Labels for the vertical scale of the graph appear to the right of each graph.

A Time pane appears at the bottom of the NetGraph window. By default, the time type is Scrolling. It shows the absolute (real) time of day.

To select a graph, click the left mouse button in its pane. The pane background turns yellow.

## NetGraph Control Panels

NetGraph provides two control panels, Edit and Parameters, for use in setting up what NetGraph monitors and how it is displayed.

To bring up a control panel, select it from the Controls menu. The control panel is automatically placed at the location specified in a *.netgraphrc* file if one is being used, or in a default location otherwise.

Each of the control panels is discussed in a section below.

## Edit Control Panel

Use the Edit control panel to specify a filter, how the traffic should be measured, graph style, graph color, and alarms for a single graph.

To edit a graph, select the graph by clicking on it in the NetGraph main window. The values in the Edit control panel are updated to match the current values of the selected graph. An example is shown in Figure 4-3. Each section of the control panel is discussed below.

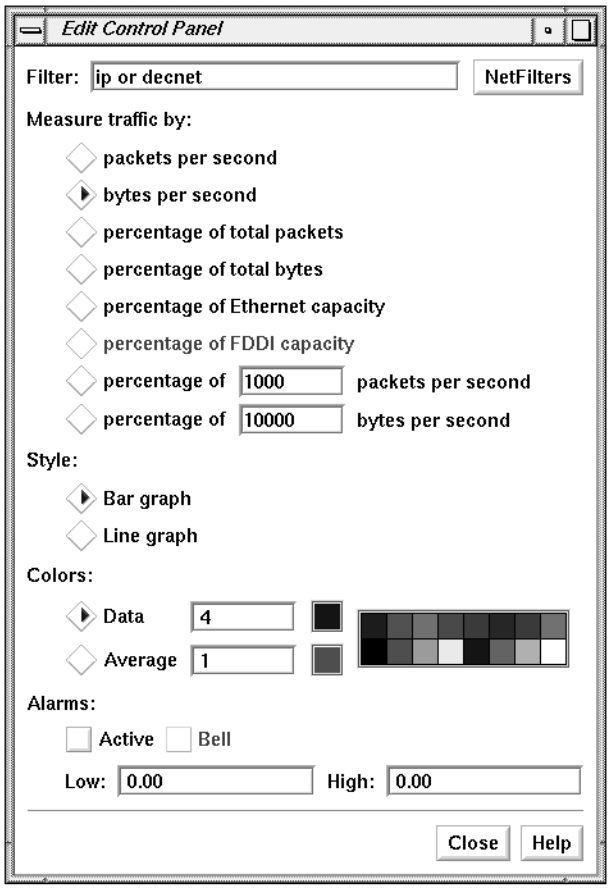



**Figure 4-3** Edit Control Panel

Perhaps you want to capture only certain network packets, such as packets destined for a particular node or packets of a particular protocol. To do this, use the Filter entry field shown in Figure 4-4 to capture only packets that are of interest to you.


**Figure 4-4** Filter Entry Field

Type the filter you want to use in the entry field and press **<Enter>** or use NetFilters. When you click the *NetFilters* button, the NetFilters main window appears. When you select a filter in NetFilters by double-clicking on it, it appears in the Filter entry field and the selected graph is automatically changed. Using NetFilters is described in Chapter 2, "NetFilters."

The NetGraph default is to monitor the total network traffic. The NetGraph-specific filter `total` is used to indicate this.

To specify how the traffic is to be measured, select the Measure traffic radio button shown in Figure 4-5 that specifies the type of graph you want. The graph types are:

packets per second
: The number of packets matching the filter in packets per second.

bytes per second
: The number of bytes matching the filter in bytes per second.

percentage of total packets
: The number of packets matching the filter as a percentage of the total number of packets.

percentage of total bytes
: The number of bytes matching the filter as a percentage of the total number of bytes.

percentage of Ethernet capacity
: The volume of traffic as a mathematically calculated percentage of Ethernet capacity. It is a percentage of the theoretical capacity of the medium.

percentage of FDDI capacity
: The volume of traffic as a mathematically calculated percentage of FDDI capacity. It is a percentage of the theoretical capacity of the medium.

percentage of *n* packets per second
: The number of packets per second as a percentage of a number of your choice. For example, if *n* is 1000, the default, 700 packets per second would be shown as 70.

percentage of *n* bytes per second
: The number of bytes per second as a percentage of a number of your choice. The default value of *n* is 10,000.

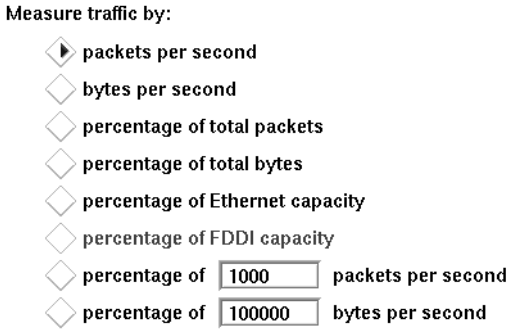



**Figure 4-5** Measure Traffic Radio Buttons

Figure 4-6 shows the Style radio buttons. Select the Bar graph radio button or the Line graph radio button to specify the graph style you want.

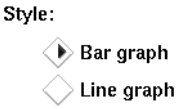



**Figure 4-6** Style Radio Buttons

For the graph and the moving average line, the Colors entry fields and left-most color squares in Figure 4-7 display their color values and colors. The array of colors on the right show some of the possible color choices. To select a new value, first select the radio button for the color you want to change, data or moving average. You can enter a color map index in the entry field and press **<Enter>**, or click the left mouse button on the color square of your choice in the array of colors on the right. The color square next to the entry field is updated to reflect the new selection and the selected graph is changed.


**Figure 4-7** Colors Radio Buttons and Entry Fields

The NetGraph alarm notifies you when a value on a graph goes above or below numbers you specify. When an alarm condition is met, NetGraph flashes pink, writes a message to a shell window or to a file you specify, and if specified emits an audible sound (a beep). The alarm is turned off by default. To set the alarm, check the "Active" check box shown in Figure 4-8. Check the "Bell" check box when you want NetGraph to beep when an alarm condition is met.


**Figure 4-8** Alarms Check Boxes and Entry Fields

The Low and High entry fields specify the conditions when you want alarm notification. If the graph goes under the Low value or above the High value, a message appears in the window from which you invoked NetGraph. If you launched NetGraph from the *netvis* directory view, a message is sent to the console. When the condition is no longer met, you also receive a message. For example, the Low entry field contains 100, the High entry field contains 1000, and the filter is `ip.host serendipity`. Each time an alarm condition is met, messages appear on the screen:

```
Alarm condition met at 19:02:10:
     filter: ip.host serendipity (packets / second)
     value: 92.00 < 100.00
Alarm condition no longer met at 19:02:22:
     filter: ip.host serendipity (packets / second)
     value: 196.00 > 100.00
Alarm condition met at 19:02:34:
     filter: ip.host serendipity (packets / second)
     value: 1105.00 > 1000.00
Alarm condition no longer met at 19:03:11:
     filter: ip.host serendipity (packets / second)
     value: 996.00 < 1000.00
```

If you do not want messages to appear on the screen, you can put them in a file by starting NetGraph with the **–l** option. See "Writing Alarm Messages to a File" in this chapter for an example.

## Parameters Control Panel

The Parameters control panel enables you to specify new values for the NetGraph time legend and graph scale parameters, to specify a new interface to collect data from, and to control various time intervals. Figure 4-9 shows the Parameter Control Panel window.

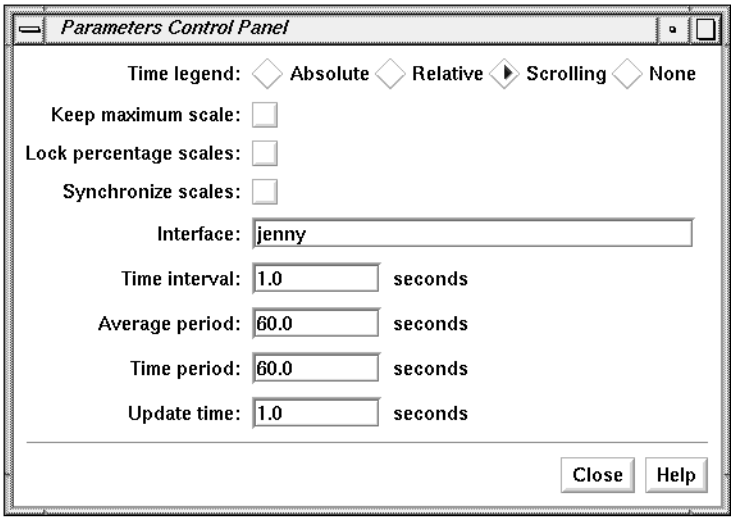



**Figure 4-9** Parameter Control Panel

Each parameter corresponds to a NetGraph command line option; for example, a check mark in the "Keep maximum scale" check box corresponds to the **–M** option. See *netgraph*(1M) in Appendix F for the options that correspond to the choices on the Parameter Control Panel.

Figure 4-10 shows the Time legend radio buttons.


**Figure 4-10** Time Legend Radio Buttons

The radio buttons are:

Absolute — Show the absolute time of day of the start and end of the displayed samples. When the period of time it takes for a single sample to scroll off the graph (Time period) is set to one minute, the time legend looks similar to Figure 4-11.

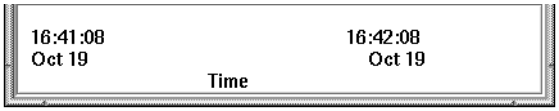



**Figure 4-11** Absolute Time

Relative — Show the time of the samples in the graphs relative to the current time. When the period of time it takes for a single sample to scroll off the graph (Time period) is set to one minute, the time legend looks similar to Figure 4-12.


**Figure 4-12** Relative Time

Scrolling — Display a scrolling time legend below the graphs showing the absolute time of day. When the period of time it takes for a single sample to scroll off the graph (Time period) is set to one minute, the time legend looks similar to Figure 4-13.

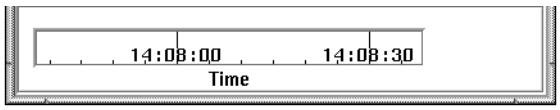



**Figure 4-13** Scrolling Time

None — Do not display a time legend.

Check the "Keep maximum scale" check box shown in Figure 4-14 to keep the vertical scale of all graphs at the maximum value they attain. When this check box is not checked, the graphs are rescaled down when NetGraph deems it appropriate. Rescaling provides better resolution of the graphs and occurs when the current value is greater than the maximum or when all values are low enough to decrease the scale. Maximum scale values are 10, 20, 50, 100, 200, 500, and so on. For example, if the maximum value displayed

reaches 81, the graph is rescaled to 100. If the maximum value drops to 50 or below for the period of time the graph is displayed on the screen, the graph is rescaled to 50.

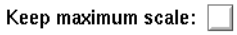



**Figure 4-14** Keep Maximum Scale Check Box

Check the "Lock percentage scales" check box shown in Figure 4-15 to lock the scale of all percentage graphs from 0 to 100. When this check box is not checked, (the default) percentage graphs scale like all the others.

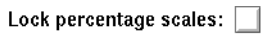



**Figure 4-15** Lock Percentage Scales Check Box

Check the "Synchronize scales" check box shown in Figure 4-16 to synchronize rescaling of all graphs of the same type. For example, graphs displaying packets per second would all be rescaled at once. When this check box is not checked (the default), graphs rescale individually.

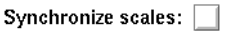



**Figure 4-16** Synchronize Scales Check Box

Leave the Interface entry field shown in Figure 4-17 blank to snoop on the local interface (the default) on the Display Station.


**Figure 4-17** Interface Entry Field

You can specify another interface, possibly on a remote Data Station, to snoop on using the format:

*station* : *interface*

The default interface is understood by NetGraph, so you don't have to specify it. For example, to snoop on the default interface on a Data Station named `reddog`, enter:

```
reddog:
```

Give the command **`netstat -i`** to see a list of configured interfaces. You can use any of the names in the Address column for *station* and the matching name in the Name column as the *interface*.

The NetGraph main window title bar includes "–i" and the interface you are snooping on.

Specify the sampling interval, in seconds, for all the graphs using the entry field shown in Figure 4-18. The default is to sample every second. A short time interval shows real-time network utilization levels that often appear as spikes. A longer interval shows average network utilization levels that seem smoother.

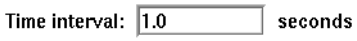



**Figure 4-18** Time Interval Entry Field

Specify the moving average calculation period of each graph using the Average period entry field shown in Figure 4-19. The default is to calculate the moving average using one minute of data. When a shorter moving average period is used, the moving average line more closely approximates the data graph. A longer average period tends to smooth out the moving average line.

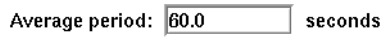



**Figure 4-19** Average Period Entry Field

Specify the period it takes for a single sample to move off the graph in seconds using the Time period entry field shown in Figure 4-20. The default is for a sample to take one minute to move off the chart. The time period cannot be less than the time interval.

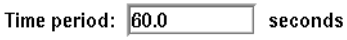



**Figure 4-20** Time Period Entry Field

If you specify a small value, the samples will scroll off the chart more quickly. If you specify a large value, the samples will display more data; this requires more memory and more time to redraw the graphs.

Set the update time in seconds using the Update time entry field shown in Figure 4-21. The default is to update every sampling interval (from the Time interval entry field). The update time determines how often graphs are redrawn. The update time cannot be less than the time interval and cannot be greater than the time period.

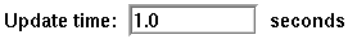



**Figure 4-21** Update Time Entry Field

For example, suppose you set the time interval and update interval to .1 so NetGraph will sample and update every .1 seconds. The drawing of the graphs may not be able to keep up with real time, especially if you have a lot of graphs. However, if you specify an update rate of 1, you will get a lot of samples and still keep up with real time. This setting causes graphs to be redrawn with 10 new samples every second rather than one new sample every .1 second.

## NetGraph Actions

Figure 4-22 shows the Actions menu. The three choices are described in the following sections.

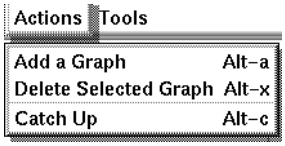



**Figure 4-22** Actions Menu

## Add a Graph

To add a graph to the NetGraph main window, select “Add a Graph” from the Actions menu. If no graph is highlighted in the NetGraph main window when you select “Add a Graph,” a new graph appears immediately above the Time pane. If a graph is selected (highlighted in yellow), the new graph appears above the selected graph. The current settings of the Edit control panel are used for the new graph. A typical sequence for adding a graph is:

1. Choose a location for the new graph by selecting the graph that you want the new graph above (do not select a graph if you want the new graph at the bottom).
2. Select “Add a Graph.”
3. Select the new graph by clicking on it.
4. Select “Edit...” from the Controls menu if the Edit control panel isn’t already open.
5. Edit the graph as described in “Edit Control Panel” in this chapter to change the filter, graph style, alarms, and so on.

## Delete Selected Graph

To delete a graph from the NetGraph main window, select the graph to highlight its pane in yellow, then select “Delete Selected Graph” from the Actions menu.

## Catch Up

The display of graphs may fall behind real time, especially if you have many graphs and a small sampling interval. To catch up to real time, select “Catch up” from the Actions menu. To catch up, NetGraph will average all data it skips, so the graphs will appear level for the skipped time. NetGraph may occasionally catch up by itself if it falls very far behind real time, or in some cases when you add or delete a graph.

## NetGraph File Menu

The NetGraph File menu lists three choices: “Save Controls,” “Save Controls As...”, and “Quit.” These choices are described below.

### Save Controls

The “Save Controls” choice enables you to save NetGraph configuration information. It saves user interface configuration information to the user interface file you last specified with “Save Controls As...” (if any) or the file read when NetGraph started up.

By default, the file *~/.netgraphrc* is read when you start NetGraph; you can specify a different user interface configuration file at startup with the **–u** command line option.

### Save Controls As

The “Save Controls As...” choice enables you to save NetGraph user interface configuration information to a file name of your choice. When you select “Save Controls As...”, a file prompter window appears. Use the procedure in the section “Using a File Prompter” in the Introduction to specify the file name for the configuration data.

By default, the file *~/.netgraphrc* is read when you start NetGraph; you can specify a different configuration file at startup with the **–u** command line option.

### Quit

To exit NetGraph, select “Quit” from the File menu. A NetGraph Question window appears. To save the current configuration in the file shown in the message and quit NetGraph, click the *Yes* button. To quit without saving configuration information, click the *No* button. If you want to write the information to another file or decide not to quit NetGraph, click the *Cancel* button.

## Playing Back a NetGraph History File

You can collect NetGraph data, put it in a history file, and review the data at a later time.

### Creating a History File

To collect data and write it to a history file, give the NetGraph command with the **–o** option:

```
netgraph -o file
```

where *file* is the name of the file in which to put the data. If the file exists, and is currently being used by another NetGraph, you cannot write history data to it; if the file exists and is not in use by another NetGraph, it will be overwritten with the new data.

For example, to save the date to a file named *netgraph_hist*, type:

```
netgraph -o netgraph_hist
```

The NetGraph window appears on the screen as described previously in this chapter. NetGraph data is written to the file you specified. To stop data collection, just quit NetGraph.

An option that can be used only when you give the **–o** option is the **–O** (capital O) option. When you give this option, traffic data is printed to standard output as well as the history file.

### Playing Back a History File

To play back the data, give the command:

**netgraph -i** *file*

For example, to play back the data from *netgraph_hist*, enter:

```
netgraph -i netgraph_hist
```

The title bar of the NetGraph main window displays the name of the file you specified on the command line (for example, *netgraph_hist*). The graph specifications and options are read from the history file and the graphs automatically replay the data that was previously captured. The bars/lines scroll from right to left, and the time legend is updated.

An additional pane with history controls appears at the bottom of the window. Figure 4-23 shows the history controls

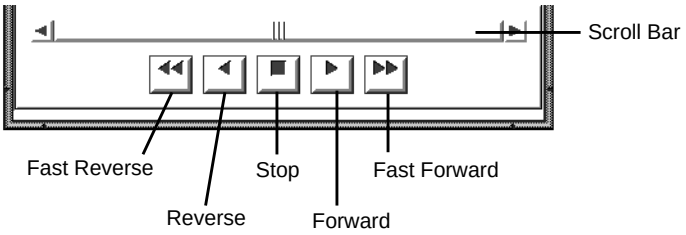



**Figure 4-23** History Playback Controls

Scroll Bar — As the data is played back, the scroll bar slider moves from left to right. The scroll bar represents the length of the file, and the slider shows the position of the graphs within the file. Press the slider and slide it to the right or left to scroll forward or backward, respectively.

Stop Button — Click the *Stop* button to stop scrolling. To continue scrolling, click any of the arrow buttons.

Right (Forward) Arrow — Click the single right arrow to cause the graphs to scroll forward slowly.

Fast Forward Arrows — Click the double right arrows button to cause the graphs to scroll forward faster.

Left (Reverse) Arrow — Click the single left arrow to cause the graphs to scroll in reverse slowly.

Fast Reverse Arrows — Click the double left arrows button to cause the graphs to scroll in reverse faster.

While you are playing back a history file, not all features of NetGraph are available since only the stored data is available. However, many options on the Edit and Parameter control panels can be used. These control panels appear in abbreviated versions when you are playing back a history file.

## NetGraph Examples

This section includes several examples of NetGraph use:

- using some simple filters
- monitoring several network segments from a single Display Station
- using NetGraph information collected over time
- monitoring internetwork traffic
- writing alarm messages to a file

### Using NetGraph with Filters

You can use any valid filter expression in the Filter entry field on the Edit control panel. For example, you may want to capture only User Datagram Protocol (UDP) packets on the network. In this case, the filter consists of a simple one-word expression: `udp`. This expression is a macro that expands to `ip.udp`. (To see the macro definition, give the command **`netsnoop -L ether`**.)

To capture all IP packets between two nodes named `gary` and `indiana`, enter this expression in the Filter entry field:

```
ip.between(gary,indiana)
```

To create a filter expression, use logical operators to combine protocols, functions, macros, and constants as described in Chapter 10, "Creating and Using Filters." Some examples of filters are:

`ip` Internet Protocol traffic

`tcp or nfs` TCP or NFS traffic

`dst=8:0:69:1:4:93`
Any traffic that goes to `8:0:69:1:4:93`

`udp and ip.src=indiana`
UDP traffic from source `indiana`

## Using NetGraph in a Distributed Environment

This section describes how to monitor three network segments from a central Display Station. You can do this provided a Data Station is connected to each network segment and you have authorization to snoop from the respective Data Stations. For information on authorization, see "Authorizing NetVisualyzer Users for Snooping" in Chapter 1.

Suppose you want to run three NetGraph sessions from the central Display Station—one that snoops locally on the Display Station and two other sessions that snoop on the two remote Data Stations. You also want to gather different information from each network. To do this, create a different configuration file for each network, and then create a script that runs the files.

For example, suppose the networks are named `building1-net`, `building2-net`, and `building3-net`. The Display and Data Stations are named `netvis-station1`, `netvis-station2`, and `netvis-station3`. Follow the steps below to monitor the three network segments.

1. Use NetGraph to create three configuration files to capture data from each Data Station. Just use the Edit and Parameters control panels to specify how you want the graphs set up. Then save the first NetGraph configuration (for example save it as *.netgraphrc1*) but do not quit NetGraph. Next edit the graphs for the second network and save the configuration as a new file name, *.netgraphrc2*, for example. If you wish, you can use an editor such as *vi*(1) to add comments to each file, for example, `# configuration for building1-net`. The files look like this:

```
# configuration for building1-net
option -i netvis-station1 -t .5
total bytes
nfs packets
“ip.between(node4, node5)” line
```

```
# configuration for building2-net
option -i netvis-station2
total bytes
nfs packets
```

```
# configuration for building3-net
option -i netvis-station3 -t 2
total bytes
“nfs or udp” %packets line
```

   Each file has the NetGraph options and graph specifications for the NetGraph session that will snoop on the respective Data Station. The key item in each file is the **–i** option that specifies the Data Station on which to snoop.

2. Create a shell script to run the three NetGraph sessions, each with the appropriate configuration file. For example, name the file *netgraph_3*; it looks like this:

```
netgraph –u .netgraphrc1
netgraph –u .netgraphrc2
netgraph –u .netgraphrc3
```

   Make sure this file is executable by typing the following at a shell prompt:

```
chmod +x netgraph_3
```

3. Now start three sessions of NetGraph, each snooping on different network segments, just by typing:

   ```
   netgraph_3
   ```

That's all there is to it. You can monitor as many networks as you want providing each network has a Data Station connected to it.

## Using the NetGraph Information

Since NetGraph shows network traffic over a period of time, it is useful in seeing how various types of traffic vary throughout the day or week.

Measuring traffic as packets per second tells you how the CPU resources in the network are being interrupted and used for protocol passing. This is often the most resource-consuming task in a network environment. For example, if you are using an NFS server for file-sharing among an increasing number of clients, you can use NetGraph to monitor the load on the NFS server and determine when subnetting and additional servers are needed.

Monitoring traffic as `%ether` or `%fddi` shows total bandwidth use. For example, if utilization is consistently above 30% for Ethernet, consider dividing the network.

Observing packet traffic between interconnected networks provides information for determining packet-routing requirements for bridges, routers, and gateways. Configuring these internetworking devices with adequate packet-processing capacity eliminates traffic congestion, improves network response, and enhances internetwork communications.

## Monitoring Internetwork Traffic

Suppose the network is divided into smaller networks and at times exhibits performance degradation. You suspect the interconnecting routers or gateways may have inadequate packet-processing capacities.

Use NetGraph to monitor the traffic going through a specific routing device. Figure 4-24 shows a router with two interfaces: `router1` with physical address `2:26:2:3:2:4` and `router2` with physical address `2:26:2:3:2:5`.

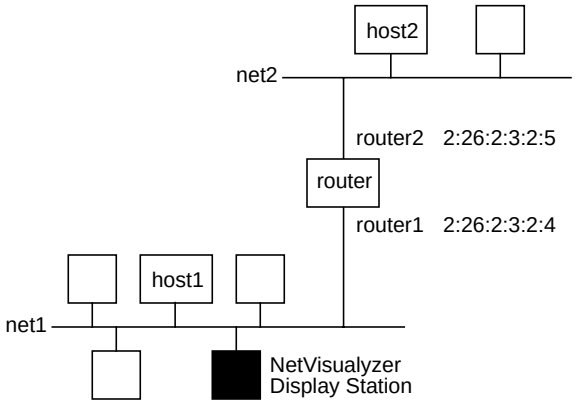



**Figure 4-24** Using NetGraph to Monitor Traffic through the Connecting Router

Use a filter expression at the command line to request that NetGraph capture and display only Ethernet packets with address `2:26:2:3:2:4` as either the source or the destination node. In other words, capture only internetwork traffic through `router1`.

Enter this filter in the Filter entry field of the Edit control panel:

```
host 2:26:2:3:2:4
```

NetGraph displays a graph of varying traffic load on `router1`, the routing device between `net1` and `net2`. This measure (in packets per second) also represents the number of internetwork packets between the two networks that router must be able to process per second. If `router1`'s traffic load exceeds its specified packet processing capacity, `router1` may be a bottleneck in passing information between the two networks.

When a router is the bottleneck in internetwork traffic, slower network response occurs. Imagine `node1` in `net1` trying to send a packet to `node2` in `net2` (see Figure 4-24) and the packet fails to enter the queue at `router1` to be processed. `node1` has to send the same packet again, thus consuming twice as much of `net1`'s bandwidth.

NetGraph, like other NetVisualyzer tools, offers filtering capabilities for isolating traffic for analysis. For example, if an NFS workgroup is part of a multivendor network, you may want to monitor only varying NFS traffic loads in the network. Enter this filter in the Filter entry field of the Edit control panel:

```
nfs
```

This filter causes only NFS traffic to be displayed. When the traffic load is measured in packets per second, the NetGraph graph shows the interrupt load on the NFS server for processing of file-sharing services. Also, the load measure in bytes per second reflects the amount of data flow between an NFS server and its clients.

As different NFS servers have different interrupt and transfer-throughput capabilities, it's up to you to decide when to divide an NFS workgroup and add servers. NetGraph helps you to predict how different NFS servers will behave in similar configurations.

## Writing Alarm Messages to a File

When you check the "Alarm" check box in the Edit control panels for a graph, alarm messages are written to your console or the shell window from where you started NetGraph. To write these messages to a file instead, start NetGraph with the **–l** option:

**`netgraph -l`** *file*

where *file* specifies the name of the message file. Messages are appended to the end of this file.

Chapter 5

# Analyzer

*Analyzer is a tool for capturing packets. This chapter explains how to configure and use Analyzer and provides examples.*

*Chapter 5*

# Analyzer

Analyzer is a graphical network protocol decoder. It is a visual interface to NetSnoop that enables you to see, at a glance, summary information about packets you capture, layer-by-layer protocol information for packets, and the contents of packets in hexadecimal format.

Analyzer is also useful in a software development environment for distributed applications. By capturing packets of the protocol you are using, you can determine the sources, destinations, and contents of packets generated by your application and the order in which packets were transmitted. This information about how the application exchanged data can be invaluable in debugging.

Once a specified number of packets or type of packet is captured and stored, Analyzer decodes each stored packet, including protocol headers and data. The decoded information that appears in the three panes of the Analyzer main window includes:

- packet information summaries
- protocol analysis
- hexadecimal and ASCII representations

The panes are synchronized for scrolling and viewing. Matching bytes are highlighted in all panes when they are selected in any pane to allow comparison of bytes between the three displays.

With Analyzer, you can analyze packets from several sources:

- the default interface on your workstation (the default)
- any interface on any Data Station that you are authorized on (see "Authorizing NetVisualyzer Users for Snooping" in Chapter 1 for details)
- a data file created by NetSnoop
- a data file created during a previous Analyzer session

By default, Analyzer captures, stores, and decodes all packets; however, you can specify a filter so that Analyzer stores only those packets that are of interest to you. In this case, Analyzer captures every packet and compares it against the filter. If a packet matches the filter, Analyzer stores that packet (for information about filters, see Chapter 10, "Creating and Using Filters"). You can also specify start and stop "trigger" conditions. Triggers enable you to specify that no packets should be stored until a particular filter is matched and that no more packets should be stored after a particular filter is matched.

Analyzer is a pure X application so you can run it on an X terminal.

This chapter explains how to:

- start Analyzer
- use the Capture control panel to specify the traffic you want to capture and store
- use the Analyzer main window to display decoded packets
- use the Analyzer File menu to save packets and configuration information
- configure Analyzer for best performance

In addition, a variety of Analyzer examples are provided. For complete information on Analyzer command line options and resources, see the *analyzer*(1M) manual page in Appendix F, "NetVisualyzer Manual Pages." Additional information about the Analyzer configuration file is provided in Appendix D, "Configuration File Formats."

**Note:** You must have authorization to use Analyzer. See "Authorizing NetVisualyzer Users for Snooping" in Chapter 1 and Appendix B, "Authorization Reference," for details. ♦

## Starting Analyzer

To start Analyzer, double-click the *analyzer* icon in the *netvis* directory view or type:

```
analyzer
```

If the configuration file *~/.analyzerrc* doesn't exist and you don't specify another file, the Analyzer main window and the Capture control panel appear in default locations; otherwise, they appear in locations specified in configuration file (their locations at the time *.analyzerrc* was created).

You can specify a configuration file on the Analyzer command line with the **–u** option or in the Analyzer resources file with the `Analyzer*controlsFile` resource.

Figure 5-1 shows the Analyzer main window, and Figure 5-2 shows the Capture control panel.


**Figure 5-1** Analyzer Main Window

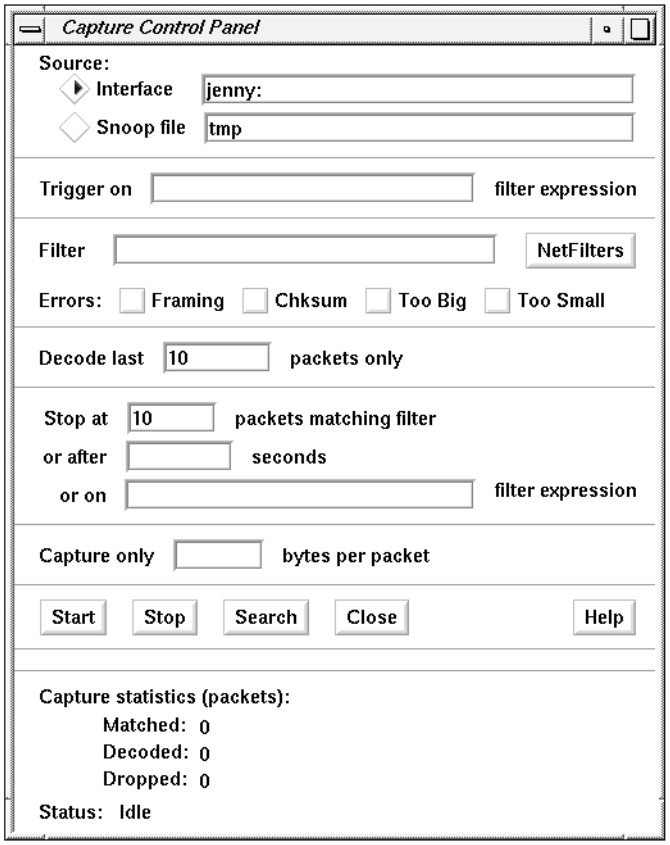



**Figure 5-2** Capture Control Panel

The next two sections describe the Capture control panel and the Analyzer main window.

## Analyzer Capture Control Panel

The Capture control panel enables you to specify options for capturing packets, to start and stop the capturing of packets, and search for packets. The following sections explain how to use this control panel to perform these tasks.

## Capture Options

This section explains the entry fields, buttons, and status area sections of the Capture control panel.

You can capture packets from either a Data Station on the network or a previously saved file that contains captured packets. To specify a Data Station, select the Interface radio button (the default) as shown in Figure 5-3. By default, its entry field contains the name of your workstation followed by a colon. This specifies that packets should be captured using the default interface on your workstation using RPC snooping.


**Figure 5-3** Source Radio Buttons

The format for specifying a Data Station and its interface is:

*station*`:`*interface*

*station* is the node name of a Data Station that you want to use to capture packets, and *interface* is the interface that you want to use on that node. (Use the command **`netstat -i`** to list the interfaces for a workstation.) If the interface is not specified, Analyzer captures from the node's default interface. For example, to capture packets from the default interface on a Data Station named `reddog`, type:

```
reddog:
```

If you want to snoop on an interface other than the default, you must specify it. For example, assume that `reddog` has an additional EFast™ Ethernet board installed and you wish to capture from it. Type:

```
reddog:fxp0
```

For a distributed environment in which a Data Station is connected to each network or segment, you can capture packets remotely. You must, however, have authorization in the */usr/etc/rpc.snoopd.auth* file on each Data Station. Thus, you can use Analyzer to collect data from remote networks and to perform remote diagnostics from one location.

To analyze captured packets that have been saved in a file, select the Snoop file radio button and enter the file name in the entry field. Previously saved files can contain the output of NetSnoop (created by redirecting the output of NetSnoop to a file) or Analyzer (see "Save Packets" in this chapter).

Use the Trigger On entry field shown in Figure 5-4 to specify a filter expression to trigger the start of packet capture. When you enter a filter expression on this line, Analyzer does not start to capture packets until it sees a packet that matches the expression. Then all packets that match the Filter entry field are captured. If this entry field is empty, Analyzer begins capturing packets when you press the *Start* button.


**Figure 5-4** Trigger On Entry Field

You can use the *NetFilters* button in this window to start up NetFilters and to copy a filter from a filter repository to this entry field. To use NetFilters, make sure the insertion point is in this entry field, click the *NetFilters* button, and select the filter you want to use by double-clicking it. The filter automatically appears in the entry field. See Chapter 2, "NetFilters," for more information about NetFilters.

A filter allows you to examine only the packets that interest you because only packets that match the given expression are stored. (Filters are described in detail in Chapter 10.) You can type the filter you want to use in the entry field shown in Figure 5-5 or you can use NetFilters. To use NetFilters, make sure the insertion point is in this entry field, click the *NetFilters* button, then select the filter you want to use by double-clicking it. The filter automatically appears in the entry field. If you leave this entry field blank, Analyzer stores all packets that it captures.


**Figure 5-5** Filter Entry Field

You can capture and store error packets by clicking one or more Errors check boxes (shown in Figure 5-6) for the desired errors. The errors you can capture are:

| | |
|---|---|
| Framing | Packet received, but with framing error |
| Chksum | Packet received, but with CRC error |
| Too big | Packet received, but truncated to fit buffer |
| Too small | Packet received, but size less than minimum |

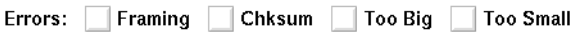



**Figure 5-6** Errors Check Boxes

If you don't specify any errors, Analyzer captures only packets without errors. See "Capturing Errors" in Chapter 10 for information on filter expressions you can use to capture errors.

The Decode Last entry field shown in Figure 5-7 allows you to specify that you do not want all captured packets stored and decoded, just the last *n* packets (a post-trigger). This feature enables you to reduce the amount of memory used for stored packets since only *n* packets are stored. For example, to capture and store 1000 packets takes 1.5 Mb of memory (1000 x 1500, the maximum size of an Ethernet packet), but if you keep only 100 packets, just .15 Mb of memory is used.

If you leave the Decode Last entry field, shown in Figure 5-7, blank, Analyzer captures and decodes the number of packets you specified in the Stop At entry field. If the Stop At and Decode Last entry fields are blank, only one packet is captured and decoded.


**Figure 5-7** Decode Last Entry Field

Three Stop At entry fields allow you to specify when to stop capturing packets. To use the first Stop At entry field, shown in Figure 5-8, specify the number of packets matching the filter that you want to store. Analyzer captures and stores the specified number and then stops. The default is to capture one packet. If you leave this entry field and the next two blank, Analyzer captures indefinitely, or until you stop it using the *Stop* button.

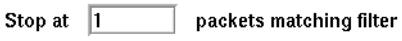



**Figure 5-8** Stop At Entry Field

To use the Or After entry field, shown in Figure 5-9, specify the length of time for the capture. For example, suppose you find that a time-out condition occurs after 30 seconds. You can capture packets for 30 seconds and then decode them to determine the cause of the time-out.


**Figure 5-9** Or After Entry Field

Enter a filter expression in the Or On entry field, shown in Figure 5-10. When a packet matches that expression, capturing stops. You can use the *NetFilters* button to start up NetFilters and copy a filter from a filter repository to this entry field. Put the insertion point in this entry field to make NetFilters copy a filter you select to this entry field.


**Figure 5-10** Or On Entry Field

You can specify more than one Stop At condition; when the first condition is met, Analyzer stops capturing packets.

By default, the Capture Only entry field shown in Figure 5-11 is blank, which means that Analyzer captures all bytes of each packet. When a number, *n*, is entered in this field, only the first *n* bytes of each captured packet are stored and decoded. The maximum number of bytes per packet for Ethernet is 1500 and for FDDI is 4352.


**Figure 5-11** Capture Only Entry Field

## Capturing and Decoding Data

To start capturing, click the *Start* button. The status information at the bottom of the control panel changes from "Idle" to "Capturing." If you entered a trigger expression in the Trigger On entry field, the status changes from "Idle" to "Waiting for Trigger."

After you press *Start*, Analyzer checks filters and other values you've specified in the Capture control panel. If it detects any errors in what you've specified, the backgrounds of entry fields with errors become red and a Capture Warning dialog box appears with an error message.

As packets are captured, the Capture statistics section at the bottom of the Capture control panel, shown in Figure 5-12, is updated to report the number of matched, decoded, and dropped packets, and the current status.

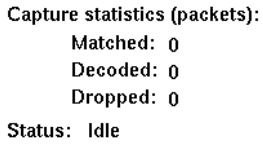



**Figure 5-12** Status Area of Capture Control Panel

Analyzer stops capturing packets when one of the three Stop At conditions is met or when you click the *Stop* button.

After the specified packets are captured, they are decoded and the status in the lower left corner changes to "Decoding." When all packets are decoded,

the status returns to “Idle” and Analyzer’s main window is updated with the newly captured packet information.

At times, Analyzer reports that packets are dropped, which means that the data you receive may not be complete. This happens when the interface or system buffers become full or other processes are using the CPU. See “Configuring Analyzer for Best Performance” in this chapter for more information.

## Searching Through Captured Packets

The *Search* button enables you to apply a filter to already captured and decoded packets. Packets that do not match the filter are not displayed. To perform a search, configure the Capture control panel for your search, for example type a filter into the Filter entry field, and click the *Search* button.

For example, suppose you capture 200 packets and then decide that only the NFS packets are of interest to you. Replace the filter in the Filter entry field, if any, with the filter `nfs` and click *Search*. The Summary pane of the Analyzer main window changes to show only the packets with `nfs` in the Type column. You may decide to look at DECnet packets next. Replace the filter with `decnet` and click *Search*. All the DECnet packets from the 200 packets are displayed.

## Analyzer Main Window

After you have captured and decoded packet data, the Analyzer main window displays packet information as shown in Figure 5-13.

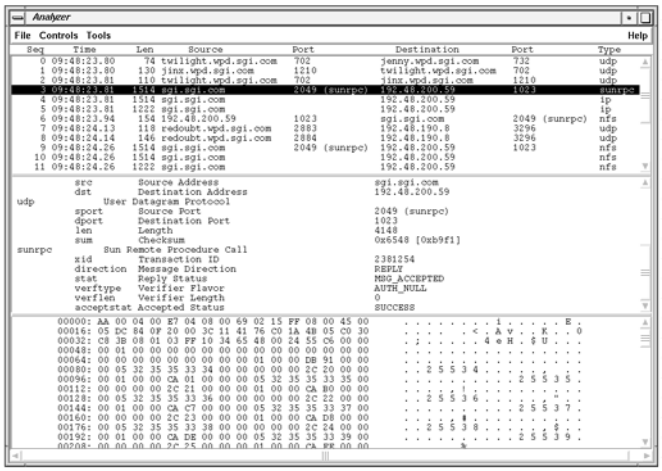



**Figure 5-13** Analyzer Main Window after Capturing Packets

Analyzer's main window has three panes. Table 5-1 describes the functions of these panes. At startup, these panes are blank.

**Table 5-1** Analyzer Main Window Panes

| Pane Position | Pane Name | Function |
|---|---|---|
| Top | Summary | Summarizes the captured packet(s) |
| Middle | Detail | Detailed protocol decoding of the captured packet(s) |
| Bottom | Hex Dump | Produces hex and ASCII dumps of the selected packet |

In the Analyzer main window, the left mouse button is your highlighting tool. In the Summary pane, use it to highlight a packet that you want to decode in the other panes. In the Detail pane, highlight an entry field, and the bytes that correspond to the entry field appear highlighted in the Hex Dump pane. In the Hex Dump pane, highlight a byte, and the entry fields that the byte represents appear highlighted in the Detail pane.

You can adjust the pane heights to maximize the display of information of interest to you without scrolling. Use the left mouse button to make the adjustment by dragging the separator up or down between the panes. As one pane grows, the pane next to it shrinks.

Scroll bars on the bottom and at the right of the main window enable you to scroll the panes. The bottom scroll bar is active only when you are displaying a packet with entry field(s) wider than the window or when you have sized the main window to less than its default width. In this case, use the bottom scroll bar to scroll all panes left and right. Each of the panes has a scroll bar on the right, which enables you to scroll the individual panes up and down.

The Detail and Hex Dump panes' scroll bars have an additional characteristic: you can use it to fix the first visible line position that you want to view for each packet you select. By default, when you select a packet in the Summary pane, the Detail and Hex Dump panes display from the first line of that packet. However, if you adjust the scroll bar so that the tenth line of a packet is the first line from the top that is visible, the next packet you select in the Summary pane is displayed starting from the tenth line. If the pane is big enough to show all the entry fields, the entire packet is displayed.

## Summary Pane

The Summary pane lists each packet in order of capture with the time of capture, the packet length, the source and destination nodes, and the protocol type.

For example, the Summary pane for a TCP packet looks like this:

```
Seq     Time      Len      Source          Port      Destination    Port   Type
  4 09:17:01.61    60 rachel.wpd.sgi.com   1017   eno.wpd.sgi.com   1016   tcp
```

Table 5-2 defines the columns of the Summary pane.

**Table 5-2** Summary Pane Columns

| Column | Description |
|---|---|
| `Seq` | Sequence in which the packet was captured |
| `Time` | Time of capture |
| `Len` | Length of the packet in bytes |
| `Source` | Source node's name or address |
| `Port` | Source port of the packet, if applicable, and possibly the port's name in parentheses |
| `Destination` | Destination node's name or address |
| `Port` | Destination port of the packet, if applicable, and possibly the port's name in parentheses |
| `Type` | Protocol type |

## Detail Pane

The Detail pane decodes the packet highlighted in the Summary pane. It decodes protocol headers layer by layer and disassembles each layer field by field. An example of the decoded layers and fields for a TCP packet looks like this:

```
ether          Ethernet
         src        Source Address                  8:0:69:6:bd:d8/SGI
         dst        Destination Address             8:0:69:2:27:73/SGI
         type       Packet Type                     ip
ip             Internet Protocol
         v          Version                         4
         hl         Header Length                   5
         tos        Type of Service                 0
         len        Total Length                    40
         id         Identification                  17141
         off        Fragment Offset                 0
         ttl        Time to Live                    60
         p          Protocol                        tcp
         sum        Header Checksum                 0x1c26
```

```
        src       Source Address                  rollon.wpd.sgi.com
        dst       Destination Address             midas.wpd.sgi.com
tcp          Transmission Control Protocol
        sport     Source Port                     1023
        dport     Destination Port                513 (rlogin)
        seq       Sequence Number                 43,968,243
        ack       Acknowledgment Number           1,841,551,347
        off       Data Offset                     5
        flags     Flags                           ACK
        win       Window                          61,440
        sum       Checksum                        0x6ad5
        urp       Urgent Pointer                  0
```

Notice how Analyzer decoded Ethernet, IP, and TCP layers. Table 5-3 lists and describes the fields decoded in this packet. When you decode a packet and want a description of a field, refer to that protocol's RFC or specification manual for information about its fields. See Appendix C, "Protocols," for a list of references for the protocol specifications.

**Table 5-3** Protocol Information in the Detail Pane

| Field | Title | Description |
|---|---|---|
| `ether` | `Ethernet` | Protocol name |
| `src` | `Source Address` | Ethernet source address |
| `dst` | `Destination Address` | Ethernet destination address |
| `type` | `Packet Type` | The packet type (`ip`) |
| `ip` | `Internet Protocol` | Protocol name |
| `v` | `Version` | Version number |
| `hl` | `Header Length` | Length of the header |
| `tos` | `Type of Service` | Normal service |
| `len` | `Total Length` | Length of the header and data |
| `id` | `Identification` | A value in assembling fragments |
| `off` | `Fragment Offset` | Indicates where in data a fragment belongs |

**Table 5-3** (continued) Protocol Information in the Detail Pane

| Field | Title | Description |
|---|---|---|
| ttl | Time to Live | Time data allowed to remain in IP system |
| p | Protocol | Protocol of next layer (tcp) |
| sum | Header Checksum | Checksum on the header |
| src | Source Address | IP source address |
| dst | Destination Address | IP destination address |
| tcp | Transmission Control Protocol | Protocol name |
| sport | Source Port | Source port number |
| dport | Destination Port | Destination port number |
| seq | Sequence Number | Sequence number of first data octet |
| ack | Acknowledgment Number | Value of next sequence number expected |
| off | Data Offset | Indicates where data begins (multiples of 4) |
| flags | Flags | Indicates Acknowledgment field significant (ACK) |
| win | Window | Data sender is willing to accept |
| sum | Checksum | Checksum of header and data |
| urp | Urgent Pointer | Current value of the urgent pointer |

You may find the protocol fields in the Detail pane useful when you construct a filter. For example, you may want to use field names such as `src`, `dst`, `sport`, and `dport` when building a filter.

### Hex Dump Pane

The Hex Dump pane provides a full listing of the packet highlighted in the Detail pane in both hexadecimal and ASCII formats. For example, the output of a TCP packet in the Hex Dump pane looks like this:

```
00000: 08 00 69 02 29 38 08 00 69 06 64 68 08 00 45 00   ..i.)8..i.dh..E.
00016: 00 28 16 D6 00 00 3C 06 23 6A C0 1A 4F 0B C0 66   .(....<.#j..O..f
00032: 75 04 03 FF 02 02 6D 44 A8 3A 6D 32 C9 FD 50 10   u.....mD.:m2..P.
00048: EF 88 29 0B 00 00 00 20 00 01 00 20               ..).... ...
```

The left column gives the byte offset of the first byte in the row. The middle section displays 16 bytes in hexadecimal. The right section displays those same bytes in ASCII. A dot (.) appears where the ASCII representation of the byte is not a printing character. Typically, the ASCII representation is useful only when the packet contains ASCII text.

You can use the left mouse button in the Hex Dump pane to select the fields of the packet. Click the left mouse button on any character in the middle or right sections of the Hex Dump pane to highlight the field that includes that byte in the hex and ASCII sections of the Hex Dump pane and in the Detail pane.

## Analyzer File Menu

The File menu enables you to save packets that you've captured and decoded to files and to exit Analyzer. The sections below describe the two choices on the "Save Packets" rollover menu, the two choices on the "Save Controls" rollover menu, and "Quit."

### Save Packets

The "As Snoop File..." choice on the "Save Packets" rollover menu enables you to save stored packets to a file. When you choose "As Snoop File...", a file prompter window appears. Use it to specify a file name (see "Using a File Prompter" in the Introduction for more details). You can view these packets with Analyzer again by selecting Snoop file radio button as the source in the Capture control panel and specifying this file name.

You can save packets captured by Analyzer in a text file whose format is similar to the three panes of Analyzer. To do this, choose "As Text File..." from the "Save Packets" rollover menu. The dialog box shown in Figure 5-14 appears. Specify the packets you want to save in one of two ways: select the packets in the Summary pane of the Analyzer main window, or enter a packet number sequence in the entry field of the dialog box. A packet number sequence can be a single packet number, a packet number range, for example `100-200`, or the word `all`. Use the check boxes to specify which information (panes) you want to save (Summary, Detail, and/or Hex Dump).

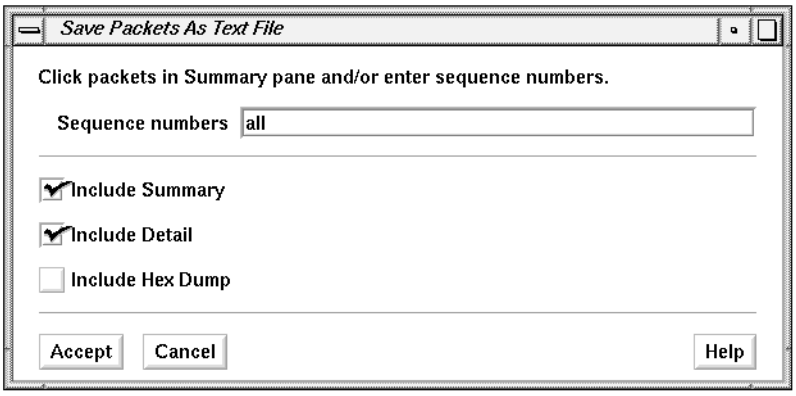



**Figure 5-14** Save Packets As Text File Dialog Box

When you click *Accept*, a file prompter window appears. Use it to specify the file to which you want to save this "Analyzer main window look-alike" version of the decoded packets. The information is appended to the file and appears one packet at a time. For example, if you request Summary, Detail, and Hex Dump information, Analyzer generates a Summary line for each packet followed by the decoding of the protocol layers from the Detail pane, and the hex and ASCII versions from the Hex Dump pane, as shown below.

```
+++

Seq     Time   Len Source        Port   Destination         Port          Type
0 15:53:09.16  154 192.26.72.5    1023   maddog.wpd.sgi.com  2049 (sunrpc) nfs

 ether          Ethernet
          src        Source Address               8:0:69:2:26:11/SGI
```

```
          dst       Destination Address          2:cf:1f:e0:81:55/CMC
          type      Packet Type                  ip
ip             Internet Protocol
          v         Version                      4
          hl        Header Length                5
          tos       Type of Service              0
          len       Total Length                 140
          id        Identification               44143
          off       Fragment Offset              0
          ttl       Time to Live                 58
          p         Protocol                     udp
          sum       Header Checksum              0x7583
          src       Source Address               192.26.72.5
          dst       Destination Address          maddog.wpd.sgi.com
udp            User Datagram Protocol
          sport     Source Port                  1023
          dport     Destination Port             2049 (sunrpc)
          len       Length                       120
          sum       Checksum                     0x33f9
sunrpc         Sun Remote Procedure Call
          xid       Transaction ID               13041
          direction Message Direction            CALL
          rpcvers   RPC Version Number           2
          prog      Program Number               nfs
          vers      Version Number               2
          proc      Procedure Number             16
          credtype  Credential Flavor            AUTH_UNIX
          credlen   Credential Length            32
          cred      Call Credentials             moose:1274.10,10
          verftype  Verifier Flavor              AUTH_NULL
          verflen   Verifier Length              0
nfs            Sun Network File System
          proc      Procedure                    READDIR
          fh        File Handle                  1e00:4794 [1e00 a0000 12ba
29ceb8f2 a 0 229ce b8f20000]
          offset    Offset                       0
          count     Count                        4096

00000: 02 CF 1F E0 81 55 08 00 69 02 26 11 08 00 45 00   .....U..i.&...E.
00016: 00 8C AC 6F 00 00 3A 11 75 83 C0 1A 48 05 C0 30   ...o..:.u...H..0
00032: 96 1E 03 FF 08 01 00 78 33 F9 00 00 32 F1 00 00   .......x3...2...
00048: 00 00 00 00 00 02 00 01 86 A3 00 00 00 02 00 00   ................
00064: 00 10 00 00 00 01 00 00 00 20 2A AD 2E D5 00 00   ......... *.....
00080: 00 05 6D 6F 6F 73 65 00 00 00 00 00 04 FA 00 00   ..moose.........
00096: 00 0A 00 00 00 01 00 00 00 0A 00 00 00 00 00 00   ................
```

```
00112: 00 00 00 00 1E 00 00 0A 00 00 00 00 12 BA 29 CE   ..............).
00128: B8 F2 00 00 00 0A 00 00 00 00 00 02 29 CE B8 F2   ............)...
00144: 00 00 00 00 00 00 00 00 10 00                     ..........
```

### Save Controls

The "Save Controls" choice enables you to save user interface configuration information. This choice has a rollover menu with two choices, "Save" and "Save As...". These choices save user interface configuration information to the file *~/.analyzerrc* or to a file name of your choice. When you choose "Save As...", a file prompter window appears. Use the procedure in the section "Using a File Prompter" in the Introduction to specify the file name for the user interface configuration data.

By default, the file *~/.analyzerrc* is read when you start Analyzer; you can specify a different network configuration file at startup with the **–u** command line option.

### Quit

To exit Analyzer, select "Quit" from the File menu. An Analyzer Question window appears. To save the current Capture control panel settings in the file shown in the message and to quit Analyzer, click the *Yes* button. To quit without saving configuration information, click the *No* button. If you want to write the information to another file or decide not to quit NetLook, click the *Cancel* button.

## Configuring Analyzer for Best Performance

This section gives recommendations designed to minimize the possibility that Analyzer will drop packets and to maximize Analyzer performance. They are conservative recommendations that allow for worst-case network traffic scenarios and should be used as starting points for adjusting the values to your needs.

When using Analyzer, the following values in the Capture control panel give the best results:

- The Capture Only entry field should be set to about 200. Seventy bytes gives optimal results, but if you are looking at NFS or AFP protocol headers, a setting of 70 does not show all of the fields.
- The Stop At entry field should be set to less than 200 packets.
- The Decode Last entry field should be set to less than 200.
- The Filter entry field should contain a simple protocol expression that can be translated into the kernel's filter string so that the kernel, instead of *snoopd*, can do filtering. Some examples are:

  ```
  src == 8:0:69:4:0:16
  ```

  ```
  ip
  ```

  ```
  host(8:0:69:1f:0:4)
  ```

Other NetVisualyzer tools should not be run when you run Analyzer because they dramatically affect Analyzer performance. Dedicating the Display Station to Analyzer (no applications or services running beyond the minimum) also improves performance.

When you specify a value in the Stop At entry field, Analyzer's statistics for the number of packets dropped are accurate. They are also accurate when the Stop At and Or After entry fields in the Capture control panel are blank (specifying free-running capture on the default interface until the *Stop* button is clicked).

When capturing is based on the Or After entry field or the Or On entry field, the number of dropped packets may be exaggerated. The number includes packets that were received but ignored after the packets of interest were detected at the interface.

## Analyzer Examples

The following sections explain three examples of using Analyzer. The first section lists the packets that are captured and stored for four different settings of the Capture control panel. The second example explains how to use the Decode Last entry field to investigate problems that happen just prior to a known event. The third example shows how to use Analyzer to record security intrusions.

### Filter Examples

Table 5-4 shows the packets that Analyzer stores during four sessions in which it captures identical packets. Each session has different specifications for the Trigger On, Filter, and Or On entry fields.

**Table 5-4** Packets Stored for Four Settings of the Capture Control Panel

| Packets Seen on the Network | Capture Control Panel Settings | Session 1 | Session 2 | Session 3 | Session 4 |
|---|---|---|---|---|---|
| | **Trigger on** | | nfs | | nfs |
| | **Filter** | | | tcp | tcp |
| | **or on filter expression** | | udp | | udp |
| udp | | udp | | | |
| tcp | | tcp | | tcp | |
| tcp | | tcp | | tcp | |
| nfs | | nfs | nfs | | |
| tcp | | tcp | tcp | tcp | tcp |
| tcp | | tcp | tcp | tcp | tcp |
| nfs | | nfs | nfs | | |
| nfs | | nfs | nfs | | |

**Table 5-4** Packets Stored for Four Settings of the Capture Control Panel

| Packets Seen on the Network | Capture Control Panel Settings | Session 1 | Session 2 | Session 3 | Session 4 |
|---|---|---|---|---|---|
| | **Trigger on** | | nfs | | nfs |
| | **Filter** | | | tcp | tcp |
| | **or on filter expression** | | udp | | udp |
| tcp | | tcp | tcp | tcp | tcp |
| arpip | | arpip | arpip | | |
| tcp | | tcp | tcp | tcp | tcp |
| tcp | | tcp | tcp | tcp | tcp |
| udp | | udp | udp | | |
| tcp | | tcp | | tcp | |
| nfs | | nfs | | | |
| tcp | | tcp | | tcp | |

In Session 1, no expressions are given so Analyzer stores all packets it captures.

In Session 2, Analyzer waits for the Trigger On expression (`nfs`), stores all packets on the network because no filter to compare the packets to is specified, and stops capturing when it sees a UDP packet because `udp` was specified in the Or On entry field.

In Session 3, Analyzer stores only TCP packets because they match the `tcp` filter.

In Session 4, Analyzer waits for an NFS packet (the trigger), stores only TCP packets because they match the `tcp` filter, and stops capturing when it sees a UDP packet.

## "Decode Last" Examples

Suppose you know that a condition occurs after a certain number of packets (say, 200) are captured. You suspect that the last few packets cause the condition. Use a value in the Decode Last entry field as a post-trigger to store (and subsequently decode) only the last few captured packets. Capture 200 packets and store the last 20 that were captured by filling in Capture control panel entry fields as shown in Figure 5-15.

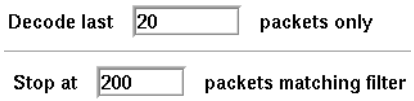



**Figure 5-15** Example Decode Last and Stop At Entry Fields

Analyzer captures 200 packets, stores the last 20 captured, and decodes these 20 packets.

As another example, suppose you know that a problem occurs before an NFS packet whose procedure field (`proc`) contains `GETATTR`. To view the last 30 packets before the `GETATTR` packet, specify Capture control panel entry fields as shown in Figure 5-16.

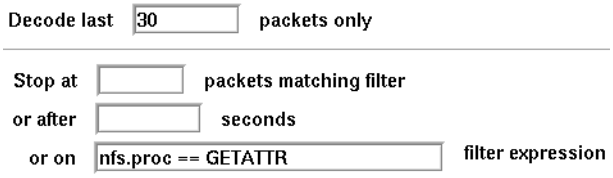



**Figure 5-16** Example Decode Last and Or On Entry Fields

Analyzer captures all packets from the time you click the *Start* button, but it stores only the last 30 packets it has seen. It stops capturing when it finds a packet with `nfs.proc` equal to `GETATTR`, then decodes the 30 stored packets.

## Using Analyzer to Record Network Security Intrusions

The example in "Using NetLook to Monitor Network Security Intrusions" in Chapter 3 describes how to use NetLook to identify a network security breach and trace its source. Often, however, just knowing that a breach has occurred and who the violator is may not be sufficient. To understand the implications of the breach, you may want to know exactly what transpired. For instance, were files copied or changed? Use Analyzer to help you rebuild what happened; it records a network intrusion in its entirety, packet by packet.

### Setting Up Analyzer to Capture Packets

To set up Analyzer to capture packets:

1. In the Filter entry field of the Capture control panel, enter this filter expression to capture all traffic packets into and out of `secret`:

   **`host secret`**

   Or instead of `secret`'s name, you can enter its IP address.
2. Make sure no Errors check boxes are checked (the default).
3. Remove any filters in the Or On entry field.
4. In the Decode Last entry field, enter **`5000`** packets (or any number that you expect to capture during the entire monitoring period).
5. Click *Start* to start capturing packets. Whenever a packet involving `secret` is captured, the count for matched packets in the Capture statistics portion of the control panel increments by 1.
6. When you return, click the *Stop* button.

### Interpreting the Captured Packets

The information in the Summary pane of the Analyzer main window tells you a great deal about the nature of the communication between `secret` and the potential intruders. The timing information tells when the intrusion occurred. The node data tells who the intruders are. The port data tells what kind of activities occurred. For example, if destination port 513 (a well-known port for *rlogin*(1C)) is detected, it is an indication that someone

tried to log in from a remote workstation. (Give the command **`cat /etc/services`** in a shell window to see a list of well-known ports.)

The middle and bottom panes provide more detailed information, such as the sequencing of the packets. You can use this information to discover how the two nodes exchanged information. For example, to determine whether the intruder succeeded in a remote login to `secret`, use the middle pane to decode the packet's protocol header layer by layer down the protocol stack, and field by field at each layer. Thus you can see the interactions between `secret` and the intruder and the status of the login attempt.

You can also use the hexadecimal and ASCII dumps in the bottom pane to look for messages sent from `secret` to the intruder. For example, a packet with the message "login incorrect" indicates that the user could not log in because he or she did not enter the correct password.

Chapter 6

# NetTop

*NetTop uses three-dimensional graphs to display the volume of network traffic between selected nodes. This chapter explains how to configure and use NetTop and provides examples.*

*Chapter 6*

# NetTop

NetTop gives you a graphical view of the volume of network traffic between selected nodes in your network. For example, it can provide you with a real-time display indicating the source and destination nodes with the highest volume of packets or bytes or the volume of traffic that matches filters you specify at particular nodes. With its rotatable 3-D bar graphs, you can see at a glance the total and relative volume of traffic at nodes of interest to you.

This chapter explains how to:

- start NetTop
- use the NetTop main window to display real-time traffic volume information
- use the NetTop control panels to specify the type of traffic you want to view and configure the display of traffic
- use the NetTop File menu

In addition, several examples are provided. For complete information on NetTop command line options and resources, see the *nettop*(1M) manual page in Appendix F, "NetVisualyzer Manual Pages." Additional information about the NetTop configuration file is provided in Appendix D, "Configuration File Formats."

**Note:** You must be authorized to use NetTop. See "Authorizing NetVisualyzer Users for Snooping" in Chapter 1 and Appendix B, "Authorization Reference," for details. ♦

## Starting NetTop

To start NetTop, double-click the *nettop* icon in the *netvis* directory view or enter:

```
nettop
```

The NetTop main window appears. An example of the NetTop main window is shown in Figure 6-1.

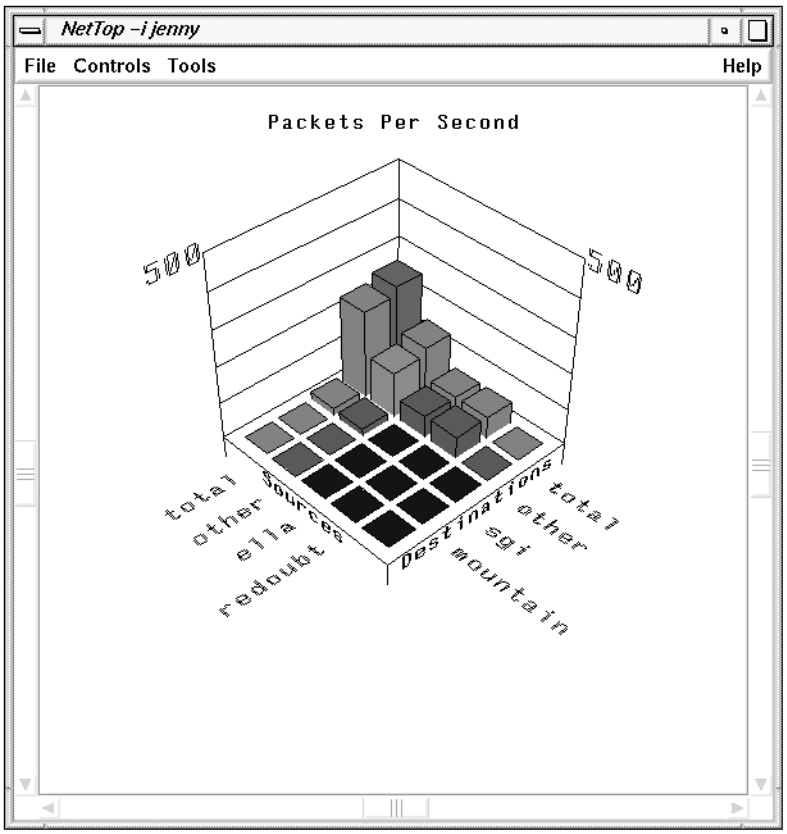



**Figure 6-1** NetTop Main Window

By default, NetTop looks for the configuration file *.nettoprc* in your home directory. You can specify a configuration file on the NetTop command line with the **–u** option or in the NetTop resources file with the `NetTop*controlsFile` resource.

By default, NetTop displays the network traffic between five source nodes and five destination nodes in packets per second. The sources and destinations shown are `total`, the total network traffic on this network segment; `other`, the difference between `total` and the specific sources or destinations shown; and three specific source and destination nodes. By default, the three source and destination nodes are read from the file *.nettoprc* or are blank if no *.nettoprc* file is found at startup.

Each tower represents the number of packets between its source and its destination nodes per second. This number is calculated over intervals (the default is 1 second) and displayed at the end of each interval. Scale lines are shown on the "back" two sides of the 3-D graph. The top scale line is labeled with the number of packets or bytes per second. This number is rescaled up as often as needed and rescaled down after 5 seconds if the traffic level drops. See "NetTop Traffic Control Panel" in this chapter for more information about the rescaling of the scale lines.

## NetTop Main Window

The NetTop main window scroll bars enable you to rotate and scale the NetTop graph. The three scroll bars are:

Right scroll bar – tilt
: The right scroll bar enables you to control the angle at which you view the graph. With the scroll bar at the bottom, you are looking at the towers from a point even with their bases. With the scroll bar at the top, you have a birds-eye view of the graph from above its center.

Bottom scroll bar – spin
: When you move the bottom scroll bar, the graph spins counterclockwise (moving left) or clockwise (moving right). The vertical scale lines adjust so that they are always at the "back" of the graph, and all of the labels adjust so that they are readable.

Left scroll bar – zoom
: To zoom in, move the left scroll bar down; to zoom out, move the left scroll bar up.

To return the graph to its initial position, press the **<Home>** key.

The horizontal axes of the graph are labeled with the node names, node addresses, or filters. Use the Nodes control panel to change the labels (see "NetTop Nodes Control Panel" in this chapter for more information). The labels can be in one of several colors:

| | |
|---|---|
| blue | Locked nodes and filters |
| yellow | Selected nodes and filters |
| green | Normal nodes and filters |
| red | An error |

You can click on node names, filters, and towers to select them or deselect them if they are already selected. Clicking on a tower is equivalent to clicking on both of its node names or its node and filter. Selected towers, names and filters appear in yellow.

For selected towers, the source (or node) name, destination name (or filter), and the value of the highlighted tower are shown at the bottom of the main window. Figure 6-2 shows an example. The numeric value of the tower is updated as the graph is changed. When an entire row or column is selected, the numeric value is the value of the "total" tower in that row or column.

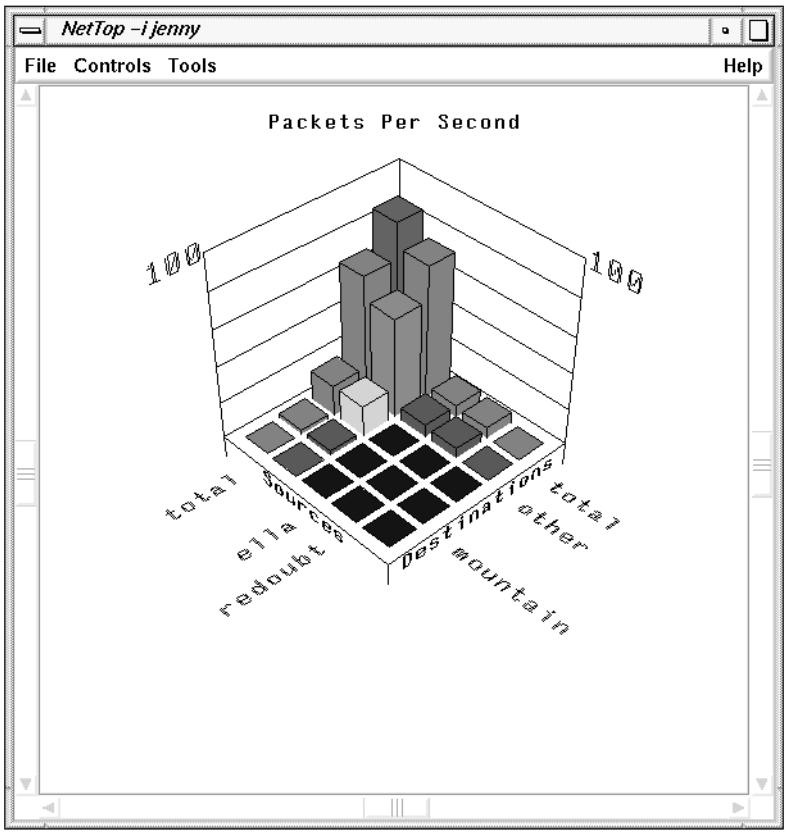



**Figure 6-2** NetTop Main Window with a Selected Tower

When NetTop is automatically updating the nodes shown on the horizontal axes with the currently busiest nodes, you can lock a particular tower by double-clicking it. Locking a node causes it to be displayed even when it is no longer among the busiest nodes or node pairs. You might want to lock a node pair, for instance, if you notice a very high volume of traffic for the pair and want to watch for a while to determine if the burst in traffic is sustained or not. See "NetTop Nodes Control Panel" in this chapter for more information about locking nodes.

The NetTop main window title bar includes "–i" and the interface on which it is snooping. The interface is in the same format as the argument to the **–i** command line option.

## NetTop Traffic Control Panel

When you select "Traffic" from the Controls menu in the NetTop main window, the control panel shown in Figure 6-3 is displayed. This control panel enables you to change the way the NetTop graph displays traffic.

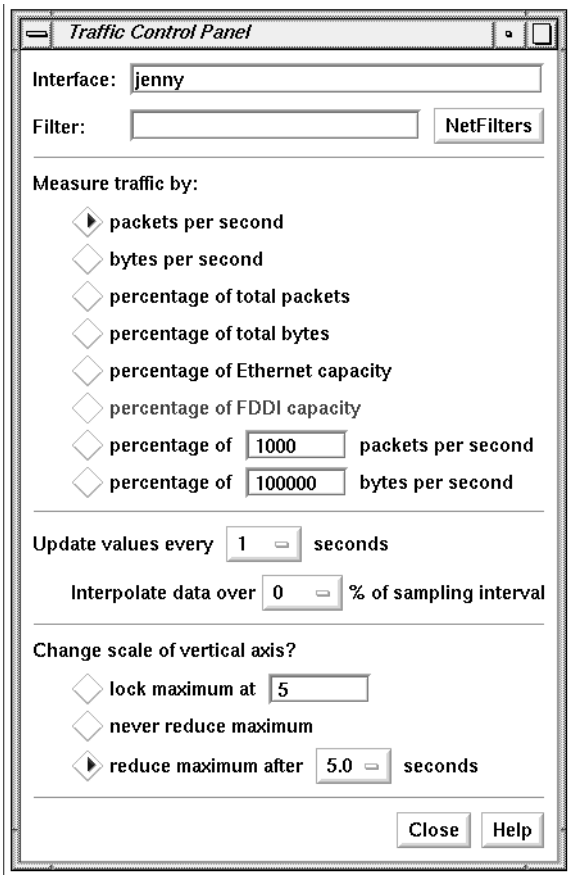



**Figure 6-3** Traffic Control Panel

The Interface entry field shown in Figure 6-4 contains the name and/or interface of the node on which you are snooping. By default, NetTop snoops on the default interface of your workstation. You can specify the Data Station on which you want to snoop by entering it in this entry field and pressing

**<Enter>**. You can specify another interface, possibly on a remote Data Station, on which to snoop using the format:

*station*:*interface*


**Figure 6-4** Interface Entry Field

You can also specify an interface by starting NetTop with the **–i** option. Give the command **`netstat -i`** to see a list of configured interfaces. The NetTop main window title bar includes "–i" and the interface on which you are snooping.

You can limit the packets that are counted to a subset of interest to you by entering a filter in the Filter entry field shown in Figure 6-5. Type in the filter and press **<Enter>** or click the *NetFilters* button to view the NetFilters main window, then click on the filter you want to select it. The filter you select will appear in the Filter entry field. Using NetFilters is described in Chapter 2, "NetFilters." Constructing filters is described in Chapter 10, "Creating and Using Filters."


**Figure 6-5** Filter Entry Field

Each tower shows the traffic from source to destination that matches the filter entered above, or if nodes and filters are shown, each tower shows the traffic to and from each node that matches both the filter above and the filter for that tower. The section of the control panel shown in Figure 6-6 enables you to select the units used in displaying the traffic:

packets per second
: Each tower displays the number of packets per second from source to destination. If the axes are labeled with nodes and filters, the towers display the number of packets per second to or from each node that match the filter for that row. The number displayed is counted over intervals whose length is specified by the Update values option button in this control panel. The number is displayed immediately or gradually as specified by the Interpolate data option button.

bytes per second
: Each tower displays the number of bytes per second from source to destination. If the axes are labeled with nodes and filters, the towers display the number of packets per second at each node that match the filter. The number displayed is counted and displayed as described above.

percentage of total packets
: Each tower displays the number of packets from source to destination (or to or from its node and matching the filter) as a percentage of the total number of all packets seen on the network.

percentage of total bytes
: Each tower displays the number of bytes from source to destination (or to or from its node and matching the filter) as a percentage of the total number of all bytes seen on the network.

percentage of Ethernet capacity
: Each tower displays the mathematically calculated percentage of the theoretical capacity of the Ethernet medium that is in use.

percentage of FDDI capacity
: Each tower displays the mathematically calculated percentage of the theoretical capacity of the FDDI medium that is in use.

percentage of *n* packets per second
: Each tower displays the number of packets per second as a percentage of a number, *n*, that you specify in the entry field on this line. The default value of *n* is 1000.

percentage of *n* bytes per second
: Each tower displays the number of bytes per second as a percentage of a number, *n*, that you specify in the entry field on this line. The default value of *n* is 100,000.

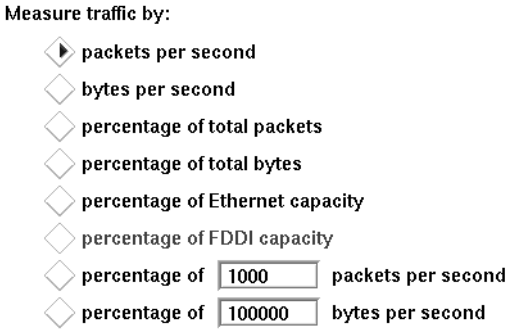



**Figure 6-6** Measure Traffic Radio Buttons

The Update values option button shown in Figure 6-7 controls how often the new values of the towers are calculated (counting period). For instance, if `1` is shown on the option box, the counting period is one second and the towers are updated once every second to show the traffic data value obtained in the previous second. To change this value, press the option button and select one of the values in the menu that pops up.


**Figure 6-7** Update Values Option Button

If the Interpolate data option button shown in Figure 6-8 has the default value, 0, the tower heights are changed each time the new values are calculated at the end of each counting period specified by the option button above this line. The option button provides other values that specify how long a gradual increase or decrease to the new values should take: 10, 25, 50, 75, or 100% of the next counting interval.


**Figure 6-8** Interpolate Data Option Button

The Change scale radio buttons shown in Figure 6-9 enable you to control how the data is scaled:

lock maximum at *n*
: When this radio button is selected, you can choose the maximum label value shown on the vertical scale by typing it in the entry field and then pressing **<Enter>**. The towers display their actual value, even if they are off the top of the scale.

never reduce maximum
: When this radio button is selected, the vertical axis is rescaled to display new maximum tower values only. It is never rescaled to a smaller number.

reduce maximum after *n* seconds
: When this radio button is selected (the default), the vertical axis is rescaled as necessary to accommodate new maximum tower values. Each new maximum value is displayed for at least the number of seconds shown in the option box on the line (the default is 5 seconds).

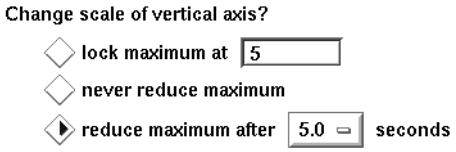



**Figure 6-9** Change Scale Radio Buttons

## NetTop Nodes Control Panel

When you select "Nodes" from the Controls menu in the NetTop main window, the control panel shown in Figure 6-10 is displayed. Its appearance depends on which radio buttons are selected. This control panel enables you to specify the source and destination nodes (or alternatively nodes and filters) shown in the NetTop graph. The Nodes control panel also enables you to specify the labels of the horizontal axes and the number of nodes and filters.

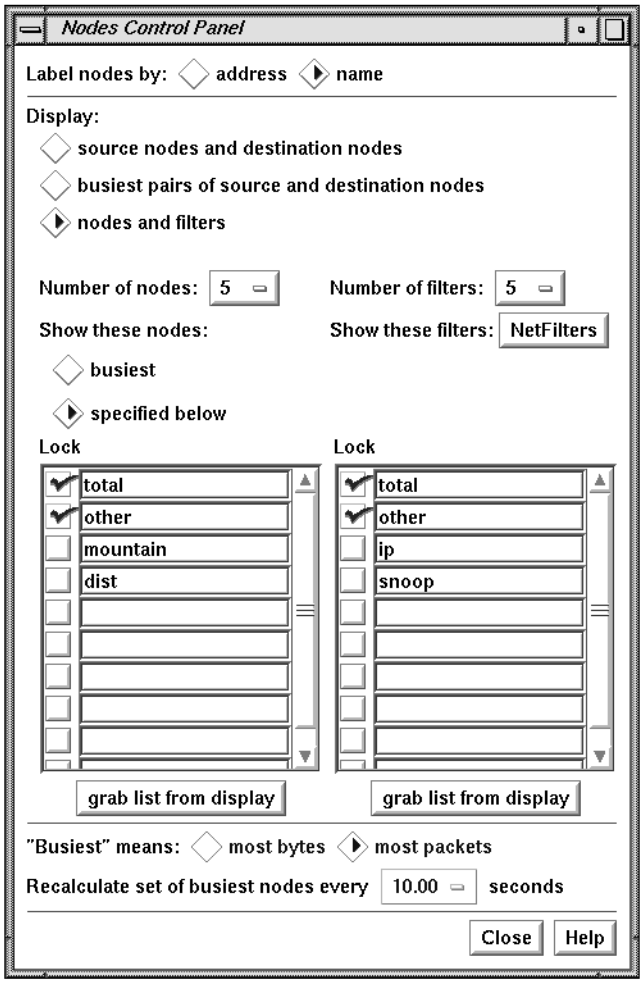



**Figure 6-10** Nodes Control Panel

In the Label nodes section of the Nodes control panel shown in Figure 6-11 you can change how nodes are labeled. The choices are:

| | |
|---|---|
| name | If this radio button is checked, node names from */etc/hosts*, NIS, or BIND are used to label one or both of the horizontal axes (for more information see "Address/Name Resolution" in Chapter 1). |
| address | Node addresses are used as labels if this radio button is checked. It can be an IP address, DECnet address, or physical address, depending on the type of traffic. |


**Figure 6-11** Label Nodes Radio Buttons

This setting is ignored for nodes whose names or addresses you've entered farther down in this control panel. For these nodes, the name or address you type is used.

The remainder of the Nodes control panel is used to specify what you want to display on the horizontal axes. The appearance of this section varies depending on which of the three Display buttons is chosen. The three versions are shown and described below.

When you choose the source nodes and destination nodes radio button as shown in Figure 6-12, two option buttons appear: one for the number of source nodes and one for the number of destination nodes. By default five sources and five destinations are displayed; you can use the option buttons to select different numbers.

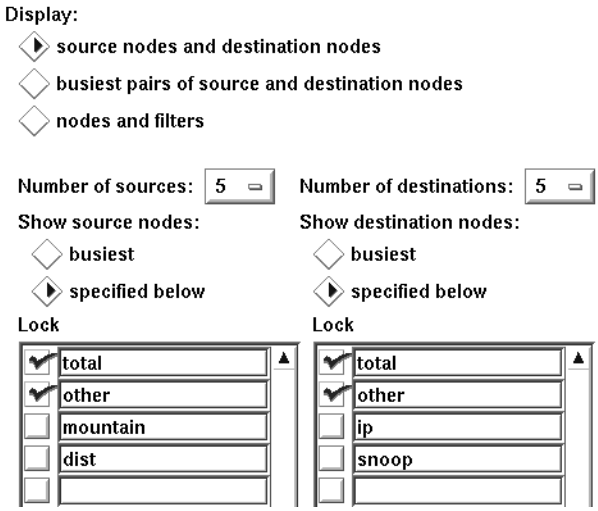



**Figure 6-12** Display Section of Nodes Control Panel (Sources and Destinations)

Using radio buttons, you can choose to display the busiest source and/or destination nodes, or specific nodes. If you select the specified below radio button, enter the node names or addresses in the entry fields below. Press **<Enter>** when you finish each name or address to make it appear on the graph in the main window.

When you choose the busiest radio button for source and/or destination nodes, you can use the "Lock" check boxes to specify that you want to continue to see a particular node in the NetTop main window, even if it is not among the busiest nodes.

When you select the busiest pairs of source and destination nodes radio button as shown in Figure 6-13, use the option button that appears to select the number of node pairs you want to view. You can show up to ten pairs. The busiest node pairs are node pairs that have the highest amount of traffic between them. How busy a node is can be measured in packets per second or bytes per second as specified by the radio buttons at the bottom of the window.

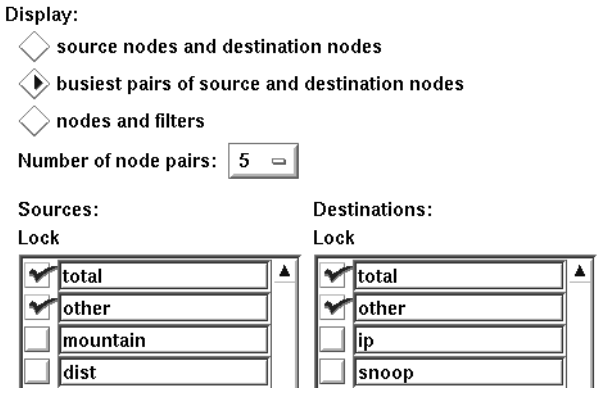



**Figure 6-13** Display Section of Nodes Control Panel (Busiest Pairs)

Two scrolling lists show source and destination node names. If the box is checked, the node is always displayed on the graph in the main window, whether it is busy or not.

When you put a check in a "Lock" check box, the node is displayed in the NetTop main window, even if it is not among the busiest nodes.

When you choose the nodes and filters radio button as shown in Figure 6-14, you can select the numbers of nodes and filters with option boxes. NetTop calculates and displays the busiest nodes if you select the busiest radio button in the "Show these nodes" section, or you can enter specific node names or addresses in the left display area as described above. In the right display area, enter each of the filters you want to use by typing them in or by starting NetFilters with the NetFilters button and copying filters from an archive. (See "Using NetFilters to Specify Filters for Other NetVisualyzer Tools" in Chapter 2 for information). By default, `total` and `other` are listed

as filters: `total` means the total for all traffic and `other` means the total for all filters not explicitly listed.

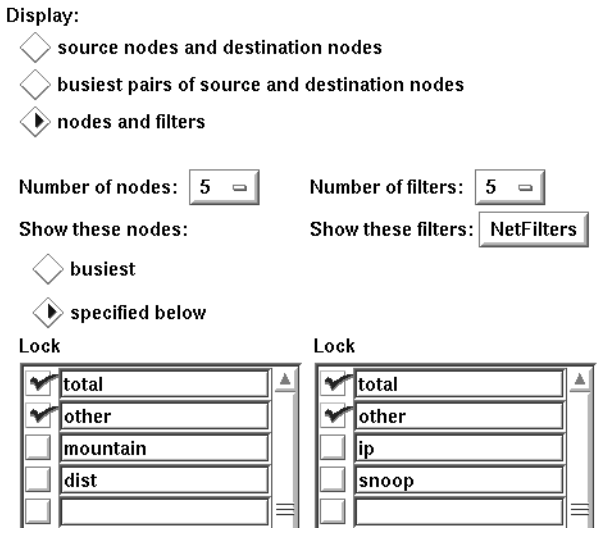



**Figure 6-14** Display Section of Nodes Control Panel (Nodes and Filters)

When you choose the busiest radio buttons, you can use the “Lock” check boxes to specify that you want to continue to see a particular node in the NetTop main window, even if it is not among the busiest nodes.

If you click one of the *grab list from display* buttons shown in Figure 6-15, the current list of nodes or filters in the display area above it is replaced with the nodes or filters currently displayed in the graph in the main window. This feature is useful when you want to fill in the list with the nodes or filters that are currently shown in the NetTop main window. If you lock these nodes and/or filters by checking their “Lock” check boxes, they remain on display even if the calculation of busiest nodes would otherwise make them disappear.


**Figure 6-15** Grab List from Display Button

The line shown in Figure 6-16 enables you to specify how you want to define the term busiest. The busiest nodes have the highest volume of traffic. You can choose to measure the traffic in bytes per second or packets per second with these radio buttons. This line is grayed out when you specify nodes rather than when NetTop displays the busiest nodes.

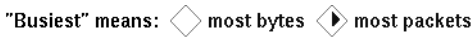



**Figure 6-16** Busiest Definition Radio Buttons

The frequency of evaluating which nodes are the busiest is controlled by the line shown in Figure 6-17. It provides an option button with a list of choices of the number of seconds between calculations. If NetTop is not calculating busiest nodes, this option button is grayed out.

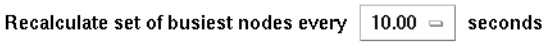



**Figure 6-17** Recalculate Busiest Nodes Option Button

## NetTop File Menu

The File menu in the NetTop main window provides you with these choices:

"Save Controls"
: The current NetTop control panel configuration is saved in the file you last read or wrote with "Save Controls" or "Save Controls As...".

"Save Controls As..."
: A file prompter window appears. Use it to specify the name of a file; your current NetTop control panel configuration is saved to that file. For more information on using this window, see "Using a File Prompter" in the Introduction.

| | |
|---|---|
| "Quit" | A NetTop Question window appears. To save the control panel settings in the file shown in the message and to quit NetTop, click the *Yes* button. To quit without saving configuration information, click the *No* button. If you want to write the information to another file or decide not to quit NetTop, click the *Cancel* button. |

## NetTop Examples

This section provides a few examples and tips for using NetTop.

NetTop complements NetLook and NetGraph. It allows you to determine in real time the top contributors to network traffic like NetLook, but adds the third dimension to provide more analytical data on traffic volume. It also captures the identities of the top sources and destinations for subsequent analysis. NetTop can be used to spot interesting patterns in hosts or protocols dynamically. This information can then be used to tailor graphs in NetGraph. For example, it can be used to tell NetGraph where to look.

### Viewing Low-volume Traffic

In your network you may have traffic patterns where a few nodes predominate: a few nodes generate traffic that is five or more times that of the average node. In this situation, the NetTop display clearly shows the very large towers for the top nodes, but the rest of the nodes appear as very short towers that move almost imperceptibly.

To create a clear display of the low towers, use the Traffic control panel and change the scale of the vertical axis. To determine an appropriate scale, select one of the taller short towers and watch the display at the bottom of the NetTop main window for a short period to get an idea of the maximum height of the short towers. Select the lock maximum radio button and enter this maximum in the entry field.

When the maximum is suited for the smaller towers, they "grow" to be visible, while the large towers "zoom" into the stratosphere. You can now use NetTop to view the average traffic nodes.

## Calculating the Busiest Nodes over Extended Periods

With the Nodes control panel, you specify the frequency that NetTop calculates the busiest nodes on the network. The selections offered range between 10 and 600 seconds because the general use of the tool is for dynamic capture and display.

NetTop can also be initialized to determine the busiest nodes over an extended period, for example hours or days. This can be handy to get a "bigger picture" of the key nodes or connections. NetCollect and NetAccount are useful in determining traffic statistics over longer intervals by source, destination and protocol. NetTop complements these tools by allowing you to specify a filter to ask questions like, "What are the top 5 NFS connections on my network over a 24-hour period?" NetTop can also be set to show the top connections rather than the individual sources and destinations as is reported by NetAccount.

To set NetTop for this mode, start it from the command line with the **–T** option and the number of seconds for the interval that you want to calculate the busiest nodes, for example 86400 for one day. Specify the **–O** option to generate a brief report of the results.

After NetTop begins, make your selections of filter, traffic measured in packets or bytes, number of source and destination, and so on according to your interest. In the Nodes control panel select either the source nodes and destination nodes radio button or the busiest pairs of source and destination nodes radio button. If you select the busiest pairs of source and destination nodes, also select the busiest radio button in the Show these nodes section. NetTop displays traffic as always and also generates a log in the window in which it was started.

An example of the log that results from executing the command:

```
nettop -T 3600 -O
```

with a filter of `nfs` and specifying the busiest five pairs of source and destination nodes is:

```
For the 3600.0 seconds ending Tue Oct 13 17:34:18 PDT 1992,
with filter: nfs
the node pairs transmitting the most packets were:
Source:   192.26.80.119     deepthought.wpd.sgi.com
```

```
Dest:     192.26.75.45      mountain.wpd.sgi.com
37090 packets   5896820 bytes
Source:   192.26.75.45      mountain.wpd.sgi.com
Dest:     192.26.80.119     deepthought.wpd.sgi.com
34046 packets   9426816 bytes
Source:   192.48.200.73     192.48.200.73
Dest:     192.26.75.5       sgi.sgi.com
21997 packets   3676242 bytes
Source:   192.26.75.5       sgi.sgi.com
Dest:     192.48.200.73     192.48.200.73
20438 packets   3208072 bytes
Source:   192.26.75.11      gate-squaw.wpd.sgi.com
Dest:     192.26.75.45      mountain.wpd.sgi.com
```

## Understanding Your Servers

Most networking environments have a few key workstations that function as servers for files, compute cycles, application, mail, or network news. Understanding the ebb and flow of their traffic and their connections to other machines can be important to maintaining an efficient network.

NetTop can be configured to always watch a server as either a source or destination and then to dynamically display its traffic.

To set NetTop to watch a specific node, in the Nodes control panel select the source nodes and destination nodes radio button, and select the specified below radio button in the Show source nodes section. Enter the name of the server in the first open field in the list. Set the Number of sources option button to 3 and the Number of destinations option button to 10.

You are now set to dynamically discover the top destinations for the server. The interval monitored can be set from the Nodes control panel or using the command line options **–T** and **–O** (described in "Calculating the Busiest Nodes over Extended Periods" in this chapter) to permit setting an extended interval.

You can also give a filter to view the server from the perspective of its NFS traffic. In the Traffic control panel enter the filter `nfs`.

Chapter 7

# NetCollect, NetPack, and NetAccount

*NetCollect, NetPack, and NetAccount are command-line tools for collecting, compressing, and displaying statistical information on network traffic data. This chapter explains how to use the tools and provides examples.*

*Chapter 7*

# NetCollect, NetPack, and NetAccount

This chapter describes how to use NetCollect, NetPack, and NetAccount to collect network traffic data and produce reports that show an accounting of network usage. You can collect information such as the total number of packets transmitted by a particular protocol. You can also find the nodes that have transmitted the most packets and bytes.

Because you can use NetCollect to collect data for days, weeks, and months, you will be able to identify the nodes that most heavily use the network as well as identify trends in network use. And because NetCollect offers the potential to collect a large amount of data, you can pack (condense) the data with NetPack so it doesn't use as much disk space. You can produce reports from this data with NetAccount.

By reviewing the reports of network traffic, you can:

- observe patterns of network use over a period of time
- plan for network expansion by locating nodes that cause bottlenecks
- track network use for accounting purposes

In addition to generating reports, you can also display and analyze the reports with a database or spreadsheet such as Wingz™. For information on Wingz, refer to the *Wingz User's Guide*.

This chapter explains how to:

- collect data with NetCollect
- combine data files created by NetCollect with NetPack
- generate traffic reports with NetAccount

In addition, examples are provided. For complete information on NetCollect, NetPack, and NetAccount command line options, see the *netcollect*(1M), *netpack*(1M), and *netaccount*(1M) manual pages in Appendix F, "NetVisualyzer Manual Pages."

**Note:** You must have authorization to use NetCollect. See "Authorizing NetVisualyzer Users for Snooping" in Chapter 1 and Appendix B, "Authorization Reference," for details. ♦

## Using NetCollect to Collect Data

To collect network traffic statistics, click the *netcollect* icon in the *netvis* directory view or enter:

```
netcollect &
```

You can optionally enter the ampersand (`&`) to make the collection process run in the background. When a process runs in the background, that process's identification number (PID) is displayed on the screen. For example:

```
[1] 43258
```

You can use the number, 43258, to stop (by using the *kill*(1) command) the *netcollect* process. Or you can use the *killall*(1M) command. For example:

```
kill 43258
```

or

```
killall netcollect
```

Until you kill *netcollect*, the data is collected from your workstation's default interface (for example, an Ethernet interface, ec0). Data is stored by default in files in a predefined directory structure in the current directory. The directory structure and file names are shown in Figure 7-1.

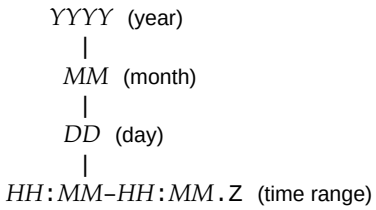



**Figure 7-1** NetCollect Data File Directory Structure

The files in the *DD* directory are named by using the starting and ending time of the sample interval. Hours are specified using a 24-hour clock. The file names end in `.Z`, indicating that they are compressed files.

The default sample interval is 1 hour. The first sample interval is adjusted, if necessary, so that the second and subsequent sample intervals begin at the start of the hour. For example, data collected starting at 10:00 a.m. is stored as:

```
10:00–10:59.Z
```

Data collected beginning at 10:35 a.m. is stored in the file:

```
10:35–10:59.Z
```

If a sample interval is not completed, a file is not generated.

For example, suppose you collect files daily in 1-hour intervals beginning at 9:30 a.m. and stopping at 18:30 (6:30 p.m.). You invoke NetCollect while in the directory named */usr/spool/netvis*. Files produced by NetCollect on June 16, 1992 are stored in the directory named */usr/spool/netvis/1992/06/16*.

A listing of the directory looks like this:

```
09:30-09:59.Z    12:00-12:59.Z    15:00-15:59.Z
10:00-10:59.Z    13:00-13:59.Z    16:00-16:59.Z
11:00-11:59.Z    14:00-14:59.Z    17:00-17:59.Z
```

Notice that there is no file for the period of 6:00 p.m. to 6:30 p.m. No file was created because the sample interval of 6:00 p.m. to 7:00 p.m. wasn't completed.

A sample report later in this chapter shows a portion of the data stored in one of NetCollect's data files. The report was generated by NetAccount. Notice the level of detail and the amount of data in this file. Be aware of the amount of disk space used by the NetCollect files. To free up disk space, it's best to periodically archive and remove the files.

You may want to create a shell script to collect the data and take advantage of *cron*(1M) to start and stop collection, pack the files, or produce reports of the data. Refer to the */usr/people/4Dgifts/examples/netvis* directory for an example of a script you can use to pack files.

The next sections explain how to use the options to NetCollect.

## Collecting Data from Another Interface

To collect data from an interface other than the default, use the **–i** option. For example, suppose you want to collect data from a Data Station that has the additional EFast Ethernet board (fxp0). To do this, type:

```
netcollect -i fxp0 &
```

It's recommended that you collect data locally. However, you may want to capture packets from the default interface on a remote Data Station. For example, to collect data from the default interface of a Data Station named `outback`, type:

```
netcollect -i outback: &
```

Notice that you do not have to type the default interface. If you want to collect data from a remote Data Station's interface that is not the default, you must specify it. For example, assume `outback` has the additional EFast Ethernet board installed. To capture from this interface, type:

```
netcollect -i outback:fxp0 &
```

## Specifying a Different Path

You can also specify a different path to store the data files. To do this, use the **–p** option. For example:

```
netcollect -p /usr/spool/netvis &
```

### Changing the Sampling Interval

You can change the sampling interval by using the **–t** option. For example, to increase the interval to 2 hours, enter:

```
netcollect -t 120 &
```

Because you are collecting large volumes of data, consider using NetPack to combine data from many files into one file.

## Using NetPack to Pack Data

Use NetPack to coalesce data files obtained from NetCollect. When files are packed, all the data from the NetCollect files you specify is added together; therefore, you lose each individual file's time resolution. For example, you may want to pack all the files for a week; thus you will have only one data file for the week. Or you may want to pack a day's worth of files (all the files collected daily between 08:00 and 18:00) into one data file.

To pack files, double-click the *netpack* icon in the *netvis* directory view or give the command using the syntax:

**netpack** *datafile1 datafile2* **...** *datafilen*

where each *datafile* is a file produced by NetCollect. When you double-click the icon, a Launch Command window appears. Complete the command line using the syntax shown above.

The following example packs NetCollect data files produced on June 16 from 09:00 to 18:00. First, change to the directory that contains the files:

```
cd /usr/spool/netvis/1992/06/16
```

You can use the wildcard character (*) to pack all the files:

```
netpack * &
```

or you can specify individual file names:

```
netpack 09:00-09:59 10:00-10:59 11:00-11:59 12:00-12:59 \
13:00-13:59 14:00-14:59 15:00-15:59 16:00-16:59 &
```

Notice that you do not have to specify a file in which to put the packed files. NetPack automatically creates the file for you. When it packs files, NetPack reads each data file in the list and merges the data into a single file. The newly created file is then written to the directory structure used by NetCollect.

By default, the current directory is checked to see if it contains a year directory or is part of a data directory structure. If it finds a data directory structure, that structure is used; otherwise, a directory structure is created in the current directory.

Files packed from the previous NetPack command are put in the directory */usr/spool/netvis/1992/06/16* in the file:

```
09:00-16:59.Z
```

If the file spans 2 days, it is placed in the *MM* directory and is named appropriately. For example:

```
04@17:00-05@08:59.Z
```

If the file spans months, it is placed in the *YYYY* directory. For example:

```
01.01@00:00-12.31@23:59.Z
```

If the file spans years, it is placed in the top-level directory. For example:

```
1990.01.01@00:00-1992.12.31@23:59.Z
```

## Removing the NetCollect Files

The **–r** option allows you to remove the original NetCollect data files after they are packed together. For example:

**`netpack -r`** *datafile1 datafile2*

packs *datafile1* and *datafile2* together and then removes them. Once you delete a NetCollect data file, you cannot retrieve that file's data because that data was merged with the other packed files' data.

### Specifying a Different Directory

The **–p** path option allows you to specify a different directory in which to put the NetPack files. For example:

**`netpack -p /usr/spool/netvis`** *datafile1 datafile2*

## Using NetAccount to Produce an Accounting of Traffic Data

After collecting the data, use NetAccount to generate a report of the network traffic. You can generate the report from files produced by either NetCollect or NetPack.

To generate a report, double-click the *netaccount* icon in the *netvis* directory view or enter:

**`netaccount`** *datafile* **`>`** *reportfile*

where *datafile* is either a NetCollect or NetPack file, and *reportfile* is the file in which to write the report. Output appears on the screen if you do not redirect it (by using the > character) to a file. When you double-click the icon, a Launch Command window appears. Complete the command line using the syntax shown above.

For example, to see the traffic statistics in the file *09:00-16:59.Z* produced by NetPack, change to the */usr/spool/netvis/1992/06/16* directory and type:

```
netaccount 09:00-16:59.Z
```

By default, the report ranks the top five nodes in order of most packets and most bytes, summarizes nodes having 25% or more of the packets of any protocol and nodes having 25% or more of the bytes of any protocol, and produces counts of total bytes and total packets by protocol. By default, a report contains the following six sections:

- Traffic Summary
- Total
- Source Ranking
- Source Summary

- Destination Ranking
- Destination Summary

Each section of the report is described below and a portion of that section of a report is shown. The report used in these examples was based on data that was collected over intervals of two minutes. The NetAccount command with no options was used to create the report.

## Traffic Summary

The Traffic Summary section lists the network segment from which the data was collected and the Data Station that collected the data. The date and time the data was collected is listed, too. For example, the Traffic Summary section is:

```
                ========  Traffic Summary  ========

Network:        b9l-pubs(192.26.79.0)
Collector:      yeti.wpd.sgi.com(192.26.79.6)
From:           Tue Jun 16 09:46:26 1992 PDT
To:             Tue Jun 16 10:29:59 1992 PDT
```

## Total

The Total list shows the total number of packets and bytes transmitted in each protocol during the report period. For example, part of the Total section might look like this:

```
              ----------       Total       ----------

Protocol       Packets                          Bytes

ether           193572                       34470280
arp                 78                           4680
arpip               78                           4680
ip              193494                       34465600
icmp              5815                         549066
igmp               102                           6120
tcp              33071                        9941938
udp             151672                       20120828
```

## Source Ranking

The Source Ranking part of the report is organized by protocol. For each protocol, NetAccount ranks the top source nodes first by the number of packets sent and second by the number of bytes sent. Nodes are listed by name (if known), IP address, and physical address. The next line after each node contains the number of packets sent, the percentage of the packets of that protocol sent by that node, the number of bytes sent, and the percentage of the bytes of that protocol sent by that node. For example, part of the Source Ranking section looks like this:

```
               ----------   Source Ranking    ----------
                               Top 5

ether
  Ranked by packets:
    1. paganini.wpd.sgi.com(192.26.79.1)[8:0:69:2:29:38/SGI]
                    65221 [ 33.69%]               6864078 [ 19.91%]
    2. eno.wpd.sgi.com(192.26.79.24)[8:0:69:6:59:22/SGI]
                    44378 [ 22.93%]               7037132 [ 20.42%]
...

  Ranked by bytes:
    1. eno.wpd.sgi.com(192.26.79.24)[8:0:69:6:59:22/SGI]
                    44378 [ 22.93%]               7037132 [ 20.42%]
    2. paganini.wpd.sgi.com(192.26.79.1)[8:0:69:2:29:38/SGI]
                    65221 [ 33.69%]               6864078 [ 19.91%]
...
```

## Source Summary

The Source Summary list is organized by source nodes. Each source node that accounts for at least 25% percent, by default, of the packets or bytes of any protocol is listed (sorted by address). For each source node, each protocol used is listed. For each protocol, the number of packets sent, the percentage of all packets of that protocol sent by this node, the number of bytes sent, and the percentage of all bytes of that protocol sent by this node is listed. For example, part of the Source Summary section looks like this:

```
               ----------   Source Summary    ----------
               Minimum 25% of protocol packets or bytes
```

```
mantis.wpd.sgi.com(192.26.79.15)[8:0:69:2:29:23/SGI]
  total:
    ether            2662 [  1.38%]              260770 [  0.76%]
    ip               2662 [  1.38%]              260770 [  0.76%]
    icmp             2629 [ 45.21%]              257246 [ 46.85%]
    igmp                1 [  0.98%]                  60 [  0.98%]
    udp                32 [  0.02%]                3464 [  0.02%]
    tsp                11 [  1.92%]                1298 [  1.92%]

eno.wpd.sgi.com(192.26.79.24)[8:0:69:6:59:22/SGI]
  total:
    ether           44378 [ 22.93%]             7037132 [ 20.42%]
    arp                 1 [  1.28%]                  60 [  1.28%]
    arpip               1 [  1.28%]                  60 [  1.28%]
    ip              44377 [ 22.93%]             7037072 [ 20.42%]
    icmp               11 [  0.19%]                 682 [  0.12%]
    tcp              9650 [ 29.18%]             2574358 [ 25.89%]
    udp             34716 [ 22.89%]             4462032 [ 22.18%]
    dns                 4 [  0.56%]                 296 [  0.35%]
    sunrpc             25 [  0.12%]                4298 [  0.09%]
    nfs                 9 [  0.04%]                1322 [  0.02%]
    pmap                8 [  3.33%]                 784 [  2.97%]
    telnet              1 [100.00%]                  60 [100.00%]
    tsp                11 [  1.92%]                1298 [  1.92%]
    rcp              3058 [ 51.80%]             2176609 [ 49.41%]
```

## Destination Ranking

The Destination Ranking part of the report is organized by protocol. For each protocol, the top destination nodes are ranked first by the number of packets received and second by the number of bytes received. Nodes are listed by name (if known), IP address, and physical address. The next line after each node contains the number of packets received, the percentage of the packets of that protocol received by that node, the number of bytes received, and the percentage of the bytes of that protocol received by that node. For example, part of the Destination Ranking section looks like this:

```
---------- Destination Ranking ----------
                    Top 5

ether
  Ranked by packets:
    1. paganini.wpd.sgi.com(192.26.79.1)[8:0:69:2:29:38/SGI]
```

```
                    64679 [ 33.41%]                 8252867 [ 23.94%]
    2. eno.wpd.sgi.com(192.26.79.24)[8:0:69:6:59:22/SGI]
                    43272 [ 22.35%]                 6137764 [ 17.81%]
...

  Ranked by bytes:
    1. paganini.wpd.sgi.com(192.26.79.1)[8:0:69:2:29:38/SGI]
                    64679 [ 33.41%]                 8252867 [ 23.94%]
    2. eno.wpd.sgi.com(192.26.79.24)[8:0:69:6:59:22/SGI]
                    43272 [ 22.35%]                 6137764 [ 17.81%]
...
```

## Destination Summary

The Destination Summary list is organized by destination nodes. Each destination node that accounts for at least 25% percent, by default, of the packets or bytes of any protocol is listed (sorted by address). For each destination node, each protocol used is listed. For each protocol, the number of packets received, the percentage of all packets of that protocol received by this node, the number of bytes received, and the percentage of all bytes of that protocol received by this node is listed. For example, part of the Destination Summary section looks like this:

```
                ---------- Destination Summary ----------
                Minimum 25% of protocol packets or bytes

paganini.wpd.sgi.com(192.26.79.1)[8:0:69:2:29:38/SGI]
  total:
    ether           64679 [ 33.41%]                 8252867 [ 23.94%]
    ip              64679 [ 33.43%]                 8252867 [ 23.95%]
    icmp              281 [  4.83%]                   17422 [  3.17%]
    udp             64398 [ 42.46%]                 8235445 [ 40.93%]
    dns               211 [ 29.63%]                   35275 [ 41.68%]
    sunrpc              8 [  0.04%]                     608 [  0.01%]
    pmap                2 [  0.83%]                     140 [  0.53%]
    tsp               280 [ 48.78%]                   33040 [ 48.78%]

calliope.wpd.sgi.com(192.26.79.9)[2:cf:1f:11:39:75/CMC]
  total:
    ether            3995 [  2.06%]                  468118 [  1.36%]
    ip               3995 [  2.06%]                  468118 [  1.36%]
    icmp             2629 [ 45.21%]                  257246 [ 46.85%]
    udp              1366 [  0.90%]                  210872 [  1.05%]
```

```
sunrpc          1332 [  6.15%]          207504 [  4.28%]
nfs             1326 [  5.52%]          206844 [  2.40%]
pmap               3 [  1.25%]             294 [  1.11%]
tsp               11 [  1.92%]            1298 [  1.92%]
```

## NetCollect, NetPack, and NetAccount Examples

The following sections contain examples for NetCollect, NetPack, and NetAccount—planning prior to running NetCollect, producing a report of a specific protocol, collecting data in a distributed environment, and producing verbose reports.

### Planning Disk Space Needs

Because you are collecting large amounts of data (about 120K per hour for NetCollect), you should consider the amount of disk space on your system. Examples of other questions to consider are:

- What time period do you want to use when collecting the data—how many hours (for example, between working hours of 8:00 a.m. to 5:00 p.m. or 24 hours a day), and on which days (for example, 5 working days or 7 days)?
- What sample interval do you want to use when collecting the data—how many minutes' worth of data do you want in each file (the default is 60 minutes)?
- How often do you want to pack the files—every day, week, or month?
- How do you want to pack the files—pack the entire day's files into one file, or pack all files within a specific time interval into one file?
- How long do you want to keep the original files after you pack them?
- How often do you want to produce reports?
- What files do you want to archive?
- When do you want to remove the data, packed data, and report files?

For information on administrative tasks, such as how to determine disk space and how to archive and back up files, see the *Personal System Administration Guide*.

## Producing a Report of a Specific Protocol

The **–p** option of the *netaccount* command allows you to produce a report of a specific protocol's network traffic. The syntax is:

**`netaccount -p`** *protocol datafile*

where *protocol* is the protocol name and *datafile* is a NetCollect or NetPack file. You can specify more than one protocol. For example, to produce a report of NFS and TCP traffic that occurred between 9:00 a.m. and 9:59 a.m., type:

```
netaccount -p nfs -p tcp 09:00-09:59
```

## Using NetCollect, NetPack, and NetAccount in a Distributed Environment

Just as with NetLook, NetGraph, Analyzer, and NetTop, you can take advantage of NetCollect, NetPack, and NetAccount to monitor networks in a distributed environment. To use NetCollect, NetPack, and NetAccount in a distributed environment, connect a Data Station to every network segment. Each Data Station can collect data and generate accounting reports that can reside on the Data Station's disk.

## Producing Verbose Output

To obtain data about remote nodes and ports, use the **–v** (verbose) option of the *netaccount* command. This option is cumulative; for example, **–vv** includes **–v** data and produces additional data. With no **–v** option, sources and destinations that are matched by IP address are shown. The additional information you get with **–v** options is:

**–v** Sources and destinations are matched by physical addresses. This provides different results from matching IP addresses when packets from one host travel through two routers.

**–vv** The sources and destinations are broken down: for each source, show statistics for each destination separately in the Source Summary; for each destination, show statistics for each source separately in the Destination Summary.

**–vvv** Break down the **–vv** information by port number. This shows individual packet transactions.

First, look at a small part of the source summary produced without the **–v** option (or with the **–v** option, they are the same) for a node named `bonnie`.

```
              ----------   Source Summary    ----------
             Minimum 25% of protocol packets or bytes

bonnie.wpd.sgi.com(192.26.61.14)[8:0:69:2:1e:dc/SGI]
  total:
    ether          211494 [ 28.79%]             127948448 [ 34.55%]
    arp                 4 [  0.37%]                   240 [  0.37%]
    arpip               4 [  0.35%]                   240 [  0.35%]
    ip             211490 [ 28.85%]             127948208 [ 34.56%]
    igmp                2 [  0.25%]                   120 [  0.13%]
    tcp             85787 [ 18.00%]               5147943 [  2.31%]
    udp             64957 [ 34.00%]              40140468 [ 64.84%]
    rip               180 [  3.51%]                 11880 [  0.59%]
    sunrpc          64777 [ 25.92%]              40128588 [ 35.10%]
    nfs            109413 [ 60.15%]             108771904 [ 87.53%]
    pmap               12 [  0.30%]                  1008 [  0.28%]
```

In this example, `bonnie` accounted for 211494 `ether` packets, which was 28.79% of all `ether` packets and 12794884 `ether` bytes, which was 34.55% of all `ether` bytes.

With the **–vv** option, the packets are broken out by destination. For example:

```
              ----------   Source Summary    ----------
             Minimum 25% of protocol packets or bytes

bonnie.wpd.sgi.com(192.26.61.14)[8:0:69:2:1e:dc/SGI]
         -> rains.wpd.sgi.com(192.26.61.4)[8:0:69:2:15:88/SGI]
    ether          209743 [ 28.55%] [ 99.17%]    127609232 [ 34.46%] [ 99.73%]
    ip             209743 [ 28.61%] [ 99.17%]    127609232 [ 34.47%] [ 99.74%]
    tcp             85754 [ 17.99%] [ 99.96%]      5145240 [  2.31%] [ 99.95%]
    udp             63292 [ 33.12%] [ 97.44%]     39871284 [ 64.40%] [ 99.33%]
    sunrpc          63292 [ 25.33%] [ 97.71%]     39871284 [ 34.88%] [ 99.36%]
    nfs            109211 [ 60.04%] [ 99.82%]    108736096 [ 87.50%] [ 99.97%]
         -> zoomer.wpd.sgi.com(192.26.61.9)[8:0:69:2:12:fc/SGI]
    ether               4 [  0.00%] [  0.00%]          474 [  0.00%] [  0.00%]
    arp                 1 [  0.09%] [ 25.00%]           60 [  0.09%] [ 25.00%]
    arpip               1 [  0.09%] [ 25.00%]           60 [  0.09%] [ 25.00%]
    ip                  3 [  0.00%] [  0.00%]          414 [  0.00%] [  0.00%]
```

```
    udp                3 [  0.00%] [  0.00%]          414 [  0.00%] [  0.00%]
    sunrpc             3 [  0.00%] [  0.00%]          414 [  0.00%] [  0.00%]
    nfs                2 [  0.00%] [  0.00%]          276 [  0.00%] [  0.00%]
```

In this example, packets from `bonnie` to `rains` accounted for 209743 `ether` packets, which is 28.55% of all `ether` packets and 99.17% of `bonnie`'s `ether` packets.

The **–vvv** option additionally lists all ports that were accessed. The size of the report substantially increases the time it takes to generate the report. For example:

```
                ----------   Source Summary    ----------
                Minimum 25% of protocol packets or bytes

bonnie.wpd.sgi.com(192.26.61.14)[8:0:69:2:1e:dc/SGI]
     2049 -> rains.wpd.sgi.com(192.26.61.4)[8:0:69:2:15:88/SGI].1023
    ether           61490 [  8.37%] [ 29.07%]     39123088 [ 10.56%] [ 30.58%]
    ip              61490 [  8.39%] [ 29.07%]     39123088 [ 10.57%] [ 30.58%]
    udp             61490 [ 32.18%] [ 94.66%]     39123088 [ 63.19%] [ 97.47%]
    sunrpc          61490 [ 24.60%] [ 94.93%]     39123088 [ 34.22%] [ 97.49%]
    nfs             52617 [ 28.93%] [ 48.09%]     32456046 [ 26.12%] [ 29.84%]
     2049 -> zoomer.wpd.sgi.com(192.26.61.9)[8:0:69:2:12:fc/SGI].1023
    ether               3 [  0.00%] [  0.00%]          414 [  0.00%] [  0.00%]
    ip                  3 [  0.00%] [  0.00%]          414 [  0.00%] [  0.00%]
    udp                 3 [  0.00%] [  0.00%]          414 [  0.00%] [  0.00%]
    sunrpc              3 [  0.00%] [  0.00%]          414 [  0.00%] [  0.00%]
    nfs                 2 [  0.00%] [  0.00%]          276 [  0.00%] [  0.00%]
```

In this example, `bonnie` sent 61490 `ether` packets from its port 2049 to `rains`, which is 8.37% of all `ether` packets and 29.07% of `bonnie`'s `ether` packets.

Chapter 8

# NetSnoop

*NetSnoop captures and displays packets. It is the command-line version of Analyzer. This chapter explains its use and provides examples.*

*Chapter 8*

# NetSnoop

NetSnoop captures and displays every packet of information that travels through the network. Analyzer is a visual interface to NetSnoop. You may find that Analyzer is easier to use than NetSnoop, because Analyzer displays a decoded packet in a way that is easy to understand.

If Analyzer is easier to use, then why use NetSnoop? With NetSnoop you can:

- capture packets, save them to a file, and analyze them later (without having to run them first through Analyzer). In a network with high-volume traffic, you can dedicate resources just to data collection.
- analyze data on a terminal that does not have graphics capabilities. You can use a workstation or any terminal in the network to access the traffic information that NetSnoop gathers.

In addition, you can use NetSnoop with IRIX utilities and the shell to:

- keep a complex NetSnoop command in a shell script
- run a NetSnoop script automatically under specified conditions
- format and display the output of NetSnoop in a particular way

This chapter explains how to:

- start and stop NetSnoop
- specify the interface you want NetSnoop to use for snooping
- interpret NetSnoop output
- get statistics on dropped packets and configure NetSnoop for best performance

In addition, a variety of NetSnoop examples are provided. Using NetSnoop to get protocol information that can be used to create filters is explained in "Using NetSnoop to Find Filter Operands" in Chapter 10. For complete information on NetSnoop command line options and resources, see the *netsnoop*(1M) manual page in Appendix F, "NetVisualyzer Manual Pages."

**Note:** You must be superuser to use NetSnoop while direct snooping. To use NetSnoop with RPC snooping, you must be authorized in the /usr/etc/rpc.snoopd.auth file. See "Authorizing NetVisualyzer Users for Snooping" in Chapter 1 and Appendix B, "Authorization Reference," for details. ♦

## Starting and Stopping NetSnoop

You can start NetSnoop from the *netvis* directory view, the command line, or a shell script. For example, to start NetSnoop from the command line give this command as *root*:

```
netsnoop
```

When you first start NetSnoop, it looks for the *.netsnooprc* file in your home directory. You can specify NetSnoop options in the *.netsnooprc* file. If the *~/.netsnooprc* file does not exist and no interface is specified with the **–i** command line option, NetSnoop starts snooping on the default interface on your workstation and captures packets until you stop it. See "NetSnoop Configuration File" in Appendix D for more information about the *.netsnooprc* file.

The command line options to NetSnoop allow you to snoop on an interface you specify, snoop for a specified time period, and collect a specified number of packets. You can also list information about a specific protocol or field of a protocol. The *netsnoop*(1M) manual page in Appendix F describes all of the NetSnoop options.

You can specify NetSnoop options in these ways:

- on the command line, for example:

  **`netsnoop -c 5`**

- in the *.netsnooprc* configuration file, for example:

  **`option -c 5`**

Options on the command line override options in the *.netsnooprc* configuration file.

To stop snooping, generate a keyboard interrupt (for example, **`<Ctrl-C>`**).

## Specifying an Interface to NetSnoop

A useful command line option is **–i** *interface*, which you can use to snoop on a remote Data Station or on a specific interface on your workstation. You do not need to be superuser when you use the **–i** option. The format is:

`netsnoop -i` *station*`:`*ifname*

where *station* is the name or IP address of a Data Station on which you have snooping authorization, and *ifname* is the name of the interface you want to use. Both *station* and *ifname* are optional. Table 8-1 shows the different forms of *station*`:`*ifname*, the type of snooping used, and the permission requirements.

**Table 8-1** *netsnoop* **–i** *station* **:** *ifname* Forms

| *station* **:** *ifname* **Form** | **Type of Snooping** | **Authorization Required** |
|---|---|---|
| *station* **:** *ifname* | RPC snooping is done on *station* using the interface *ifname*. | User must be authorized in */usr/etc/rpc.snoopd.auth* on *station.* |
| *station* | If station is the name or address of the local host (from the Name or Address columns of `netstat -i` output or an IP address), direct snooping is used. If *station* is a remote host, RPC snooping on the default interface of *station* is used. | User must be superuser for direct snooping; user must be authorized in */usr/etc/rpc.snoopd.auth* on *station* for RPC snooping. |
| *station* **:** | RPC snooping is done on the default interface of *station*. | User must be authorized in */usr/etc/rpc.snoopd.auth* on *station.* |
| *ifname* | Direct snooping (on the local host) using the interface *ifname*. | User must be superuser. |

For example:

```
netsnoop -i reddog:fxp0
```

snoops on the additional EFast Ethernet board installed on the Data Station named `reddog`.

If you want to snoop on the default interface of a Data Station you do not need to specify *ifname*. Two examples are:

```
netsnoop -i yeti:
```

```
netsnoop -i yeti
```

In the first example, RPC snooping is used regardless of whether `yeti` is the local host or not. In the second example, direct snooping is used if `yeti` is the local host and RCP snooping is used if `yeti` is a remote host. In all cases, the default interface is used.

When only an interface is given, direct snooping on the local host is used. For example:

```
netsnoop -i fxp0
```

## Interpreting NetSnoop Output

If you specify a filter on the NetSnoop command line, NetSnoop prints an internal, "corrected" version of the filter enclosed in brackets ([]) before it starts capturing so you can verify the filter.

An example of NetSnoop output looks like this:

```
0198:   len  166   time 15:49:27.446
    ether:  src 8:0:69:2:29:38/SGI       dst ff:ff:ff:ff:ff:ff
    ip:     src paganini.wpd.sgi.com     dst 192.26.79.255
    udp:    sport 520 (rip)              dport 520 (rip)
    rip:    cmd RESPONSE

0199:   len   86   time 15:49:27.456
    ether:  src 8:0:69:6:8e:f8/SGI       dst 8:0:69:6:17:88/SGI
    ip:     src whizkid.wpd.sgi.com      dst outland.wpd.sgi.com
    tcp:    sport 6000 (x11)             dport 4984

0200:   len  118   time 15:49:27.476
    ether:  src 8:0:69:6:8e:f8/SGI       dst 8:0:69:6:17:88/SGI
    ip:     src whizkid.wpd.sgi.com      dst outland.wpd.sgi.com
    tcp:    sport 6000 (x11)             dport 4984

0201:   len  150   time 15:49:27.476
    ether:  src 2:cf:1f:11:46:32/CMC     dst 2:cf:1f:11:9:13/CMC
    ip:     src bigsur.wpd.sgi.com       dst magrathea.wpd.sgi.com
    udp:    sport 1023                   dport 2049 (sunrpc)
    sunrpc: xid 151111         direction CALL      credtype AUTH_UNIX
    nfs:    proc LOOKUP

0202:   len  170   time 15:49:27.476
    ether:  src 2:cf:1f:11:9:13/CMC      dst 2:cf:1f:11:46:32/CMC
    ip:     src magrathea.wpd.sgi.com    dst bigsur.wpd.sgi.com
    udp:    sport 2049 (sunrpc)          dport 1023
    sunrpc: xid 151111         direction REPLY     stat MSG_ACCEPTED
    nfs:    proc LOOKUP        status NFS_OK
```

For each packet captured, NetSnoop shows the sequence number of the packet, the length of the packet in bytes, and the reception time.

Next, NetSnoop decodes protocol data. Data is grouped into frames labeled by protocol (in the example, frames are labeled as `ether`, `ip`, `udp`, `rip`, `tcp`, `sunrpc`, and `nfs`).

Each frame is decoded as a sequence of fields. In the previous example, `src`, `dst`, `sport`, and `dport` are a few of the fields in packet 0198. `src` shows the source address, `dst` shows the destination address, `sport` shows the UDP source port, and `dport` shows the UDP destination port. The source's physical address is `8:0:69:2:29:38`.

## Getting Statistics on Dropped Packets from NetSnoop

To get statistics on dropped and captured packets from NetSnoop, use the **–s** option. It lists the number of packets received by the network interface and records how many were subsequently dropped due to kernel resource limits. Keep the following points in mind when interpreting the results:

- When snooping on a specified interface, the statistics you get with the **–s** option often show more packets seen at the interface than are returned. This result stems from the lag between NetSnoop getting all the packets it wants and signaling *snoopd* that it then wants statistics. The statistics may show that packets were dropped, but this is not a problem because the packets were dropped after NetSnoop received all the packets of interest.
- When NetSnoop is snooping locally on the default interface, dropped packets are indicated by gaps in the sequence numbers of the packets captured. The sum of the values for `packets dropped at network interface` and `packets dropped at socket buffer` equals the number of missing sequence numbers.
- When RPC snooping is used, there are no gaps in the packet sequence numbers because sequence numbers are re-numbered on the receiving side, even though packets may have been dropped. When direct snooping is used, packets are not re-numbered.

## Configuring NetSnoop for Best Performance

This section gives recommendations designed to minimize the possibility that NetSnoop will drop packets and to maximize NetSnoop performance. They are conservative recommendations for worst-case network traffic scenarios. They should be used as starting points for adjusting the values to your needs.

NetSnoop command lines with the following options give the best results:

`netsnoop -o` *output* `-b` *buffer_size* `-l` *packet_length simple_filter*

The **–o** option causes the raw packets to be put into the file *output*. There is no decoding or formatting of the data.

*buffer_size* should be less than 200 for networks with traffic that is mostly MTU-size and less than 50 for network traffic made up of small packets. With large memory configurations, this argument can produce dramatic improvements. Leaving it out altogether or using the **–c** option with these values is also satisfactory.

*packet_length* is the number of bytes of a captured packet saved by NetSnoop. It does not include the MTU header (14 bytes for Ethernet and 16 bytes for FDDI). A reasonable length for many cases is about 200 bytes. This number allows NetSnoop to capture all the protocol headers currently supported plus some user data.

Using NetSnoop with direct snooping rather than RPC snooping is the optimal configuration for packet capture because there is no interprocess communication overhead. There is no communication latency due to stopping and getting statistics. RPC snooping on a remote interface gives lower performance than RPC snooping on a local interface.

There are two types of *simple_filters*: a filter that can be translated into the kernel's filter string so that the kernel can do filtering instead of *snoopd*, or an Ethernet address that enables the Ethernet hardware to do the filtering instead of *snoopd*. Examples of *simple_filter* expressions that can be filtered by the kernel are:

```
src == 8:0:69:4:0:16
```

```
ip
```

```
host(8:0:69:1f:0:4)
```

Other NetVisualyzer tools should not be run while running NetSnoop because they will dramatically affect NetSnoop performance. Performance is improved if the Data Station is dedicated to NetVisualyzer, and no applications or services beyond the minimum are running.

## NetSnoop Examples

The following examples show how to use NetSnoop to track an overloaded Ethernet gateway and to monitor remote use of resources.

### Using NetSnoop to Track an Overloaded Ethernet Gateway

This example explains how to troubleshoot a router or gateway that displays performance problems due to an unusually large amount of traffic. To begin, use NetSnoop to track down the node from which the packets originate:

```
netsnoop -c 20 -t 2 -i ec0 -y dst=BROADCAST or \
dst=8:0:69:2:26:11
```

where:

**`dst=BROADCAST or dst=8:0:69:2:26:11`**
: filters all broadcast packets plus packets specifically destined for one of the Ethernet controllers in the gateway (physical address `8:0:69:2:26:11`).

**`-c 5`**
: captures 20 packets that match the filter. All 20 packets are buffered before they are decoded.

**-t 2** limits captures to at most 2 seconds; capture may stop sooner if the 20 packets specified by the **-c** option are received.

**-i ec0** specifies snooping on the interface ec0.

**-y** causes NetSnoop to consult NIS in order to translate IP addresses to node names. The output from this command may contain node names that are not translated. This may be because they are new nodes and therefore unknown to NIS.

A portion of the output looks like this:

```
0002:   len  310   time 16:26:45.437
    ether:  src mountain.wpd.sgi.com     dst 8:0:69:2:26:11/SGI
    ip:     src littlesnoop.wpd.sgi.com  dst yeti.wpd.sgi.com
    tcp:    sport 1036                   dport 639

0003:   len   60   time 16:26:45.635
    ether:  src 8:0:69:6:2c:83/SGI       dst ff:ff:ff:ff:ff:ff
    arp:    op REQUEST
    arpip:  sha 8:0:69:6:2c:83/SGI       spa whizzer.wpd.sgi.com
            tha 0:0:0:0:0:0              tpa ram.wpd.sgi.com

0004:   len   60   time 16:26:46.093
    ether:  src 8:0:69:6:1c:99/SGI       dst 8:0:69:2:26:11/SGI
    ip:     src patton.wpd.sgi.com       dst babylon.wpd.sgi.com
    tcp:    sport 1021                   dport 513 (rlogin)

0005:   len   60   time 16:26:46.301
    ether:  src 8:0:69:6:1c:99/SGI       dst 8:0:69:2:26:11/SGI
    ip:     src patton.wpd.sgi.com       dst babylon.wpd.sgi.com
    tcp:    sport 1021                   dport 513 (rlogin)
```

The `ether: src` lines show the nodes that generate the traffic. In this example the traffic originates from several sources.

You can use a variation of this command to find only the traffic forwarded by the router:

```
netsnoop -c 20 -t 2 -i ec0 -y '(dst=BROADCAST or \
dst=8:0:69:2:26:11) and ip.dst != gateway' >> /tmp/snoop.log
```

The ignored destination, `gateway`, is the node name of the IP router. This filter captures traffic received by the router but not destined for it at the IP layer (and therefore probably destined for a node on the other side). The Ethernet destination (`dst`) is the router's Ethernet address.

## Using NetSnoop to Track Remote Use of Resources

A common source of poor system performance is the use of system resources from a remote system. In many cases, this use of resources is either unnecessary or completely unknown to the user. For example, you can accidentally impact the resources of a remote system by running the *find*(1) command from the root directory while a remote file system is mounted.

The following example uses NetSnoop to compute a histogram of the traffic from an NFS server's clients. The most active client is displayed first. Note the use of *awk*(1) and *sort*(1) in ordering and managing NetSnoop's output.

```
netsnoop -c 100 ip.dst=server and nfs | awk \
'$1 == "ip:" { hist[$3]++ } \
END { for (client in hist) \
printf "%-25.25s %d\n", client, hist[client] }' \
| sort +1 -n -r
```

In this command, *server* is the server's name. The *awk* script looks at NetSnoop output lines that begin with `ip:`. They have this form:

`ip: src` *client* `dst` *server*

The *awk* command interprets the variable $1 as `ip:`, $2 as `src`, and $3 as *client*. Given traffic from three clients, `192.82.172.25`, `squaw`, and `stingray`, the *awk* code builds an associative array, `hist`, indexed by client node name:

```
hist["192.82.172.25"]
hist["squaw"]
hist["stingray"]
```

The *awk* code prints the histogram at the end of input, sorted numerically based on the second field, and displays the output in reverse. The most active client appears first in the sorted output. An example of the output is:

```
squaw.eng.sgi.com          70
stingray.eng.sgi.com       24
192.82.172.25              6
```

## Using NetSnoop for Error Snooping

You can start the following NetSnoop command in a window and use it to collect data for a period of days. A large sample, such as the sample created after days of monitoring for errors, can be useful in tracking down intermittent network problems.

```
netsnoop -vv -x -e any > /usr/tmp/error.trace &
```

This command produces a hexadecimal dump (`-x`) of any packet with an error (`-e any`). An example of the output looks like this:

```
0000:!  len  100   time 15:25:03.840 CHECKSUM
    ether:  src 2:cf:1f:10:41:73/CMC      dst 8:0:69:2:19:58/SGI
            type ip
    ip:     v 4       hl 5      tos 0     len 84    id 21896  off 0
            ttl 60    p udp     sum 0x2e06
            src gate-orange.wpd.sgi.com   dst work.wpd.sgi.com
    udp:    sport 691                     dport 887
            len 64    sum 0xa293
    00042:                                2a d2 0d ce 00 00           *.....
    00048:  00 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
    00064:  00 00 00 00 00 01 00 00 00 0c 74 63 70 6d 75 78  ..........tcpmux
    00080:  20 31 2f 74 63 70 00 00 00 05 31 2f 52 55 b5 aa   1/tcp....1/RU..
    00096:  52 d5 aa 55                                     R..U
```

Chapter 9

# Browser

*Browser enables you to view the Management Information Bases (MIBs) for a node, and, if authorized, change the values of MIB variables. If you are unfamiliar with MIBs, see Appendix E for an introduction before reading this chapter.*

*Chapter 9*

# Browser

Browser enables you to select a node on your network and view and change the contents of one or more Management Information Bases (MIBs) for that node. Browser communicates with a node that you select using Simple Network Management Protocol (SNMP). The node can be a workstation, router, bridge, hub, or gateway—any device that has an IP address and implements the SNMP protocol and agent.

Browser enables you to walk the tree of information represented by the MIBs, the SNMP Containment Tree, and get the values of MIB variables. While you are using Browser, you can save the variable values that you receive to a file. You can set MIB variables if the SNMP and MIB implementations on the node you are browsing allow it and the community string you provide authorizes it.

Several MIB specifications are provided with Browser; you can easily supply additional specifications to Browser so that it can present MIB information for devices not supported by the supplied specifications. The supplied MIB specifications are:

- `mib-2`
- `rmon`
- `cisco`
- `cabletron`

Browser is designed to be used by network managers experienced in managing various devices on the network. This chapter assumes that you are familiar with SNMP management terminology and technology, especially the MIBs for different devices. If you are not familiar with this terminology, “SNMP Management Reference” in Appendix E defines the basic terms and should be read before this chapter.

This chapter explains how to:

- start Browser
- use the Browser main window to specify the node you want to browse and begin navigating the SNMP Containment Tree
- navigate the SNMP Containment Tree to view subtrees, tables, and variables
- get descriptions of variables
- get and set the values of variables
- use the Browser File menu

In addition, an example of using Browser is provided. For complete information on Browser command line options, see the *browser*(1M) manual page in Appendix F, "NetVisualyzer Manual Pages." "Adding a MIB Specification" in Appendix E explains how to provide additional MIB specifications to Browser.

**Note:** To enable Browser to get and set MIB variables on a Silicon Graphics workstation, that workstation must be running the SNMP daemon snmpd(1M), and your Display Station must be authorized in the file /usr/etc/snmpd.auth on that workstation. See "Authorizing Browsing" in Chapter 1 and Appendix B, "Authorization Reference," for details. ♦

**Caution:** With proper authorization, Browser lets you change some MIB variable values on devices you browse. Because MIB variable values can be critical to the operation of a device and your network, do not change values unless you understand the effects of your changes.

## Starting Browser

To start Browser, double-click the *browser* icon in the *netvis* directory view or type:

```
browser
```

The Browser main window appears. An example is shown in Figure 9-1.

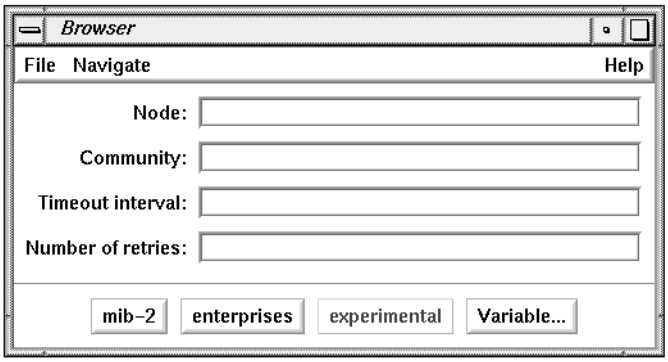



**Figure 9-1** Browser Main Window

## Browser Main Window

The entry fields in the Browser main window enable you to specify the node you wish to browse, a community string, a time-out value for accessing the SNMP agent on the node, and the number of retries to make when attempting to access a remote node.

When you invoke Browser, the entry field shown in Figure 9-2 contains the name of your workstation. You can replace it with the name or address of the node you want to browse. A blank entry field is the same as specifying the name of your workstation.


**Figure 9-2** Node Entry Field

The Community entry field shown in Figure 9-3 contains the community string that is to be used in the SNMP packets sent to the node. The community string is an authorization password for the node you browse on. On Silicon Graphics workstations, valid community strings and other authorization information is specified in the file */usr/etc/snmpd.auth*. The default community string on Silicon Graphics workstations is "public". For other types of nodes, such as routers and bridges, the community string is specified for each device by a system administrator. A valid community string must be supplied in order to use Browser to view MIB information.


**Figure 9-3** Community Entry Field

If Browser doesn't receive a reply from the SNMP agent on the specified node within the time-out value, it will try again. The default time-out value, shown in Figure 9-4, is 5 seconds.


**Figure 9-4** Timeout Interval Entry Field

The Number of retries entry field, shown in Figure 9-5, specifies the number of retries when there has been no reply from the node. The default is 3.


**Figure 9-5** Number of Retries Entry Field

The *mib-2*, *enterprises*, and *experimental* buttons in the Browser main window, shown in Figure 9-6, provide quick ways to specify what you want to browse: the `mib-2` MIB, or the `enterprises` or `experimental` nodes in the SNMP Containment Tree, respectively (see Figure E-1). When you click these buttons, a Subtree window appears. Subtree windows and a second type of window that displays MIB information, Table windows, are described in the next section. These buttons are grayed-out if no MIB specifications in that portion of the SNMP Containment Tree are available to Browser (see "Adding a MIB Specification" in Appendix E for more information).


**Figure 9-6** *mib-2*, *enterprises*, and *experimental* Buttons

When you click the *Variable...* button, shown in Figure 9-7, a Variable window appears. This window is used to get and set the values of specific MIB variables. It is explained in detail in "Getting and Setting Variable Values Using the Variable Window" in this chapter.


**Figure 9-7** *Variable...* Button

## Browser Subtree and Table Windows

To display MIB information, Browser uses two types of windows, Subtree windows and Table windows. For every nonleaf node in the SNMP Containment Tree, Browser displays one of these types of windows:

- a Subtree window showing the subtrees of that node
- a Subtree window showing the variables and/or tables of that node
- a Table window showing the array of table variables in that table

The remainder of this section discusses examples of these windows.

## Subtree Windows that Show Subtrees

Figure 9-8 shows the Subtree window for `mib-2`. It is an example of a Subtree window for a subtree that contains other subtrees.

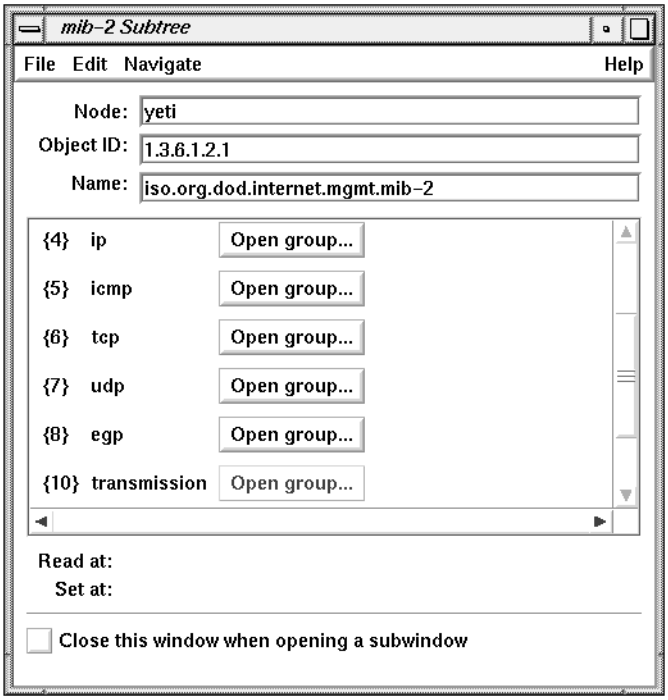



**Figure 9-8** Subtree Window Showing Subtree Objects

The Node entry field, shown in Figure 9-9, contains the node name or address you specified in the Browser main window.


**Figure 9-9** Node Entry Field

The Object ID and Name entry fields, shown in Figure 9-10, contain two different representations of the name of the subtree displayed in the

window. The Object ID entry field contains the numeric representation of the name (dot separated object numbers) and the Name entry field contains the text string representation (dot-separated object names).


**Figure 9-10** Object ID and Name Entry Fields

The scrolling display area in the center of the window contains one line for each object in the subtree, such as the `udp` line shown in Figure 9-11. The line begins with the object's number in curly braces followed by its object name. Clicking the *Open group...* button brings up a Subtree window for the object on this line. Its use is described more fully in "Navigation Using Buttons in the Subtree and Table Windows" in this chapter. A grayed-out button means there are no variables under this object in the MIB.


**Figure 9-11** Object in a Display Area

The Read At line provides status information during a "Get" operation (see "Getting, Setting, and Saving Variable Values" in this chapter), which is replaced by the current time after the operation is completed. Two examples are shown in Figure 9-12.

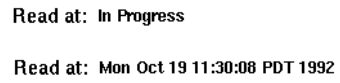



**Figure 9-12** Read At Lines

The current time is displayed on the Set At line after a “Set” operation (see “Getting, Setting, and Saving Variable Values” in this chapter). An example is shown in Figure 9-13.

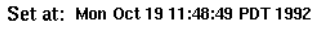



**Figure 9-13** Set At Line

Checking the “Close this window when opening a subwindow” check box, shown in Figure 9-14, specifies that you want this Subtree window to be closed when a new Subtree or Table window for a node in this subtree is opened. By default, each of the Subtree or Table windows you open for subtrees or tables within this subtree will have the same setting.


**Figure 9-14** Close This Window When Opening a Subwindow Check Box

## Subtree Windows that Show Variables and Tables

Figure 9-15 shows the Subtree window for `mib-2.udp`. It is an example of a Subtree window that shows the variables and/or tables of that subtree (in MIB terminology, this type of subtree is called a *group*).

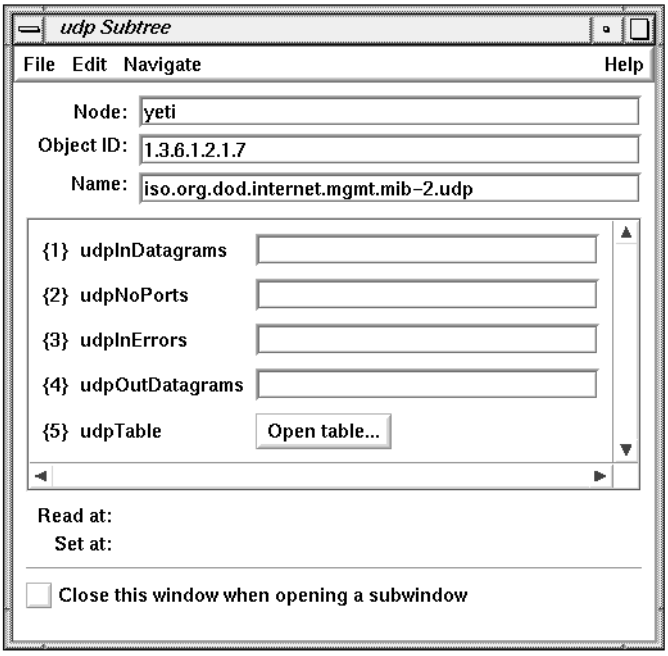



**Figure 9-15** Subtree Window Showing Variables and a Table

Most portions of this type of Subtree window are the same as the Subtree windows described in "Subtree Windows that Show Subtrees" in this chapter. However, the display area of this type of Subtree window contains entry fields and *Open table...* buttons rather than *Open group...* buttons.

Variables in the subtree are shown in the display area as an object number, an object name, and an entry field, as shown in Figure 9-16. When the object number (in braces) is appended to the object ID at the top of the window, it forms the complete object ID for the object. The entry field is gray for variables whose values are defined as read-only in the MIB and pink for variables that are defined as read-write or write-only in the MIB. If the entry field is pink, you can set the value of that variable (see "Getting and Setting Variable Values Using the Edit Menu of a Subtree Window" in this chapter).


**Figure 9-16** Variable Line in a Subtree Display Area

Tables in the subtree have lines that include their object number, their name, and the *Open table...* button, as shown in Figure 9-17. When you click an *Open table...* button, a Table window, described in the next section, appears.


**Figure 9-17** Table Line in a Subtree Display Area

### Table Windows

Figure 9-18 shows the default Table window for `mib-2.udp.udpTable`. When a Table window appears, the display area contains only the names of the table variables. Entry fields appear for the variables as you retrieve their values with "Get next row" in the Edit menu (see "Getting and Setting Variable Values Using the Edit Menu of a Table Window" in this chapter).

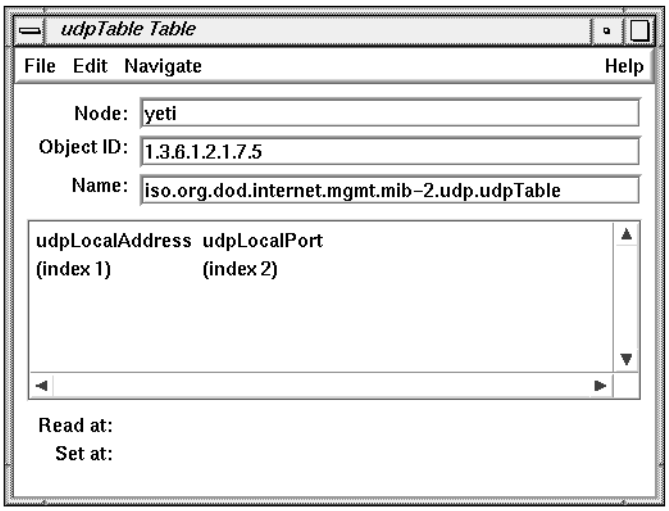



**Figure 9-18** Table Window

## Navigating the SNMP Containment Tree

With Browser you can open a Subtree window for each subtree in a MIB you want to browse and a Table window for each table in a MIB you want to browse. Browser buttons and menus enable you to specify the subtree or table you want to view. When you use these buttons and menus, you are "navigating" the MIB Tree. The three navigation methods are described in the following sections.

## Navigation Using the *mib-2*, *enterprises*, and *experimental* Buttons in the Main Window

The *mib-2*, *enterprises*, and *experimental* buttons in the Browser main window provide three starting points for browsing the SNMP Containment Tree. Clicking the *mib-2* button brings up a Subtree window for the MIB-II MIB. Clicking the *enterprises* and *experimental* buttons brings up Subtree windows for the Enterprises and Experimental subtrees respectively.

## Navigation Using the Navigate Menu

To use the Navigate menu from any window, follow these steps:

1. Press the left mouse button on Navigate in the menu bar.

   In the menu that appears, each choice except the last is the name of a subtree or table that is an object in the subtree in the window. Choices are highlighted as you move the cursor on them; if they have a rollover menu, it appears automatically. Figure 9-19 shows an example of the Navigate menu at the `mib-2` subtree with the cursor on `udp`.

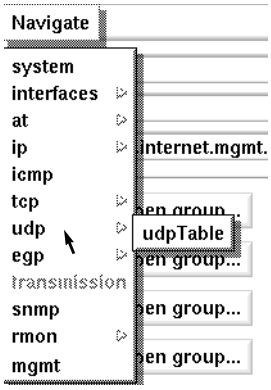



**Figure 9-19** Navigate Menu

2. To view one of the subtrees of the subtree in the window, select one of the choices on the Navigate menu (not on a rollover menu).

   If you make the subtree selection shown in Figure 9-19, the window shown in Figure 9-15 appears.

3. To view subtrees or tables farther down in the hierarchy, move the cursor to a choice on a rollover menu and release the mouse button. In this way you can traverse the entire breadth and depth of the subtree in the window.

4. To view the parent of the current subtree, select the last choice on the Navigate menu. It is the name of the parent of the subtree in the window.

## Navigation Using Buttons in the Subtree and Table Windows

Figure 9-8 shows an example of *Open group...* buttons in the display area of the `mib-2` Subtree window. When you click one of these buttons, a new Subtree window appears for this object. It is equivalent to choosing this subtree from the Navigate menu.

In Figure 9-15, `udpTable` is a table and has an *Open table...* button. Clicking an *Open table...* button is equivalent to choosing the table from the Navigate menu. Figure 9-18 shows the Table window for `udpTable` that appears when you click this button.

## Getting Descriptions of Variables

To get a description of each of the objects in a subtree or each of the variables in a table, select "Description" from the Help menu of the Subtree or Table window. A Description window appears. Figure 9-20 shows an example.

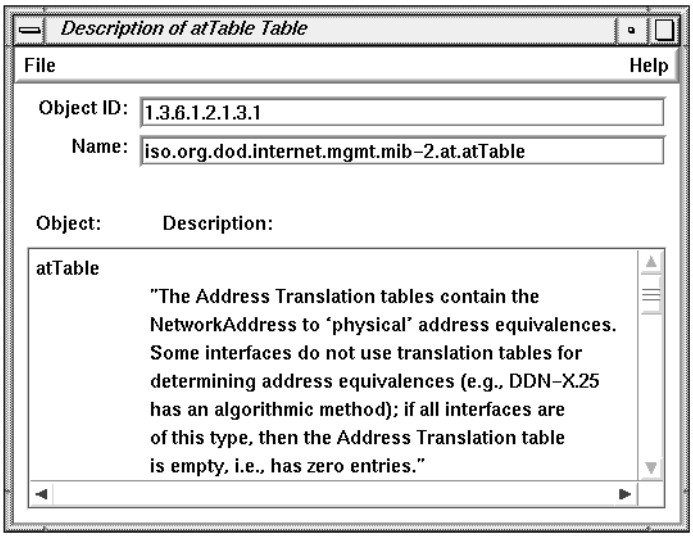



**Figure 9-20** Description Window

## Getting, Setting, and Saving Variable Values

Browser enables you to get the values of MIB variables and set them if you have write access. (Write access is determined by the type of the variable and by your community. See "Authorizing Browsing" in Chapter 1 and "SNMP Management Reference" in Appendix E) Three types of Browser windows can be used to get and set variables:

- The Variable window enables you to get and set individual variables. The Variable window is described in "Getting and Setting Variable Values Using the Variable Window" in this chapter.

- Subtree windows enable you to get and set variables that aren't part of tables. "Getting and Setting Variable Values Using the Edit Menu of a Subtree Window" in this chapter describes how to do this.
- Table windows enable you to get and set variables that are part of tables. "Getting and Setting Variable Values Using the Edit Menu of a Table Window" in this chapter describes how to do this.

## Getting and Setting Variable Values Using the Variable Window

Follow the steps below to use the Variable window to get and set variable values.

1. Click the *Variable...* button in the Browser main window. The window shown in Figure 9-21 appears.

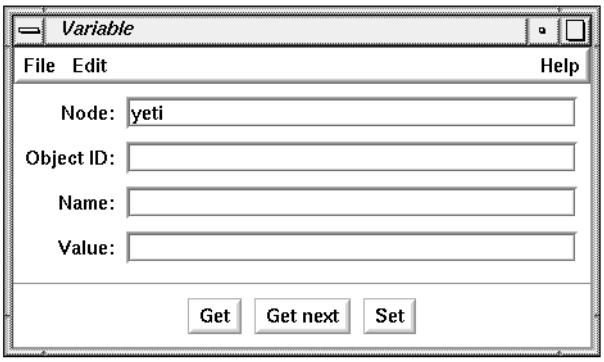



**Figure 9-21** Variable Window

2. If you want to specify a variable by object identifier, fill in the Object ID entry field with the object identifier and append **`.0`** (dot zero). `.0` specifies that you want the value of the object; if you forget to use `.0`, Browser adds it automatically. For example, to specify `mib-2.ip.ipForwarding` (`1.3.6.1.2.1.4.1`), the Object ID entry field should look like the one shown in Figure 9-22.


**Figure 9-22** Object ID Entry Field

3. To specify a variable in a table, enter its object identifier in the Object ID entry field. To construct its object identifier, you can use the object identifier of the table and append ".1.*x*.*y*". *x* is the column number (beginning with 1), and *y* is the value of `index` for the row you want. For example, the object identifier for the `ifDescr` variable (column 2) in the first row (index value of 1) of the `mib-2.interfaces.ifTable` (`1.3.6.1.2.1.2.2`) table is `1.3.6.1.2.1.2.2.1.2.1`. If the table you are using has more than one `index` column, create *y* by specifying each `index` value in order and separating them with periods. For example, if the value of `index1` is `127.1.9` and the value of `index2` is `7`, *y* is `127.1.9.7`.
4. If you want to specify the variable by name, fill in the name in the Name entry field. You need not type in the complete hierarchical name, just the last component of the name. Adding `.0` to the name is optional. If the Object ID and the name you fill in don't match, the Object ID is used.
5. To get the value of the variable, click the *Get* button. The value of the variable appears in the Value entry field. The Name entry field is automatically modified so that it contains the complete hierarchical name.
6. To set the value of a variable, enter the value in the Value entry field and click the *Set* button.
7. To get the value of the next variable, click the *Get next* button. Depth-first search is used to determine the next variable, so the right-most component of the object identifier varies fastest as the tree is traversed with *Get next*.
8. Continue getting and setting variables as necessary by modifying the Object ID, Name, and/or Value entry fields and using the *Get*, *Get next*, and *Set* buttons.

## Getting and Setting Variable Values Using the Edit Menu of a Subtree Window

You can get and set the values of variables from a Subtree window using the Edit menu:

1. Bring up the Subtree window that contains the variable whose value you want to get or set (see "Navigating the SNMP Containment Tree" in this chapter).
2. Select "Get" from the Edit menu to get the values of all of the variables. The current time is displayed on the Read At line.
3. Make changes in the entry fields for any variables whose values you want to change. Only variables whose entry fields are pink may be changed.
4. Select "Set" from the Edit menu to change variable values. The current time is displayed on the Set At line.

## Getting and Setting Variable Values Using the Edit Menu of a Table Window

You can get and set the values of variables in a table from its Table window using the Edit menu:

1. Bring up the Table window for the table you are interested in (see "Navigation Using the Navigate Menu" in this chapter).
2. To get the first row of variables in the table, select "Get next row" from the Edit menu. The current time is displayed on the Read At line.
3. To get other rows for the table, select "Get next row" from the Edit menu as many times as necessary.
4. Make changes in the entry fields for any variables whose values you want to change.
5. Select "Set" from the Edit menu to change variable values. The current time is displayed on the Set At line.

## Browser File Menu

The File menu in each window gives you one or more of the window management and quitting choices listed below.

"Save MIB Values"
: Save MIB values for this subtree for all nodes in the subtree that have open windows to a file. The values are appended to the file that you last specified with "Save MIB Values As...".

"Save MIB Values As..."
: Save MIB values for this subtree for all nodes in the subtree that have open windows to a file. When you select this choice, a file prompter appears. Use it to specify a file name (see "Using a File Prompter" in the Introduction). They are sorted by object identifier.

"Pop Main Window"
: Display the Browser main window. This is useful if you have many windows open and want to locate the Browser main window quickly.

"Close Lower Level Windows"
: Close the windows for all subtrees and tables below the subtree in this window. (To close windows automatically as new windows are opened, see the discussion of the check box "Close this window when opening a subwindow" in "Subtree Windows that Show Subtrees" in this chapter.)

"Close"
: Close this window.

"Quit"
: Quit Browser (available only from the File menu in the Browser main window).

## Browser Example

This section contains an example Browser session on a network that contains a Cisco router with IP address `192.26.51.27`.

To begin this session, invoke Browser with this command:

```
browser
```

The Browser main window is placed on the screen. To browse the MIB for the Cisco router, first fill in the entry fields in the Browser main window:

Node — Enter the Cisco router's IP address, `192.26.51.27`.

Community — Enter the community string your workstation is authorized to use. In this example, the string `public` is used.

Time-out interval — Use the default value, 5 seconds, for the time-out interval. If the Browser doesn't receive a reply from the Cisco SNMP agent in 5 seconds, it will try again.

Number of retries — Use the default number of retries, which is 3. This entry field specifies the number of times that Browser tries again if no reply is received from the Cisco agent within the time-out interval.

Figure 9-23 shows the Browser main window after you've filled in the entry fields.

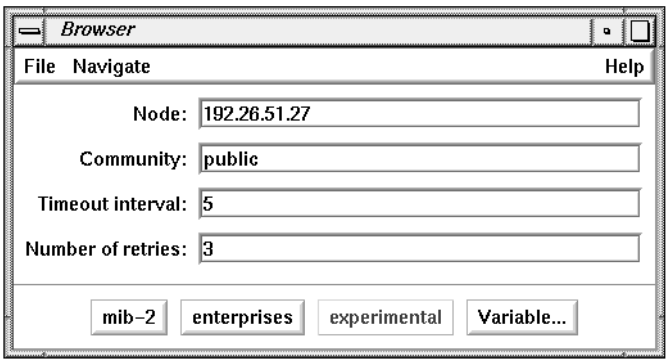



**Figure 9-23** Example Browser Main Window

Suppose you want to get the values for the `lsystem` group in the Cisco MIB. To display the variables in this group, use the Navigate rollover menus to navigate through the MIB hierarchy to `lsystem`, as shown in Figure 9-24.

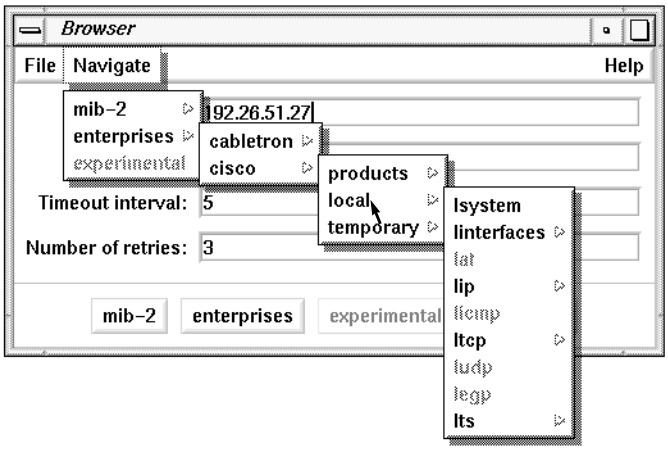



**Figure 9-24** Navigate Rollover Menus for `cisco.local.lsystem`

When you release the mouse button, the Subtree window shown in Figure 9-25 appears.

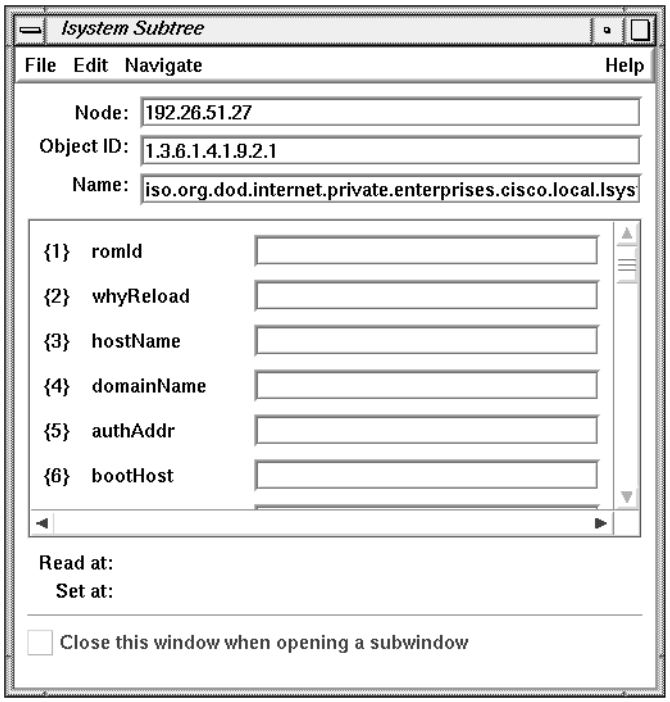



**Figure 9-25** Subtree Window for `cisco.local.lsystem`

To get the values for these variables, select “Get” from the Edit menu. The entry fields for the variables are filled in with their current values, and the current time is indicated at the bottom of the window. Figure 9-26 shows an example.

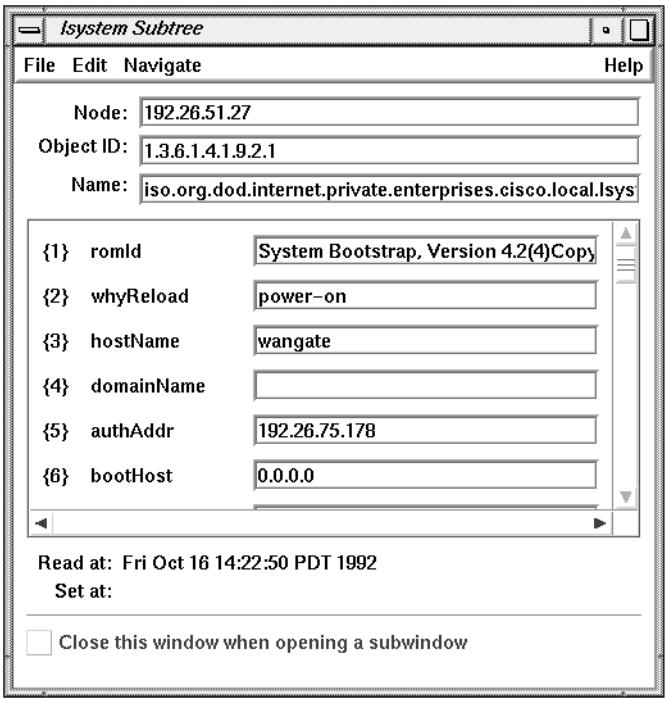



**Figure 9-26** Subtree Window with Values for `cisco.local.lsystem`

Use the right scroll bar to adjust the display area so that you can examine the values of the variables that don’t fit in the default-size display area.

Chapter 10

# Creating and Using Filters

*This chapter explains the syntax of filters and the operators and operands you can use. It also provides many examples of useful filters.*

*Chapter 10*

# Creating and Using Filters

This chapter describes filter expressions, explains how to build them and use them with NetVisualyzer tools, and gives examples of various types of filters.

Filters enable you to focus the NetVisualyzer tools; they allow you to "zoom and pan" around your network dynamically. You can shift your attention to specific nodes or connections between a pair of nodes. You can zoom your view from all packets of all protocols to a specific protocol, and then to certain packet types of that protocol.

A filter screens out certain types of data and allows only the data you specify to be captured. Instead of capturing all network packets, you capture only selected packets. For example, you can use a filter expression to capture only the packets between two nodes or to capture only NFS packets.

When you specify a filter, captured packets are compared to the filter; if the packet matches the filter, the packet is stored. If the packet does not match the filter, it is not stored.

As your understanding of the problem being analyzed increases, you can modify the filters to gather an increasingly refined view of the network. Filters thus enable a highly interactive and iterative approach to planning or troubleshooting your network. The filter selects the subset of the network and its traffic that you want to view, graph, or capture.

This chapter explains:

- what a filter is
- the syntax of filters
- how to find protocol-specific information that can be used in filters
- how to use filters with NetVisualyzer tools

In addition, many example filters are described in this chapter. Additional information on the components of filter expressions is available in the *netsnoop*(1M) manual page in Appendix F, "NetVisualyzer Manual Pages.". The filter file supplied with NetVisualyzer contains many additional examples of filters. Use NetFilters to view this file.

This chapter assumes that you are familiar with the network protocols that NetVisualyzer supports; if not, refer to Appendix C, "Protocols." A basic understanding of the C programming language is also helpful. For details on the C programming language, refer to *The C Programming Language* by Brian Kernighan and Dennis Ritchie.

## What Is a Filter?

A filter is an expression formed from protocol packet fields, network device names, and logical operators. The filter expression determines which subset of all network traffic is to be captured and processed by the tools. Some examples of filters and their semantics are:

`ether` Select all packets on the Ethernet.

`ip` Select only IP packets.

`ip.dst = squaw`
Select IP packets with the destination host name `squaw`.

All of the protocol header fields for the protocols decoded by NetVisualyzer can be used as operands in the filter.

Operands can also be constants or macros. A set of predefined constants are provided for ease of use. Some examples of constants and their values are:

| | | |
|---|---|---|
| `BROADCAST` | `0xffffffff` | Ethernet broadcast address |
| `ACK` | `0x10` | TCP acknowledge |

Macros are defined for the supported protocols to allow more convenient reference to commonly requested protocol strings or relations. For example:

```
nfs
```

is a macro for

```
ip.udp.sunrpc.nfs
```

In this example, the macro `nfs` is a shorthand for the complete protocol description for the NFS layer, `ip.udp.sunrpc.nfs`. The complete protocol description, except for the physical layer, is required for NetVisualyzer to correctly determine the scope of operands. The `nfs` macro can be used alone as a filter to select only NFS traffic, or it can be used as a component in a larger expression.

An example of a macro that takes one parameter is:

```
host
```

This macro is a shorthand for

```
src == $1 || dst == $1
```

and is defined for several protocols. To use this macro, you must preface it with the protocol you are interested in and give an argument ($1). For example,

**`ip.host(yosemite)`**

All IP packets whose source or destination is the node `yosemite` match this filter.

The operators supported are a subset of the C programming language relational operators. Arbitrarily complex expressions can be constructed and passed to the tools. The syntax of filter expressions is explained in the next section.

## Filter Syntax

A filter is a logical expression (sometimes referred to as a Boolean expression). The expression consists of *operands* joined by *operators*.

### Operands

*Operands* are subexpressions, path expressions, C integer constants, or protocol-specific strings. NetVisualyzer has many one-word macros and protocol-specific strings that you can use as simple filters. You can also, if necessary, define your own macros. The subsections that follow give brief descriptions of each type of operand you can use.

#### Subexpressions

A subexpression is an expression such as `ip.src == yeti` that is joined to another expression with an operator. An example is:

```
ip.src == yeti and ip.dst == squaw
```

#### Macros

For each protocol, macros are defined to provide names for magic numbers, shorthands for common lengthy expressions, and nicknames for long protocol path expressions. For example, the Ethernet protocol defines `BROADCAST` as the Ethernet broadcast address `ff:ff:ff:ff:ff:ff`, and IP defines `nfs` as a nickname for `ip.udp.sunrpc.nfs`. Macros can have arguments. They follow the macro name, either separated by white space or in a parenthesized, comma-separated list containing arbitrary white space.

#### Path Expressions

A path expression is a period-separated sequence of components, for example, `ip.udp.sunrpc.nfs`. All but the last component must be legally-formed C identifiers. Each identifier except the last must name a protocol encapsulated by the preceding component's protocol or a structured field in the last protocol. The first identifier in a path names a field in the network interface's data link protocol or a network protocol encapsulated by the data link protocol. Supported data link protocols

include Ethernet, FDDI, Serial Line IP, Token Ring, and the Loopback pseudo-protocol. The last identifier can name a protocol macro or be a protocol-specific string such as the name of a well-known port.

#### C Integer Constants

Integers, such as port numbers, can be used as operands in filters. Hex and octal integers (0x*nnn* and 0*nnn*, respectively) can be used as well as decimal integers.

#### Protocol-specific Strings

Protocol-specific strings are described in "Finding Protocol-specific Operands" in this chapter.

#### Macro Definitions

Macros can be defined for NetSnoop only. Macros are defined in NetSnoop configuration (*.netsnooprc*) files. To define a macro, use this format:

`define(`*name*`, `*def*`)`

Strings of the form $*n* within *def* represent formal arguments to *name*. Each such formal argument is replaced by the *n*th actual argument supplied when *name* is called. The number of formal and actual arguments must agree.

## Operators

An *operator* specifies an operation to be performed, such as addition or subtraction. All C operators except the assignment operators and ?: are supported. In addition, these simplifications are provided:

- You can use the keywords *and* and *or* in place of && and `||`, respectively.
- You can use a single = in place of ==.

- In some instances, you can omit == (=) and *and* (&&) operators from filter expressions. For example:

  ```
  nfs ip.dst alpine
  ```

  is equivalent to:

  ```
  nfs and ip.dst == alpine
  ```

Other characteristics of filter syntax are:

- You must precede the subtraction operator (–) with white space; otherwise, it will be taken as part of a protocol-specific string, for example, an IP node name such as `gate-firefly`.
- If you use a special character in a filter expression that has meaning to the shell (such as !, >, or &), enclose the expression in quotation marks when you use it in a command line.
- The maximum length of filters used in the Filter entry fields of NetLook, NetGraph, Analyzer, NetFilters, NetTop, and in files read by these programs is 255 characters.
- When you use a filter expression on the command line (for example, with *netsnoop*), the length is restricted only by command line limitations.

Table 10-1 gives the list of operators you can use in filters. They are listed in order of highest to lowest precedence. Unless otherwise noted in the table, operators have left-to-right associativity. Parentheses can be used to specify precedence.

**Table 10-1** Filter Operators

| Operator | Definition | Comments |
|---|---|---|
| [ ] | array element | Must occur as a pair of brackets separated by an expression |
| ! | logical not | Right-to-left associativity |
| ~ | one's complement | Right-to-left associativity |
| – | minus | Right-to-left associativity |
| ++ | increment | Right-to-left associativity |

**Table 10-1** (continued) Filter Operators

| Operator | Definition | Comments |
|---|---|---|
| – – | decrement | Right-to-left associativity |
| * | multiply | |
| / | divide | |
| % | remainder | |
| + | add | |
| – | subtract | Must be preceded by white space |
| << | left shift | |
| >> | right shift | |
| < | less than | |
| <= | less than or equal | |
| > | greater than | |
| => | greater than or equal | |
| ==, = | equal | Can be replaced by a space in some expressions |
| != | not equal | |
| & | bitwise and | |
| ^ | bitwise exclusive or | |
| \| | bitwise or | |
| &&, and | logical and | Can be replaced by a space in some expressions |
| \|\|, or | logical or | |

Several NetVisualyzer tools print a “corrected” version of your filter. This corrected version contains parentheses and operators that were implied in the original version. These corrected versions are printed so that you can verify the filter; they don’t imply that there was an error. If a filter contains a syntax error, an error message is given.

## Finding Protocol-specific Operands

This section explains how to find protocol-specific information to use as operands in filters. For purposes of illustration, examples in this section use the Internet Protocol (IP).

There are two steps to finding protocol-specific information:

1. Understand the protocol layering so that you can determine which protocols you want to capture. This is discussed in “Understanding Protocol Layer Relationships” in this chapter.
2. Use protocol information from NetSnoop or Analyzer to construct filters to capture specific data. Using NetSnoop output is discussed in “Using NetSnoop to Find Filter Operands” in this chapter and using Analyzer output is discussed in “Detail Pane” in Chapter 5.

## Understanding Protocol Layer Relationships

First, take a look at the IP diagram shown in Figure 10-1. (Diagrams for all protocols are in Appendix C, "Protocols.") The diagram illustrates the IP relationship to other network layers when the physical layer is Ethernet.

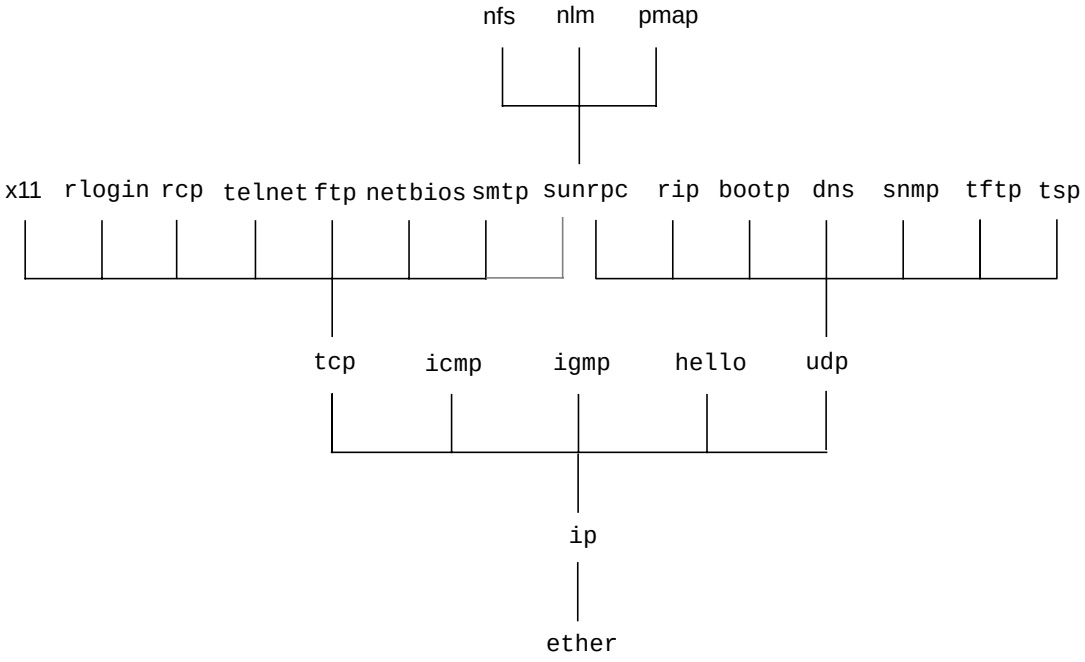



**Figure 10-1** IP Protocol Diagram

Suppose you want to capture packets using the IP protocol. In this case, you can use a simple filter, `ip`. As another example, suppose that you want to capture packets using the TCP protocol. The filter for this protocol is `ip.tcp`. This filter is `ip.tcp` because each layer above the physical layer must be qualified with the protocol it is defined in. The physical layer, such as `ether` or `fddi`, is implied by the type of the interface you are snooping on. You could also use `tcp` as the filter because the `ether` protocol defines `tcp` as a macro whose definition is `ip.tcp`.

Figure 10-1 shows some of the protocols that an IP packet can contain. Assuming that the packet comes over an Ethernet interface, each packet is of the protocol `ether` and `ip`. Many of these packets will also contain other protocols above IP such as UDP or TCP.

To capture NFS packets, refer to Figure 10-1 to see which protocols are used by the NFS packet: `ether`, `ip`, `udp`, `sunrpc`, and `nfs`. You would therefore use `ip.udp.sunrpc.nfs` as the filter. You can also use the filter `nfs` because `nfs` is a predefined macro that expands to `ip.udp.sunrpc.nfs`.

## Using NetSnoop to Find Filter Operands

This section explains how you can use NetSnoop to find NetVisualyzer protocol-specific strings, macros, and so forth. To see these, use *netsnoop* with the **–L** option. The syntax is:

`netsnoop -L [ all | `*protocol*` ...]`

With the **–L** option, NetSnoop lists information about the options and symbols defined by the protocol(s). `all` lists information for all protocols, and *protocol* is a protocol name, such as `ip`. You need not be superuser or have authorization to use NetSnoop when you use the **–L** option.

For example, to list IP information, type:

**`netsnoop -L ip`**

The screen displays lists of fields, protocols, functions, macros, constants, and so forth. An example of the IP information looks similar to this output:.

```
ip (Internet Protocol):

  o Field           type            level    title
    v               u_int:4         -vv      Version
    hl              u_int:4         -vv      Header Length
    tos             u_char          -vv      Type of Service
    len             u_short         -v       Total Length
    id              u_short         -v       Identification
    off             u_short         -v       Fragment Offset
    ttl             u_char          -vv      Time to Live
    p               u_char          -vv      Protocol
    sum             u_short         -vv      Header Checksum
    src             u_long          -        Source Address
```

```
  dst               u_long            -         Destination Address
  opt               u_char            -v        Option Type
  optlen            u_char            -vv       Option Length
  rtoff             u_char            -vv       Route Offset
  rthop             u_long            -         Route Address
  tsptr             u_char            -vv       Timestamp Pointer
  tsoflw            u_int:4           -vv       Timestamp Overflow
  tsflg             u_int:4           -vv       Timestamp Flag
  tstime            u_long            -v        Timestamp Time
  tsaddr            u_long            -         Timestamp Address

o Protocol          typecode  (decimal)
  hello             0x003f    63
  icmp              0x0001    1
  igmp              0x0002    2
  tcp               0x0006    6
  udp               0x0011    17

o Function          description
  badsum            Match packet if header checksum is incorrect

o Macro             definition
  HELLO             hello
  ICMP              icmp
  IGMP              igmp
  NFS               udp.sunrpc.nfs
  TCP               tcp
  UDP               udp
  between           src == $1 && dst == $2 || dst == $1 && src == $2
  host              src == $1 || dst == $1
  nfs               udp.sunrpc.nfs

o Constant          value     (hexadecimal)
  BROADCAST         -1        ffffffff
  DF                16384     4000
  EOL               0         0
  LSRR              131       83
  MAXPRIVPORT       1023      3ff
  MF                8192      2000
  MINHL             20        14
  MINUSERPORT       5001      1389
  MSS               576       240
  NOP               1         1
  RR                7         7
  SATID             136       88
```

```
   SECURITY          130       82
   SSRR              137       89
   TS                68        44

o Option             description
   etherupdate       Update Ethernet hostname/address cache
   hostbyname        Decode IP addresses into hostnames
   hostresorder      Hostname resolution order; see resolver(4)
```

The first column of each section of the output consists of words that you can use in filter expressions. For example, you can create the filter:

```
ip.dst = BROADCAST
```

where `ip` is the protocol name, `dst` is from the Field section, and `BROADCAST` is a Constant.

This output contains Field, Protocol, Function, Macro, Constant sections that are described briefly in Table 10-2. The next subsections explain each section of the output in detail and how you can use the information in them.

**Table 10-2** NetSnoop Protocol Output

| Section | Description |
|---|---|
| Field | Lists all the fields of the protocol from top to bottom in an abbreviated form. |
| `type` | Shows the type of variable; for example, `v` (version) is `u_int:4`, which is an unsigned integer four bits in length. |
| `level` | Lists the level of verbosity (**–v**) you must use with the NetSnoop command for this field to be decoded. |
| `title` | Lists the expanded description of the short form of the field. When entering a field as part of a filter, use the abbreviated name; Analyzer prints both the name and the title. |
| Protocol | Shows all protocols at the level above the current protocol. |
| `typecode` | A value used to identify upper layer protocol types within this protocol. |
| `(decimal)` | Shows the decimal equivalent of the typecode. |
| `Function` | Shows functions defined for the protocol. |

**Table 10-2** (continued) NetSnoop Protocol Output

| Section | Description |
|---|---|
| `description` | Describes each function. |
| Macro | Lists macros defined for the protocol and their expansions. |
| `definition` | Shows the definition of each macro. |
| Address | Defines common addresses such as the broadcast address. For example, `dst=BROADCAST`. |
| Constant | Lists constant values defined for the protocol. For example, MAXPRIVPORT is 1023, a port number defined by IP. Only the superuser can access a port less than or equal to port 1023 (see the example in the "Constants" section below). |
| `value` | Shows the value of the constant as a decimal number. |
| `(hexadecimal)` | Shows the value of the constant as a hexadecimal number. |
| Option | Defines any options you can specify for the protocol. Manipulate protocol options by using the **–p** option to NetSnoop. See the *netsnoop*(1M) man page for its use. Options cannot be used in a filter. |
| `description` | Describes the option. |

### Fields

The Field section lists the abbreviated names of the fields of the protocol. Fields are listed in the order they are decoded by the Analyzer. In the `ip` example, the field listing starts with `v` (for Version) at the top of the listing and ends with `tsaddr` (Timestamp Address) at the bottom.

Other columns list the type of variable of the field, the level of verbosity that you must use with NetSnoop to produce output for the field, and the title (full name) of the field. Table 10-3 lists types of variables and their sizes.

**Table 10-3** Field Types

| Type | Size (in bytes) |
|---|---|
| `void` | 0 |
| `char` | 1 |
| `u_char` | 1 |
| `short` | 2 |
| `u_short` | 2 |
| `int` | 4 |
| `u_int` | 4 |
| `long` | 4 |
| `u_long` | 4 |
| float | 4 |
| `double` | 8 |
| `address` | 8 |

To capture packets for a particular field, precede the field with the protocol name. For example, to capture IP packets destined for a node, use the format:

**`ip.dst=`***nodename*

where `dst` (an abbreviation for *destination address*) is a field defined for IP, and *nodename* is the name of the node. For example, to capture packets destined for a node named `indiana`, use the filter:

```
ip.dst=indiana
```

To capture IP packets whose source is a node named `gary`, use the filter:

```
ip.src=gary
```

You can also substitute IP addresses in place of the node names (for example, `192.26.75.10` as found in the */etc/hosts* file):

```
ip.src=192.26.75.10
```

Note that different protocols use different names for similar fields. For example, if you want to capture the same type of data from TCP, use the destination port and source port specified as `dport` and `sport`, respectively. Give the command **`netsnoop -L tcp`** to see the differences in field names specified for TCP.

### Protocols

The Protocols section lists the protocols that NetVisualyzer decodes above the current protocol. For example, for IP higher-level protocols include the `hello`, `icmp`, `igmp`, `tcp`, and `udp` protocols. The protocol diagrams in "Protocol Layers" in Appendix C give you this same information in diagram form.

### Functions

Some protocols define functions that perform some analysis of the packet and return a Boolean value. An example is `ip.badsum`, which matches IP packets with bad checksums.

### Macros

A macro is a shorthand form of a longer expression. Each NetVisualyzer protocol has predefined macros. You can also define your own macros (see "Macro Definitions" in this chapter and *netsnoop*(1M) in Appendix F). Table 10-4 lists macros defined for IP.

**Table 10-4** IP Macros

| Macro | Definition |
|---|---|
| `HELLO` | `hello` |
| `ICMP` | `icmp` |
| `IGMP` | `igmp` |
| `NFS` | `udp.sunrpc.nfs` |
| `TCP` | `tcp` |
| `UDP` | `udp` |
| `between` | src == $1 && dst == $2 \|\| dst == $1 && src == $2 |
| `host` | src == $1 \|\| dst == $1 |
| `nfs` | `udp.sunrpc.nfs` |

Various protocols can have various definitions for the same macro; for example, the `between` macro captures packets between nodes. As you can see, this macro is defined for `ip`; it is also defined for `ether`, `tcp`, and `udp`. You must precede the macro name with the protocol name for all protocols except the physical layer protocols such as `ether` and `fddi`. For example, to capture IP packets between nodes, use `ip.between`, and then specify the source and destination as IP addresses or node names. If you use this macro with `ether`, the protocol is understood, but you must supply the Ethernet addresses as the arguments. You can use names if they can be mapped to the Ethernet address using */etc/ethers* (or NIS or BIND).

The Macro section has a column that defines how the macro is expanded on execution. For example, the IP `HELLO` macro (see Table 10-4) is expanded to `hello`, the NetVisualyzer name for the HELLO protocol. In this example, a macro enables you to capitalize the protocol name instead of using the all-lowercase format for protocol names used by NetVisualyzer tools.

The IP `NFS` macro is an example of a macro that simplifies specifying protocol layers; it expands to `upd.sunrpc.nfs`. A filter that captures NFS packets using this macro is:

```
ip.NFS
```

rather than the longer form:

```
ip.udp.sunrpc.nfs
```

For another type of example, look at the definition of the host macro:

`host` `src =` *host1* `or dst =` *host1*

The host, *host1*, is either the source (`src`) or destination (`dst`). To capture all IP packets going to or coming from a host named `gary`, you can use the `host` macro by entering:

```
ip.host gary
```

You wouldn't use `ip.host=gary` because `host` is a one-argument macro. Another way to write the filter is `ip.host(gary)`.

If you did not use the `host` macro, you would have to enter:

```
ip.src=gary or ip.dst=gary
```

You can combine components from different sections, such as a component from the Macro section and a component from the Protocol section. For example, to capture only TCP packets to and from `gary`, use the filter:

```
tcp and ip.host gary
```

where `tcp` is a protocol and `host` is a macro.

Next, suppose you want to capture packets between two nodes. To do this, use the filter:

```
ip.src=gary and ip.dst=indiana
```

However, an easier way to do this is to use the `between` macro that captures packets between two nodes. For example:

```
ip.between gary indiana
```

captures only packets between `gary` and `indiana`. Do not use the operator *and* (&&) with the `between` macro. If you are familiar with the C programming language, you may want to use an alternative way of entering this filter:

```
ip.between(gary,indiana)
```

To see most of the macros understood by NetVisualyzer tools, give this NetSnoop command:

```
netsnoop -L ether fddi ip llc
```

### Addresses

Addresses are multibyte strings up to 8 bytes in length. The `ether` protocol uses 6-byte (48-bit) addresses and defines a constant address called `BROADCAST` for the broadcast address `ff:ff:ff:ff:ff:ff`.

### Constants

You can capture packets by using constants in filter expressions. A constant is a string that translates to a predefined value; the constant is often easier to remember than the value. For example, `BROADCAST` is a constant defined for IP. To capture all IP broadcast packets, you can use either the constant or its value. For example, you can enter:

```
ip.dst = BROADCAST
```

or

```
ip.dst = -1
```

Another constant is MAXPRIVPORT, which defines ports 1023 and below as restricted to superuser access. To show the nodes that have logged in to any of the ports restricted to the superuser, use the filter:

```
udp.sport <= ip.MAXPRIVPORT
```

or you can use the value of MAXPRIVPORT, and enter:

```
udp.sport <= 1023
```

You can use some constants such as SECURITY and RR with the opt (option type) field. For example, to show using the IP record route option, use the constant RR:

```
netsnoop -i gary: ip.opt=ip.RR
```

## Using Filters in NetVisualyzer Tools

You can specify filters when you use NetVisualyzer tools in these ways:

- Enter them in Filter entry fields in NetLook, NetGraph, Analyzer, and NetTop to capture only packets of interest to you.
- Use them on the command line of Analyzer, NetCollect, NetGraph, NetLook, NetTop, and NetSnoop to capture only packets of interest.
- Put them in configuration files so that they are read automatically when you use the tools that read these files. See Appendix D, “Configuration File Formats,” for information about configuration files.
- Create a collection of filters with NetFilters so that you have a library of filters for your site.

NetLook, NetGraph, NetTop, and Analyzer each allow the dynamic creation and use of a filter. Each of these tools has one or more Filter entry fields where you can enter a filter. Alternatively, you can call up the NetFilters tool and retrieve one of the filters that has been stored in a filter archive. NetVisualyzer comes with a large set of standard, generally useful filters in the archive. You can adapt the filters in the archive to your local needs and create new ones. NetFilters allows you to easily share the same filter between different tools or to use different, complementary filters in multiple tools.

## Example Filters

This section starts with a simple one-word filter. Other examples in this section describe more complex filters.

### Capturing IP Packets

Suppose that you want to capture only packets using the IP protocol. The filter expression looks like this:

```
ip
```

This simple filter captures every packet that has the IP layer embedded in it. When you use this filter with Analyzer, the Summary window displays:

```
Seq     Time      Len      Source          Port      Destination    Port   Type
  4 09:17:01.61    60 rachel.wpd.sgi.com   1017   eno.wpd.sgi.com   1016   tcp
```

Notice that the Type column, which tells the type of packet captured, shows that a `tcp` packet was captured, not an `ip` packet. This is because the `tcp` packet has `ip` embedded in it. Next, output in the Detail window looks like this:

```
ether          Ethernet
          src        Source Address                  2:cf:1f:11:1:51/CMC
          dst        Destination Address             8:00:69:1:9:7e/SGI
          type       Packet Type                     ip
ip             Internet Protocol
          v          Version                         4
          hl         Header Length                   5
          tos        Type of Service                 0
          len        Total Length                    552
          id         Identification                  61487
          off        Fragment Offset                 0
          ttl        Time to Live                    29
          p          Protocol                        tcp
          sum        Header Checksum                 0xaef3
          src        Source address                  gary.wpd.sgi.com
          dst        Destination Address             192.26.56.11
tcp            Transmission Control Protocol
          sport      Source Port                     514 (shell)
          dport      Destination Port                1023
          seq        Sequence Number                 966,794,335
          ack        Acknowledgement Number          28,226,517
          off        Data Offset                     5
          flags      Flags                           ACK
          win        Window                          24,576
          sum        Checksum                        0xc004
          urp        Urgent Pointer                  0
```

Notice how the output of this window displays the Field section information for each protocol that was captured. This display also uses the full name description (taken from the "title" column of **`netsnoop -L`** output).

## Capturing Only TCP or UDP Packets

To create a filter that captures only TCP or UDP data, use protocol filters. Protocol filters are found in the Protocol section of the output of **`netsnoop -L`**. For TCP and UDP, which are higher level protocols of IP, the *netsnoop* command is **`netsnoop -L ip`**. The output includes the protocols `tcp` and `udp`. To create the filter, enter:

```
ip.tcp or ip.udp
```

The output of Analyzer with this filter shows `ether` and `ip` information (shown in the previous example) and `udp` and `dns` information:

```
...
udp          User Datagram Protocol
         sport      Source Port                    53 (dns)
         dport      Destination Port               53 (dns)
         len        Length                         52
         sum        Checksum                       0z686e
dns          Domain Name System Protocol
         id         Identifier                     43212
         qr         Query/Response Flag            0 (query)
         opcode     Operation Code                 QUERY
         aa         Authoritative Answer           0
         tc         Truncation                     0
...
```

## Capturing TCP or UDP Packets from a Specific Node

Suppose you want to constrict the filter to see TCP or UDP packets coming to or from an IP host named `gary`. Use the protocols `tcp` and `udp` and the `host` macro defined for IP to construct a filter like this:

```
(tcp or udp) and ip.host gary
```

or you can use a longer filter:

```
(tcp or udp) and (ip.src=gary or ip.dst=gary)
```

You must include parentheses to ensure the correct order of evaluation. This time the filter isolated data to and from `gary`. In addition to `ether` and `ip` information, the Detail window lists the `udp` information that was captured:

```
...
udp          User Datagram Protocol
         sport      Source Port                    525 (timed)
         dport      Destination Port               525 (timed)
         len        Length                         84
         sum        Checksum                       0xe28b
```

## Capturing TCP or UDP Packets between Two Specific Nodes

Perhaps you want to capture only TCP or UDP packets between nodes named `gary` and `indiana`. This filter uses the `between` macro defined for IP:

```
(tcp or udp) and ip.between gary indiana
```

You can achieve the same result by using a longer filter:

```
(tcp or udp) and (ip.src=gary and ip.dst=indiana or ip.dst=gary
and ip.src=indiana)
```

## The Snoop Filter

A header field is prepended by IRIX when packets delivered to the snoop socket are received. This header consists of the following data:

- title
- sequence number of the packet
- state flags (including error flags)
- packet length
- reception time

To see a list of header fields and errors you can capture, type:

```
netsnoop -L snoop
```

You can use the words in the Field and Constant sections of the output in filter expressions. For example, to capture packets with a length less than 100 bytes, use the filter:

```
snoop.len < 100
```

## Capturing Errors

Perhaps you want to capture errors on the network. As described in the previous subsection, a list of errors to capture appears in the Field column that you see when you type:

```
netsnoop -L snoop
```

Types of errors include “frame,” “checksum,” “toobig,” “toosmall,” and “nobufs.” You can use these error types in a filter expression. For example, to capture framing errors, use the filter:

```
snoop.frame
```

To capture all errors, enter the filter:

```
snoop.error
```

For a description of flags and errors, see “Capture Options” in Chapter 5.

## Monitoring a Router

Suppose your workstation, with a physical (Ethernet) address of `8:0:69:2:9:2f`, serves as a router on the network. To capture all Ethernet packets to and from this router, use the `host` macro:

```
host 8:0:69:2:9:2f
```

You do not need to precede this macro with the protocol name. If you use a macro to capture `ether` packets, the protocol (`ether`) is understood and you must supply the physical address and not the symbolic name unless the name is in */etc/ethers*.

## Capturing Remote Logins

To capture remote logins (*rlogin*(1)) between two nodes named `buffalo` and `newyork`, use the filter:

```
ip.between buffalo newyork and rlogin
```

## Capturing a String

Use `string` in a filter expression to obtain a string from a protocol. For example, you want to capture the string "hello" and you know it begins at byte 38 in the packet. To do so, use the filter:

```
string(5,38) == "hello"
```

where `(5,38)` specifies the size to capture (5 bytes) and where to begin the capture. `38` is the offset (offsets begin at 0), which specifies the place to start capturing (position 39 in the packet). You can use `\x01` and `\001` to represent hex and octal.

An easy way to determine the offset is to use Analyzer. Highlight the field you wish to see in the Detail pane. The Hex Dump panes hexadecimal characters that correspond to the field also become highlighted. Thus you can determine the offset of a field by counting (from left to right) the number of bytes.

## Capturing Data

Use `fetch` in a filter expression to obtain information from a protocol that is not currently supported or to obtain information that is in the data content portion of the packet.

For example, suppose your site runs a proprietary protocol over UDP. The first byte of your proprietary protocol represents a count of users logged in on the sending system. The UDP header consists of 8 bytes, so the eighth byte from the start of UDP is your count. To capture packets only from systems that your proprietary protocol reports one user is on, enter:

```
udp.fetch(1,8) == 1
```

where `(1,8)` specifies the size to capture (1 byte) and the offset 8 (where to begin the capture), and `1` specifies one user.

Chapter 11

# Using NetVisualyzer in a DECnet Environment

*NetVisualyzer can be used in a DECnet environment. This chapter discusses topics that are specific to DECnet.*

*Chapter 11*

# Using NetVisualyzer in a DECnet Environment

This chapter explains how to use NetVisualyzer tools in a DECnet environment. It describes:

- setting up NetVisualyzer Data and Display Stations
- using 4DDN™ to resolve DECnet addresses and names
- resolving DECnet addresses and names without 4DDN
- dividing a DECnet network
- suppressing the DECnet HELLO message

You can use NetVisualyzer tools (such as NetLook and Analyzer) with DECnet as well as LAT™ protocols.

## Setting Up Stations in a DECnet Environment

Your Display Station and Data Stations use Transmission Control Protocol/Internet Protocol (TCP/IP) as the standard networking software. If you have not already done so, set up the network for your Display Stations and Data Stations as an IP network. For information, see the *IRIX Advanced Site and Server Administration Guide*. For example, some of the tasks you will need to do include obtaining an IP network number for each Display and Data Station and setting up network files such as */etc/ethers* and */etc/hosts*.

## Resolving DECnet Addresses and Names

For NetVisualyzer tools to resolve (map) node addresses to names, you can:

- use 4DDN to resolve addresses/names
- enter each node's address/name manually in the */etc/ethers* file

### Using 4DDN to Resolve DECnet Addresses and Names

4DDN is a Silicon Graphics software option that connects IRIS workstations and servers to a DECnet network. It provides DECnet connection and data transfer service. For more information on 4DDN and the setup requirements, see the *4DDN Network Management Guide*.

Use 4DDN to download a remote DECnet node database to the Display Station or Data Station. NetVisualyzer tools access this database to automatically resolve addresses to the DECnet node names. By using 4DDN, you eliminate the time-consuming process of entering the DECnet addresses and names into the */etc/ethers* database file. For example, with 4DDN installed, type:

```
netsnoop -vv decnet
```

where `-vv` produces verbose output. For example, output looks like this:

```
0000:   len   60   time 16:06:57.311
   ether:  src aa:0:4:0:a8:4     dst aa:0:4:0:28:4
           type decnet
   decnet: size 34   pad 0     flags 0x26=(I-E:1,RTS:0,RQR:0,LFDP:LONG_FORMAT)
           d_area 0            d_subarea 0         d_id 1.40 (SIERRA)
           s_area 0            s_subarea 0         s_id 1.232 (CHEESE)
           nl2 0     visit 0   s_class 0           pt nsp
   nsp:    msgflag LINK_SERVICE          dstaddr 8212         srcaddr 857
           acknum ACK 1        ackdat ACK 1599     segnum 149
           lsflags 0 =(FCVAL-INT:data req cnt, FC MOD: no change)
           fcval 0

0001:  len    60   time 15:58:05.869
   ether:  src aa:0:4:0:28:4 dst ab:0:0:3:0:0/DECnet
           type decnet
   decnet: size 34   pad 0      flags 0xd=(RES:0,TYPE:ENDNODE_HELLO)    ver 2
           eco 0     user_eco 0            id 1.40 (SIERRA)
```

```
        iinfo 0x3           blksize 1498        e_area 0
        seed 00:00:00:00:00:00:00:00   neighbor aa:0:4:0:0:0
        e_timer 15          e_mpd 0
```

The output shows the DECnet source and destination node names, `sierra` and `cheese`, in parentheses after the DECnet addresses:

```
d_area 0            d_subarea 0         d_id 1.40 (SIERRA)
s_area 0            s_subarea 0         s_id 1.232 (CHEESE)
```

These names are displayed by NetVisualyzer tools; for example, the names appear on the perimeter of the network circle when you use NetLook.

Without 4DDN installed, the output looks like this:

```
0000:    len   60   time 16:06:57.311
   ether:  src aa:0:4:0:e8:4/DEC       dst aa:0:4:0:28:4/DEC
           type decnet
   decnet: size 34   pad 0     flags 0x26=(I-E:1,RTS:0,RQR:0,LFDP:LONG_FORMAT)
           d_area 0            d_subarea 0         d_id 1.40 (1.40)
           s_area 0            s_subarea 0         s_id 1.232 (1.232)
           nl2 0     visit 0   s_class 0           pt nsp
   nsp:    msgflag LINK_SERVICE          dstaddr 8212       srcaddr 857
           acknum ACK 1        ackdat ACK 1599     segnum 149
           lsflags 0 =(FCVAL-INT:data req cnt, FC MOD: no change)
           fcval 0

0001:   len   60   time 15:58:05.869
   ether:  src aa:0:4:0:28:4/DEC       dst ab:0:0:3:0:0/DECnet
           type decnet
   decnet: size 34  pad 0      flags 0xd=(RES:0,TYPE:ENDNODE_HELLO)   ver 2
           eco 0    user_eco 0          id 1.40 (1.40)
           iinfo 0x3           blksize 1498        e_area 0
           seed 00:00:00:00:00:00:00:00 neighbor aa:0:4:0:0:0
           e_timer 15          e_mpd 0
```

The packets captured without 4DDN show the source and destination nodes as physical (Ethernet) and DECnet addresses, for example:

```
ether:  src aa:0:4:0:e8:4/DEC       dst aa:0:4:0:28:4/DEC
...
        d_area 0            d_subarea 0         d_id 1.40 (1.40)
        s_area 0            s_subarea 0         s_id 1.232 (1.232)
```

## Entering DECnet Addresses Manually

If you don't have 4DDN, you can still resolve DECnet node names by using NetSnoop and editing the */etc/ethers* file to contain the DECnet addresses and node names. First, obtain DECnet output by typing:

```
netsnoop -vv decnet
```

which displays a node's physical as well as DECnet address. For example, partial output looks like this:

```
ether:  src aa:0:4:0:e8:4/DEC       dst aa:0:4:0:28:4/DEC
        type decnet
decnet: size 34   pad 0     flags 0x26=(I-E:1,RTS:0,RQR:0,LFDP:LONG_FORMAT)
        d_area 0            d_subarea 0       d_id 1.40 (1.40)
        s_area 0            s_subarea 0       s_id 1.232 (1.232)
```

where `aa:0:4:0:e8:4` is the physical address of the source node named `cheese` and `aa:0:4:0:28:4` is the physical address of the destination node named `sierra`.

Next, edit the */etc/ethers* file. Enter one address/name pair per line by using the format:

*x:x:x:x:x:x  node_name*

where *x:x:x:x:x:x* is the physical address and *node_name* is the node name. The DECnet network administrator will be able to supply this name.

For example, an entry in the */etc/ethers* file for a node named `cheese` with physical address `aa:0:4:0:e8:4` and a DECnet address `1.232` (shown in the previous example) looks like this:

```
aa:0:4:0:e8:4 cheese
```

You do not need to include the DECnet address. To add a node named `sierra` with physical address `aa:0:4:0:28:4`, edit the */etc/ethers* file to contain:

```
aa:0:4:0:28:4 sierra
```

You may wonder how a physical address is translated to a DECnet address. Figure 11-1 shows how `cheese`'s physical address (`aa:0:4:0:e8:4`) is translated to its DECnet address (`1.232`).

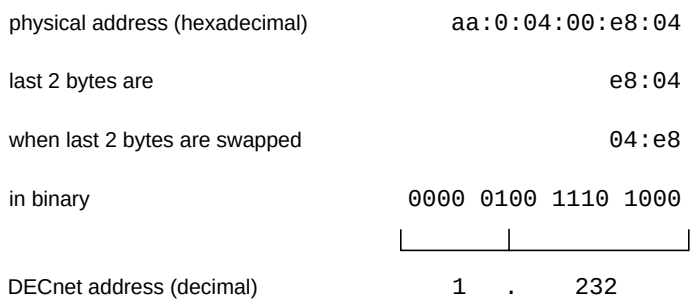



**Figure 11-1** Physical-to-DECnet Address Translation

The last 2 bytes of the physical address is the DECnet address swapped byte for byte. Note that the first 4 bytes of a DECnet address consist of the standard DECnet addressing scheme.

## Dividing a DECnet Network

NetLook typically displays a bridged DECnet network as a large circle, making it difficult to differentiate nodes. To make a DECnet network easier to visualize and monitor, divide it so that each new network circle contains nodes that are grouped by physical segmentation or other criteria. You can do so by editing the NetLook configuration file *network.data*. See "NetLook Network Data File" in Appendix D for more information.

## Suppressing the DECnet HELLO Message

Use the filter `nsp` to suppress the DECnet HELLO messages. It matches all DECnet packets except HELLOs.

# Appendices

*The appendices in this guide are:*

- *Appendix A, "Error Messages"*
- *Appendix B, "Authorization Reference"*
- *Appendix C, "Protocols"*
- *Appendix D, "Configuration File Formats"*
- *Appendix E, "Introduction to MIBs"*
- *Appendix F, "NetVisualyzer Manual Pages"*

*Appendix A*

# Error Messages

This appendix contains information about some of the error messages from NetVisualyzer tools. These error messages are displayed in Warning and Error windows for graphical programs or are written to standard output by nongraphical programs. Additional error messages (not described here) are sometimes written to the file */usr/adm/SYSLOG*.

The first section contains error messages that are common to Analyzer, NetGraph, NetLook, and NetTop. The remaining sections list error messages for individual tools in alphabetical order by tool name.

## Messages Common to Analyzer, NetGraph, NetLook, and NetTop

During the startup of most NetVisualyzer tools, a connection must be established between the Display Station tool and the local or remote Data Station Snoop daemon, *snoopd*(1M) (see "Snooping" in Chapter 1). This section lists problem messages and their remedies in the sequence that they are typically encountered during the connection process.

*DataStation* `name not recognized.`

`Check that` *DataStation* `is a valid name and is entered in the appropriate data base. See Appendix A of the NetVisualyzer User's Guide for detailed help.`

By default NetVisualyzer uses entries in */etc/hosts* to resolve Data Station names into their IP addresses. The name provided was not found in this file or the entry was invalid.

If the Data Station name was entered as an Internet address in the standard dot notation (for example, `191.26.45.75`), that IP address was not reachable on the network. The address may have been keyed in error or a route to the host may not be known.

If the **–y** option was specified on the command line when the tool was started, name resolution will use the network hosts databases, NIS and BIND, if the `hostresorder` resource includes `NIS` and `BIND`. This request failed; check with your system administrator to verify that this node is known to the network's name server.

See also *gethostbyname*(3N), *resolver*(4), and *ypbind*(1M).

`Could not connect to snoopd on` *DataStation*`: Program not registered.`

`Check that snoopd is installed and configured correctly on the Data Station. See Appendix A of the NetVisualyzer User's Guide for detailed help.`

For NetVisualyzer to snoop on a Data Station, it must establish an RPC connection with the Snoop daemon, *snoopd*, on that node. Normally, following the directions in "Enabling Network Snooping" in Chapter 1 is sufficient. This error message indicates that the connection wasn't made. To diagnose the problem, verify the following:

- Data Station software is installed.

  Check that the *snoopd* program is installed on *DataStation* by giving the command:

  **`ls -L /usr/etc/rpc.snoopd`**

  You should see this response:

  `/usr/etc/rpc.snoopd`

  If you do not see this message, repeat the software installation of *netvis_data.sw.data* and *netvis_data.sw.links*.

- The RPC service is configured for snooping.

  Check that the service is known on the Data Station by giving this command as *root*:

  **`rpcinfo -p`** *DataStation*

  You should see a line in the output similar to:

  `391000 1 tcp 1044 sgi_snoopd`

  If you do not see this line, follow the procedure in "Enabling Network Snooping" in Chapter 1.

- The network daemon *inetd*(1M) is configured to initiate snooping.

  Check that this line is present in the file */usr/etc/inetd.conf*:

  ```
  sgi_snoopd/1 stream rpc/tcp wait    root
  /usr/etc/rpc.snoopd     snoopd
  ```

  As *root*, restart *inetd*(1M) by typing:

  **`killall -HUP inetd`**

- The Data Station node is reachable and fully operational. Possible causes for a Data Station not being reachable or operational are listed below under the `Contact lost` message.

  `Could not connect to snoopd on` *DataStation*`: Error in subscribe: No permission match.`

  `Check that snoopd is installed and configured correctly on the Data Station. See Appendix A of the NetVisualyzer User's Guide for detailed help.`

To protect the tools from illegal users, NetVisualyzer allows the superuser to define which users and hosts have permission to use the tools on each Data Station.

You must have been granted authorization in the file */usr/etc/rpc.snoopd.auth* to snoop on a Data Station. See "Authorizing NetVisualyzer Users for Snooping" in Chapter 1 for more information.

`Could not find a valid NetVisualyzer license`

NetVisualyzer requires the user to purchase a license for the total number of concurrent Data Stations that are to accessed from the Display Station. The license is checked by the Display Station software as each tool is invoked. Possible reasons for this message are:

- This software was provided for evaluation and the temporary license has expired. If you wish to purchase NetVisualyzer, contact your sales representative.
- The license file */usr/netls/nodelock* has been corrupted or deleted.

Examine the license file and make sure that the correct password from your software license appears. It may be necessary for the superuser to re-enter the password string.

`NetVisualyzer license limit reached`

The Display Station has attempted to connect to more concurrent Data Stations than it is authorized for. You can either retry later when demand has lessened or consider upgrading the license to permit a larger number of simultaneous users.

`Contact lost with Data Station` *DataStation*`. Check network connection and remote node. See Appendix A of the NetVisualyzer User's Guide for detailed help.`

Typically, this error message results from a problem with either the network or the Data Station node. Either software or hardware failure is possible:

- The network connection to the node may no longer be available or reliable. Possible causes to be investigated are loss of end-to-end network connectivity due to bridge or router failure and connection time-outs due to high network congestion.
- The Snoop daemon on the Data Station is not responding due to system hardware or software failure or *snoopd* software failure.

A first, simple test is to ping the node to determine whether the node responds and if the link is congested.

## Analyzer Messages

The following messages can occur when you are using Analyzer:

`Could not snoop from` *filename*`:` *errmsg.*

Cannot open the specified file for reading.

`No captured data available.`

No snoop data available for searching. Capture some data before using the *Search* button.

`Could not write to` *filename*`:` *errmsg*`.`

An error occurred when writing to the file.

`Unresolved String:` *filter*

Cannot interpret the filter expression.

`Snoop length error.`

Length of data packet to capture is not acceptable.

`Cannot keep` *n* `packets because only` *m* `packets to stop at. Can keep only` *m*`.`

The value in the Stop At entry field is less than the value specified to keep for decoding. Only the smaller number of packets can be decoded.

`Value too big. Only` *n* `packets are kept.`

The number of packets specified to keep for decoding is too big. The bigger default value is used.

`analyzer: Could not start NetFilters:` *errmsg*`.`

NetFilters is not installed or some other error occurred that is described in *errmsg*.

`Could not write to .analyzerrc.`
`Could not write to` *user_input_rcfilename*`.`

No permission to write or the directory does not exist.

`Could not read .analyzerrc.`
`Could not read` *user_input_rcfilename*`.`

No permission to read or the file does not exist.

`Could not transfer filter expression to a non-filter field.`

A filter expression can only be transferred to the Trigger On, Filter, or Stop On entry fields.

`Could not open help file` *filename*`:` *errmsg*`.`

Cannot open the help file listed in the error message.

## Browser Messages

The following messages can occur when you are using Browser:

```
Time expired. Agent did not respond. Check whether the agent
is running on the node; check whether the community name
that you entered in the main window corresponds to what the
agent expects and also whether your host is authorized for
the service you requested on the remote host
```

This message can occur when Browser fails to establish a connection with the SNMP agent on the node you want to browse. Some possible causes are networking problems, incorrect community string, and your host is not authorized to browse on this node.

```
Send Error: Unable to communicate with the remote node
```

The remote node is down or the communication link to that node is down. Check whether the node is up by issuing *ping*(1M) command.

```
MIB tree was not created
```

The MIB database was not created because there was a problem with reading and compiling the MIB specifications in the */usr/lib/netvis/mibs* directory.

```
No such name
```

The specified variable is not found in the MIB. Check whether you are entering the correct object identifier or name for the variable. This error is also caused by trying to write to a read-only object. If you are entering the right name (or object id) and that object has read-write permissions, then that variable is not implemented by the agent.

```
Bad value
```

The requested operation contained incorrect syntax or an incorrect value when trying to modify a variable.

```
SNMP protocol error
```

The SNMP agent's response did not conform to the SNMP protocol specification.

```
Nothing to set in this table yet, do a Get first
```

When a table window comes up, it displays only the columns of the table. To set a particular value in the table, you have to do a Get so that the entry fields for the table appear on the screen.

```
SNMP Generic error
```

Got genErr from the agent.

```
Variable is a Table
```

You probably want to get a value for a variable in this table. Specify the object identifier and an instance for the variable.

```
Variable is a Group
```

You probably want to get a value for a variable in this group. Specify the object identifier and instance for the variable.

```
Variable is in a Table. It requires an instance
```

You have to specify the instance for this variable.

```
Variable name not in the MIB tree
```

You are trying to specify a name for a variable that is not known to the Browser database. Specify the object identifier for the variable.

## NetAccount Messages

The following messages can occur when you are using NetAccount:

`usage: netaccount [-p proto] [-tvy] [-r nrank] [-s nsumm] file`

Bad arguments given.

`netaccount: Could not find a valid NetVisualyzer license.`

No license found.

`netaccount: NetVisualyzer license limit reached.`

Maximum number of data stations active.

`netaccount: unknown protocol.`

A bad value for protocol passed with **–p**.

`netaccount: invalid rank count.`

A bad value for rank passed with **–r**.

`netaccount: invalid summarize count.`

A bad value for summarize passed with **–s**.

`netaccount: error reading` *filename*`:` *message*`.`

Error occured reading the file.

## NetCollect Messages

The following messages can occur when you are using NetCollect:

`usage: netcollect [-h hashsize] [-i interface] [-p path] [-t interval]`

Displayed when a bad argument is given.

`netcollect: couldn't get hostname:` *message*`.`

The name of the host NetCollect is running on could not be obtained.

`netcollect: interval must be > 0 and < 1440.`

A bad value for interval was given.

`netcollect: error getting host information for` *hostname*`:` *message*`.`

The host name passed through a **–i** option could not be found.

`netcollect: interface: Invalid argument.`

A bad value was passed for interface.

`netcollect: error in open:` *message*`.`

Could not open a socket.

`netcollect: error in subscribe: snoopd on` *hostname*`:` *message*`.`

Could not connect to *snoopd*.

`netcollect: error in setsnooplen:` *message*`.`

Could not set the length of packets to capture.

`netcollect: error in getaddr:` *message*`.`

Could not get the address or netmask of the interface being snooped on.

`netcollect: error in compile:` *message*`.`

Could not compile filter.

`netcollect: error creating data directory:` *message*`.`

Could not create the directory to store data files.

`netcollect: error in start:` *message*`.`

Could not turn on snooping.

`netcollect: error in read:` *message*`.`

Error while reading from snoopd.

`netcollect: error in write:` *message*`.`

Could not write data file.

`netcollect: snoopd on` *hostname*`: Error in subscribe:` *message*`.`

The Snoop daemon on *hostname* rejected the subscribe.

## NetFilters Messages

The following messages can occur when you are using NetFilters:

`Couldn't load file` *filename*`.`

Error in reading the filters file. The directory or file doesn't exist.

`No current file name. Use the "Save As" command.`

There is no filters file in use currently. Use the "Save As" menu choice to specify a file name.

`Couldn't save file` *filename*`.`

Error in writing the filters to *filename*. You do not have permission to write to the directory or it doesn't exist.

## NetGraph Messages

The following messages can occur when you are using NetGraph:

```
Could not start snooping
Error in snoop read
```

An error was returned by *snoopd*. It may have lost contact with the Data Station.

```
Could not add promiscuous snoop filter
```

*snoopd* did not accept a promiscuous snoop filter, so NetGraph couldn't run.

```
Cannot continue -- too many errors
```

The content of the NetGraph user interface configuration file (*.netgraphrc*) has more than 10 errors. See "NetGraph User Interface Configuration File" in Appendix D for the correct format, or start NetGraph without a *.netgraphrc* file.

```
Illegal graph specification token
```

An illegal keyword was given in the specification for a graph. Valid keywords include `line`, `bar`, `packets`, `noalarm`, and so on. See "NetGraph User Interface Configuration File" in Appendix D for the correct keyword.

```
Error in opening history file, so can not save history
```

Bad directory name or no permission to write to that directory.

```
Error in opening history file, so can not play back history
```

The file you specified as a history file doesn't exist, isn't a valid NetGraph history file, or is read-protected.

```
History file cannot be specified here
```

You can start NetGraph with the **–i** *history_file* option, but you cannot specify a history file as the interface after NetGraph is already running.

```
When playing back history, cannot simultaneously record
history
```

The data being displayed are coming from a history file; it is not necessary to save it again.

`Could not open alarm log file, so alarms will go to standard output`

The log file specified does not exist or permission denied.

`ControlsFile has too many lines. It probably isn't in the correct format`

A correct user interface configuration file has one line per graph desired, plus a few extra lines. The file you specified has so many lines that it probably isn't a valid user interface configuration file.

`netgraph: Could not start NetFilters:` *errmsg*`.`

NetFilters is not installed or some other error occurred that is described in *errmsg*.

## NetLook Messages

This section lists messages displayed by NetLook. "Messages Common to Analyzer, NetGraph, NetLook, and NetTop" in this chapter lists additional messages from NetLook.

### Progress Message

The NetLook Progress window appears with the following message while NetLook is being initialized at startup:

`Starting NetLook...`

A progress indicator is displayed as the program is started.

### Errors at Startup

The following messages can appear when you attempt to begin snooping. The messages listed in "Messages Common to Analyzer, NetGraph, NetLook, and NetTop" in this chapter can also appear.

`Buffer size must be greater than zero.`

The buffer size resource was set to a value <= 0.

`Interval size must be greater than zero.`

The interval resource was set to a value <= 0.

`Could not get this host's name.`

The name of the Display Station could not be obtained.

`Could not get this host's entry.`

The host table entry for the Display Station could not be obtained.

`Could not open socket.`

A socket could not be created.

`Could not get interface configuration.`

The configuration of the network interface on the Display Station could not be obtained.

## Warnings

NetLook Warning windows contain informational messages. You can close the windows by pressing the *Continue* button.

`Could not create node` *node*`.`

An error occurred when NetLook tried to create a node from the string *node*.

`Could not start snooping on` *node*`:` *errmsg*`.`

The error described in *errmsg* occurred when NetLook tried to start snooping on *node*.

`Could not stop snooping on` *node*`:` *errmsg*`.`

The error described in *errmsg* occurred when NetLook tried to stop snooping on *node*.

`Could not change filter on the following DataStations:`
*node*`:` *errmsg*

The errors described occurred when NetLook tried to change the filter on *node*.

*object* `is hidden.`

The node or network requested in the Find action was hidden.

`Could not find` *object*`.`

No matches could be found for the Find action or the *Hide* button action on *object*.

*object* `is already hidden.`

The object given for a *Hide* button action is already hidden.

*object* `is not hidden.`

The object given for an *Unhide* button action is not hidden.

`Could not start ping:` *errmsg*`.`

The *ping* command, */usr/etc/ping* by default, could not be started.

`Could not start traceroute:` *errmsg*`.`

The *traceroute* command, */usr/etc/traceroute* by default, could not be started. You must be *root* to use */usr/etc/traceroute*.

`netlook: Could not start NetFilters:` *errmsg*`.`

NetFilters is not installed or some other error occurred that is described in *errmsg*.

`Could not open` *datafile*`:` *errmsg*`.`

The error described occurred when NetLook tried to open *datafile*.

`Could not read` *datafile*`:` *errmsg*`.`

The error described occurred when NetLook tried to read *datafile*.

*datafile* `is not a NetLook data file.`

The file *datafile* is not a NetLook network data file.

*datafile*`: line` *line*`:` *parse_error*`.`

The error described occurred while parsing *datafile* at line *line*.

`Could not write` *datafile*`:` *errmsg*`.`

The error described occurred when trying to write *datafile*.

`No object to hide.`

The *Hide* button of the Hide control panel was pressed with no object given or no applicable object selected.

`No hidden objects.`

The *Unhide* button of the Hide control panel was pressed with no hidden objects to unhide.

## Questions

NetLook Question windows appear when a confirmation is required before continuing.

`Save data to` *datafile* `before quitting?`

A prompt suggests that you save network configuration data and user interface configuration data before quitting.

`Are you sure that you want to delete` *object*`?`

A confirmation of the Delete action.

## NetPack Messages

The following messages can occur when you are using NetPack:

`usage: netpack [-p path] [-rv] file1 file2 ...`

Bad arguments given.

`netpack: could not read` *filename*`:` *message*`.`

Error occurred reading the file.

`netpack: could not unlink` *filename*`:` *message*`.`

Error occurred removing a file.

`netpack: could not get current directory:` *message*`.`

Could not find the current directory.

`netpack: could not create data directory:` *message*`.`

Could not create the directory to store data files.

`netpack: could not save` *filename*`:` *message*`.`

Error occurred writing the file.

## NetSnoop Messages

The following messages can occur when you are using NetSnoop:

`netsnoop:` *interface_or_filename*`: No such file or directory.`

An invalid interface was specified using the **–i** option. NetSnoop expects a valid interface name, such as ec0, or the name of a snoop file with the full path if it is not in the current directory.

`netsnoop: cannot snoop to tracefile` *filename*`: Cannot write to the named file.`

Bad directory name or no permission to write to that file.

```
netsnoop: cannot snoop on default interface: Permission
denied.
```

You must be superuser to run NetSnoop when snooping from a network interface.

`netsnoop: cannot snoop on` *interface*`: Not enough space.`

Not enough memory left to buffer snoop data.

## NetTop Messages

The following messages can occur when you are using NetTop:

```
Error in snoop read
Could not delete snoop filter
Could not start snooping
Could not stop snooping
Could not set snooping interval
Error in snoop unsubscribe
```

An error was returned by *snoopd*. Contact with the Data Station may have been lost.

`Error in snoop read - wrong number of bins`

NetTop couldn't run any more because *snoopd* returned bad data.

`nettop: Could not start NetFilters:` *errmsg*`.`

NetFilters is not installed or some other error occurred that is described in *errmsg*.

*Appendix B*

# Authorization Reference

Most NetVisualyzer tools require authorization to use them. Authorization takes several different forms; in some cases you must be superuser, and in other cases authorization files such as */usr/etc/rpc.snoopd.auth* and */usr/etc/snmpd.auth* must contain appropriate entries. This appendix lists the authorization required to use each tool and describes the syntax of */usr/etc/rpc.snoopd.auth* (*/usr/etc/snmpd.auth* is described in the *snmpd*(1M) manual page in Appendix F, "NetVisualyzer Manual Pages").

## Tool Authorization Summary

Table B-1 lists each NetVisualyzer tool and the authorization required to use it. In cases where the authorization required depends upon the command line options given or which feature of the tool is used, each type of line options given or which feature of the tool is used, each type of authorization required is listed.

**Table B-1** Tool Authorization Summary

| Tool | Option | Authorization |
|---|---|---|
| Analyzer | Snoop on local workstation on default interface | User must be superuser or be authorized in */usr/etc/rpc.snoopd.auth* for `netsnoop` service. |
| | **–i** *interface* (snoop on interface) or *interface* specified in Interface entry field on Capture control panel | If RPC snooping is used, user must be authorized in */usr/etc/rpc.snoopd.auth* for `netsnoop` service on the Data Station used for snooping. If direct snooping is used, user must be superuser. See Table 8-1 for more information; it applies to Analyzer as well as to NetSnoop. |
| | **–i** *packet_file* (snoop from file) or *packet_file* specified in Snoop file entry field on Capture control panel | No special authorization is required. |
| Browser | any | The host running Browser must be authorized in */usr/etc/snmpd.auth* on Silicon Graphics nodes browsed; user must specify appropriate community string for all types of nodes. |
| NetAccount | any | No special authorization is required. |
| NetCollect | any | User must be authorized in */usr/etc/rpc.snoopd.auth* for `netlook` service. |
| NetFilters | any | No special authorization is required. |
| NetGraph | any (snoop on any interface or from a file) | User must be authorized in */usr/etc/rpc.snoopd.auth* for `histogram` service on the Data Station used for snooping. |

**Table B-1** (continued) Tool Authorization Summary

| Tool | Option | Authorization |
|---|---|---|
| NetLook | any (snoop on any interface or from a file) | User must be authorized in */usr/etc/rpc.snoopd.auth* for `addrlist` service on each Data Station used for snooping. |
| | Trace Route Action | User must invoke NetLook as *root* if the `traceRouteCommand` resource is set to *traceroute* and *traceroute* is not setuid; requirements above also apply. |
| NetPack | any | No special authorization is required. |
| NetSnoop | **–L** | No special authorization is required. |
| | Snoop on local workstation on default interface | User must be superuser. |
| | **–i** *interface* (snoop on interface) | If RPC snooping is used, user must be authorized in */usr/etc/rpc.snoopd.auth* for `netsnoop` service on the Data Station used for snooping. If direct snooping is used, user must be superuser. See Table 8-1 for more information. |
| | **-i** *packet_file* (snoop from file) | No special authorization is required. |
| NetTop | any (snoop on any interface or from a file) | User must be authorized in */usr/etc/rpc.snoopd.auth* for `addrlist` and `histogram` services on the Data Station used for snooping. |

## /usr/etc/rpc.snoopd.auth

On each Data Station, the file */usr/etc/rpc.snoopd.auth* contains authorization information that specifies which users using which hosts are authorized for *snoopd*(1M) services. You must be superuser (*root*) to read or write */usr/etc/rpc.snoopd.auth*. For security reasons, the owner and permissions of this file should not be changed.

A simple authorization line in */usr/etc/rpc.snoopd.auth* has the form:

`accept localhost:`*user*

This line authorizes the user *user* to use all *snoopd* services on this Data Station when he or she starts NetVisualyzer tools from this Data Station. *user* can be a login name, a numerical user id, or an asterisk (*), which stands for all users. To use `localhost` in */usr/etc/rpc.snoopd.auth*, `localhost` must be defined in */etc/hosts*.

Another simple authorization line is:

```
accept *
```

This line authorizes all users on all hosts to use all services on this Data Station. The asterisk on this line is a wild card that stands for all hosts. Specifying *user* is optional; if no user is given, the authorization line applies to all users.

You do not need to have a separate line for each user; you can replace *user* with a comma-separated list of users:

```
accept localhost:belle,ariel,snowwhite
```

To authorize users to snoop on this Data Station while they use NetVisualyzer tools on another workstation, use a line of this form:

`accept` *host*`:`*user*

*host* can be a workstation name or Internet address. It can be replaced with an asterisk (*), to indicate that *user* has authorization from any host, or it can be a comma-separated list of workstation names or Internet addresses.

To specify that a user must be a member of a specific group, use the form:

`accept` *host*`:`*user*`.`*group*

*group* can be a group name or group id. Examples of this form and several variations include:

`accept reddog:root.net`
: The user *root* in group *net* on the host `reddog` is authorized to use all services on this Data Station.

`accept *:root.*`
: A user named *root* from any workstation in any group is authorized on this Data Station.

`accept wookie:*.user,engr`
: All users from the workstation `wookie` who are in group *user* or group *engr* are authorized.

`accept *:joe.engr+net`
: The user *joe* is authorized to use all services on this Data Station from any workstation when he is in both of the groups *engr* and *net*.

It is possible to restrict access to just some of the *snoopd* services. Authorization lines that restrict services have the basic form:

`accept` *host*`:`*user*`.`*group*`/`*service*

These are the possible values of *service* and the tools that require them:

| | |
|---|---|
| `netlook` | Required by NetCollect. |
| `netcollect` | Equivalent to `netlook`. |
| `histogram` | Required by NetGraph and NetTop. |
| `netgraph` | Equivalent to `histogram`. |
| `netsnoop` | Required by Analyzer and NetSnoop. |
| `analyzer` | Equivalent to `netsnoop`. |
| `addrlist` | Required by NetLook and NetTop. |

A comma-separated list of services can be used to authorize several services. Examples are:

`accept yeti:joe/netlook,netgraph`
: Authorize *joe* to use `netlook` and `netgraph` services on this Data Station from the workstation `yeti`.

`accept */netlook`
: Authorize all users from all workstations for `netlook` service only.

Authorization lines can prohibit access to *snoopd* services. These lines begin with `reject`, but otherwise have the same basic form as lines that begin with `accept`:

`reject` *host*`:`*user*`.`*group*`/`*service*

Some examples of lines that prohibit the use of the Data Station are:

`reject aztec,maya,inca`
: Prohibit the use of *snoopd* services on this Data Station by any user on the workstations `aztec`, `maya`, or `inca`.

`reject *:guest`
: Prohibit the use of *snoopd* services by any user named *guest*.

`reject */addrlist`
: Prohibit the use of `addrlist` service on this Data Station.

Finally, several `accept` or `reject` lines can be combined on one line by separating the specifications with white space. For example:

`accept curly:* moe:*.net larry/netgraph`
: Authorize all users from the workstation `curly`, users in the group *net* from the workstation `moe`, and users who want `netgraph` service from the workstation `larry`.

See the *snoopd*(1M) manual page in Appendix F for other examples of entries in the */usr/etc/rpc.snoopd.auth* file. For information on the */etc/passwd* file, see *passwd*(1M).

*Appendix C*

# Protocols

NetVisualyzer supports many network protocols. Packet headers for supported protocols can be fully decoded by Analyzer and NetSnoop, and information about packets using supported protocols is available from NetAccount, NetGraph, and NetTop. In addition, other protocols are recognized, although not fully decoded, by NetVisualyzer tools.

This appendix lists the protocols supported and partially supported by NetVisualyzer tools and contains diagrams showing how these protocols relate to one another. You can use this information to create filters to capture the protocol packets of interest to you. This chapter also provides references for further information about the protocols.

## Supported Protocols

You can see a list of protocols supported by NetVisualyzer tools by giving the command:

```
netsnoop -L all
```

Table C-1 lists fully supported protocols. It contains protocol name acronyms, the names used by NetVisualyzer tools, and the full protocol names. Where available, the RFC sources for the protocols are listed.

**Table C-1** Supported Protocols

| Protocol Name | NetVisualyzer Name | Description |
|---|---|---|
| AARP | `aarp` | AppleTalk™ Address Resolution Protocol |
| ADSP | `adsp` | AppleTalk Data Stream Protocol |
| AEP | `aep` | AppleTalk Echo Protocol |
| AFP | `afp` | AppleTalk Filing Protocol |
| ARP | `arp` | Address Resolution Protocol (RFC 826) |
| ARPIP | `arpip` | IP to Ethernet ARP (RFC 826) |
| ASP | `asp` | AppleTalk Session Protocol |
| ATP | `atp` | AppleTalk Transaction Protocol |
| BOOTP | `bootp` | Bootstrap Protocol (RFCs 951 and 1084) |
| DDP | `ddp` | AppleTalk Datagram Delivery Protocol |
| DECnet | `decnet` | DECnet Phase IV protocol |
| DNS | `dns` | Domain Name System protocol (RFC 1035) |
| ECHO | `echo` | XNS® Echo protocol (RFC 862) |
| ELAP | `elap` | AppleTalk EtherTalk Link Access Protocol |
| ERROR | `error` | XNS Error protocol |
| Ethernet | `ether` | Ethernet version 2 protocol |
| FDDI | `fddi` | Fiber Distributed Data Interface protocol |
| FTP | `ftp` | File Transfer Protocol (RFC 959) |
| HELLO | `hello` | DEC™ HELLO routing protocol (RFC 891) |
| ICMP | `icmp` | Internet Control Message Protocol (RFCs 792 and 950) |
| IDP | `idp` | XNS Internetwork Datagram Protocol |

**Table C-1** (continued) Supported Protocols

| Protocol Name | NetVisualyzer Name | Description |
|---|---|---|
| IGMP | `igmp` | Internet Group Management Protocol (RFC 1112) |
| IPX™ | `ipx` | NetWare® Internetwork Packet Exchange protocol |
| IP | `ip` | Internet Protocol (RFC 791) |
| LAT™ | `lat` | DEC Local Area Transport protocol |
| LLC | `llc` | Logical Link Control protocol |
| MAC | `mac` | Media Access Control protocol |
| NBP | `nbp` | AppleTalk Name Binding Protocol |
| NFS | `nfs` | Sun Network File System protocol (RFC 1094) |
| NLM | `nlm` | Network Lock Manager protocol |
| NSP | `nsp` | DECnet IV Network Services Protocol |
| PAP | `pap` | AppleTalk Printer Access Protocol |
| PEP | `pep` | XNS Packet Exchange Protocol |
| Portmap | `pmap` | Sun RPC Portmap protocol (RFC 1057) |
| RARP | `rarp` | Reverse Address Resolution Protocol (RFC 903) |
| rcp | `rcp` | BSD Remote Copy protocol (RFC 1282) |
| RIP | `rip` | Routing Information Protocol (RFC 1058) |
| RIP | `novellrip` | Novell Routing Information Protocol |
| RIP | `xnsrip` | XNS Routing Information Protocol (Xerox) |
| rlogin, rsh | `rlogin` | BSD Remote Login/Remote Shell protocol (RFC 1282) |
| RTMP | `rtmp` | AppleTalk Routing Table Maintenance Protocol |

**Table C-1** (continued) Supported Protocols

| Protocol Name | NetVisualyzer Name | Description |
|---|---|---|
| SMT | `smt` | Station Management protocol |
| SNMP | `snmp` | Simple Network Management Protocol (RFC 1157) |
| SPP | `spp` | XNS Sequenced Packet Protocol |
| SPX | `spx` | NetWare Sequenced Packet Exchange protocol |
| SunRPC | `sunrpc` | Sun Remote Procedure Call protocol (RFC 1057) |
| TCP | `tcp` | Transmission Control Protocol (RFC 793) |
| TELNET | `telnet` | Telnet protocol (RFC 854) |
| TFTP | `tftp` | Trivial File Transfer Protocol (RFC 783) |
| Token MAC | `tokenmac` | Token Ring Media Access Control protocol |
| Token Ring | `tokenring` | Token Ring protocol |
| TSP | `tsp` | Time Synchronization Protocol |
| UDP | `udp` | User Datagram Protocol (RFC 768) |
| XTP | `xtp` | Xpress Transfer Protocol |
| ZIP | `zip` | AppleTalk Zone Information Protocol |

Table C-2 lists the protocols that are partially supported by NetVisualyzer. Analyzer and NetSnoop recognize these protocols, but don't decode them. These protocols are fully supported by the other NetVisualyzer tools.

**Table C-2** Partially Supported Protocols

| Protocol Name | NetVisualyzer Name | Description |
|---|---|---|
| NetBIOS™ | `netbios` | NetBIOS Services protocol (RFC 1002) |
| OSI | `osi` | Open Systems Interconnection protocols |
| SMTP | `smtp` | Simple Mail Transfer Protocol (RFC 821) |
| SNA | `sna` | System Network Architecture protocol |
| VINES® | `vines` | Banyan® VINES protocol |
| X | `x11` | X network protocol |

## Protocol Layers

This section contains diagrams showing the supported protocols listed in Table C-1 and Table C-2 and their relationships to each other. In these diagrams, arrows pointing up indicate that the layers above this protocol are shown in a diagram later in this section.

These diagrams are useful when constructing filters since each protocol must be completely specified except for its physical layer. These diagrams also show all of the packet types that are captured by Analyzer and other tools if you specify a lower-level protocol as a filter. For example, if you use `ip.tcp` (or the macro `tcp`) as a filter in Analyzer, the Type column in the Summary pane can show `rcp`, `rlogin`, `telnet`, and other protocols as well as `tcp`. You find out the complete list of possible protocol types by looking at Figure C-8 and noting the protocol layers above `tcp`.

Figure C-1 shows the Snoop pseudo-protocol, `snoop`, and the three physical layer protocols, `ether`, `fddi`, and `tokenring`, above it. `loop` is a pseudo-protocol.

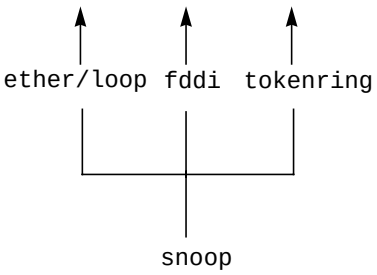



**Figure C-1** Snoop Pseudo-protocol Diagram

Figure C-2 shows the Ethernet physical layer, `ether`, and the supported layers of protocols above it. The Loopback pseudo-protocol `loop` is also shown as the bottom layer because it has the same layers as `ether` above it.

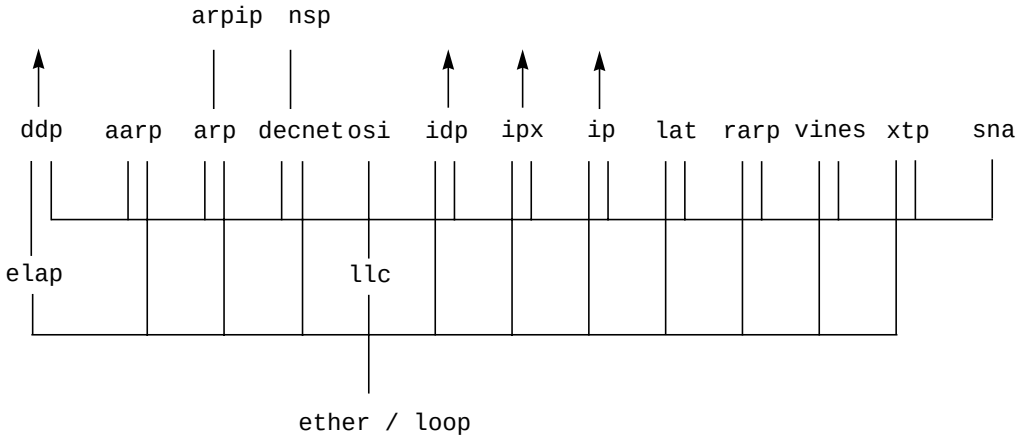



**Figure C-2** Ethernet Protocol Diagram

Figure C-3 shows the FDDI physical layer, `fddi`, and the supported layers of protocols above it.

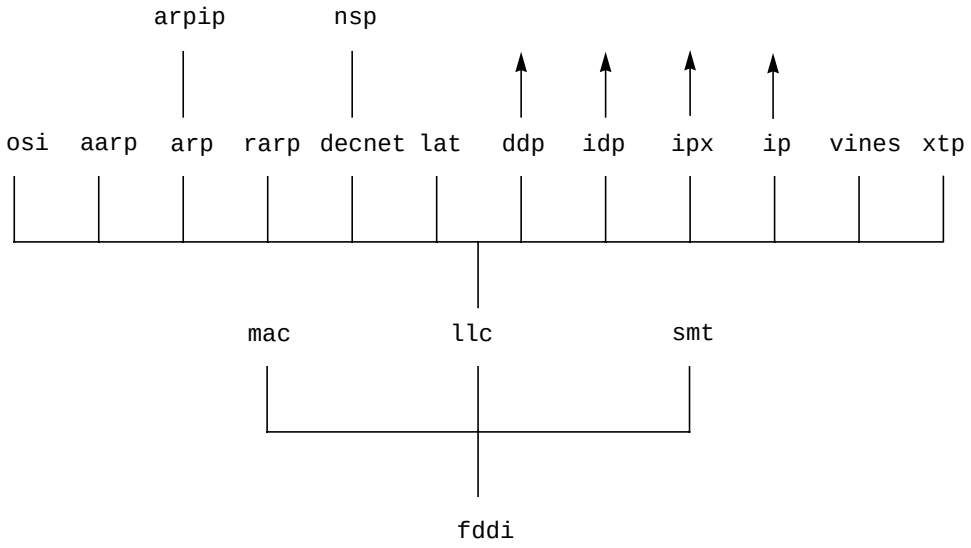



**Figure C-3** FDDI Protocol Diagram

Figure C-4 shows the third supported physical layer, `tokenring`, and the supported layers of protocols above it.

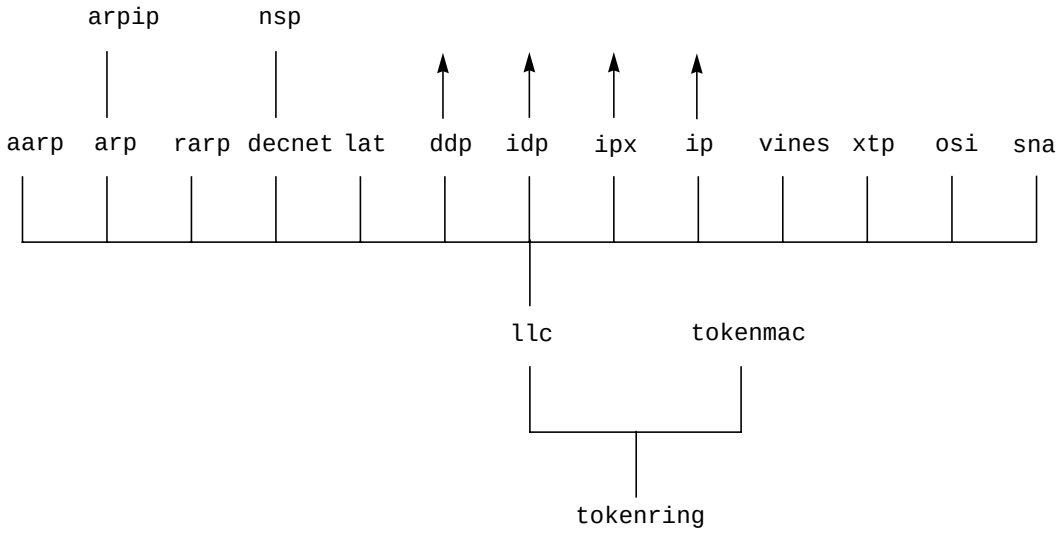



**Figure C-4** Token Ring Protocol Diagram

Figure C-5 shows the Datagram Delivery Protocol, `ddp`, and layers above it.

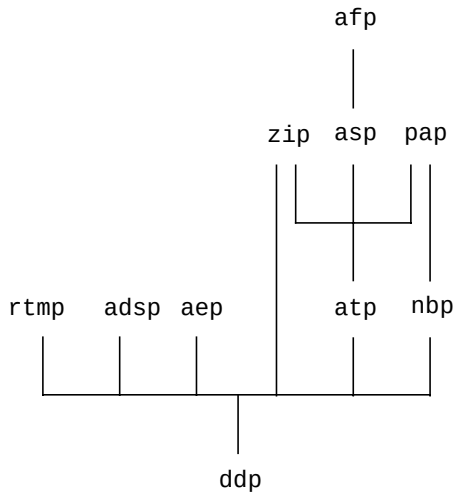



**Figure C-5** Datagram Delivery Protocol Diagram

Figure C-6 shows the AppleTalk protocols, Phases 1 and 2. EtherTalk™ is supported and decoded.

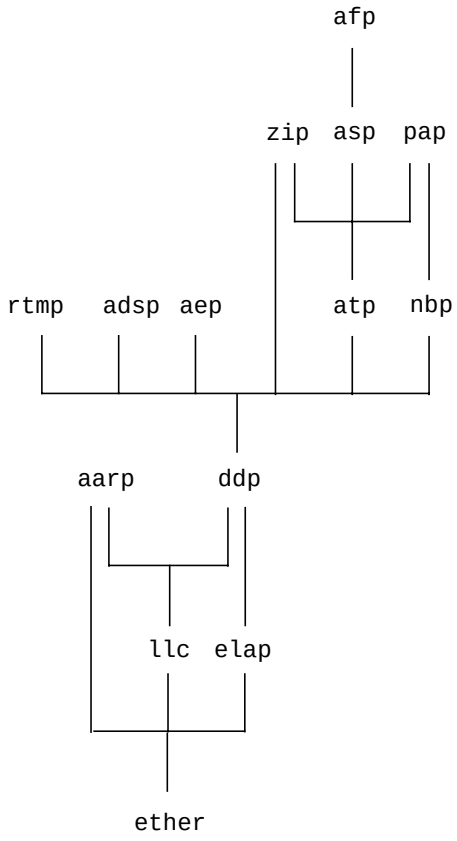



**Figure C-6** AppleTalk Protocols Phase 1 and 2 Protocol Diagram

Figure C-7 shows the Xerox Network Systems (XNS) Internetwork Datagram, `idp`, and NetWare Internetwork Packet Exchange, `ipx`, protocols. The two protocols are very similar; differences lie in the naming conventions used for the layers.

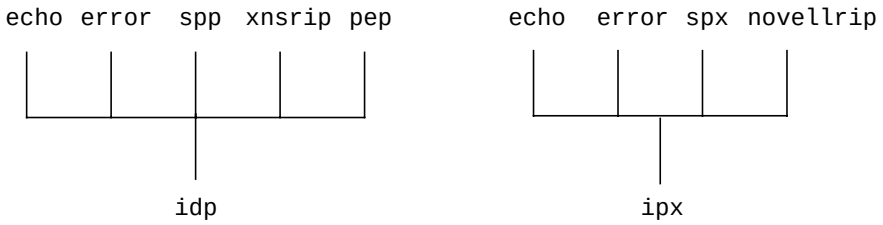



**Figure C-7** Internetwork Datagram and Internal Packet Exchange Protocol Diagrams

Figure C-8 shows the Internet Protocol, `ip`, and the layers above it. The dashed line from `tcp` to `sunrpc` indicates that the `sunrpc` protocol can be used over either `tcp` or `udp`.

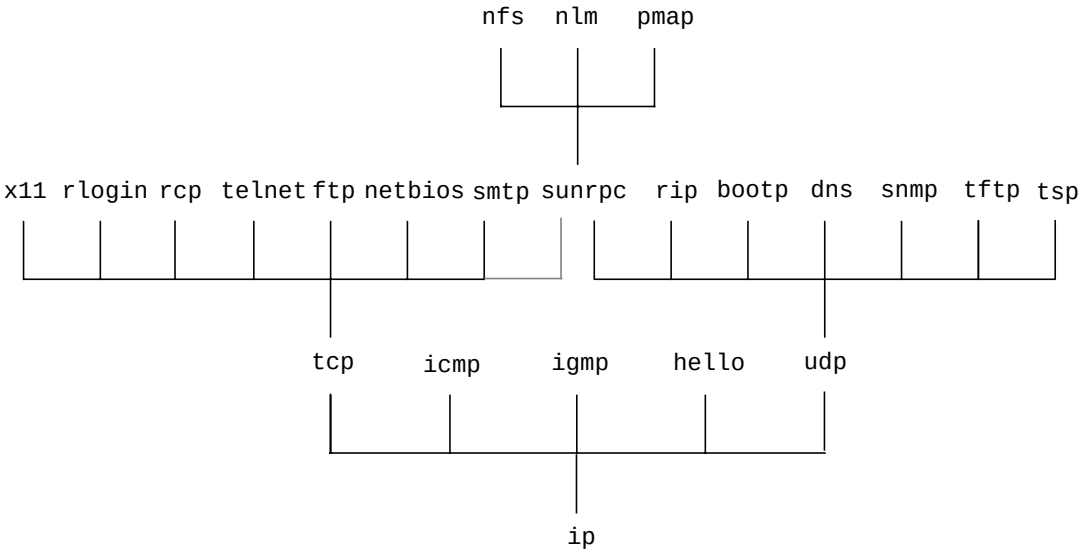



**Figure C-8** Internet Protocol Diagram

As an example of how to use these diagrams, assume that you want to capture FTP packets and that Ethernet is your physical routing layer. Find `ftp` (in Figure C-8) and trace the layers down through `tcp` and `ip` until you come to the physical layer `ether` in Figure C-2. Putting this into the syntax for specifying protocol scoping, use `ip.tcp.ftp` as your filter to capture FTP packets (`ether` isn't included because you don't need to specify physical layers). To see if there is a macro that you can use for `ip.tcp.ftp`, give the command:

```
netsnoop -L ether
```

In the Macro section of the output, notice that there is a macro called `ftp`:

```
ftp                 ip.tcp.ftp
```

So, instead of using `ip.tcp.ftp` as your filter for FTP packets, you can use just `ftp`.

## Protocol References

This section lists documentation sources for protocols supported by NetVisualyzer. The addresses for Request for Comment, ANSI, and ISO documents are listed at the end of this section.

AppleTalk (AARP, ADSP, AEP AFP, ASP, ATP, DDP, NBP, PAP, RTMP, ZIP)
See ELAP.

ARP — David C. Plummer, "Ethernet Address Resolution Protocol." *Request For Comment 826*. November 1982.

ARPIP — David C. Plummer, "Ethernet Address Resolution Protocol." *Request For Comment 826*. November 1982.

BOOTP — W.J. Croft and J. Gilmore, "Bootstrap Protocol." *Request For Comment 951*. September 1985.

J.K. Reynolds, "BOOTP Vendor Information Extensions." *Request For Comment 1084*. December 1988.

DECnet — DNA Routing Layer Functional Specification, Version 2.0. Digital Equipment Corporation. Part Number AA-K1821-TK.

| | |
|---|---|
| DNS | P. Mockapetris, "Domain Names—Implementation and Specification." *Request For Comment 1035*. November 1987. |
| ECHO | J.B. Postel, "Echo Protocol." *Request For Comment 862*. May 1983. |
| ELAP (AppleTalk) | Gursharan S. Sidhu, Richard F. Andrews, Alan B. Oppenheimer, Apple Computer, Inc., *Inside AppleTalk*. Menlo Park, California; Addison-Wesley Publishing Company, Inc. May 1990. |
| ERROR | See XNS. |
| Ethernet | *Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications*. ANSI/IEEE Standard 802.3-1985. |
| FTP | J.B. Postel, "File Transfer Protocol." *Request For Comment 959*. October 1985. |
| HELLO | D.L. Mills, "DCN Local-Network Protocols." *Request For Comment 891*. December 1983. |
| ICMP | J.B. Postel, "Internet Control Message Protocol, DARPA Internet Program Protocol Specification." *Requests For Comment 792 and 950*. Information Sciences Institute, University of Southern California. September 1981. |
| IDP | See XNS. |
| IGMP | S. Deering, "Host Extensions for IP Multicasting." *Request For Comment 1112*. August 1989. |
| IP | "Internet Protocol, DARPA Internet Program Protocol Specification." *Request For Comment 791*. Information Sciences Institute, University of Southern California. September 1981. |
| IPX | "NetWare System Interface Technical Overview." Novell, Inc. June 1989. |
| LAT | "Local Area Transport (LAT) Specification." Digital Equipment Corporation. Part Number AA-NL26A-TE. |
| LLC | See Token Ring. |

| | |
|---|---|
| MAC | *ANSI/FDDI Media Access Control (MAC) X3.139:1987 ISO 9314-2: 1989*, Information Processing Systems—Fibre Distributed Data Interface (FDDI) – Part 2: Token Ring Media Access Control (MAC). |
| NetBIOS | "Protocol Standard for a NetBIOS Service on a TCP/UDP Transport: Detailed Specifications." *Request for Comment 1002*. March 1987. |
| NFS | "NFS: Network File System Protocol Specification." *Request For Comment 1094*. Sun Microsystems, Inc., Mountain View, California. March 1989. |
| NLM | See SunRPC. |
| NSP | "DNA Network Services Protocol Functional Specification," Version 4.0. Digital Equipment Corporation. Part Number AA-X439A-TK. |
| OSI | "Open Systems Interconnection." International Standards Organization, Technical Committee ISO/TC 97. |
| PEP | See XNS. |
| PHY | *ANSI/FDDI Physical Layer (PHY) X3.148:1988 ISO 9314-1: 1989*, Information Processing Systems—Fibre Distributed Data Interface (FDDI) – Part 1: Token Ring Physical Layer Protocol (PHY). |
| PMD | *ANSI/FDDI Physical Medium Dependent (PMD) X3.166:1990 ISO 9314-3: 1990*, Information Processing Systems—Fibre Distributed Data Interface (FDDI) – Part 3: Token Ring Physical Layer Medium Dependent (PMD). |
| Portmap | "RPC: Remote Procedure Call Protocol Specification: Version 2." *Request For Comment 1057*. Sun Microsystems, Inc. June 1988. |
| RARP | Finlayson, Mann, Mogul, and Theimer, "A Reverse Address Resolution Protocol." *Request For Comment 903*. Stanford University, Palo Alto, California. June 1984. |
| rcp | B. Kantor, "BSD Rlogin." *Request for Comment 1282*. September 1991. |
| RIP | C. Hendrick, "Routing Information Protocol." *Request For Comment 1058*. June 1988. |

| | |
|---|---|
| RIP (Novell) | *Novell's Portable Transports*. Novell Part # 183-000347-001. Novell, Inc. Provo, Utah. April 1990. |
| RIP (XNS) | See XNS. |
| rlogin | B. Kantor, "BSD Rlogin." *Request for Comment 1282*. September 1991. |
| SMT | *ANSI/FDDI Station Management (SMT) X3T9.5/84-49*, Rev. 6.2, May 18, 1990. |
| SMTP | J.B. Postel, "Simple Mail Transfer Protocol." *Request For Comment 821*. August 1982. |
| SNA | *Systems Network Architecture Network Product Formats*. Part Number LY43-0081-1. International Business Machines. June 1989. |
| SNMP | Case, Fedor, Schoffstall, and Davin, "A Simple Network Management Protocol." *Request For Comment 1157*. May 1990. |
| SPP | See XNS. |
| SPX | See IPX. |
| SunRPC | *Remote Procedure Calls: Protocol Specification*. RPC4.0. *Request For Comment 1057*. Sun Microsystems, Inc., Mountain View, California. |
| TCP | "Transmission Control Protocol, DARPA Internet Program Protocol Specification." *Request For Comment 793*. Information Sciences Institute, University of Southern California. September 1981. |
| TELNET | J.B. Postel, "Telnet Protocol Specification." *Request For Comment 854*. May 1983 |
| TFTP | K. R. Sollins, "The TFTP Protocol (Revision 2)." *Request For Comment 783*. June 1981. |
| Token MAC | See Token Ring. |
| Token Ring | "Token-Ring Network Architecture Reference." Part Number SC30-3374-02. International Business Machines. 1989. |

| | |
|---|---|
| TSP | Ricardo Gusella, Stephano Zatti, and James M. Bloom, "The Berkeley UNIX Time Synchronization Protocol." Computer Systems Research Group, Computer Science Division, Department of Electrical Engineering and Computer Science, University of California at Berkeley, Berkeley, CA 94720. |
| UDP | J.B. Postel, "User Datagram Protocol." *Request For Comment 768*. August 1980. |
| X | "X Protocol Reference Manual for Version 11 of the X Window System." O'Reilly & Associates, Inc. 1990. |
| XNS | "Xerox Internet Transport Protocols, Xerox System Integration Standard." Xerox Corporation, Sunnyvale, California. XNSS 028112. December 1981. |
| XTP | "XTP Protocol Definition." Protocol Engines. Santa Barbara, California. Revision 3.5, September 1990. |
| VINES | "VINES Protocol Definition." Order Number DA254-00. Banyan Systems, Inc. February 1990. |


Internet *Request For Comment* (RFC) documents, in the *Defense Data Network Protocol Handbook*, are available from:

DDN Network Information Center
14200 Park Meadow Dr., Suite 200
Chantilly, VA 22021

To order ANSI and ISO documents, contact:

American National Standards Institute
1430 Broadway
New York, NY 10018
Telephone: (212) 354-3300
Fax: 212/302-1286
Telex: 42 42 96 ANSI UI

*Appendix D*

# Configuration File Formats

This appendix gives information about the formats of the configuration files created and read by NetVisualyzer tools. They are presented in alphabetical order by tool name.

By default, NetVisualyzer tools look for configuration files in your home directory. You can specify a different location in these ways (in descending order of precedence):

- on the command line with the **–u** option (for configuration files) or the **–f** option (for *network.data* files)
- in a resources file in your home directory (the file must have the same name as the resources file in */usr/lib/X11/app-defaults* that it overrides) with the `*controlsFile` resource for user interface configuration files and `*networksFile` for the NetLook network data file
- in a resources file in */usr/lib/X11/app-defaults* with the `*controlsFile` and/or `*networksFile` resources

Since the `*controlsFile` and `*networksFile` resources are standard X resources, you can use all of the standard methods for setting X resources as well.

## Analyzer Configuration File

The Analyzer configuration file, typically called *.analyzerrc*, specifies Analyzer configuration information such as the settings of all control panels.

You can save Analyzer user interface configuration information such as control panel settings in the *~/.analyzerrc* file (or a file you specify). The next time you invoke Analyzer, it uses *~/.analyzerrc* by default to configure the display and control panel settings. You can override the control panel settings in a configuration file you specify with command line options.

Each line in the configuration file has two columns separated by a tab. The first column is the name of a configurable feature of Analyzer. The second column is the value. A blank value is acceptable.

The following is an example of the *.analyzerrc* file (reformatted slightly):

```
AnalyzerGeometry     1008x728+260+35
InterfaceField       jenny:
FilenameField        snoop.file
Network              Yes
StartOnField
FilterField
Framing              No
Chksum               No
Toobig               No
Toosmall             No
DecodeBufferField    10
StopAtField          10
StopOrField
StopOnField
CaptureLengthField
CaptureGeometry      500x612+769+405
```

```
AnalyzerGeometry     1008x728+260+35
```

Defines the location of the analyzer window: width x height + X_origin + Y_origin.

```
InterfaceField  jenny:
```

Defines the network interface.

```
FilenameField   snoop.file
```

Defines the name a previously saved snoop data file.

```
Network Yes
```

`Yes` means capture from the network interface using the name in the InterfaceField. `No` means capture from a file using the name in the filenameField.

```
StartOnField    ip
```

When the filter expression, in this case `ip`, is encountered, begin to save the packets for decoding.

```
FilterField     tcp
```

Save packets that matches the filter expression, in this case `tcp`, for decoding.

```
Framing No
```

`Yes` means capture and save packets with framing error.

```
Chksum  No
```

`Yes` means capture and save packets with checksum error.

```
Toobig  No
```

`Yes` means capture and save packets that are too big.

```
Toosmall        No
```

`Yes` means capture and save packets that are too small.

```
DecodeBufferField       10
```

Number of packets to decode when capture is complete. The last 10 packets are decoded in this case. This number should be less than or equal to the StopAtField.

```
StopAtField     10
```

Number of packets to save for decoding. It is 10 packets in this case.

```
StopOrField     5
```

The duration of time to capture packets from start. It is 5 seconds in this case.

```
StopOnField     decnet
```

When the filter expression, in this case `decnet`, is encountered, stop capturing packets.

```
CaptureLengthField      50
```

Defines the maximum number of bytes to capture for each packet. It is 50 in this case.

```
CaptureGeometry 500x612+769+405
```

Defines the location of the capture window. Width x Height + X_origin + Y_origin.

## NetGraph User Interface Configuration File

You can save NetGraph user interface configuration information such as options to NetGraph and control panel settings in the *~/.netgraphrc* file (or a file you specify). The next time you invoke NetGraph, it uses *~/.netgraphrc* by default to configure the display and control panel settings. You can override the control panel settings in a configuration file you specify with command line options.

It's easy to create a user interface configuration file using the "Save Controls," "Save Controls As...", or "Quit" choices on the NetGraph File menu. You can also create a configuration file with a text editor. See "Using NetGraph in a Distributed Environment" in Chapter 4 for information on multiple configuration files.

A *.netgraphrc* file can have multiple lines; the format can be one or multiple lines of the following:

`option` *options*
*filter* [*graph_type*] [*color_index*] [*avg_color_index*] [*graph_style*] [*alarm_info*]
`#` *comment*

where:

| | |
|---|---|
| *options* | is any NetGraph option except **–u**. Options must be at the top of the file before any graph specifications. |
| *filter* | is a filter understood by NetSnoop. If *filter* contains white space, enclose the filter in single or double quotes. |
| *chart_type* | is `packets` (the default), `bytes`, `%packets`, `%bytes`, `%ether`, `%fddi`, `%npackets` *N*, or `$nbytes` *N*, where *N* is the base rate for the percentage calculation. |
| *color_index* | is a color map index. The default is color number 4, blue. |
| *avg_color_index* | is a color map index used for the moving average line. The default is color number 1, red. |
| *graph_style* | is the type of graph, either `bar` or `line`. The default is `bar`. |
| *alarm_info* | is the alarm information—see the description below. |

The format for the alarm information is:

`[alarm | noalarm] [silent | bell]` *nnnn.nn nnnn.nn*

where

| | |
|---|---|
| `alarm` | means the alarm is turned on. |
| `noalarm` | means the alarm is turned off. |
| `silent` | means the alarm is silent. |
| `bell` | specifies that a bell sounds when an alarm condition is met. |
| *nnnn.nn* | is a low or high number at which the alarm becomes active. |

## NetLook User Interface Configuration File

When you save the NetLook controls, the settings and positions of the control panels— your user interface configuration—are automatically saved in the *~/.netlookrc* file.

You can save NetLook user interface configuration information such as control panel settings in the *~/.netlookrc* file (or a file you specify). The next time you invoke NetLook, it uses *~/.netlookrc* by default to configure the display and control panel settings. You can override the control panel settings in a configuration file you specify with command line options.

An example of the NetLook user interface configuration file is:

```
NetworkDisplay  Name
NetworkShown    Active
NetworkNew      Add
NodeDisplay     Local
NodeShown       Active
NodeNew         Add
Rescale         5
Timeout         60
TrafficMode     Endpoint
TrafficBasis    Packets
PacketColorStep 1
ByteColorStep   1024
MinColorValue   8
MaxColorValue   15
DataStation     -yeti
Hide            mountain.wpd.sgi.com
Hide            192.26.61
NetLookGeometry 442x453+611+90
NetLookSnoopGeometry    334x138+363+374
NetLookTrafficGeometry  325x574+679+84
NetLookHideGeometry     310x223+570+213
```

The column on the left lists the user interface features; the column on the right lists settings.

You probably won't need to change this file manually; NetLook changes it automatically whenever you change any of the user interface features and save the configuration.

## NetLook Network Data File

The NetLook network data file, typically called *network.data*, contains network configuration and node information that has been accumulated by NetLook. It is saved when you use the "Save Networks" choices on the File menu or when you quit NetLook. The following sections describe the format of the file and explain two example files.

### Format

The *network.data* file is composed of an object-oriented language made up of statements that bind values to objects. Two types of objects are defined: simple and complex.

A simple object has a single value bound to it. The format of a simple object statement is:

*object value*

An example of a simple object is `PhysAddr`, which is used to define a physical address of an interface. To define a physical address of `8:0:69:2:0:f9`, the statement is:

```
PhysAddr 8:0:69:2:0:f9
```

A complex object has other objects bound to it. Complex objects may optionally be named. The format of a complex object is:

*object name* `{` *object value object value* `...` `}`

Objects are case insensitive; names are case sensitive.

An example of a complex object is `Interface`, which defines an endpoint for communication. An `Interface` object can be bound with a `PhysAddr` object to define its physical address, an `IPAddr` object to define its IP address, and a `DNAddr` to define its DECnet address.

To define an `Interface` named `cheese` with a physical address of `aa:0:4:0:e8:4`, an IP address of `192.26.75.14`, and a DECnet address of `1.232`, the complex object statement looks like this:

```
Interface cheese { PhysAddr aa:0:4:0:e8:4 IPAddr 192.26.75.14 DNAddr 1.232 }
```

In the file, a pound sign (#) denotes a comment. White space, such as new-lines, tabs, and spaces, is ignored. Thus the complex object statement can also look like this:

```
Interface cheese {
        PhysAddr        aa:0:4:0:e8:4
        IPAddr          192.26.75.14
        DNAddr          1.232
}
```

Table D-1 defines the *network.data* objects.

**Table D-1** *network.data* Objects

| Complex Object | Simple Object That Can Bind to the Complex Object | Definition |
|---|---|---|
| `Network` | | A logical network of bridged segments |
| | `IPNet` | IP network number of the network |
| | `IPMask` | IP subnet mask to use for the network |
| `Segment` | | A network segment |
| | `IPNet` | IP network number of the segment |
| `Node` | | A point of communication attached to one or more segments |
| | `NISServer` | A node declared as an NIS server (does not take a value) |
| | `NISMaster` | A node declared as an NIS master (does not take a value) |
| `Interface` | | A point of communication attached to a single segment |
| | `PhysAddr` | Physical address of an interface |
| | `IPAddr` | IP address of an interface |
| | `DNAddr` | DECnet address of an interface |

Complex objects bind to other complex objects. For example, `Interface` binds to `Node`, `Node` binds to `Segment`, and `Segment` binds to `Network`.

## A Simple Example

This example uses a simple *network.data* file that defines a configuration of a single network composed of one segment with two nodes named `cheese` and `squaw`.

All *network.data* files begin with a file format version number; in this case, `1.10`. After the version is the object `Network`. The object `Network` is a complex object, so it can take on a name; in this case the name is `engineering`. The file looks like this:

```
NetLook 1.10
Network engineering {
        IPNet   192.26.75
        Segment  {
                IPNet   192.26.75
                Node  {
                        Interface cheese  {
                                PhysAddr        aa:0:4:0:e8:4
                                IPAddr          192.26.75.14
                                DNAddr          1.232
                        }
                }
                Node  {
                        Interface squaw  {
                                PhysAddr        8:0:69:2:0:f9
                                IPAddr          192.26.75.11
                        }
                }
        }
}
```

After the `Network` object, a left brace marks the beginning of object-value pairs that are bound to this network. The first object assigned to the network is the simple object `IPNet`, with a value of `192.26.75`. This value defines the network as representing the IP network of `192.26.75`. The next line contains the complex object `Segment`. In this case, `Segment` is not named and is followed by a left brace. On the next line, another `IPNet` simple object also binds the IP network number of `192.26.75` with this segment.

The next line contains the complex object `Node` (not named) and an opening left brace. This node is defined to have an interface named `cheese` with three simple objects bound to it:

- a PhysAddr of `aa:0:4:0:e8:4`
- an IPAddr of `192.26.75.14`
- a DNAddr of `1.232`

Next are the closing right brace of the `Interface` complex object and the closing right brace of the `Node` complex object.

A second node is defined with an interface named `squaw`. This interface has only a `PhysAddr` of `8:0:69:2:0:f9` and an `IPAddr` of `192.26.75.11` bound to it; no `DNAddr` is defined.

Lastly, the closing right braces of the `Interface`, `Node`, `Segment`, and `Network` complex objects end the complex object definitions.

## A Network Using an IP Netmask

It is a common practice at large sites running TCP/IP to use a netmask to divide a network. NetLook reads the netmask that is in use on the Display Station and uses it in its logic when it configures nodes on the networks. The netmask is written to the network data file, *network.data*.

The following example shows a *network.data* file for a network named `usa` that uses a netmask.

```
NetLook 1.10
Network usa {
        IPNet   131.131
        IPMask  255.255.255.0
}
Network boston {
        IPAddr  131.131.1
        Segment boston {
                IPAddr 131.131.1
                Node  {
                        Interface gate-boston {
                                IPAddr         131.131.1.1
                        }
                }
        }
}
Network dallas {
        IPAddr        131.131.2
        Segment dallas {
                IPAddr 131.131.2
                Node  {
                        Interface gate-dallas {
                                IPAddr         131.131.2.1
                        }
                }
        }
}
```

The first network object defines the Class B network and binds the netmask `255.255.255.0` with that network. The following network objects, which are subnetworks of the `131.131` network, use the netmask that was defined for the `131.131` network. You can also establish a netmask in a remote network or use a different netmask (for example, `255.255.252.0`).

## NetSnoop Configuration File

The NetSnoop configuration file, typically called *.netsnooprc*, consists of command line options, protocol option settings, macro definitions, comment lines beginning with a pound sign (#), and blank lines.

You can override the control panel settings in a configuration file you specify with command line options.

An example *.netsnooprc* file is:

```
define ip.udst ip.dst == mountain
option -x -n ether
set ip.noetherupdate arpip.noetherupdate
```

The meaning of each line is:

`define ip.udst ip.dst == mountain`
: A user-defined macro. In this case, the macro `ip.udst` is defined to be equal to `ip.dst == mountain`.

`option -x -n ether`
: All the NetSnoop command line options can be given on an `options` line. In this case, `-x -n ether` is used as if it had been entered on the command line when NetSnoop is run.

`set ip.noetherupdate arpip.noetherupdate`
: Protocol options are set with a `set` line as if they were specified with the **–p** option on the command line. White space may be used as well as commas to separate the protocol options.

## NetTop Configuration File

The NetTop configuration file, typically called *.nettoprc*, contains NetTop configuration information.

You can save NetTop user interface configuration information such as control panel settings in the *~/.nettoprc* file (or a file you specify). The next time you invoke NetTop, it uses *~/.nettoprc* by default to configure the

display and control panel settings. You can override the control panel settings in a configuration file you specify with command line options.

Each line of the NetTop configuration file has two or three columns separated by a space or a tab. The first column is the name of a configurable feature of NetTop. These names are defined by NetTop. The second and third columns are values. A blank value is acceptable.

The following is an example of a *.nettoprc* file (slightly reformatted):

```
SnoopInterface          jenny
SnoopFilter             tcp
MeasureTraffic          Packets
DataUpdateTime          1.000000
InterpolatePercent      50
VerticalScale           Reduce
LockAt                  5
ReduceAfter             5.000000
LabelBy                 Name
Display                 SourcesAndDestinations
Sources                 5
Destinations            5
ShowSourceNodes         Specific
ShowDestinationNodes    Specific
BusiestMeans            MostPackets
Source 2                diamond
Source 3
Source 4
Source 5
Source 6
Source 7
Source 8
Source 9
Destination 2           ruby
Destination 3
Destination 4
Destination 5
Destination 6
Destination 7
Destination 8
Destination 9
NetTopGeometry          592x603+313+33
TrafficControlGeometry  390x632+8+183
NodeControlGeometry     460x720+767+263
```

The remainder of this section explains each of these lines and others that can appear in the file:

```
SnoopInterface  jenny
```

Defines the snooping interface.

```
SnoopFilter             tcp
```

Specifies a filter expression.

```
MeasureTraffic  %NPackets
```

Specifies how traffic should be measured. The possibilities are: `Packets`, `Bytes`, `%Packets`, `%Bytes`, `%Ether`, `%FDDI`, `%Npackets`, and `%Nbytes`.

```
Npackets
Nbytes
```

If `MeasureTraffic` is `%Npackets` or `%Nbytes`, the traffic is calculated as a percentage of the value of one of these resources.

```
DataUpdateTime  1.000000
```

Specifies the time interval in seconds to recalculate and update the tower display.

```
InterpolatePercent      50
```

Specifies the percentage of *n* `DataUpdateTime` seconds it takes to change from one value to a new value. In this case, it is 50%.

```
VerticalScale   Reduce
```

Specifies how to scale the vertical axis. The options are: `Lock`, `NeverReduce`, and `Reduce`.

```
LockAt          5
```

Sets the maximum value for the vertical axis. This value is used when `VerticalScale` is set to `Lock`. In this case, the maximum value is 5.

```
ReduceAfter     5.000000
```

Sets the time after which the maximum vertical scale is reduced. This value is used when `VerticalScale` is equal to `Reduce`. In this case, it is set to 5 seconds.

```
LabelBy         Name
```

Specifies how one or both of the horizontal axis are labeled. If the value is `Name`, node names are used. If the value is `Address`, IP addresses, physical addresses, or DECnet addresses are used as labels.

```
Display         SourcesAndDestinations
```

Specifies the type of graph. The options are: `SourcesAndDestinations`, `BusyPairs`, and `NodesAndFilters`.

```
Sources         5
Pairs           5
Nodes           5
```

Specifies number of source nodes to display. Depending upon which `Display` option is specified, one of these line appears in the file.

```
Destinations    5
Filters         5
```

Specifies the number of destination nodes or filters to display. Depending upon which `Display` option is specified, one of these line appears in the file.

```
ShowSourceNodes Specific
ShowDestinationNodes    Specific
ShowNodes       Busy
```

Specifies the radio buttons chosen for type of source nodes, destination nodes, and nodes. `Specific` means display nodes that are listed in the scrolling list. `Busy` means display the busiest source nodes.

```
BusiestMeans    MostPackets
```

Specifies the radio button chosen for the definition of the term busiest. `MostPackets` means traffic volume is measured in packets. `MostBytes` means traffic volume is measured in bytes.

```
Source 2        diamond
Source 3
```

```
Source 4
Source 5
Source 6
Source 7
Source 8
Source 9
```

Specifies source node names.

```
Destination 2   ruby
Destination 3
Destination 4
Destination 5
Destination 6
Destination 7
Destination 8
Destination 9
```

Specifies destination node names.

```
Filter 2 ip
Filter 3
Filter 4
Filter 5
Filter 6
Filter 7
Filter 8
Filter 9
```

Specifies the filters to be used on one horizontal axis.

```
NetTopGeometry  592x603+313+33
```

Defines the location of the main window.

```
TrafficControlGeometry    390x632+8+183
```

Defines the location of the Traffic control panel.

```
NodeControlGeometry       460x720+767+263
```

Defines the location of the Nodes control panel.

*Appendix E*

# Introduction to MIBs

This appendix contains information that may be useful to Browser users:

- a glossary of SNMP management terms
- information about the Silicon Graphics SNMP agent
- instructions for providing additional MIB specifications to Browser

## SNMP Management Reference

This section describes some basic terms used in the SNMP management framework. It begins with basic concepts. Later definitions build on terms defined previously. Terms in *italics* are defined elsewhere in this section.

### Network Management Model

The Network Management Model contains three components:

- A *network management station* running *network management applications*.
- A *network management protocol*.
- *Managed nodes*, each containing the corresponding *agent*.

### Network Management Station

A Network Management Station is a host that runs *network management applications* and is responsible for managing the network. For example, a Silicon Graphics workstation running NetVisualyzer Display Station software is a network management station.

### Network Management Applications

Network management applications typically reside in the *network management station* and collect data from various *agents* in the network. They process and present that data to the user. The applications may also control or configure the data and thus affect the behavior of the underlying network entity or device. NetVisualyzer tools are examples of network management applications.

### Agent

An agent is software or firmware that gathers the information important to the device on which it resides. It also implements a protocol that exchanges that information with a *network management station*. *snmpd*(1M) is an example of an SNMP agent.

### Network Management Protocol

The network management protocol is the protocol used to convey management information between an *agent* and a *network management application*. The protocol used by Browser to query information from various agents is *SNMP*.

### Simple Network Management Protocol (SNMP)

SNMP is the *network management protocol* used by Browser to talk to *agents* on remote *managed nodes*. It is defined in *RFC* 1157 (see "Protocol References" in Appendix C). For Silicon Graphics workstations, the SNMP agent is *snmpd*(1M). Agents for other types of nodes may be implemented in software or firmware and are vendor-specific.

### Managed Node

A managed node is a device such as workstation, router, or hub that has an SNMP *agent* implemented.

### Management Information Base (MIB)

A MIB is the specification for the virtual store of the information supported by an *agent*. Some MIBs have *RFC* status, which means they have been approved by the *IETF*.

MIBs are defined in *SMI* format. For instance, a router MIB is a collection of important information about a router defined in *SMI* format. An SNMP *agent* typically implements two MIBs, *MIB-II* and a device-specific MIB (an *Enterprise MIB*). However, the *agent* may not implement all of the objects defined in each MIB.

### SNMP Containment Tree

The vast information that *agents* gather about devices is organized into a hierarchical tree called the SNMP Containment Tree. A portion of this tree is shown in Figure E-1. Some of the *subtrees* in this tree are *MIBs*. In Figure E-1, `mib-2`, `cisco`, and `cabletron` are *MIBs*.

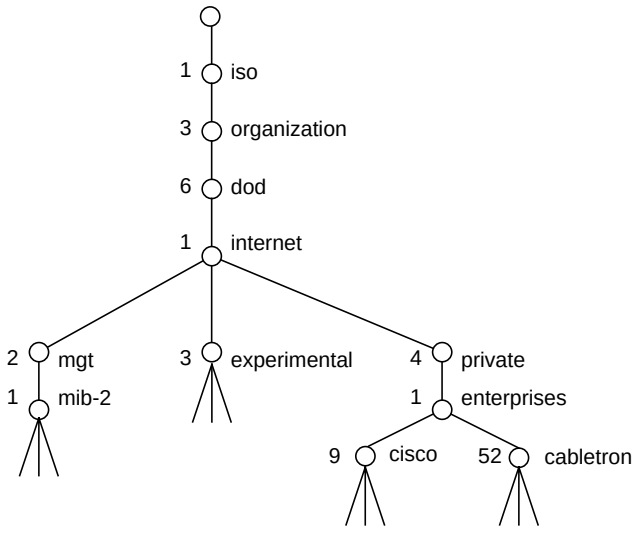



**Figure E-1** SNMP Containment Tree

### Subtree

Each node with children in the *SNMP Containment Tree* (Figure E-1) is called a subtree.

### MIB-II

MIB-II is the Internet-standard *MIB (RFC* 1213). This MIB has *managed objects* that are important for managing the TCP/IP suite of protocols. The *subtrees* of MIB-II—`system`, `interfaces`, `at`, `ip`, and so on—and some of their *variables* are shown in Figure E-2.

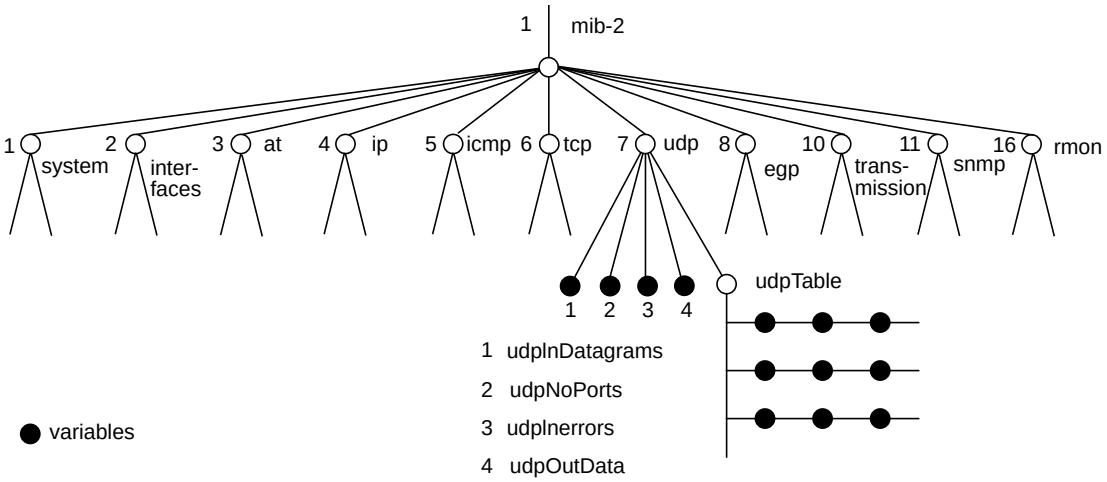



**Figure E-2** `mib-2` Portion of the SNMP Containment Tree

### Managed Object

A managed object is also known as a variable. Figure E-2 shows an additional portion of the *SNMP Containment Tree*. In this figure, the managed objects are solid circles. For example, `sysContact` is an object in `mib-2` that contains the name of the contact person for that system.

### Variable

A leaf node in the *SNMP Containment Tree* is called a variable. In the *SMI* definition for each *MIB*, each variable is defined to be read-write, read-only, or write-only.

### Object Identifier (Object ID)

An object identifier is a name in a dot notation that uniquely identifies an object (*subtree* or *managed object*) in the *MIB*. It is a sequence of numbers that traverses the *SNMP Containment Tree* shown partially in Figure E-1. For example, the object ID for `sysContact` is `1.3.6.1.2.1.1.4`.

### Structure of Management Information (SMI)

SMI is a textual format for the schema that is the precise definition of a *managed object*.

### Enterprise MIBs

Enterprise MIBs are *MIBs* defined by different vendors for managing their devices. They can be very device or vendor specific. For example, the CISCO *MIB* has objects for the Cisco router.

### Community String

A community string is a password used by an *SNMP agent* for authentication purposes. The default string for Silicon Graphics workstations is “public”. Community strings are typically defined by system administrators.

### Request for Comment (RFC)

A Request for Comment is an Internet standards document for a technology, concept, or protocol.

#### Internet Engineering Task Force (IETF)

The Internet Engineering Task Force is a community of network designers, operators, vendors, and researchers whose purpose is to coordinate the operation, management, and evolutions of the Internet, and to resolve short- and mid-range protocol and architectural issues.

## The Silicon Graphics SNMP Agent

The Silicon Graphics SNMP agent is the SNMP daemon, *snmpd*(1M). Its default community string is `public`. Information on this agent is provided in the *snmpd* manual page in Appendix F, "NetVisualyzer Manual Pages." See also "Authorizing Browsing" in Chapter 1 for information on *snmpd* authorization.

## Adding a MIB Specification

Browser is supplied with MIB specifications for MIB-II, RMON, Cisco, and Cabletron. These specifications are in the files */usr/lib/netvis/mibs/mib2, /usr/lib/netvis/mibs/rmon.mib, /usr/lib/netvis/mibs/cisco.mib,* and */usr/lib/netvis/mibs/cabletron.mib* respectively. Browser is designed so that you can add MIBs for any additional devices you want to browse to the directory */usr/lib/netvis/mibs.*

Some guidelines for the */usr/lib/netvis/mibs* directory are:

- Only MIB files in SMI format should be present in the */usr/lib/netvis/mibs* directory.
- Do not make any subdirectories in the */usr/lib/netvis/mibs* directory.
- Do not edit or move the files in */usr/lib/netvis/mibs* that were provided with NetVisualyzer.

To add MIB specifications to */usr/lib/netvis/mibs,* follow this procedure:

1. For vendor-specific MIBs, contact the vendor for that device to get the MIB in SMI format.
2. For MIBs that aren't vendor specific, obtain the MIB specifications from the RFC for that MIB. (See the address for obtaining RFCs in "Protocol References" in Appendix C.)
3. Edit the MIB specification and comment out the following lines by preceeding each line you want to comment out with two dashes, "`--`".
   - Definition lines:

     `DEFINITIONS ::= BEGIN`

     or

     *XXXXXXX*`-MIB DEFINITIONS ::= BEGIN`
   - IMPORTS sections:

     `IMPORTS`
     `...`
   - Lines such as:

*XXXXX*`-MIB { iso org(3) dod(6) internet(1) private(4) enterprises(1) 9 }`

4. Copy that MIB specification file to the directory */usr/lib/netvis/mibs.* You can choose any file name you want.

*Appendix F*

# NetVisualyzer Manual Pages

This appendix contains Display Station manual pages and Data Station Manual pages.

The Display Station manual pages are:

- *analyzer*(1M)
- *browser*(1M)
- *netfilters*(1M)
- *netgraph*(1M)
- *netlook*(1M)
- *nettop*(1M)
- *netvis*(1M)
- *nveventd*(1M)
- *nvlicense*(1M)

The Data Station manual pages are:

- *netaccount*(1M)
- *netcollect*(1M)
- *netpack*(1M)
- *netsnoop*(1M)
- *snoopd*(1M)

Other useful manual pages included are:

- *snmpd*(1M)
- *traceroute*(1M)
- *ping*(1M)

The manual pages are presented in alphabetical order.

# Index

## F



## G



## H



## I

## K



## L



## M



## N

## O



## P

## R



## S

**T**



**U**



**V**



**W**



**X**

## Tell Us About This Manual

As a user of Silicon Graphics products, you can help us to better understand your needs and to improve the quality of our documentation.

Any information that you provide will be useful. Here is a list of suggested topics:

- General impression of the document
- Omission of material that you expected to find
- Technical errors
- Relevance of the material to the job you had to do
- Quality of the printing and binding

Please send the title and part number of the document with your comments. The part number for this document is 007-0812-040.

Thank you!

## Three Ways to Reach Us

- To send your comments by **electronic mail**, use either of these addresses:
  - On the Internet: techpubs@sgi.com
  - For UUCP mail (through any backbone site): *[your_site]*!sgi!techpubs
- To **fax** your comments (or annotated copies of manual pages), use this fax number: 650-932-0801
- To send your comments by **traditional mail**, use this address:

  Technical Publications
  Silicon Graphics, Inc.
  2011 North Shoreline Boulevard, M/S 535
  Mountain View, California  94043-1389