OWS-E Operator Workstation Administrator's Guide

SG–3079 2.0

Cray Research, Inc.

Because of space restrictions, the following abbreviations are used in place of specific system names:

CX	Includes all models of CRAY X-MP computer systems
CEA	Includes all models of the Extended Architecture (EA) series, including CRAY Y-MP and CRAY X-MP EA computer systems
CRAY-2	Includes all models of CRAY-2 computer systems
CX/CEA	Includes all models of CRAY X-MP computer systems plus all models of CRAY Y-MP and CRAY X-MP EA computer systems

Requests for copies of Cray Research, Inc. publications should be sent to the following address:

Cray Research, Inc.
Distribution Center
2360 Pilot Knob Road
Mendota Heights, MN 55120
USA

Order desk (612) 681–5907
Fax number (612) 681–5920

Cray Research Software Documentation Map

The illustration on the following pages highlights the major body of documentation available for Cray Research (CRI) customers. The illustration is organized into categories by audience designation:

Audience	Description
General users	Those who use the UNICOS operating system, products, applications, or linking software
Application and system programmers	Those who write or modify program code on a CRI system for the purpose of solving computer system, scientific, or engineering problems
System administrators	Those who perform system administration tasks, such as installation, configuration, and basic troubleshooting
System analysts	Those who perform advanced troubleshooting, tuning, and customization
Operators	Those who perform operational functions, such as performing system dumps, and those who administer an operator workstation

To use the map, find the audience designation closest to your specific needs or role as a CRI system user. Note that manuals under other audiences may also be of interest to you; manuals are listed only once, underneath the audience to which they most directly apply. Some manual titles are abbreviated. The date in the footer tells you when the information was last revised.

For more information

In addition to the illustration, you can use the following publications to find documentation specific to your needs:

- *Software Documentation Ready Reference,* publication SQ–2122, serves as a general index to the CRI documentation set. The booklet lists documents and man pages according to topic.
- *Software Overview for Users*, publication SG–2052, introduces the UNICOS operating system, its features, and its related products. It directs you to documentation containing user-level information.
- *User Publications Catalog*, publication CP–0099, briefly describes all CRI manuals available to you, including some not shown on the map, such as release notices and training workbooks.

Ordering

To obtain CRI publications, order them by publication number from the Distribution Center:

Cray Research, Inc.
Distribution Center
2360 Pilot Knob Road
Mendota Heights, MN 55120
USA

Order desk (612) 681-5907
Fax number (612) 681-5920

GENERAL USERS

Introductory
UNICOS Primer (SG–2010)*

Software Overview (SG–2052)*

General
Software Documentation Ready Reference* (SQ–2122)

UNICOS Message Reference (SR–2200)

User Commands Reference (SR–2011)§

User Commands Ready Reference (SQ–2056)

UNICOS Shells Ready Reference (SQ–2116)

UNICOS Environment Variables Ready Reference (SQ–2117)

Index for CRAY-2 Man Pages (SR–2048)

Index for CRAY Y-MP, CRAY X-MP EA, and CRAY X-MP Man Pages (SR–2049)

General (continued)
Docview Guide (SG–2109)*

Visual Interfaces Guide (SG–3094)*

Tape Subsystem Guide (SG–2051)*

TCP/IP and OSI Network Guide (SG–2009)*

NQS Guide (SG–2105)*

Security (MLS) Guide (SG–2111)

Kerberos User's Guide (SG–2409)

Text Editing
Text Editors Primer (SG–2050)

`vi` Reference Card (SQ–2054)

`ed` Reference Card (SQ–2055)

MVS Link
SUPERLINK General Information Manual (SI–0194)

SUPERLINK Messages (SI–0176)

SUPERLINK User's Guide (SI–0195)

MVS Station Messages (SI–0108)

Station Reference (SI–2066)

Station Ready Reference (SI–0104)

RQS User's Guide (SG–2405)

NOS/VE Link
NOS/VE Reference (SC–0270)

UNIX Link
RQS User's Guide (SG–2119)

CLS-UX User's Guide (SU–3121)

VAX/VMS Link
SUPERLINK User's Guide (SV–3153)

RQS User's Guide (SV–3151)

Station Primer (SV–0361)

Station Reference (SV–0020)

Station Ready Reference (SV–0102)

VM Link
RQS VM User's Guide (SI–0170)

Station Primer (SI–0167)

Station Reference (SI–0168)

Station Messages and Codes (SI–0165)

Station Reference Summary (SI–0169)

* Available on-line with Docview

§ Man pages available with the `man` command

APPLICATION AND SYSTEM PROGRAMMERS

C
Cray Standard C Reference (SR–2074)*

Cray Standard C Ready Reference (SQ–2076)

Ada
Cray Ada Reference (SR–3014)

Cray Ada Programming Guide (SR–3082)

Pascal
Pascal Reference (SR–0060)*

Fortran 77
CF77 Fortran Reference (SR–3071)*

CF77 Compiler Message Manual (SR–3072)

CF77 Vectorization Guide (SG–3073)*

CF77 Parallel Processing Guide (SG–3074)*

CF77 Ready Reference (SQ–3070)

UNICOS Libraries
System Calls (SR–2012)§

Fortran Library (SR–2079)§

C Library (SR–2080)§

Math & Scientific Library (SR–2081)§

Specialized Libraries (SR–2057)§

I/O User's Guide (SG–3075)*

Advanced I/O Guide (SG–3076)*

Loaders
Loader Reference (SR–0066)*

SEGLDR Reference Card (SQ–0303)

Performance Utilities
Performance Utilities Reference (SR–2040)*

Debuggers
CDBX Debugger Reference (SR–2091)*

CDBX Debugger User's Guide (SG–2094)

Cray Assembly Language (CAL)
CAL Reference (SR–2003)*

CAL for CRAY Y-MP and CRAY Y-MP C90
Reference (SR–3108)

Symbolic Machine Instructions (SR–3109)

Ready Reference (SQ–3110)

CAL for CRAY X-MP and CRAY X-MP EA
Macros and Opdefs Reference (SR–0012)

Symbolic Machine Instructions (SR–0085)

Ready Reference (SQ–0083)

CAL for CRAY-2
CAL Ready Reference (SQ–2002)

Macros and Opdefs Reference (SR–2082)§

Linking Software
SUPERLINK MVS AAC Reference (SI–0197)

VAX/VMS Station Common Access Facilities (SN–0362)

SUPERLINK Programmer's Guide VAX/VMS (SV–3155)

Source Control
USM User's Guide (SG–2097)*

Networking
RPC Reference (SR–2089)

Visualization
UNICOS X Window System Reference (SR–2101)*

Other
Support Tools Guide (SG–2016)*

UNICOS Message System Programmer's Guide (SG–2121)*

Compiler Information File (CIF) Reference (SM–2401)

SYSTEM ADMINISTRATORS

General
UNICOS Installation Guide (SG–2112)

UNICOS System Administration (SG–2113)*

Administrator Commands Reference (SR–2022)§

Defining and Compiling Terminal Definitions (SN–2067)

Docview Writer's Guide (SG–2118)*

C2 Functionality on MLS Systems (SN–2407)

IOS Models B – D
IOS Guide (SG–0307)

IOS Messages (SR–2240)

MVS Link
Station Installation (SI–0078)

SUPERLINK MVS Installation, Tuning, & Customization (SI–0188)

RQS Administrator's Guide (SG–2406)

VM Link
Station Installation & Maintenance (SI–0162)

SUPERLINK Administrator's Guide (SI–0171)

VAX/VMS Link
Station Installation (SV–0100)

Station Administration (SV–0363)

RQS Administrator's Guide (SV–3152)

SUPERLINK Installation (SG–5091)

SUPERLINK Administrator's Guide (SV–3154)

UNIX Link
RQS Administrator's Guide (SG–2120)

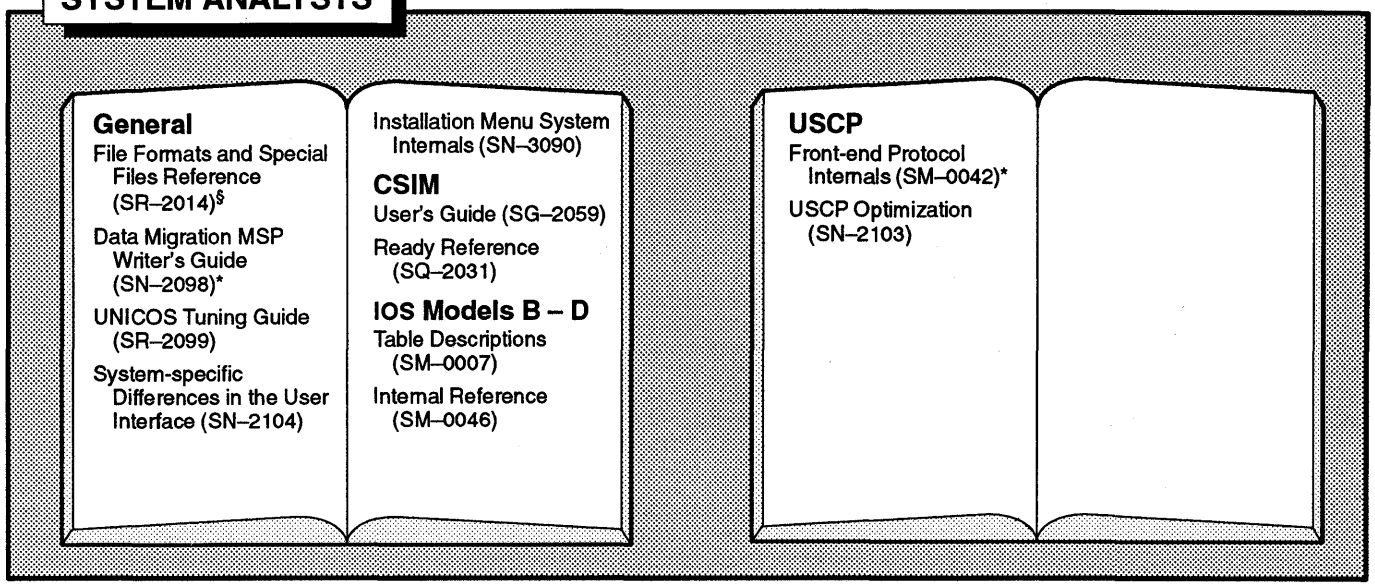
CLS-UX Installation & Configuration (SU–3123)

NOS/VE Link
NOS/VE Operator and Administrator Guide (SC–0271)

* Available on-line with Docview

§ Man pages available with the `man` command

SYSTEM ANALYSTS



General

File Formats and Special Files Reference (SR–2014)§

Data Migration MSP Writer's Guide (SN–2098)*

UNICOS Tuning Guide (SR–2099)

System-specific Differences in the User Interface (SN–2104)

Installation Menu System Internals (SN–3090)

CSIM

User's Guide (SG–2059)

Ready Reference (SQ–2031)

IOS Models B – D

Table Descriptions (SM–0007)

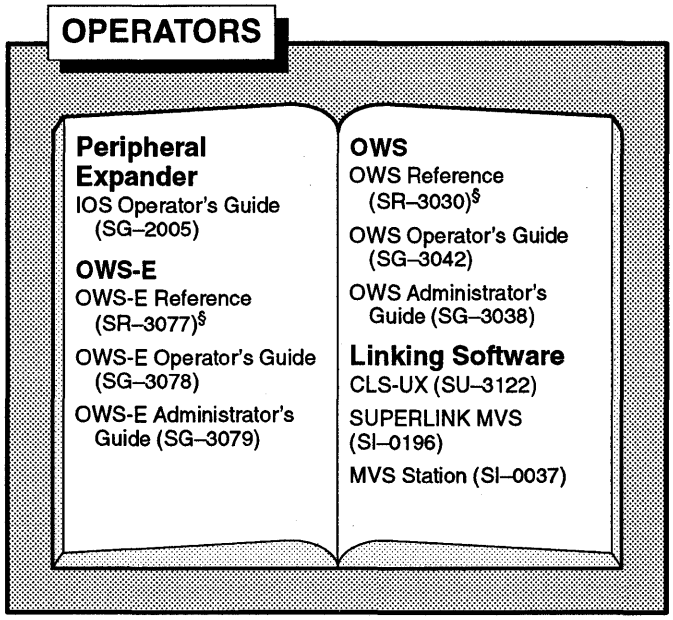
Internal Reference (SM–0046)

USCP

Front-end Protocol Internals (SM–0042)*

USCP Optimization (SN–2103)

OPERATORS



Peripheral Expander

IOS Operator's Guide (SG–2005)

OWS-E

OWS-E Reference (SR–3077)§

OWS-E Operator's Guide (SG–3078)

OWS-E Administrator's Guide (SG–3079)

OWS

OWS Reference (SR–3030)§

OWS Operator's Guide (SG–3042)

OWS Administrator's Guide (SG–3038)

Linking Software

CLS-UX (SU–3122)

SUPERLINK MVS (SI–0196)

MVS Station (SI–0037)

* Available on-line with Docview

§ Man pages available with the `man` command

New Features

OWS-E Operator Workstaion Administrator's Guide SG–3079 2.0

This rewrite of *OWS-E Operator Workstation Administrator's Guide* supports the 2.0 release of OWS-E. It contains the following changes:

- The new `edump`(8) file format is explained.
- The `bootios`, `booteiop`, `bootmux`, and `bootall` scripts have been deleted.
- You can now use `/etc/lapfile` to set line arbitration priorities.
- Many parameters have been added and deleted in `/etc/configfile`; additions are noted in the "System Configuration File" section. Other changes are noted.
- Various changes due to the new `fy` driver are noted. (The `cy` and `cz` drivers are no longer supported).
- Three new buttons have been added to the `xsnmpmon`(8) network monitor `Setup` window.

Record of Revision

The date of printing or software version number is indicated in the footer. In reprints with revision, changes are noted by revision bars along the margin of the page.

Version	*Description*
1.0	April 1991. Original printing.
1.1	September 1991. Reprint with revision to include OWS-E release 1.1 changes.
2.0	May 1992. Reprint with revision to include OWS-E release 2.0 changes.

Preface

This guide provides an overview of the Cray Research, Inc. (CRI) OWS-E operator workstation software and tells you how to perform OWS-E administrative duties for a Cray Research computer system with an I/O subsystem model E (IOS-E).

Assumptions

This guide was written for administrators of the OWS-E operator workstation. Readers should have a minimum of 16 hours training in either the UNICOS or the UNIX operating system; if you have no experience with UNICOS or UNIX, you should complete the CRI UNICOS Command Language (UCL–1) course.

It is assumed that you are running UNICOS operating system release 6.0 or later.

Conventions

The following typographic conventions are used throughout this manual:

Convention	Description
[ ]	Brackets enclose optional elements in syntax lines.
`typewriter font`	Typewriter font denotes literal items such as command names, file names, routines, directory names, path names, signals, messages, and programming language structures.
italic font	Italic font denotes variable entries and words or concepts being defined.

Convention	Description
`bold typewriter font`	In screen drawings of interactive sessions, bold typewriter font denotes literal items entered by the user. Output is shown in nonbold typewriter font.
KEY	Boxed words indicate keys on the keyboard, such as DELETE.

In this publication, *Cray Research*, *CRI*, and *Cray* refer to Cray Research, Inc. and/or its products.

Note

In this manual, *CRAY Y-MP* always refers to a CRAY Y-MP mainframe with an I/O subsystem model E (IOS-E).

Man page references

Throughout this document, reference is made to the on-line man pages available through the `man` command. A *man page* is a discussion of a particular element of the software.

Each man page includes a general description of one or more commands, routines, or other topics and provides details of their usage (command syntax, routine parameters, system call arguments, and so on). If more than one topic appears on a page, the entry will appear in the printed manual alphabetized only under its major name.

You can access a man page by entering the following:

```
man subject
```

For example, to access the `ls` man page, enter the following:

```
man ls
```

Man pages are grouped into numbered sections; each section contains entries of a particular type. Types of entries include user commands, administrator commands, system calls, library routines, file formats, and device descriptions.

The following table lists the type of entry associated with each section number shown and the manual in which the section is published.

Section	Subject	Publication
1	SunOS user commands	*SunOS Reference Manual* (Vol. I)
	UNICOS user commands	*UNICOS User Commands Reference Manual* (SR–2011)
	Simple Network Management Protocol (SNMP) user commands	*OWS-E Operator Workstation Reference Manual* (SR–3077)
5	OWS-E file formats	*OWS-E Operator Workstation Reference Manual* (SR–3077)
7	OWS-E topics	*OWS-E Operator Workstation Reference Manual* (SR–3077)
8	OWS-E administrator commands	*OWS-E Operator Workstation Reference Manual* (SR–3077)
	SunOS administrator commands	*SunOS Reference Manual* (Vol. III)
	UNICOS administrator commands	*UNICOS Administrator Commands Reference Manual* (SR–2022)

Section numbers appear in parentheses after man page names. Man pages are referenced in text by entry name and section number, as shown in the following example:

> To take a system dump, enter the `dumpsys`(8) command in an OWS-E window.

For more information

The following table lists related publications by topic; assume that a manual is a CRI publication unless it is otherwise identified.

Topic	Sources of information
Operator training	*UNICOS Operator Training* (TR–UOT)
	UNICOS Command Language (TR–UCL–1)
	Cray Research Software Training Catalog for Customers (TR–CUSTCAT)
OWS-E commands	*OWS-E Operator Workstation Reference Manual* (SR–3077)
	OWS-E Operator Workstation Ready Reference (SQ–3080)
OWS-E installation	*OWS-E 2.0 Release and Installation Notes* (RN–5060)
SunOS user information	*SunOS 4.1 User's Guides*, order number 851–1028–01 (Sun Microsystems, Inc.); also available on-line through AnswerBook.
	SunOS Reference Manual (Vol. I), order number 825–1244–01 (Sun Microsystems, Inc.)
	Using AnswerBook, order number 800–6908–10 (Sun Microsystems, Inc.)
SunOS system administrator information	*System Network Administration* (Vols. II and III), order number 800–3805–10 (Sun Microsystems, Inc.); also available on-line through AnswerBook.
	SunOS Reference Manual (Vols. II and III), order number 825–1244–01 (Sun Microsystems, Inc.)
OpenWindows	*Sun OpenWindows Version 3 End User's Manuals*, order number 851–1035–01 (Sun Microsystems, Inc.); also available on-line through AnswerBook.
UNICOS operating system, user information	*UNICOS User Commands Reference Manual* (SR–2011)
	UNICOS User Commands Ready Reference (SQ–2056)
	UNICOS Message Reference Manual (SR–2200)

Topic	Sources of information
UNICOS operating system, administrator information	*UNICOS System Administration* (SG–2113) *UNICOS Administrator Commands Reference Manual* (SR–2022)
IOS-E administration	*I/O Subsystem Model E (IOS-E) Guide,* (SD–2107)§ *IOS-E 3.0 Release and Installation Notes* (RN–5045)

§ This document is CRAY RESEARCH PRIVATE. It can be distributed to non-CRI personnel only with approval of the appropriate Cray Research manager.

For a more detailed list of Sun Microsystems, Inc., documentation, see *OWS-E Release and Installation Notes.*

Ordering publications

The *User Publications Catalog*, publication CP–0099, lists all Cray Research hardware and software manuals that are available to customers.

To order a manual, either call the Distribution Center in Mendota Heights, Minnesota, at (612) 681–5907 or send a facsimile of your request to fax number (612) 681–5920. Cray Research employees may choose to send electronic mail to `order.desk` (UNIX system users) or `order desk` (HPDesk users).

Reader comments

If you have comments about the technical accuracy, content, or organization of this manual, please tell us. You can contact us in any of the following ways:

- Send us electronic mail from a UNICOS or UNIX system, using the following UUCP address:

  `uunet!cray!publications`

- Send us electronic mail from any system connected to Internet, using the following Internet addresses:

  `pubs3079@timbuk.cray.com` (comments specific to this manual)

  `publications@timbuk.cray.com` (general comments)

- Contact your Cray Research representative and ask that a Software Problem Report (SPR) be filed. Use `PUBLICATIONS` for the group name, `PUBS` for the command, and `NO-LICENSE` for the release name.

- Call our Software Information Services department in Eagan, Minnesota, through the North American Support Center, using either of the following numbers:

  (800) 950–2729 (toll free from the United States and Canada)

  (612) 683–5600

- Send a facsimile of your comments to the attention of "Software Information Services" in Eagan, Minnesota, at fax number (612) 683–5599.

- Use the postage-paid Reader's Comment form at the back of this manual.

We value your comments and will respond to them promptly.

Contents

Page

v **Preface**

v Assumptions
v Convention
vi Man page references
viii For more information
ix Ordering publications
x Reader comments

1 **Introduction [1]**

1 What is the OWS-E?
1 What tasks can be done with the OWS-E?
2 What information is in this guide?
2 Where can I find more information?

3 **Hardware Environment [2]**

3 Hardware components
5 Tapes
5 Compact disk (CD)
5 Hardware connections

9 **OWS-E Software System Overview [3]**

9 Software components
10 Disk partitions
11 Directory structure

Page

13 **Initialization Files [4]**

13 Required files
14 Changed files
15 Environment variable settings

17 **System Configuration File [5]**

17 How does `/etc/configfile` work?
18 How can I make changes to `/etc/configfile`?
18 Order of parameters in `/etc/configfile`
18 Tokens
23 Configurable parameters set during installation
35 Configurable parameters set at release time

39 **Security [6]**

39 Passwords
39 User and group identification
40 OWS-E permissions file

43 **Setting Line Arbitration Priorities [7]**

45 **edump(8) File Format [8]**

47 File header structure
48 Section header structure
49 Register section structure

Page

51 **Customizing CRI Template Scripts [9]**

52 cpupanic

52 iophalt

53 OWS-E backup scripts: dumpdly and dumpwkly

55 **Shutting Down and Halting the OWS-E [10]**

55 Normal circumstances

57 When the system is hung

59 If all else fails: forcing a halt and reboot of the OWS-E

61 **Manually Dumping the IOS-E and Mainframe [11]**

63 **CPU Monitors [12]**

63 What do the CPU monitors do?

63 How can I use the CPU monitors?

65 How do the CPU monitors work?

67 **Network Monitor [13]**

67 What is xsnmpmon?

67 Background information on SNMP

69 How can I invoke xsnmpmon(8)?

70 What does the xsnmpmon(8) monitor look like?

71 Buttons

71 Entering text

72 Synopsis information

72 Message line

72 Modifying the monitor environment with the Setup button

74 Obtaining an overview with the System Info button

74 Using the statistics buttons

Page

74 I/F Stats

74 Media Stats

75 ICMP Stats

75 IP Stats

75 TCP Stats

75 UDP Stats

75 EGP Stats

75 SNMP Stats

76 Table buttons

76 AT Table

76 Net/Media Tbl

76 IP Addr Tbl

77 IP Route Tbl

77 TCP Connection

77 UDP Listeners

77 EGP Neigh Tbl

77 FDDI SMT

77 FDDI MAC

77 FDDI PORT

78 FDDI Attach

78 Using the Error Log button

78 Displaying messages with the Trap Log button

79 Displaying messages with the State Chg Log button

79 Creating a network with the Create Net button

80 Create Network Configuration window

81 Create Network Configuration buttons

84 Example

84 Scenario

85 Steps to create a network map

94 Loading process

96 Scanning with the Net Status button

Page

97 Network Status window

98 Buttons in the Network Status window

98 Statuses

99 Agent status

99 Interface status

100 Area status

101 Mouse button functions

102 Displaying performance information with the Perf Monitor button

102 Using the Notification button

104 Using the Telnet button

104 Log files

105 **Index**

Figures

4 Figure 1. Hardware components

6 Figure 2. Example connection of CRAY Y-MP computer systems

7 Figure 3. OWS-E/IOS-E interaction

8 Figure 4. OWS-E backplane connections

10 Figure 5. Disk partitions

11 Figure 6. Directory structure

46 Figure 7. edump file format

51 Figure 8. Interrelationships among scripts and commands

56 Figure 9. Shutting down and halting the OWS-E under normal circumstances

58 Figure 10. Shutting down and rebooting the OWS-E when it is hung

64 Figure 11. Monitor output for idle, user, system-wait, and system time averages for all CPUs

70 Figure 12. SNMP Network Monitor window

Page

73 Figure 13. Setup window buttons

80 Figure 14. Create Network Configuration window

85 Figure 15. Displaying the Create Network Configuration window

86 Figure 16. Selecting the Area button

87 Figure 17. Dialog box

88 Figure 18. bldg.net map

90 Figure 19. firstfloor.net map

91 Figure 20. lab.net map

93 Figure 21. secondfloor.net map

94 Figure 22. Interrelationships among the files (maps)

97 Figure 23. Network Status window

103 Figure 24. Event Notification window showing normal status

103 Figure 25. Event Notification window showing a change in status

Tables

19 Table 1. Tokens

23 Table 2. Configurable parameters set during the installation process

35 Table 3. Configurable parameters set at release time

Introduction [1]

Introduction [1]

This section describes the OWS-E, lists operational tasks that can be performed with it, and explains where to find information.

What is the OWS-E?
1.1

The OWS-E is a workstation that enables you to operate, administrate, and monitor your CRAY Y-MP mainframe and I/O subsystem model E (IOS-E). It uses the SunOS operating system, the OpenWindows graphical user interface, and the OWS-E software provided by CRI.

If your site has a maintenance contract with CRI, you will also have an MWS-E maintenance workstation; this machine is owned by CRI and is used to maintain and monitor the CRI computer system. The MWS-E and OWS-E cabinets look similar, but they have different hardware components and serve different purposes.

What tasks can be done with the OWS-E?
1.2

You or the operator can perform the following tasks from the OWS-E:

- Boot the IOS-E and the mainframe
- Dump the IOS-E and the mainframe
- Back up and restore files on the OWS-E
- Use the CPU and network monitors
- Respond to messages

Details for performing these tasks are provided in *OWS-E Operator Workstation Operator's Guide*.

What information is in this guide?

1.3

If you are a new administrator of the OWS-E, you will want to read through sections 2 through 5, which describe the following:

- Hardware and software environment
- Changes that CRI has made to the initialization files required for support of the OpenWindows environment (this is for your information only and may be of interest if you are familiar with OpenWindows)
- Contents of the system configuration file, which allows changes to machine dependencies to be set at execution rather than at compile time

Sections 6 through 13 discuss the administrative tasks specific to the OWS-E:

- Determining user access to sensitive commands, and other security issues
- Setting line arbitration priorities
- Customizing the template scripts and system configuration file provided by CRI
- Shutting down and halting the OWS-E
- Manually dumping the IOS-E and mainframe
- Using the CPU and network monitors

Where can I find more information?

1.4

This guide discusses only topics specific to administration of the OWS-E. *OWS-E Operator Workstation Operator's Guide* discusses operational tasks. *OWS-E Operator Workstation Reference Manual* discusses the OWS-E commands in detail.

Refer to *System and Network Administration*, provided by Sun Microsystems, Inc., for information about adding and deleting users, maintaining network files, and other tasks that you must perform as an administrator of a Sun system. For a detailed list of stock numbers, see *OWS-E 2.0 Release and Installation Notes*.

For information about OWS-E installation, see *OWS-E 2.0 Release and Installation Notes.*

Hardware Environment [2]

Hardware Environment [2]

This section describes the OWS-E hardware environment.

Hardware components 2.1

The OWS-E contains the following hardware components:

- Sun-4/370 Workstation (SPARCstation) with 16-Mbyte memory
  - VME chassis
  - Color graphics monitor
  - Keyboard
  - Optical mouse and pad
  - CPU board that contains memory, Ethernet controller, SCSI controller, and serial ports
  - Removable 669-Mbyte hard disk drive (SCSI)
  - Compact-disk read-only memory (CD-ROM) reader (SCSI)
  - 150-Mbyte 1/4-in. streaming tape drive (SCSI)
  - 2 FEI-3 board sets
- Laser printer

Figure 1 and Figure 4 show the hardware components.

Note

If your site has a maintenance contract with CRI, you will also have an MWS-E maintenance workstation. The MWS-E and OWS-E cabinets look similar, but they have some different hardware components and serve different purposes. The MWS-E is owned by CRI and is used to maintain and monitor the CRI computer system; the OWS-E is owned by your site and is used to operate and administrate the CRI computer system.

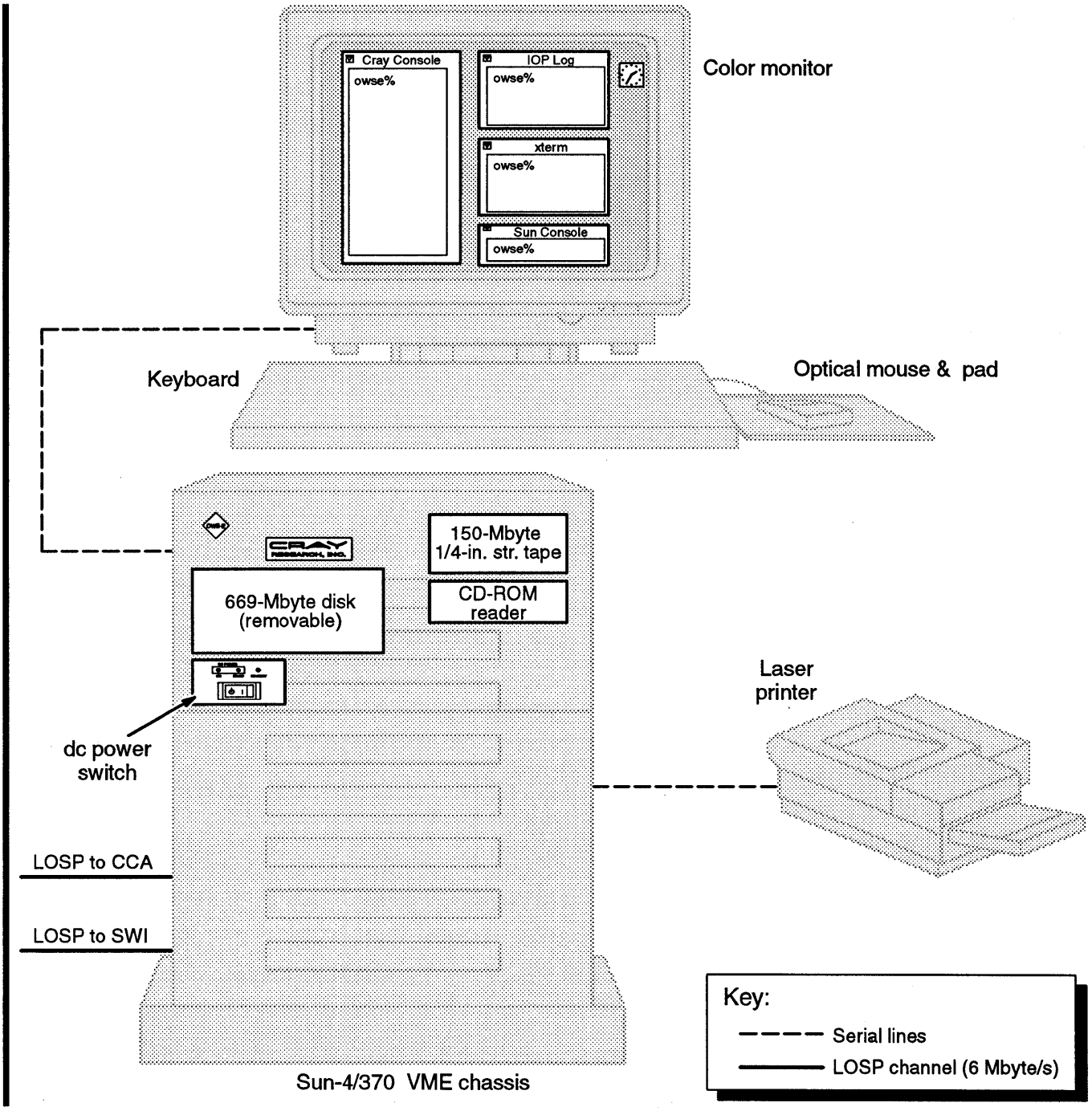


Figure 1. Hardware components

Tapes
2.2

The OWS-E uses 1/4-in. streaming tapes. It writes QIC-150 format tapes and reads both QIC-150 and QIC-24 format tapes. The tape device is usually `/dev/rst0`.

Compact disk (CD)
2.3

The OWS-E is equipped with a compact-disk, read-only memory (CD-ROM) drive. The SunOS operating system,OpenWindows, NEWSprint, AnswerBook, and OWS-E software are released on CD.

Note

All CDs must be loaded by use of the CD caddy. For more information, see *OWS-E 2.0 Release and Installation Notes.*

Hardware connections
2.4

Each CRI mainframe has its own OWS-E. If there are several OWS-E machines and they are connected by Ethernet, you can view the messages from all of the systems and perform functions from any of the OWS-E machines by logging into the other machines. The OWS-E is connected to the I/O subsystem model E (IOS-E) through two front-end interfaces (FEIs) to the service workstation interface (SWI) and to the Cray channel adaptor (CCA). For example, if your site has three CRI computer systems, they might be connected as shown in Figure 2.

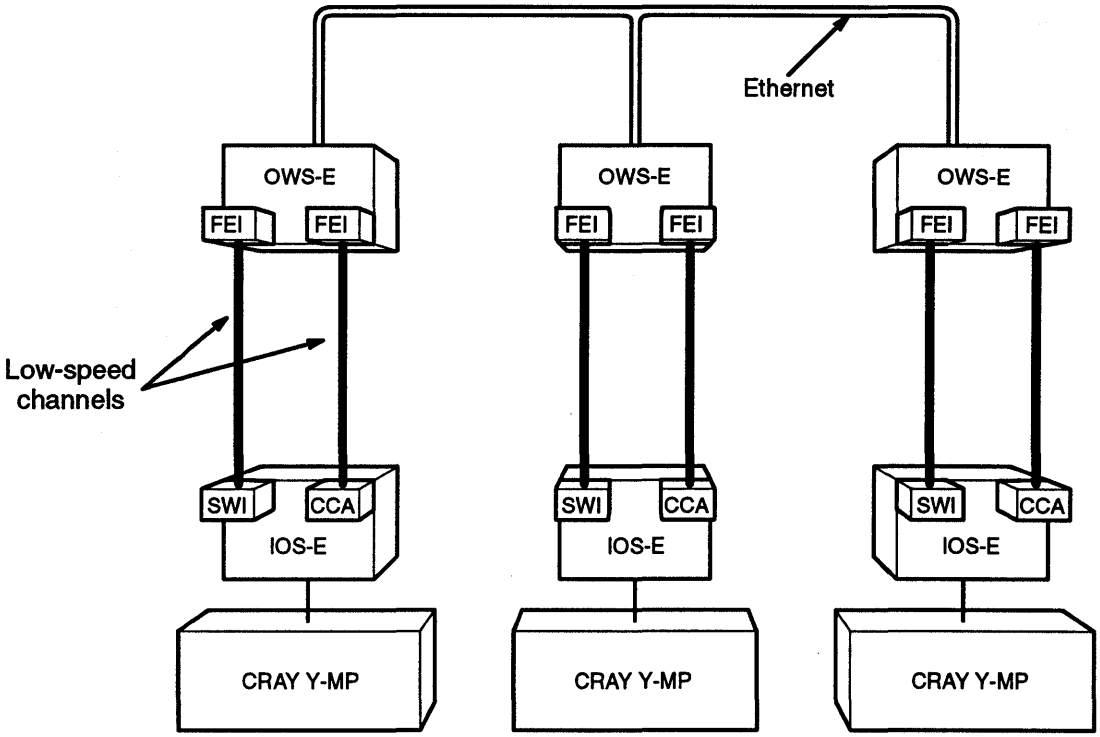


Figure 2. Example connection of CRAY Y-MP computer systems

The IOS-E consists of up to 16 I/O clusters, each with up to five I/O processors: one MUXIOP and one to four EIOPs. A MUXIOP communicates with the mainframe by sending response packets across the low-speed channel after request packets have been received and processed by the IOS-E. The MUXIOP also controls the high-speed data transfer channels to mainframe central memory and the SSD-E. An EIOP supports peripheral devices through channel adapters.

For more information about the IOS-E, see *I/O Subsystem Model E (IOS-E) Guide*§ and *IOS-E 3.0 Release and Installation Notes*.

§ This document is CRAY RESEARCH PRIVATE. It can be distributed to non-CRI personnel only with approval of the appropriate Cray Research manager.

The OWS-E software communicates with the IOS-E software through the `fy` driver, the front-end interface (FEI), and the service workstation interface (SWI), which is composed of a workstation interface (WIN) for the OWS-E and a cluster interface (CIN) for each cluster. Figure 3 illustrates the interaction between the OWS-E software and the IOS-E.

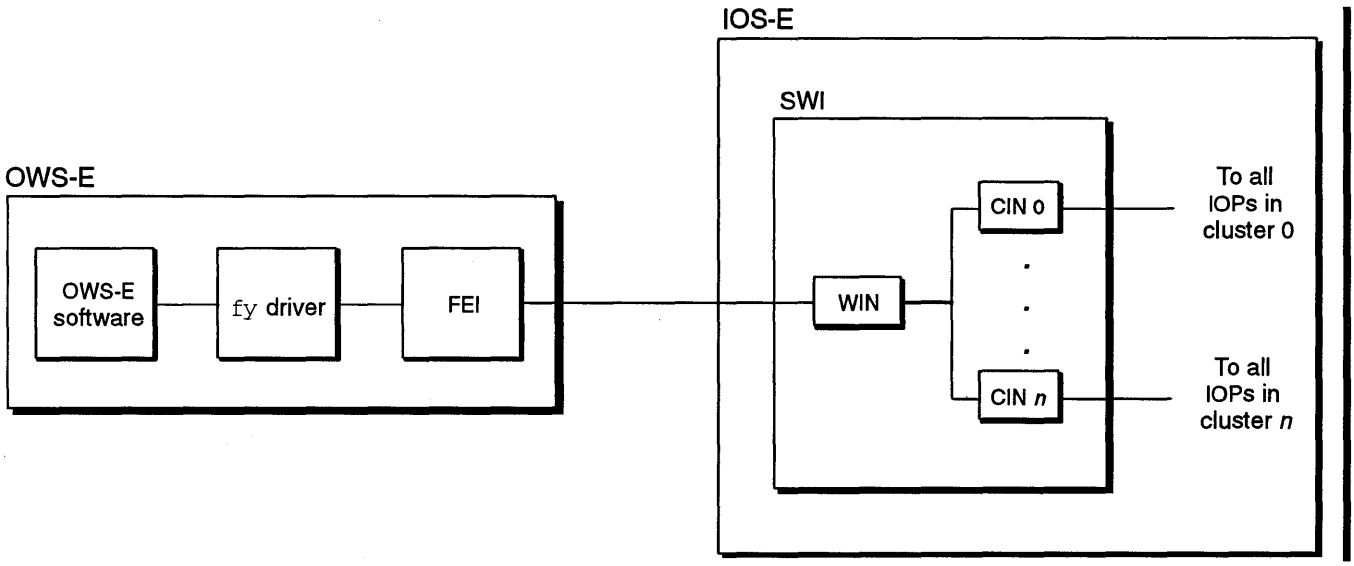


Figure 3. OWS-E/IOS-E interaction

As shown in Figure 2, page 6, the second FEI connects to the CCA, in order to support TCP/IP.

Figure 4 shows the OWS-E backplane connections.

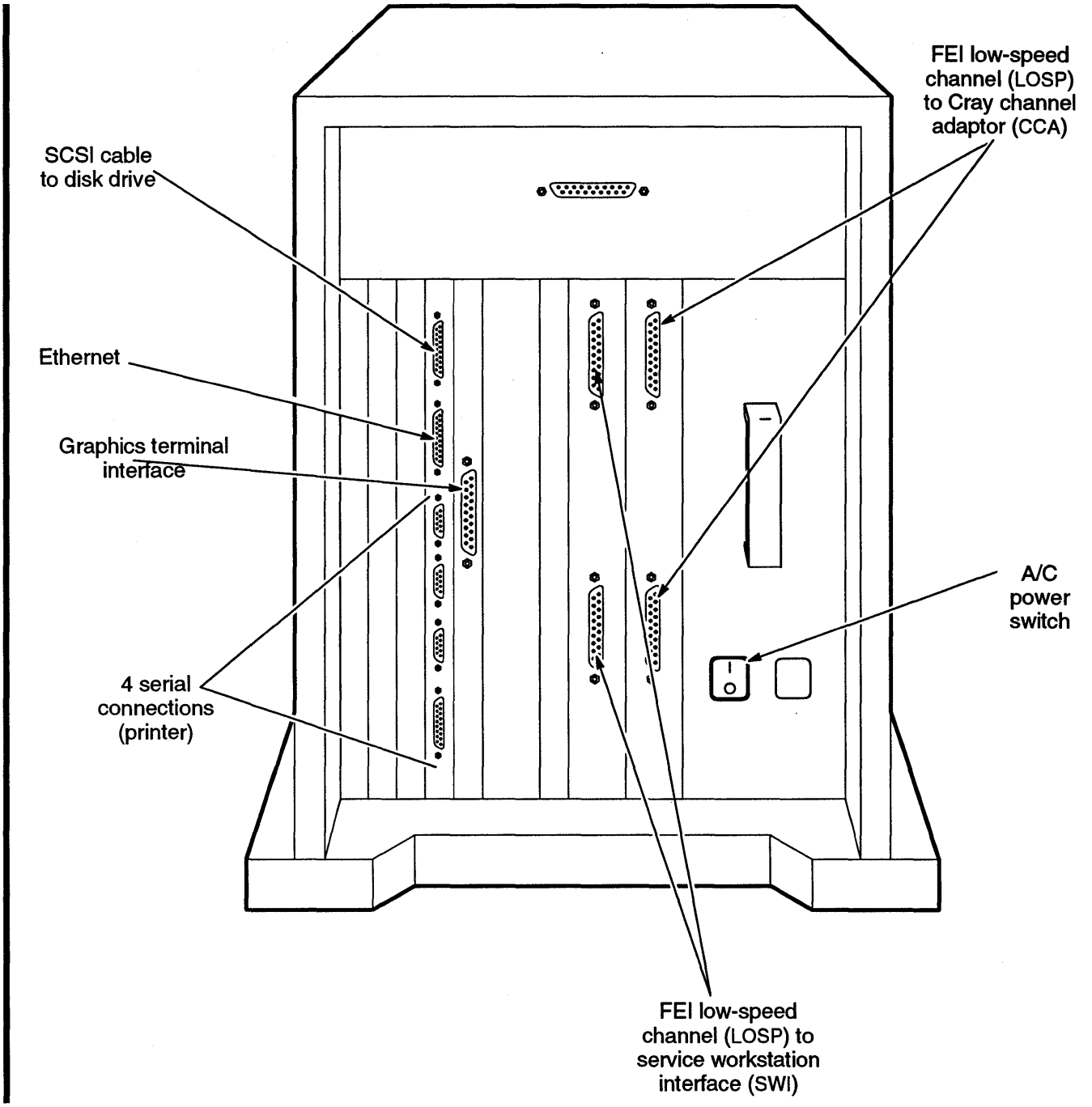


Figure 4. OWS-E backplane connections

OWS-E Software System Overview [3]

This section provides an overview of the OWS-E software system.

Software components 3.1

The OWS-E runs the following software:

- SunOS, the Sun Microsystems, Inc., operating system based on the UNIX operating system
- OpenWindows, a graphical user interface based on the OPEN LOOK standard
- NeWSprint, Sun Microsystems, Inc., printer formatting software
- AnswerBook, the Sun Microsystems, Inc., on-line documentation
- OWS-E software provided by CRI

The following subsections describe the OWS-E disk partitions and directory structure.

Disk partitions
3.2

The OWS-E disk contains a total of 669 Mbytes. Figure 5 shows the partition names and sizes; it gives a graphic representation of size, although it is not drawn exactly to scale. The letters at the left correspond to the names of the various partitions, in the form `/dev/sd0`*n*; the whole disk is `/dev/sd0c`. There is no `e` partition.

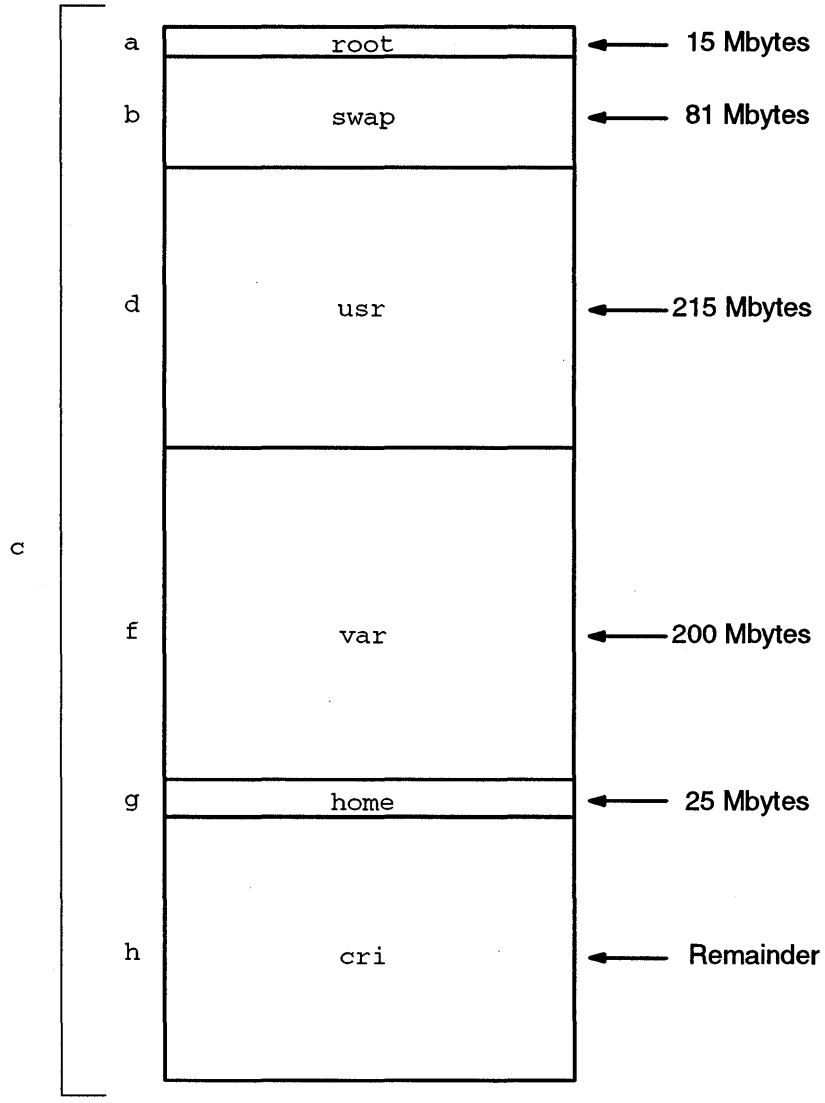


Figure 5. Disk partitions

Directory structure 3.3

Figure 6 shows some of the important files and directories in the OWS-E software. Most OWS-E commands are located under `cri`. The `.version` file contains the version number of the OWS-E software loaded on your system.

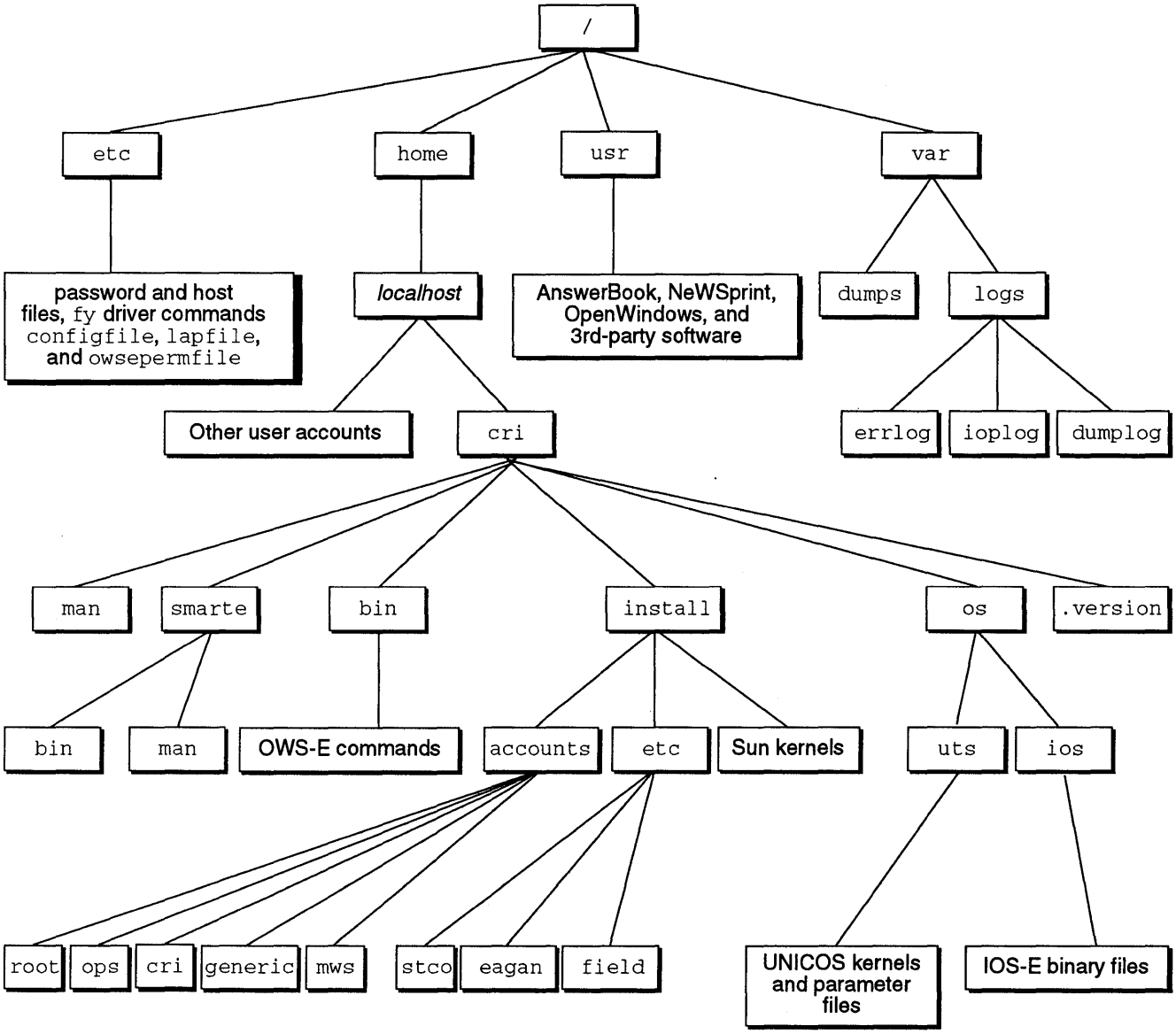


Figure 6. Directory structure

Initialization Files [4]

Initialization Files [4]

The default graphical user interface for the OWS-E is OpenWindows from Sun Microsystems, Inc. The components of this software include an X11/NeWS merged server, the XView toolkit, and the OPEN LOOK window manager (`olwm`).

This section describes the initialization files required by the OpenWindows environment and changes that have been made by CRI to other files and variables in order to support the OpenWindows environment.

Required files

4.1

The following files are required for support of the OpenWindows environment:

File	Description
`.openwin-init`	Provides the OpenWindows system with information about which windows should be displayed initially, where they are placed, and other window-specific details. If you do not have a `.openwin-init` file in your login directory, OpenWindows will use a generic one from its library.
`.openwin-menu`	Specifies session environment aspects, such as the entities that appear in the root menu. This file is read by the OPEN LOOK window manager (`olwm`).

Changed files
4.2

The following SunOS files have been changed to support the OpenWindows environment:

File	Description of changes

`.cshrc` (C shell)	Sets the `OPENWINHOME` variable to the directory in which the OpenWindows software is installed and adds the following path names to the `PATH` variable:
	`$OPENWINHOME/bin`
	`$OPENWINHOME/bin/xview`

`.login` (C shell)	Executes `openwin`, which then proceeds to read the `.xinitrc` file in setting up the user session environment.
`.profile` (Bourne shell)	Performs the same actions as `.login` and `.cshrc` combined.
`.xinitrc`	Reads the `.Xdefaults` file to the resource database, starts the `olwm` window manager, and then calls and executes the `.openwin-init` file to start the user's default display configuration. Normally, the operator should have the following line set in the `.xinitrc` file: `craymon &` This specifies that `craymon`(8) will execute in the background for as long as the operator is logged in.
`.Xdefaults`	Specifies various environmental aspects for the window manager, such as how many lines should be saved when scrolling. Several items not required for a pure X11 environment have been added for the OpenWindows system environment. For example, the `OpenWindows.SetInput` attribute is set so that the cursor follows the mouse as it moves across the screen; the alternative would be a click to focus option.

Environment variable settings 4.3

By default, the following environment variable is set to the value shown in `.cshrc` (for the C shell) or `.profile` (for the Bourne shell):

Variable	Value
MANPATH	`/usr/man:/usr/contrib/man:/usr/man/mann:$OPENWINHOME/share/man:/home/$HOST/cri/man`

For more information about these environment variables and settings, see the SunOS manual pages for `openwin`(1) and `xnews`(1) in *SunOS Reference Manual.*

System Configuration File [5]

System Configuration File [5]

The system configuration file, `/etc/configfile`, contains the system parameter labels and their corresponding values used by the OWS-E software. This file was created to allow changing machine dependencies at the time of execution. This file also contains other volatile variables, such as path names to the various binary and log files.

How does `/etc/configfile` work?

5.1

Whenever the OWS-E system needs to know the value of a particular variable that is supplied in `/etc/configfile` (such as the default path name to the UNICOS kernel), it calls a library routine. The `config` routine searches `/etc/configfile` for the label and then returns the associated string value. The `getconfig`(8) command is the shell interface to the `config` routine.

For more information about these commands, see *OWS-E Operator Workstation Reference Manual.*

How can I make changes to /etc/configfile?
5.2

Before making changes to /etc/configfile permanent, you should test those changes. To do this, follow these steps:

1. Copy /etc/configfile to a temporary location, such as your home directory.
2. Edit the file in this new location.
3. Set the OWSECONFIG environment variable to be the full path name of this edited configuration file.

The config routine checks the environment variable and uses the new configuration file when answering system parameter requests; other users (who do not have the OWSECONFIG variable set) continue to get their defaults from the original /etc/configfile. When you are satisfied that your changes work properly, you can replace the original /etc/configfile with your edited version and unset OWSECONFIG.

Order of parameters in /etc/configfile
5.3

The system configuration file /etc/configfile is divided into two sections, as follows:

1. Configurable parameters set during the install process to site-specific values
2. Configurable parameters set at release time

The parameters are described here in Table 1, Table 2, and Table 3 in the order in which they are found in /etc/configfile.

The term IOP refers to either an EIOP or a MUXIOP.

Tokens
5.4

Many of the parameters contain tokens that are changed during the installation process to reflect the specific machine being installed. These tokens are designated by *__token__* (that is, they are preceded and followed by underscores). If any of these tokens remain in /etc/configfile following completion of the installation, they should be changed as instructed in Table 1, which lists the tokens in the order in which they appear in the file.

Table 1. Tokens

Token	Description
__HOSTNAME__	Replace this token with the name of the OWS-E. For example, if the serial number of the mainframe were 1600, __HOSTNAME__ could be ows1600.
__SERIALNUMBER__	Replace this token with the serial number of the CRI mainframe to which the OWS-E is attached. For example, if the mainframe serial number were 1600, __SERIALNUMBER__ should be 1600.
__DUMPTYPE__	Replace this token with the number that specifies the type of the disk to which the mainframe memory will be dumped. The disk drives and their valid numbers are as follows: Disk device / Number DD-40 6 DD-41 9 DD-42 13 DD-49 3 DD-50 7 DD-60 10 DD-61 11 DD-62 12
__DUMPIO__	Replace this token with a 4-digit octal number specifying the cluster, IOP, and channel path down which the memory will proceed to the disk. For example, 0130 specifies that the dump will be transferred to disk through cluster 0, IOP 1, and channel 30. The disk attached to the specified channel must be of the type specified in __DUMPTYPE__.

Table 1. Tokens
(continued)

Token	Description
__DUMPUNIT__	Replace this token with the default dump device unit number. DD-40s, DD-42s, DD-60s, and DD-62s have unit numbers; everything else uses 0.

Disk drive	Valid unit numbers
DD-40	0, 1
DD-41	0, 1
DD-42	0, 1
DD-49	0
DD-50	0
DD-60	0, 1, 2, 3, 4, 5, 6, 7
DD-61	0, 1, 2, 3, 4, 5, 6, 7
DD-62	0, 1, 2, 3, 4, 5, 6, 7

__DSTARTBL__	Replace this token with the start of the slice on the disk to which the memory will be dumped. This value is designated in decimal blocks.
__DUMPLEN__	Replace this token with the length of the slice on the disk to which the memory will be dumped. This value is designated in decimal blocks.
__MAINFRAME__	Replace this token with the number that specifies the type of the mainframe to which the OWS-E is attached. The mainframe types and their respective numbers are as follows:

Mainframe type	Number
CRAY Y-MP C90 (1 to 16 CPUs)	100
CRAY Y-MP 8I (1 to 8 CPUs)	7
CRAY Y-MP 8E (1 to 8 CPUs)	8
CRAY Y-MP 4E (1 to 4 CPUs)	9
CRAY Y-MP 2E (1 or 2 CPUs)	6
CRAY Y-MP (8 CPUs) (only certain serial numbers)	0

Table 1. Tokens
(continued)

Token	Description
__MEMORY__	Replace this token with the number that specifies the memory size of the mainframe to which the OWS-E is attached. The numbers are as follows:

Memory size	Number
4 Mwords	4
8 Mwords	8
16 Mwords	16
32 Mwords	32
64 Mwords	64
128 Mwords	128
256 Mwords	256
512 Mwords	512
1024 Mwords	1024
2048 Mwords	2048
4096 Mwords	4096

Table 1. Tokens
(continued)

Token	Description
__SSD_MEMORY__	Replace this token with the number that specifies the memory size of the SSD-E solid-state storage device attached to the mainframe to which the OWS-E is attached. The numbers are as follows:

SSD memory	Number
No SSD	0
4 Mwords	4
8 Mwords	8
16 Mwords	16
32 Mwords	32
64 Mwords	64
128 Mwords	128
256 Mwords	256
512 Mwords	512
1024 Mwords	1024
2048 Mwords	2048
4096 Mwords	4096

__CPUDHOSTNAME__	Replace this token with the name of the machine on which cpud(8) is running. This passive CPU monitor normally executes on the OWS-E. For example, if the serial number of the mainframe were 1600, __CPUDHOSTNAME__ could be replaced with ows1600.

Configurable parameters set during installation

5.5

This subsection describes the parameters that are set during the installation process (see the *OWS-E 2.0 Release and Installation Notes* for more information). Parameters that you should not change are noted. Parameters are listed in the order in which they appear in the file.

Table 2. Configurable parameters set during the installation process

Parameter	Description
ROOTDIR	Defines the base directory that scripts use to find the CRI commands that they execute during processing. At release, this directory is the `cri` home directory: `/home/__HOSTNAME__/cri` **Note** Do **not** change this parameter. The location of the `cri` directory is used by many different pieces of the OWS-E software, and it should not be changed.
MAIL_CPUFAIL	Defines the login name to which mail is sent if a CPU panics. At release, this parameter is set to `cri`. To specify more than one user, use the SunOS `aliases`(5) file.
MAIL_IOPFAIL	Defines the login name to which mail is sent if an IOP halts. At release, this parameter is set to `cri`. To specify more than one user, use the SunOS `aliases`(5) file.

Table 2. Configurable parameters set during the installation process (continued)

Parameter	Description
DEFAULTUKERNFILE	Defines the path name to the default UNICOS binary. This parameter is used by the system start utility, bootsys(8), when the default UNICOS kernel is to be started on the mainframe. At release, this parameter is set to the name of the default UNICOS binary initially installed: /home/__HOSTNAME__/cri/os/uts/unicos It is likely that you will change this parameter. Although the kernel file should reside under the cri home directory, it is likely that the actual name of the default kernel to be booted will change from site to site. For example, if unicos.prod.day were the name of the default kernel at your site, this parameter would be set to the following: /home/__HOSTNAME__/cri/os/uts/unicos.prod.day
DEFAULTUPARAMFILE	Defines the path name to the default UNICOS parameter file. This parameter is used by the bootsys(8) command when the default parameter file is requested. At release, this parameter is set to the name of the default UNICOS parameter file: /home/__HOSTNAME__/cri/os/uts/param It is likely that you will change this parameter. Although the parameter file should reside under the cri home directory, the actual name of the default parameter file will change from site to site. For example, if param.prod.day were the name of the default parameter file at your site, this parameter would be set to the following: /home/__HOSTNAME__/cri/os/uts/param.prod.day

Table 2. Configurable parameters set during the installation process (continued)

Parameter	Description
DEFAULTIKERNDIR	Defines the path name to the directory in which all the IOS-E binary files are kept. This parameter is used by the `bootsys`(8) command upon a request that the default IOS-E binary files be started. At release, this parameter is set to the following: `/home/__HOSTNAME__/cri/os/ios` **Note** Do **not** change this parameter. To ensure consistency and the availability of support, you should not change the directory structure underneath the `cri` home directory.

CPUPANIC	Defines the path name of the `cpupanic`(8) script. The `rcpud`(8) command executes this script when it receives a panic packet. This script checks the dump lock file, described under the `ADUMPDIR` and `DUMP` parameters (page 27); if the lock is set, the `cpupanic` script proceeds to boot the IOS-E and take a dump image of the mainframe, using `mfdump`(8). At release, this parameter is set to the following:
	`/home/__HOSTNAME__/cri/bin/cpupanic`
	You should change this parameter if you create a different panic script or if you change the name of the script. You should keep all scripts and commands in the following directory:
	`/home/__HOSTNAME__/cri/bin`

Table 2. Configurable parameters set during the installation process (continued)

Parameter	Description
IOPHALT	Defines the path name of the iophalt(8) script. This script is executed by the hbeat(8) command whenever it detects an IOP halt or failure. This script checks the dump lock file, described under the ADUMPDIR and DUMP parameters (page 27); if the lock is set, the iophalt script proceeds to get dumps of the IOPs. At release, this parameter is set to the following: /home/__HOSTNAME__/cri/bin/iophalt You should change this parameter if you create a different halt script or if you change the name of the script. You should keep all scripts and commands in the following directory: /home/__HOSTNAME__/cri/bin
SERIALNUMBER	Defines the serial number of the CRI mainframe to which the OWS-E is attached. This parameter is used by the shell scripts. At release, this parameter is set to the __SERIALNUMBER__ token and is replaced during the installation. If this token has not been replaced after completion of the installation, proceed with the directions for replacing the __SERIALNUMBER__ token (see page 19). **Note** You should change this parameter only if a change is made to the serial number of the mainframe to which the OWS-E is attached.
DEFAULTIOP	Defines the default IOP through which the IOS-E is booted. This parameter is used by the bootsys(8) command and shell scripts. At release, it is set to IOP 0. This parameter may be set to any of the configured IOPs within the cluster specified by DEFAULTCLSTR, as long as the specified IOP has the deadstart capability.

Table 2. Configurable parameters set during the installation process (continued)

Parameter	Description
DEFAULTIDUMPDIR	Defines the default dump directory path in which the dump shell script is created. This parameter is used by the `bootsys`(8) command. At release, this parameter is set to the following: `/var/dumps` This parameter may be set to any existing directory desired; however, it is important to organize all dump information into a coherent directory structure.
ADUMPDIR	Defines the path name of the directory holding the dump lock file mentioned in the descriptions of the `CPUPANIC` and `IOPHALT` parameters. The dump lock file indicates to the `cpupanic`(8) and `iophalt`(8) scripts whether or not they should automatically take dumps upon receipt of halts or panics. The `autodump`(8) command uses this parameter when setting the lock file on or off. At release, this parameter is set to the following: `/home/__HOSTNAME__/cri/bin/adm`
DUMP	Defines the name of the dump lock file mentioned in the descriptions of the `ADUMPDIR`, `CPUPANIC`, and `IOPHALT` parameters. The dump lock file indicates to the `cpupanic` and `iophalt` scripts whether or not they should automatically take dumps upon receipt of halts or panics. The `autodump`(8) command uses this parameter when setting the lock file on or off. At release, this parameter is set to the following: `dump.on`
ERRLOGD	Defines the path name of the error logging daemon. This daemon serves as the hardware error log program for the IOS-E. It opens up an error channel, sends the messages it reads to the console, and also logs them. At release, this parameter is set to the following: `/home/__HOSTNAME__/cri/bin/errlogd`
SMDEMON	Defines the path name of the daemon that monitors the OWS-E for SMARTE. At release, this parameter is set to the following: `/home/__HOSTNAME__/cri/smarte/bin/smdemon`

Table 2. Configurable parameters set during the installation process (continued)

Parameter	Description
HBEAT	Defines the path name of the IOP monitor. This monitor opens up a heartbeat channel to the IOS-E, polls for halted or hung IOPs, and invokes the `iophalt` script. Messages are logged and sent to the console for operator intervention. At release, this parameter is set to the following: `/home/__HOSTNAME__/cri/bin/hbeat`
RCPUD	Defines the path name of the remote CPU request daemon. This daemon handles requests from the mainframe to perform workstation functions on behalf of the mainframe. It opens up a CPU channel and receives requests in the form of O packets. This command invokes the `cpupanic`(8) script. At release, this parameter is set to the following: `/home/__HOSTNAME__/cri/bin/rcpud`
CPUD	Defines the path name of the command that gathers data and disperses CPU time statistics. At release, this parameter is set to the following: `/home/__HOSTNAME__/cri/bin/cpud`
EDIAG	Defines the path name of the command that boots deadstart diagnostics into a specified IOP. At release, this parameter is set to the following: `/home/__HOSTNAME__/cri/smarte/bin/ediag`
EBOOT	Defines the path name of the command that boots an IOP from the OWS-E. At release, this parameter is set to the following: `/home/__HOSTNAME__/cri/bin/eboot`
AUTODUMP	Defines the path name of the command that controls automatic dumping of the mainframe and the IOS-E. At release, this parameter is set to the following: `/home/__HOSTNAME__/cri/bin/autodump`

Table 2. Configurable parameters set during the installation process (continued)

Parameter	Description
ECON	Defines the path name of the command that configures a MUXIOP-to-EIOP low-speed channel up or down. At release, this parameter is set to the following: `/home/__HOSTNAME__/cri/bin/econ`
HCON	Defines the path name of the command that configures a MUXIOP high-speed channel up or down. At release, this parameter is set to the following: `/home/__HOSTNAME__/cri/bin/hcon`
MFINIT	Defines the path name of the command that runs a mainframe and IOS-E initialization and confidence test. At release, this parameter is set to the following: `/home/__HOSTNAME__/cri/bin/mfinit`
MFSTART	Defines the path name of the command that starts the mainframe CPU from the OWS-E. At release, this parameter is set to the following: `/home/__HOSTNAME__/cri/bin/mfstart`
IOSDPATH	Defines the path name of the IOP deadstart diagnostic. The `ediag`(8) command uses this parameter when booting `dsdiag` into a specified IOP. At release, this parameter is set to the following: `/home/__HOSTNAME__/cri/os/ios/dsdiag`
IOSCPATH	Defines the path name of the I/O clear diagnostic. The `ediag`(8) command uses this parameter when booting `cleario` into a specified IOP. At release, this parameter is set to the following: `/home/__HOSTNAME__/cri/os/ios/cleario`

Table 2. Configurable parameters set during the installation process (continued)

Parameter	Description
IOPDEBUG	Defines the path name of a temporary file that the ecrash(8) utility uses during its processing. At release, this parameter is set to the following: /home/__HOSTNAME__/cri/os/ios/iopdebug **Note** Do **not** change this parameter. To ensure consistency and the availability of support, you should not change the directory structure underneath the cri home directory. Also, this file is an internal holder that is visible here only because of its dependency on the specific host name of the OWS-E at each specific site.
IOPSAVE	Defines the path name of a temporary file used by the edump(8) utility during its processing. At release, this parameter is set to the following: /home/__HOSTNAME__/cri/os/ios/iopsave **Note** Do **not** change this parameter. To ensure consistency and the availability of support, you should not change the directory structure underneath the cri home directory. Also, this file is an internal holder that is visible here only because of its dependency on the specific host name of the OWS-E at each specific site.

Table 2. Configurable parameters set during the installation process (continued)

Parameter	Description
MFSYSDMP	Defines the path name of the CPU-resident program used by the mfdump(8) command. mfdump loads this program onto the mainframe to save the exchange package registers and any other requested memory types. At release, this parameter is set to the following: /home/__HOSTNAME__/cri/os/uts/mfsysdmp **Note** Do **not** change this parameter. To ensure consistency and the availability of support, you should not change the directory structure underneath the cri home directory. This program is placed in its directory during the installation process.
MFBOOT	Defines the path name of the bootstrap loader program used by the mfdump(8) command. At release, this parameter is set to the following: /home/__HOSTNAME__/cri/os/uts/mfboot **Note** Do **not** change this parameter. To ensure consistency and the availability of support, you should not change the directory structure underneath the cri home directory. This program is placed in its directory during the installation process.

Table 2. Configurable parameters set during the installation process (continued)

Parameter	Description
MFIPATH	Defines the path name of the diagnostic program used by the `mfinit`(8) command. This diagnostic program loads data into registers and stores the results in memory; `mfinit` then validates the results. At release, this parameter is set to the following: `/home/__HOSTNAME__/cri/os/uts/mfchkye` **Note** Do **not** change this parameter. To ensure consistency and the availability of support, you should not change the directory structure underneath the `cri` home directory. This program is placed in its directory during the installation process.
DEF_MFCHAN	Defines the mainframe channel number of the low-speed channel attached to the cluster that deadstarts the mainframe. It is used to route the time-and-date and terminal I/O packets until the parameter file is processed. At release, this parameter is set to the following: `020`
DTYPE	Defines the type of the disk in which the mainframe memory will be dumped. The `mfdump`(8) command uses this parameter when dumping mainframe memory. At release, this parameter is set to the `__DUMPTYPE__` token and is replaced during the installation. If this token has not been replaced after completion of the installation, proceed with the directions above for replacing the `__DUMPTYPE__` token (see page 19).
DIOPATH	Defines the path that the memory dump will take from the mainframe to the disk. The `mfdump`(8) command uses this parameter when dumping mainframe memory. At release, this parameter is set to the `__DUMPIO__` token and is replaced during the installation. If this token has not been replaced after completion of the installation, proceed with the directions above for replacing the `__DUMPIO__` token (see page 19).

Table 2. Configurable parameters set during the installation process (continued)

Parameter	Description
DUNIT	Defines default dump device unit that the mfdump(8) command uses when routing the mfsysdmp binary to the mainframe prior to the dump. The value set for this parameter must match the hardware address. At release, this parameter is set to the __DUMPUNIT__ token.
DSTART	Defines the starting sector of the slice on the disk to which the memory will be dumped. The mfdump(8) command uses this parameter when dumping mainframe memory. At release, this parameter is set to the __DSTARTBL__ token and is replaced during the installation. If this token has not been replaced after completion of the installation, proceed with the directions for replacing the __DSTARTBL__ token (see page 20).
DLEN	Defines the length, in sectors, of the slice on the disk where the memory will be dumped. The mfdump(8) command uses this parameter when dumping mainframe memory. At release, this parameter is set to the __DUMPLEN__ token and is replaced during the installation. If this token has not been replaced after completion of the installation, proceed with the directions above for replacing the __DUMPLEN__ token (see page 20).
D0FWA D0LWA D1FWA D1LWA D2FWA D2LWA D3FWA D3LWA	These parameters define the actual mainframe memory ranges to be dumped. An error will not occur if these ranges overlap, but overlapping them will produce redundant results. Four ranges are available for specification. At release, only the first range is specified, and the other ranges are set to 0. This first range is set to start at word address 0 and end at word address 020000000 (4 million).
MAINFRAME	Defines the type of mainframe to which the OWS-E is attached. It is used in conjunction with the machd.h header file to define the machine-specific characteristics of the mainframe. At release, this parameter is set to the __MAINFRAME__ token and is replaced during the installation. If this token has not been replaced after completion of the installation, proceed with the directions for replacing the __MAINFRAME__ token (see page 20).

Table 2. Configurable parameters set during the installation process (continued)

Parameter	Description
M_MEMORY	Defines the memory size of the mainframe to which the OWS-E is attached. It is used in conjunction with the machd.h header file to define the machine-specific characteristics of the mainframe. At release, this parameter is set to the __MEMORY__ token and is replaced during the installation. If this token has not been replaced after completion of the installation, proceed with the directions for replacing the __MEMORY__ token (see page 21).
SSD_MEMORY	Defines the memory size of the SSD attached to the mainframe to which the OWS-E is attached. It is used in conjunction with the machd.h header file to define the machine-specific characteristics of the SSD. At release, this parameter is set to the __SSD_MEMORY__ token and is replaced during the installation. If this token has not been replaced after completion of the installation, proceed with the directions for replacing the __SSD_MEMORY__ token (see on page 22).
CPUD_HOSTNAME	Denotes the name of the machine to which the CPU monitor, cpud(8), is running. The graphs(8) utility uses this parameter as a default host name when the -h option is not specified on the command line. At release, this parameter is set to the __CPUDHOSTNAME__ token and is replaced during the installation. If this token has not been replaced after completion of the installation, proceed with the directions for replacing the __CPUDHOSTNAME__ token (see page 22).

Note

Do **not** change this parameter. This parameter should always be the name of the OWS-E attached to the mainframe. cpud must execute on the OWS-E because it must use the low-speed channel (from the OWS-E to the IOS-E) for viewing mainframe memory when gathering statistics.

Configurable parameters set at release time 5.6

This subsection describes the parameters that are set at release time but are configurable if changed with caution. Parameters that you should not change are noted. Parameters are listed in the order in which they appear in the file.

Table 3. Configurable parameters set at release time

Parameter	Description
UPDATESECS	Defines (in seconds) the polling rate for the passive CPU monitors. Every UPDATESECS seconds, the pw kernel data structure is read from mainframe memory across the MUXIOP. At release, this rate is set to 5 seconds. **Note** This value must be set to at least 1 second.
BASEPORT	Defines the starting port value used by the cpud(8) daemon, which adds 2 to the value for BASEPORT and uses that value as its port on which to listen. At release, BASEPORT is set to the following: 4370 This value would be changed if the particular range that the ports are using is already designated for a different use by the site. Changing BASEPORT to an acceptable starting port value will move all ports used into that range.
SSTBACKUP	Specifies the back-up hbeat(8) status table. At release, this parameter is set to the following: /var/logs/sstbackup

Table 3. Configurable parameters set at release time (continued)

Parameter	Description
ERRLOG	Defines the path name of the error log file. This log file is written to by errlogd(8), which is the hardware error log program for the IOS-E. At release, this parameter is set to the following: /var/logs/errlog **Note** Do **not** change this parameter. The /var file system has been allocated the space necessary to serve as a log file location. This may not be true of other file systems; /home, for example, would be filled quickly if log files were being placed in that file system.
IOPLOG	Defines the path name of the IOP log file. This log file is used by all of the OWS-E commands for logging errors, warnings, and informational messages regarding the interaction of the commands with the IOS-E and CRI mainframe. At release, this parameter is set to the following: /var/logs/ioplog **Note** Do **not** change this parameter. The /var file system has been allocated the space necessary to serve as a log file location. This may not be true of other file systems; /home, for example, would be filled quickly if log files were being placed in that file system.
IOP_DIAGNOSTICS	Controls whether diagnostics are run by bootsys(8) before booting the IOS-E. Valid values are on, off, and only. CRI recommends that you run diagnostics before each boot. At release, this parameter is set to the following: on

Table 3. Configurable parameters set at release time (continued)

Parameter	Description
LAPFILE	Specifies the location of the line-arbitration priority file used by `lapdaemon`(8). At release, this parameter is set to the following: `/etc/lapfile`
CRAYMON	Defines the colors of the OWS-E that denote whether the mainframe is up (first color) or down (second color). At release, this parameter is set to the following: `SkyBlue,red` Note that the two colors must be separated by a comma (,); there cannot be an intervening space.

Security [6]

Security [6]

The following suggestions for passwords and user and group identification will help keep the OWS-E more secure. (For more details about security, see *System and Network Administration*, produced by Sun Microsystems, Inc.) The OWS-E permissions file (described in subsection 6.3, page 40) allows you to specify who can access particular commands.

Passwords
6.1

Each person who logs into the OWS-E should have a unique password. Passwords should be at least 6 characters long, and at least one of those characters must be nonalphabetic.

The following are general guidelines you can give your operators so that they can choose secure passwords:

- DO NOT use your login name, first name, or last name, in any form.
- DO NOT use easily obtained personal information (such as telephone number, type of automobile, child's name, license plate number, and so on).
- DO NOT use any words found in the dictionary or any proper names.
- DO NOT repeat the same letter or number as the entire password.

When choosing a password, please keep in mind that the object is to make it as difficult as possible for someone to guess what you have chosen or a password.

User and group identification
6.2

In general, a person's login name should be the same on the OWS-E as it is on the CRI systems. To simplify administration in a networked environment, group IDs should also be the same.

OWS-E permissions file 6.3

The OWS-E permissions file, `/etc/owsepermfile`, allows you to specify those who can access the following OWS-E commands:

autodump	edump	mfdump
bootsys	ehalt	mfinit
cpuhalt	emon	mfstart
craymon	eping	peek
dumpsys	errlogd	poke
eboot	estat	rcpud
econ	hbeat	zip
ecrash	hcon	

When a user logs in, the `valid_user` library routine examines the access list in `/etc/owsepermfile` to determine which commands the user may execute. If the user tries to execute a command for which he or she does not have access, the following error message is issued:

```
ERROR: progname: User username not validated for use
```

There must be an entry in `/etc/owsepermfile` for every account that wants to access these commands. The user ID must be the first item on a line, followed by the commands and scripts that the user is allowed to access. You can use space, tabs, or a colon to separate the user ID from the list, and you can separate items within the list by spaces, tabs, or commas. An asterisk (*) indicates that the user is permitted to access all commands and scripts. If you want to include comments, precede them by the pound sign (#).

When you assign permission, you must be aware of the hierarchy of commands; that is, you must know which commands call other commands. For this information, see the command man pages in *OWS-E Operator Workstation Reference Manual.*

Note

All users included in `/etc/owsepermfile` must also be included in the password file. If a specified user is not in the password file, errors will result.

The following is an example of an access list in `/etc/owsepermfile`:

```
bgj     mfstart,edump,eboot
swj:mfstart,edump  eboot
emh mfstart   edump    eboot
elw     *                    # do anything!
```

This file allows `bgj`, `swj`, and `emh` to access only the `mfstart`(8), `edump`(8), and `eboot`(8) commands; `elw` can access all commands.

Setting Line Arbitration Priorities [7]

Setting Line Arbitration Priorities [7]

The lapdaemon(8) program uses a priority file to determine the priority of users. The location of this file is specified by the LAPFILE ("line arbitration priority file") parameter in /etc/configfile; by default, LAPFILE is set to /etc/lapfile.

Users not listed in /etc/lapfile have a default priority of 0; that is, they have no priority and cannot usurp a tty line.

You can specify a priority higher than 0 for particular users by including their priority numbers and login IDs in /etc/lapfile, as follows:

prioritynumber login [, *login* ...]

The priority number must be separated from the login ID by white space, such as a tab or space. You can include comments in the file by beginning the comment with a pound sign (#); blank lines are ignored. You can specify a single priority level for multiple users by placing their logins on one line, separating the logins with commas, as follows:

```
5 john, mary, louise
```

Logins used in /etc/lapfile must begin with an alphabetic character. The priority can be any positive integer; the higher the number, the higher the priority. (Negative numbers are not allowed.)

To put changes to /etc/lapfile into effect, you must send the lapdaemon(8) program a HUP (hang up) signal. To do this, find the process identification (PID) number of lapdaemon with the SunOS ps(1) command and then kill the PID with the following command line (in which *lpid* is the lapdaemon PID number):

```
ows1600% kill -HUP lpid
```

For example, if you wanted `chris` and `terry` to have a priority greater than `pat` but less than `cri`, your `/etc/lapfile` file might contain the following:

```
# /etc/lapfile PRIORITY FILE

# Logins not listed have a default priority
# of 0.

1     root
4     chris, terry
20    cri
3     pat   #pat should be lower than chris
```

With OWS-E 2.0, the edump(8) file format has been changed so that one file can hold multiple clusters.

Note

If you have unprocessed dumps from previous OWS-E releases, you must convert them to the new edump file format with the conv(8) command before processing them with ecrash(8).

Figure 7 on the following page shows the sections that constitute a dump file. The remaining sections show the C language structures for the file header, section header, and register section of the dump file.

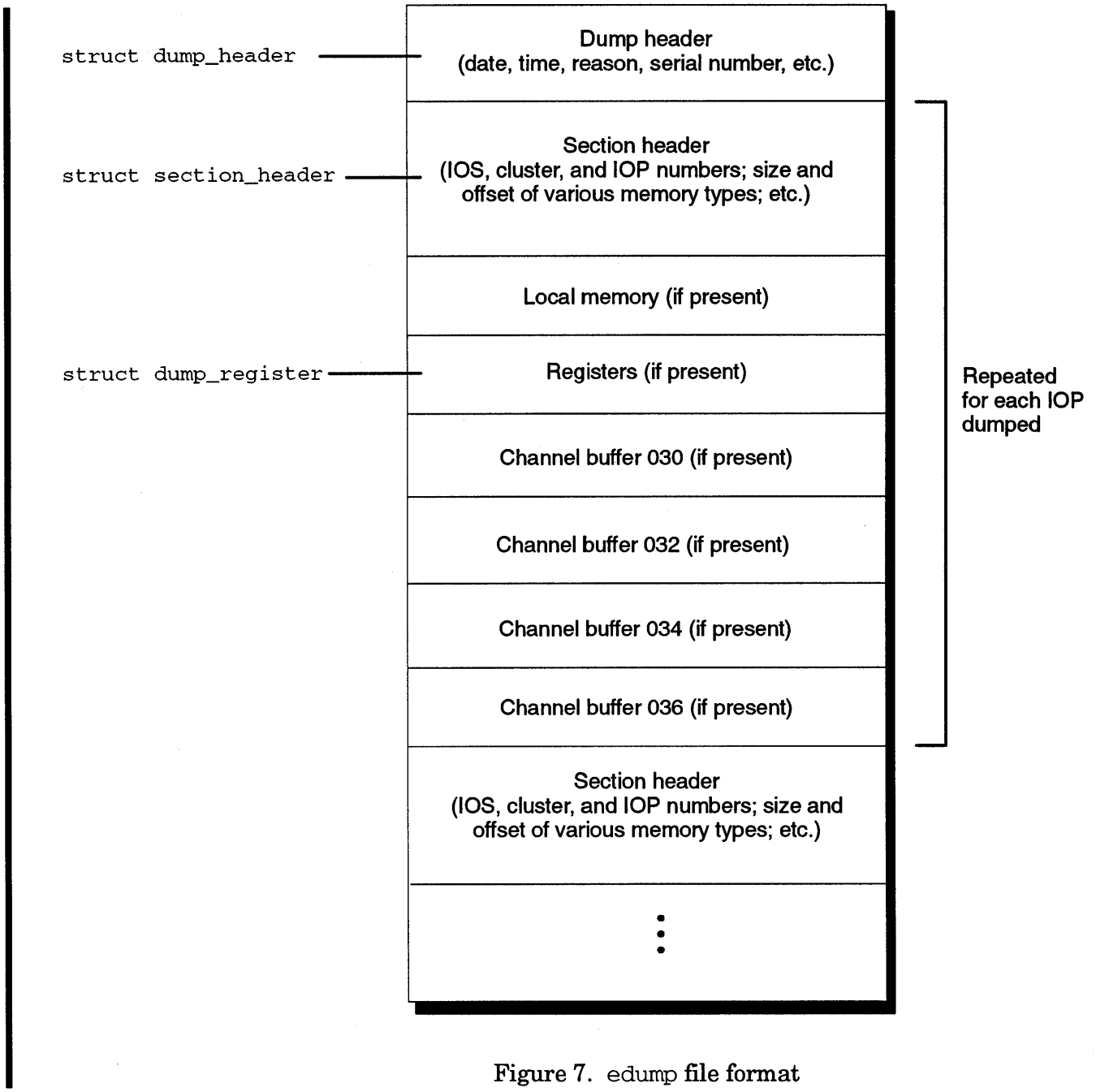


Figure 7. edump file format

File header structure

8.1

The following is the C code structure for the header of an edump(8) file:

```
#define SERIAL_SIZE             8       /* Size of the serial # field*/
#define DUMP_REASON_SIZE        80      /* Size of the reason field */
/*
 * Definition of the header of an edump file.
 */
struct dump_header {
        int     magic ;                         /* Magic number for dumpfiles */
        char    serial[SERIAL_SIZE] ;           /* serial # of machine dumped */
        int     year:16 ;
        int     month:8 ;                           /* Date stamp */
        int     day:8 ;
        int     hour:8 ;                            /* Time stamp */
        int     min:8 ;
        int     completed:8 ;                       /* Completed flag */
        int     spare[4] ;
        char    reason[DUMP_REASON_SIZE] ;          /* Reason for dump */
} ;
```

Section header structure 8.2

The following is the C code structure for the section header of an edump(8) file:

```
#define NCB             4               /* Maximum number of CBs per IOP */

/*
 * Definition of memory section.
 */
struct  memory  {
        int     offset ;
        int     size ;
} ;

typedef struct memory mem_t ;


/*
 * Definition of the section header of an edump file.
 *
 * There seems to be no sensible common denominator, so we just use
 * the most natural unit for that target.
 *
 * Note that the mem_t for local memory is referenced in parcels, the
 * mem_t for cb in words, and register_size is in bytes.
 *
 */
struct section_header {
        int     ios:8 ;                 /* IOS # */
        int     cluster:8 ;             /* Cluster # */
        int     iop:8 ;                 /* IOP # */
        int     type:8 ;                /* IOP type (currently unused)*/
        mem_t   local ;                 /* details of local mem */
        int     register_size ;         /* size of registers dumped */
        mem_t   cb[NCB] ;               /* details of channel buffers */
        int     spare[4] ;
} ;
```

Register section structure

8.3

The following is the C code structure for the register section of an edump(8) file:

```
#define NESTACK         32              /* Size of IOP exit stack */
#define NIOPCHAN        32              /* Number of channels on an IOP */
#define NIOPREGS        128             /* Number of registers in an IOP */
/*
 * Definition of the register section (if present)
 */
struct  dump_register {
        parcel  a ;                     /* A register */
        parcel  c ;                     /* C register */
        parcel  b ;                     /* B register */
        parcel  e ;                     /* E register, also contains flags */
        parcel  intr ;                  /* Interrupt enable flag */
        parcel  base ;                  /* BASE register */
        parcel  p ;                     /* P register */
        parcel  unused ;
        parcel  es[NESTACK] ;           /* e[0] -> e[n] */
        parcel  flags[NIOPCHAN/8] ;     /* channel flags (8 per parcel) */
        parcel  padding[4] ;
        parcel  r[NIOPREGS] ;           /* r[0] -> r[n] */
} ;
```

Customizing CRI Template Scripts [9]

Customizing CRI Template Scripts [9]

If you use the scripts provided by CRI, you should customize them for your site by editing them and the system configuration file (`/etc/configfile`). You should also provide your operators with guidelines for using these scripts.

The following are the template scripts:

Script	Description
`cpupanic`(8)	Takes a UNICOS panic dump image
`dumpdly`(8)	Performs an incremental (level-9) backup of the OWS-E file systems
`dumpwkly`(8)	Performs a full (level-0) backup of the OWS-E file systems
`iophalt`(8)	Dumps an IOP in the event of an IOP failure

Figure 8 shows the interrelationships of these scripts and the OWS-E commands. (Scripts are shown in boxes.)

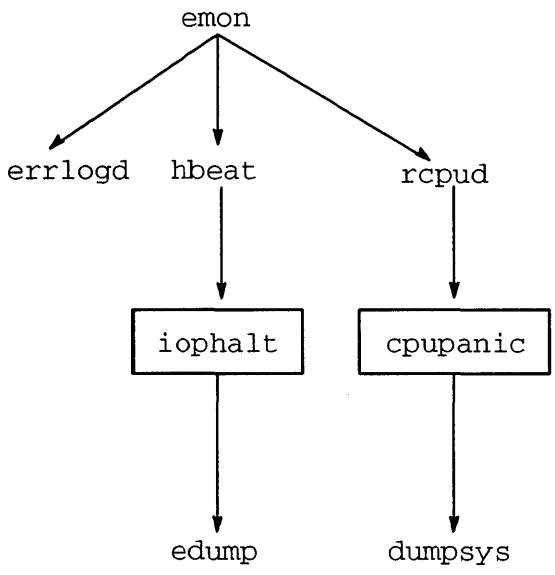


Figure 8. Interrelationships among scripts and commands

The subsections that follow describe the scripts you might want to modify and the modifications you should make to them and to `/etc/configfile`. You will have already made many changes to `/etc/configfile` when you install your OWS-E system; these changes are described in *OWS-E 2.0 Release and Installation Notes*.

For more information about these scripts and commands, see *OWS-E Operator Workstation Reference Manual* or enter the following at the system prompt:

```
man command          (or script)
```

cpupanic
9.1

The `cpupanic`(8) script takes a UNICOS panic dump image. The only modifications you will have to make for the `cpupanic` script are to set the `MAIL_CPUFAIL` parameter (in `/etc/configfile`) to the user ID of the person who should receive mail when a UNICOS panic occurs; to specify more than one person, you must use an alias (see the SunOS `aliases`(5) command). You might also want to reword the message sent.

iophalt
9.2

The `iophalt`(8) script dumps an IOP in the event of an IOP failure. You will not have to modify the script for your site unless you want to change the message sent. You will want to set the `DEFAULTIDUMPDIR` and `MAIL_IOPFAIL` parameters in `/etc/configfile` to the appropriate site-specific values:

- `DEFAULTIDUMPDIR` specifies the default dump directory on the OWS-E.
- `MAIL_IOPFAIL` specifies the person who should get mail about IOP failures. To specify more than one person, you must use an alias. See the SunOS `aliases`(5) command.

OWS-E backup scripts: `dumpdly` and `dumpwkly` 9.3

The `dumpdly`(8) and `dumpwkly`(8) scripts perform daily and weekly backups of the OWS-E, respectively. These scripts use the SunOS `dump`(8) command.

The `dumpwkly` script creates level-0 (full) dumps for each file system on the workstation. The `dumpdly` script creates level-9 (incremental) dumps for each file system. This weekly/daily backup routine ensures that complete recovery is possible.

The `dumpdly`(8) and `dumpwkly`(8) scripts assume the following file structure:

File system	Mounted on
`/dev/sd0a`	`/`
`/dev/sd0d`	`/usr`
`/dev/sd0g`	`/home`
`/dev/sd0h`	`/home/`*`localhost`*`/cri`
`/dev/sd0f`	`/var`

If your site's configuration does not match this, you must modify the scripts to ensure valid backups. Also, if your site does not use `/export`, you may want to comment it out of the scripts.

You should determine when it will be most advantageous for your site to execute these scripts. For example, if most of the work on the computers is done between 8:00 A.M. and 5:00 P.M. Monday through Friday, you might want to execute `dumpdly` at 6:00 P.M. every day and `dumpwkly` at 6:00 P.M. every Friday.

You may want to create a `crontab`(5) entry to run these scripts automatically, which will serve as a reminder for your operator to perform backups; an operator must be present to load tapes when the scripts are run. Your `crontab` entries for a daily and weekly dump at midnight could be as follows:

```
0 0 * * * /bin/sh /home/localhost/cri/bin/dumpdly
0 0 * * 7 /bin/sh /home/localhost/cri/bin/dumpwkly
```

For more information, see the man pages for SunOS `dump`(8) and `crontab`(5).

Shutting Down and Halting the OWS-E [10]

Shutting Down and Halting the OWS-E [10]

This section tells you how to perform shutdown and halt procedures for the OWS-E machine under normal circumstances and when the system is hung.

Normal circumstances 10.1

Under normal circumstances, you will shut down and halt the OWS-E before installing new software or removing power from the machine. Do the following:

1. Exit from OpenWindows by placing the mouse pointer on the workspace background, holding down the right button, and sliding down the menu to `Exit`. Confirm your decision to exit by clicking on `Exit` in the notice box that pops up.
2. Log out and log back in as the root user. You will be prompted to enter the root password. The password is not echoed.
3. Flush the disk buffers to disk by entering the following command:

   ```
   sync
   ```

4. Bring the machine into single-user mode by entering the following command:

   ```
   shutdown now
   ```

5. Enter the following command to halt the machine:

   ```
   halt
   ```

You will see the > prompt after you execute the `halt` command. When you see this prompt, you can safely power-off the machine. Figure 9 shows an example of steps 2 through 4. What you type is shown in boldface.

```
ows1600 login: root
Password: (what you type is not echoed)
Last login: Sun Aug 11 19:12:39 from console
SunOS Release 4.1.2 (CRI_FY) #1 Mon April 6 15:00:27 CDT 1992
Start OpenWindows now? [y] n

ows1600# sync
ows1600# shutdown now
          ***Final System Shutdown Message from root@ows1600 ***
                    .
                    .
                    .

ows1600# halt
syncing file systems . . .
Halted

>
```

Figure 9. Shutting down and halting the OWS-E under normal circumstances

For more information, see `halt`(8) and `shutdown`(8) in *SunOS Reference Manual.*

When the system is hung

10.2

If the OWS-E system is hung, you may have to shut down and immediately reboot it. Do the following:

1. Exit from OpenWindows by placing the mouse pointer on the workspace background, holding down the right button, and sliding down the menu to `Exit`. Confirm your decision to exit by clicking on `Exit` in the notice box that pops up.
2. Log out and log back in as the root user. You will be prompted to enter the root password. The password is not echoed.
3. Flush the disk buffers to disk by entering the following command:

   ```
   sync
   ```

4. Bring the machine into single-user mode by entering the following command:

   ```
   shutdown now
   ```

5. Enter the following command to reboot the machine immediately:

   ```
   fastboot
   ```

The machine will now reboot. Figure 10, page 58, shows an example of steps 2 through 5. What you type is shown in boldface.

```
ows1600 login: root
Password: (what you type is not echoed)
Last login: Sun Aug 11 19:12:39 from console
SunOS Release 4.1.2 (CRI_FY) #1 Mon June 3 15:00:27 CDT 1992
Start OpenWindows now? [y] n

ows1600# sync
ows1600# shutdown now
          ***Final System Shutdown Message from root@ows1600 ***
                    .
                    .
                    .

ows1600# fastboot
syncing file systems . . .
rebooting . . .
Boot: sd(0,8,0)
    .
    .
    .

ows1600 login:
```

Figure 10. Shutting down and rebooting the OWS-E when it is hung

For more information, see `fastboot`(8) and `shutdown`(8) in *SunOS Reference Manual.*

If the OWS-E system still does not respond, continue with the procedures outlined in section 10.3.

If all else fails: forcing a halt and reboot of the OWS-E

10.3

If the procedures outlined in section 10.2 are not successful or if the OWS-E is completely unresponsive, you can use the following procedures to force a halt and reboot of your OWS-E machine:

Caution

Reboot your OWS-E only as a last resort; this action does not sync the disks, which may cause damage to your file systems.

1. Press the [L1] and [a] keys simultaneously to clear the machine and bring up the > prompt.
2. Either press the [b] key to reboot the machine or ask the service engineer to perform diagnostics on the machine.

   If you accidentally press [L1] [a] and do not want to reboot, press the [c] ("continue") key. This will return you to where you were when you pressed [L1] [a] .

Manually Dumping the IOS-E and Mainframe [11]

Manually Dumping the IOS-E and Mainframe [11]

Your system may be set so that dump images of memory are taken automatically whenever UNICOS panics.

However, if `autodump`(8) is not set to `on`, the operator must take dumps manually by using the `dumpsys`(8) command.

To dump the entire system, the operator should first halt the CPUs with `cpuhalt`(8), then dump IOS-E memory with `edump`(8), and finally dump the mainframe with `dumpsys`(8). If the `cpuhalt` and `edump` commands are not executed, IOS-E memory information that may be useful in a postdump analysis will be lost.

The `edump`(8) command, without options, has the following format:

```
edump cluster:iop[:reason]
```

The operator must specify values for *cluster* and *iop* (the *reason* is optional). You should tell the operator which values should be used for *cluster* and *iop* to dump the entire IOS-E system. There is a place in *OWS-E Operator Workstation Operator's Guide* for the operator to write down this information.

If you want to specify something other than the defaults for `dumpsys`, you may want to verify your command line before you execute it. To do this, use the `-n` (no-execution) and `-v` (verbose) options. For example, suppose you have a parameter file named `param.test` that contains the following `dumpinfo` section:

```
dumpinfo {
        memory range is 0 to 2 Mwords;
        memory range is 30 to 32 Mwords;
        SSD range is 0 to 4 Mwords;
}
```

For example, if you want to specify cluster 1 rather than cluster 0, a parameter file named `param.test`, and verify your syntax, you can enter the following:

```
ows1600% dumpsys -c 1 -p param.test -nv
INFO: dumpsys: no-execute mode - mainframe will not be dumped.
INFO: dumpsys: Analyzing the parameter file 'param.test'.
Bootstrap binary : /home/ows1600/cri/os/uts/mfboot
Mfsysdump binary : /home/ows1600/cri/os/uts/mfsysdmp
Dump via cluster : 1
Boot cluster  1, iop 4 with /home/ows1600/cri/os/ios/iopmux
Boot cluster  1, iop 0 with /home/ows1600/cri/os/ios/eiop.dca2
Dump 2 CPUs, 3 cluster registers
Dump table memory, cluster registers, v, b, t registers - without force
Dump CPU from 00000000000 to 00010000000
Dump CPU from 00170000000 to 00200000000
Dump SSD from 00000000000 to 00020000000
Dump date : 03/02/92 time : 16:34:46
Dump device information :-
0: channel 022, iopath 00601034, type 10, unit 3, start 0, length 17250
```

If you want to include the `dumpinfo` section in the UNICOS parameter file, you must be running UNICOS 6.1.6 or later; if you are running UNICOS 6.0, you must use a separate parameter file that contains the `dumpinfo` section.

CPU Monitors [12]

CPU Monitors [12]

This section tells you how to monitor the CPUs from the OWS-E.

What do the CPU monitors do?
12.1

The CPU monitors allow you to determine whether the CPUs are spending their time on user work, on system work, in system-wait mode, or in idle mode.

How can I use the CPU monitors?
12.2

To use the CPU monitors, execute the `graphs` client by using the `graphs`(8) command, which is described in *OWS-E Operator Workstation Reference Manual.*

The `graphs` display consists of a line graph (similar to an EKG); it starts at the right side of the window and is initialized to a zero value. As the `graphs` client reads data from the data server's socket (see `cpud`(8)), the graph shifts to the left, with the latest value drawn in at the right side of the display. Upon filling the entire length of the display window, the graph continues to shift left and be updated on the right.

Descriptive text is shown beneath each graph displayed. The title bar of the window contains a `System Monitors` header, and additional text if specified on the `graphs` command line.

Note

If the OPEN LOOK window manager (`olwm`) is being used, the resize corners will also appear on the base frame. At this time, you cannot resize the graphs.

Figure 11, page 64, shows an example graph.

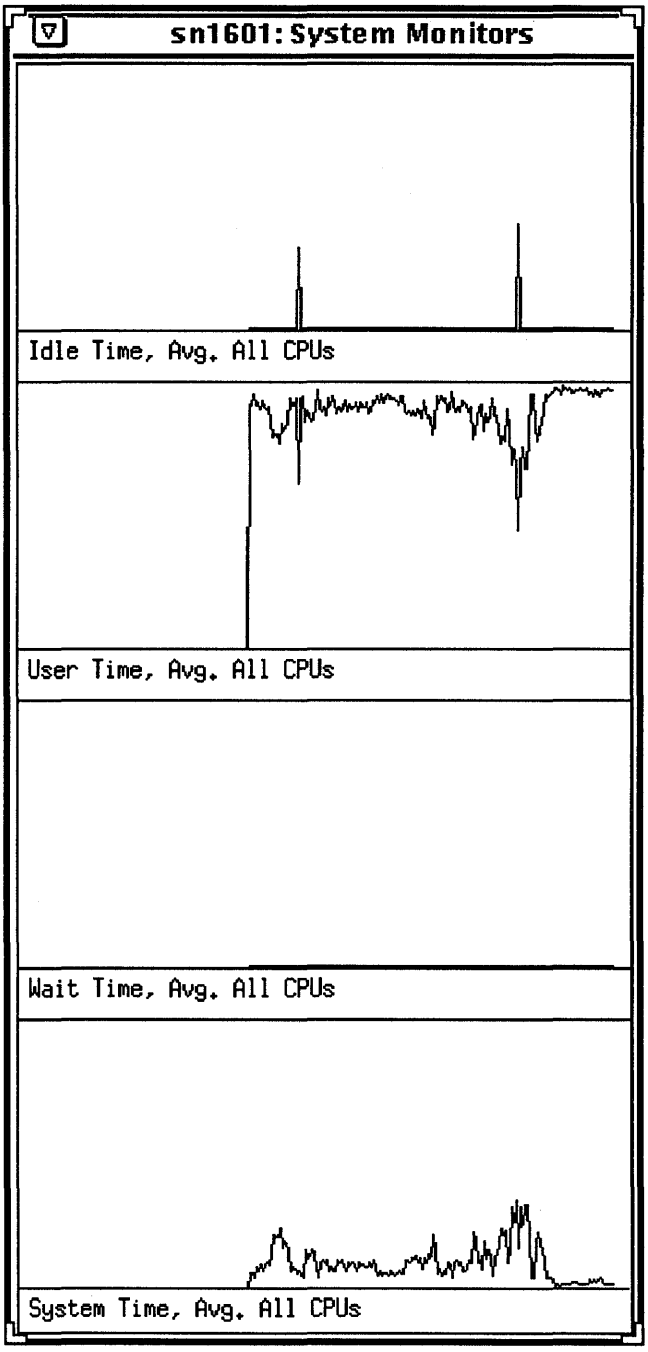


Figure 11. Monitor output for idle, user, system-wait, and system time averages for all CPUs

How do the CPU monitors work?

12.3

The CPU monitors that have been created for the IOS-E systems are unique in their implementation. Unlike other available monitors, such as `crayperf`(8), these monitors do not use CPU time themselves. This is because there is no monitoring process on the mainframe itself.

If the mainframe is down, the IOS-E is still able to peek (observe) mainframe memory. If the IOS-E is down, the monitors will fail with I/O errors when trying to peek through the IOP. Note that, as released, IOP 0 in cluster 0 is used as the peeking path in the monitor sources.

The monitor system consists of two main components:

- CPU daemon (`cpud`(8))
- `graphs` client (`graphs`(8))

The `cpud`(8) and `graphs`(8) commands run on the OWS-E. `graphs`(8) can be displayed on any X Window System, but can be run only from a node that supports OpenWindows or XView.

The CPU daemon interacts with an OWS-E library and the graphs client. First, the CPU daemon sets up a socket upon which it listens for client connections. Upon receiving a `graphs` client, it then searches through mainframe memory until it finds the `PWS` eyecatcher. With the address of the `PWS` thus found, `cpud` uses a library routine to peek at the element in the `cpuw` structure on which the number of configured CPUs is stored. This structure is used in determining the length in Cray words of the subsequent peeks (those that actually are gathering the desired statistics), which saves on the amount of data being channeled through the IOS-E. For example, if there were eight CPUs, the `pw` kernel data structure in question would have valid data of a length four times longer than if there were two CPUs.

The CPU daemon then proceeds to peek into mainframe memory every *update* seconds, as specified by the `UPDATESECS` parameter in `/etc/configfile` (see page 35 for more information about `UPDATESECS`). The default rate is 5 seconds. The `cpuw` is peeked and then dumped to a buffer consisting of a structure of unsigned integers. The elements in this structure are then converted from the 64-bit Cray word size to Sun 32-bit floating-point words. The current value is subtracted from the last value, and the result of the subtraction is divided by the time delta and then sent to the `graphs` client. As released, the `graphs` client multiplies these values by 100 in order to display them in percentage form.

The valid graphs are described by means of an array of graph structures. A single graph structure includes the name of the graph (used in the `graphs` client to determine which graph to display), the offset of its data into the unconverted data structure into which the mainframe peek is dumped, and the current data value for the graph in question.

The following graphs are supported:

Graph	Description
`idle`	Percentage of idle time over all CPUs
`idlec-`*n*	Percentage of idle time per CPU specified
`sysw`	Percentage of system wait time over all CPUs
`unix`	Percentage of system time over all CPUs
`unixc-`*n*	Percentage of system time per CPU specified
`user`	Percentage of user time over all CPUs
`userc-`*n*	Percentage of user time per CPU specified

n is the CPU number; the CPUs begin with number 0. If you had eight CPUs and you wanted to see graphs showing user time for the first CPU and the last CPU, you would enter `userc-0` and `userc-7`.

Network Monitor [13]

Network Monitor [13]

This section tells you how to use the xsnmpmon(8) network monitor from the OWS-E.

What is xsnmpmon?
13.1

The xsnmpmon(8) monitor is a program that uses the Simple Network Management Protocol (SNMP) to monitor a network on Transmission Control Protocol / Internet Protocol (TCP/IP). The monitor uses an X Window System graphical user interface. This monitor allows you to obtain the status of any SNMP-compliant machine on your network. (The CRI mainframe is SNMP-compliant, but the OWS-E is not.)

Background information on SNMP
13.2

Simple Network Management Protocol (SNMP) is a protocol that has become a *de facto* standard for managing networks based on TCP/IP. In a heterogeneous network environment, it is very difficult to monitor and manage the network components manufactured by different vendors without having a standard mechanism for conversing with the different nodes. Just as the UNIX system has made it easier for users to log in to a wide variety of machines made by a wide variety of computer manufacturers and do their work using the same user interface, SNMP has made it much easier to monitor and manage a multivendor network.

SNMP is defined by three Request for Comment (RFC) documents: RFC 1157, which defines the protocol itself; RFC 1155, which describes the structure in which management information is represented to the protocol; and RFC 1156, which defines all of the variables that make up the Management Information Base (MIB). The original MIB (called MIB-I) has been extended and clarified; MIB-II is a superset of MIB-I, and is in the process of becoming a standard. Not all vendors have implemented MIB-II as of yet, but the xsnmpmon monitor does support both MIB-I and MIB-II.

In SNMP, there are entities referred to as *clients* and *agents*. The client makes requests of the agent; the agent performs the requested action (after any necessary authentication) and responds to the client. SNMP relies on some other transport protocol to deliver the request/response packets between the client and the agent. In most cases, the protocol used is User Datagram Protocol / Internet Protocol (UDP/IP). Using a protocol like UDP, which does not guarantee data delivery, actually makes for a more robust environment for SNMP, because network management is needed most when the network is having problems. If the network is having problems, packets are probably going to get lost and/or corrupted. (Therefore, a transport protocol that guarantees data delivery may, in fact, be more a hindrance than a help.) Network management must continue to function, if at all possible, when nothing else does. The transport for network management data must be the simplest possible pass-through service available on the network. Using UDP, the client can continue to issue the requests to the agent; probability says that at least one of the packets will make it successfully to the agent, and the management action will be performed.

In the case of `xsnmpmon`(8), the client is implemented as a set of library routines (obtained from Carnegie-Mellon University) that are called by `xsnmpmon` to send and receive the SNMP requests and responses, respectively.

How can I invoke xsnmpmon(8)?

13.3

To invoke xsnmpmon(8), use the following syntax:

```
xsnmpmon [-d display | -display display]
  [-bd bordercolor]  [-bg backgroundcolor]
  [-fg foregroundcolor] [-fn font] [-lfn largefont]
  [-ib file] [-iconic]
```

-d *display* | -display *display*
: Specifies the name of the terminal on which you want to display the network monitor. You can enter either -d or -display. The default is the current value of the DISPLAY environment variable.

-bd *bordercolor*
: Specifies the border color of each window within xsnmpmon. This can also be set using the Colors button of the Setup window from within the monitor interface. The default is black.

-bg *backgroundcolor*
: Specifies the background color of each window within xsnmpmon. This can also be set using the Colors button of the Setup window from within the monitor interface. The default is dimgray.

-fg *foregroundcolor*
: Specifies the foreground color (that is, the color of the text) of each window within xsnmpmon. This can also be set using the Colors button of the Setup window from within the monitor interface. The default is black.

-fn *font*
: Specifies the normal font; the default is 6-by-13 pixels.

-lfn *largefont*
: Specifies the large font, which is used for highlighting text; the default is 6-by-13 pixels, bold.

-ib *file*
: Specifies the icon bit-map file. This allows you to specify your own icon.

-iconic
: Starts xsnmpmon as an icon.

What does the xsnmpmon(8) monitor look like?

13.4

When you enter the xsnmpmon(8) command, you will get the SNMP Network Monitor main window. Figure 12 shows an example of this window.

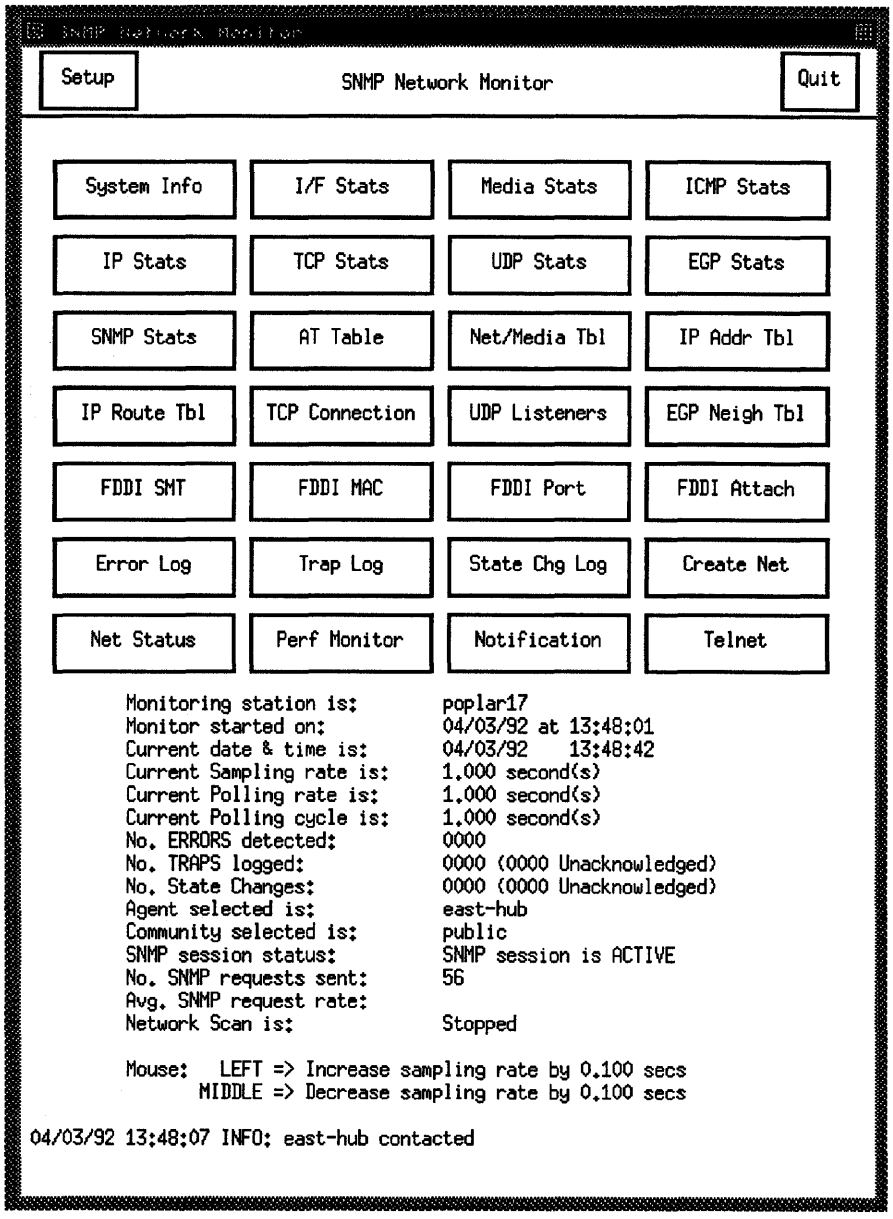


Figure 12. SNMP Network Monitor window

The main window consists of a set of buttons and synopsis information; the buttons invoke other windows that allow you to set up your monitor environment, control the monitor's actions, and perform various functions. To invoke one of these windows, place the mouse pointer on top of the button (highlighting it) and click any mouse button. You can have several windows open at one time.

Buttons
13.4.1

At the top left corner of the main window is the `Setup` button, which invokes a tool that allows you to change the default settings of the program.

At the top right corner is the `Quit` button, which allows you to exit from `xsnmpmon`. Most subwindows contain a `Quit` button that allows you to exit that function.

The first 16 buttons in the block of buttons represent the 16 SNMP MIB-II variable groups; when you click on one of these buttons, you invoke a window that contains a number of variables that are maintained by all compliant SNMP agents. In this context, *compliant* refers to agents that support the variables as they are defined in the Request for Comment documents RFC 1156 and RFC 1158.

The eight remaining buttons allow you to perform the following functions: display the error log, trap log, and state change log; create a network; graphically show the status of the network and perform a route-trace function; run a performance monitor; start an operator notification window; and use the `telnet`(1) command to contact the selected SNMP agent.

Entering text
13.4.2

Whenever the program requires that you enter text, a dialog window containing a question appears at the top of your screen; the mouse pointer must be within the window when you enter text there. Answer the question and press RETURN; to exit from the dialog window without changing anything, press RETURN without entering text. If you enter inappropriate text, the window will exit without making any changes to the monitor and issue an error message.

Synopsis information 13.4.3

Beneath the block of buttons are lines of synopsis information. This information is useful when you want to determine the monitor's current status. Clicking the left mouse button in the main window increases the sampling rate; clicking the middle mouse button decreases it. The minimum is 100 ms.

Message line 13.4.4

At the bottom of the window is a highlighted line used to display messages from the monitor. Any messages from the monitor will be echoed to this line, to standard error, and to a log file.

Modifying the monitor environment with the `Setup` button 13.5

The `Setup` button invokes a window containing several buttons, as shown in Figure 13. These buttons allow you to modify your monitor environment, either for this particular session (by clicking on the `Done` button) or permanently (by clicking on the `Save` button). The lines that follow the buttons show the current settings of all of the program's options.

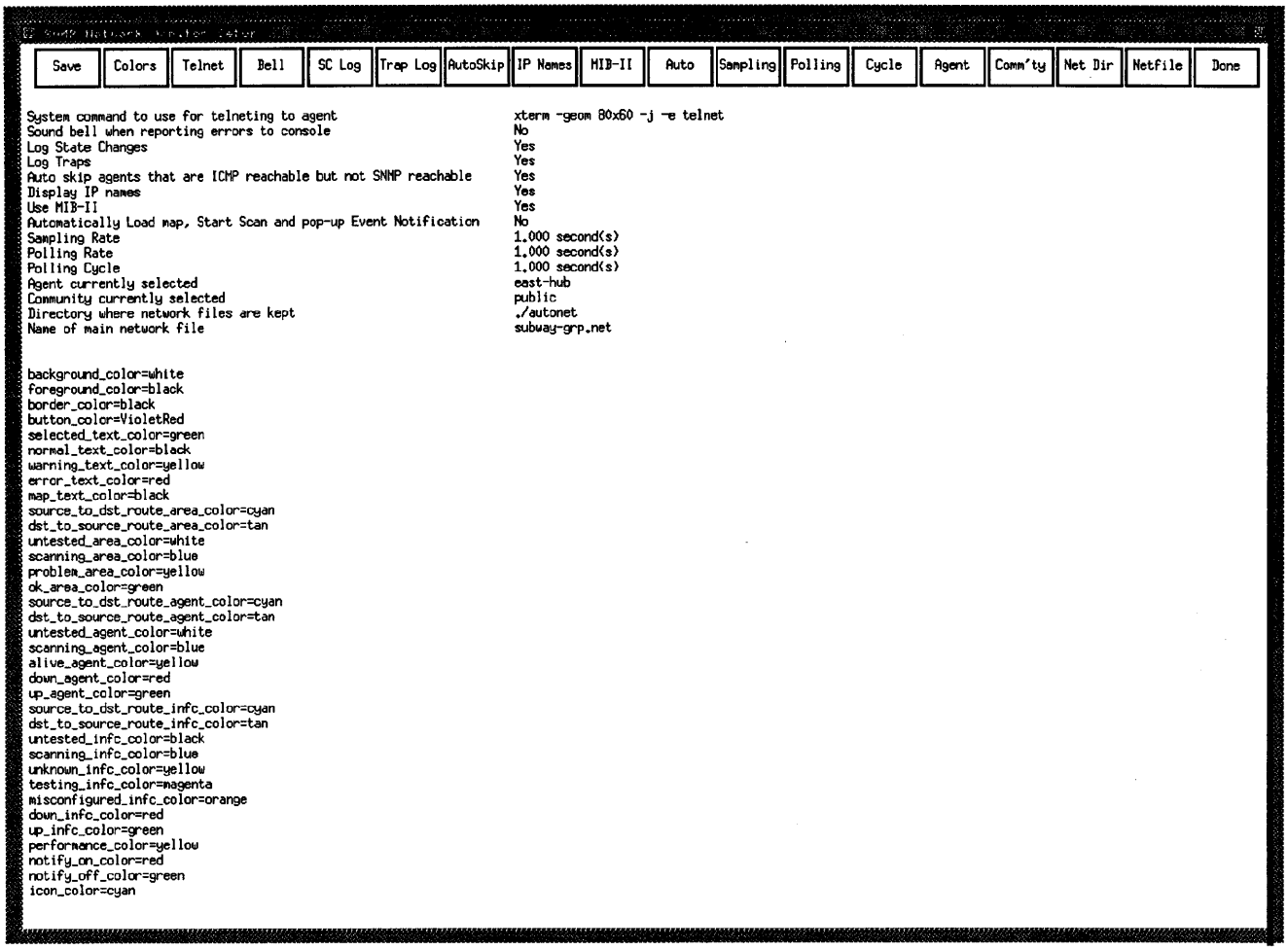


Figure 13. `Setup` window buttons

Typically you will want to specify the sampling rate with the `Sampling` button, the polling cycle time with the `Polling` button, the network directory (where network files are placed) with the `Net Dir` button, and the network file (the top "layer" file) with the `Netfile` button.

When you click on the `Colors` button, you will be asked whether you want to use a color palette program; by default, this program is `hyperview xnewsdemo.hv`. If you want to use another color palette program, set the `SNMP_COLORPROG` environment variable to the path name of the program.

Obtaining an overview with the `System Info` button
13.6

The `System Info` button gives you a quick overview of what is contained in a given agent. It provides enough information to determine the following for each agent:

- The kind of machine it is (manufacturer's information)
- The networks to which it is attached
- The number of network interfaces installed
- The status (up, down, and so on) of each interface
- The length of time the machine has been up since the last reinitialization

Using the statistics buttons
13.7

The statistics buttons display the contents of several MIB-II variables for a given agent. You can use these buttons to help determine where problems are. For example, if you notice that there are many input and output errors in the `Interface Statistics` display (accessed through the `I/F Stats` button), it may be an indication of a hardware problem.

The following subsections briefly describe each statistics button.

`I/F Stats`
13.7.1

The `I/F Stats` button dynamically shows the state of the variables associated with each interface within an agent. Click the left or middle mouse button to scan through each interface. Included here are counters and variables that show the activity of a given network interface.

`Media Stats`
13.7.2

This function has not yet been implemented. Statistics unique to different types of media, such as Ethernet and Fiber Distributed Data Interface (FDDI), will be provided here. The MIBs that define these media-specific variables are still in the experimental stage; when they are supported by a larger group of vendors, they will be supported by this program.

ICMP Stats
13.7.3

The `ICMP Stats` button dynamically displays the state of variables (statistics) associated with Internet Control Message Protocol (ICMP) packets that enter and leave the agent. The Internet uses ICMP to perform tasks such as testing readability and route redirection.

IP Stats
13.7.4

The `IP Stats` button dynamically displays the state of variables (statistics) associated with Internet Protocol (IP) packets that enter and leave the agent. TCP and UDP use IP for routing packets between destinations.

TCP Stats
13.7.5

The `TCP Stats` button dynamically displays the state of variables (statistics) associated with Transmission Control Protocol (TCP) packets that enter and leave the agent. TCP is an end-to-end protocol; applications such as `ftp` and `telnet` use it to converse across a network.

UDP Stats
13.7.6

The `UDP Stats` button dynamically displays the state of variables (statistics) associated with User Datagram Protocol (UDP) packets that enter and leave the agent. UDP is similar to TCP except that it does not guarantee data delivery, as TCP does.

EGP Stats
13.7.7

The `EGP Stats` button dynamically displays the state of variables (statistics) associated with Exterior Gateway Protocol (EGP) packets that enter and leave the agent. Gateways and routers use EGP to discover routes to various destinations.

SNMP Stats
13.7.8

The `SNMP Stats` button dynamically displays the state of variables (statistics) associated with SNMP packets that enter and leave the agent. SNMP is used to manage networks.

Table buttons
13.8

The table buttons display the contents of several MIB-II variables that are maintained in tabular form by an agent. You can use these buttons to help determine where problems are. For example, if you want to know the physical 48-bit IEEE address of a low-speed channel on an Ethernet, you could use the `AT Table` function to question any agent connected to that Ethernet.

The following subsections briefly describe each of the table buttons.

AT Table
13.8.1

The `AT Table` button displays the Address Translation table, which gives the binding between each IP address, media address, and interface. Each agent, in order to map from IP addresses to physical addresses, uses some mechanism to perform the binding and discovery of these addresses. On media such as Ethernet and FDDI, a protocol called Address Resolution Protocol (ARP) is used along with the broadcast feature of the medium to derive physical media addresses from IP addresses. On CRI mainframes, for example, a static mechanism (`hyroute`) is used.

If the window displays `More...` in the bottom right hand corner, it indicates that there is more information than would fit on a single screen. To display the rest of the information, click any mouse button while the mouse pointer is in the window.

Net/Media Tbl
13.8.2

The `Net/Media Tbl` button conveys the same information as the `AT Table` button if you are connected to an agent that supports MIB-II; if you are connected to an agent that supports only MIB-I, you will get an error message. (Therefore, you can use this button to determine whether or not an agent supports MIB-II.) In MIB-II, the Address Translation table is listed as deprecated; therefore, the Network Media table will replace the Address Translation table if a MIB-III is developed. However, because they are both part of MIB-II, both tables are supported.

IP Addr Tbl
13.8.3

The `IP Addr Tbl` button displays each interface's IP address and subnet mask, and the polarity of the least significant bit of the broadcast address for the medium.

IP Route Tbl
13.8.4

The `IP Route Tbl` button displays the IP routing entries for the agent. The table is organized as follows: destination IP address; the interface through which the packets will be routed; the value of the metrics for each route hop; the next hop in the route; whether the route is a local or remote route; how the route was learned; and the age of the route entry.

TCP Connection
13.8.5

The `TCP Connection` button displays the state of all the TCP connections in the agent. The table is organized as follows: the state of the connection; the IP address within the agent to which the connection applies; the port within the agent to which the connection was made; the IP address of the connected entity; and the port number of the connected entity.

UDP Listeners
13.8.6

The `UDP Listeners` button displays all of the UDP applications listening within the agent. The table is organized as follows: the IP address that is listening and the port number on which it is listening.

EGP Neigh Tbl
13.8.7

The `EGP Neigh Tbl` button displays information about all of the agent's EGP neighbors.

FDDI SMT
13.8.8

The `FDDI SMT` button displays all of the station management parameters for a given station management (SMT) entity in an FDDI station.

FDDI MAC
13.8.9

The `FDDI MAC` button displays all of the media-access control (MAC) parameters for a given media-access control entity in an FDDI station.

FDDI PORT
13.8.10

The `FDDI PORT` button displays all of the port parameters for a given port entity in an FDDI station.

FDDI Attach 13.8.11

The FDDI Attach button displays all of the attachment parameters for a given attachment entity in an FDDI station.

Using the Error Log button 13.9

The Error Log button displays errors that occur when xsnmpmon tries to issue SNMP requests to an agent. The message text includes the date and time a given message was received by the monitor; the latest message is indicated by the > symbol.

Displaying messages with the Trap Log button 13.10

The Trap Log button displays trap messages. A *trap message* is a message from an agent to the monitor, indicating that a significant event has occurred. The message text includes the date and time a given message was received by the monitor; the latest message is indicated by the > symbol. The message text also includes the name of the agent that sent the trap message, the message itself, and the length of time the machine has been up since the last event occurred.

For example, an agent may send a Coldstart message when it is rebooted, or an agent may send an Authentication Failure message if it suspects a security violation.

To acknowledge a trap message, first select the entry by clicking the left mouse button on it (highlight it); you can select multiple traps. When you have selected a trap, either acknowledge it by clicking the right mouse button or cancel the select on that entry by clicking the middle button. One click of the right mouse button acknowledges all selected traps. When the trap log fills up with unacknowledged traps, new traps are sent to the log file but are not placed into the trap log.

Displaying messages with the `State Chg Log` button

13.11

The `State Chg Log` button displays changes detected by the network scan process. For example, when the monitor detects that an agent has gone down, it logs the change here.

To acknowledge the message, first select an entry by clicking the left mouse button on it (highlight it); you can select multiple entries. When you have selected the state change entry, either acknowledge it by clicking the right mouse button or cancel the select on that entry by clicking the middle button. One click of the right mouse button acknowledges all selected entries. When the state change log fills up, the network scan is automatically stopped.

Note

This button does not work on monochrome displays.

Creating a network with the `Create Net` button

13.12

The `Create Net` button displays the `Create Network Configuration` window, which allows you to create a network that will be monitored by `xsnmpmon`(8). When you use the `Create Net` button, you are creating a graphic representation of the network you want `xsnmpmon` to monitor.

You can create layers as you go, starting with the top layer, the bottom layer, or some layer in between. Each picture is stored in a file that ends with the `.net` suffix and is stored in what is called the *net directory*. The top "layer" file is called the *net file*; that is, the monitor will recursively load files whose names begin with names of areas within the network file (that is, *area*`.net`).

Before you use this tool, you should know which hardware and networks you want to monitor. You will be asked to supply the names of the agents and their communities; *community* is the SNMP authentication term; it is something like a password.

The following subsections describe the buttons and discuss an example of a network.

`Create Network Configuration` *window* 13.12.1

Figure 14, page 80, shows the `Create Network Configuration` window; when you click on the `Create Net` button for the first time, the map area will be blank.

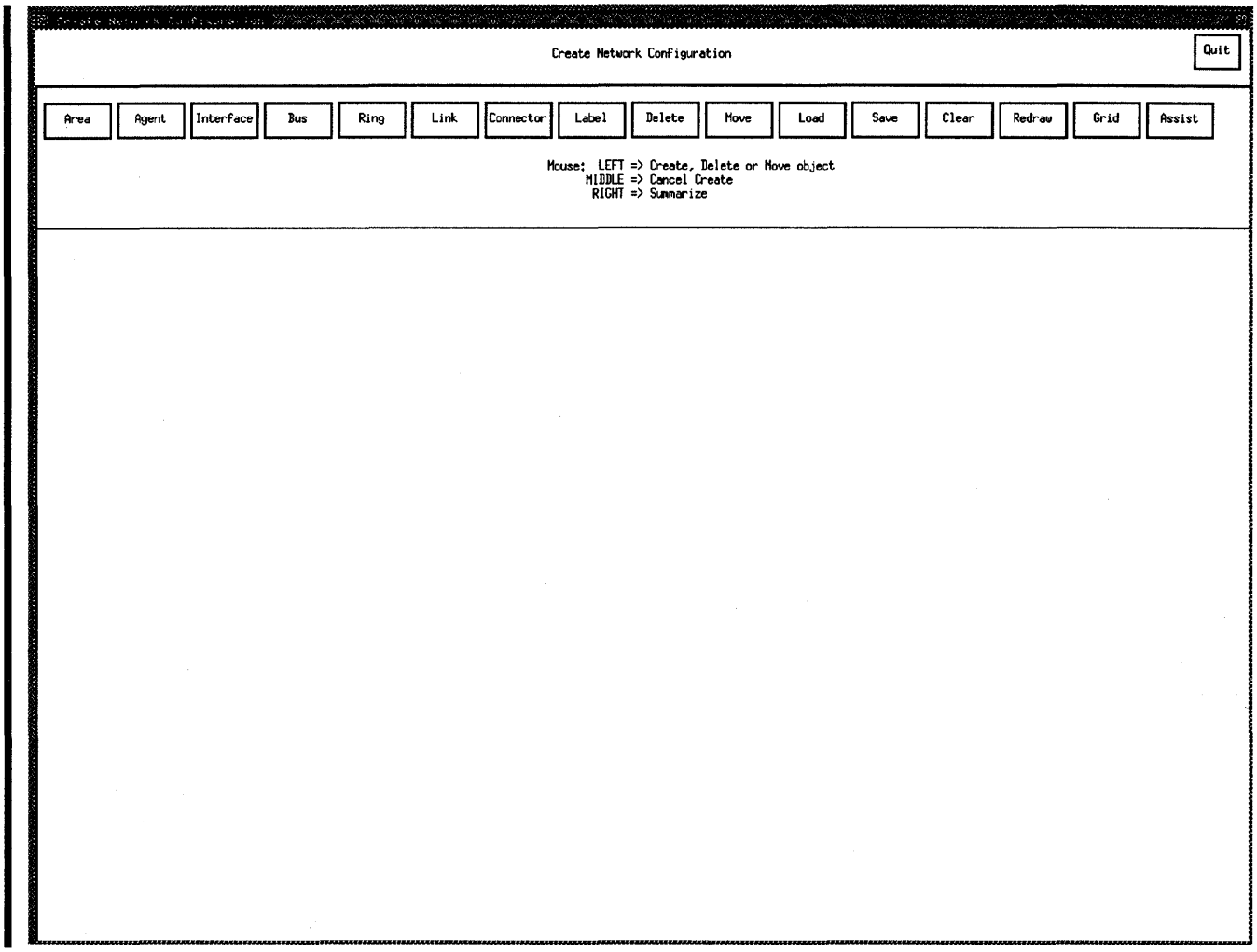


Figure 14. `Create Network Configuration` window

`Create Network Configuration` *buttons* 13.12.2

The buttons on the `Create Network Configuration` window and their functions are as follows:

Button	Description
`Area`	Creates and names a trapezoid that represents an area. An area is a box that represents any entity that is manageable by SNMP (such as a host or gateway). An area can be thought of as a room, building, city, network, and so on, that contains other areas and agents. You are asked to provide a name and description for the area. When you have moved the figure where you want it on the map, click the left mouse button to place it there.
`Agent`	Creates and names a box that represents an agent. An agent is a trapezoid that allows you to create a layered effect in the network map; an area implies that there is another map by the name *area*`.net` that contains more agents. An area can be thought of as a room, building, floor, city, network, etc., that contains other areas and agents. You can have several nested areas. You are asked to provide the name of the agent, its community, and a brief description. If the name does not appear in the `/etc/hosts` file, the program will also ask you to provide the agent's IP address. (The description is used for the summary utility in the network scan process.) When you have moved the figure where you want it on the map, click the left mouse button to place it there.
`Interface`	Creates and names a line or set of line segments that represents a physical network interface. (You must create an agent with the `Agent` button and a network with the `Bus`, `Ring`, or `Link` button before you can create an interface.) The line representing the interface must touch the agent and the network to which the agent is connected: to start the line, click the left mouse button; to change its direction, click the left mouse button; to complete the line, click the middle or right mouse button. When you have completed your drawing of the interface, the monitor attempts to contact the SNMP entity in the agent to which this interface is connected in order to obtain the interface number (index); if it is not successful, the monitor asks you

Button	Description
	to enter the number. You can find the number of the interface by connecting to the agent and displaying the system information; the leftmost column shows the interface numbers (or indexes).
`Bus`	Creates and names a line or set of line segments that represents a bus, which is a network topology in which all the nodes connect to a single wire. Examples of bus-type network media are Ethernet and HYPERchannel. To start the line, click the left mouse button; to change its direction, click the left mouse button; to complete the line, click the middle or right mouse button. If the name you entered does not appear in the `/etc/networks` file, the program will also ask you to provide the bus's IP address. When your drawing of the bus is complete, you will be prompted to place the name, IP address, and description of the bus on the map.
`Ring`	Creates and names an oval that represents a ring, which is a network topology in which all the nodes connect together in a closed loop. Examples of ring networks are FDDI and TOKEN Ring. Click the left mouse button to place the center of the ring on the map. You can change the size of the ring by dragging the mouse in any direction. Click the left mouse button a second time to affix the ring to a certain spot on the map. If the name you entered does not appear in the `/etc/networks` file, the program will also ask you to provide the ring's IP address. When your drawing of the ring is complete, you will be prompted to place the name, IP address, and description of the ring on the map.

Button	Description
`Link`	Creates and names a jagged line (similar to a lightning bolt) that represents a link, which is a network topology that involves a local and remote side, usually point-to-point in nature (such as DS1 and DS3). T1, T3, and satellites are examples of link media. Click the left mouse button to start and end the link. If the name you entered does not appear in the `/etc/networks` file, the program will also ask you to provide the link's IP address. When your drawing of the link is complete, you will be prompted to place the name, IP address, and description of the link on the map.
`Connector`	Creates a dot that represents a physical connection point of an interface to a network when an interface crosses multiple networks. This is a graphical representation only. Click the left mouse button to affix the connector to the map.
`Label`	Creates a text string. You can use this function to place comments anywhere on the network map. Click the left mouse button to affix the label to the map.
`Delete`	Deletes objects on the map. Click on the `Delete` button, place the pencil cursor on the object, click the left mouse button to highlight the object, and click the left mouse button to delete the object; if you do not want to delete the object, click the right mouse button to cancel the operation.
`Move`	Moves objects on the map. To move an object, click on the `Move` button, place the pencil cursor on the object, click the left mouse button to highlight the object, move it where you want it to be, and click the left mouse button to affix the object to its new spot.
`Load`	Loads an existing file in order to modify it, delete it, or copy it to a different file name.
`Save`	Saves the map to a disk file for later retrieval.

Button	Description
`Clear`	Clears the current map. If you have made changes to the map and have not clicked the `Save` button, you will be prompted to save or clear the map.
`Redraw`	Refreshes the contents of the window.
`Grid`	Turns on/off a grid of lines that is helpful when you are drawing and placing objects on the map. Each time you click this button, you toggle the state of the option.
`Assist`	Accesses the assist file, which defines a layered network topology to the `xsnmpmon` drawing facility. This allows you to specify a general view of the network to `xsnmpmon`, which in turn uses SNMP and the `/etc/networks` and `/etc/hosts` files to obtain information about the network.

Example 13.12.3

The example in this subsection describes one way to create a network map and illustrates how the layering effect works.

Scenario 13.12.3.1

Suppose your site is a building with two floors. On the first floor, you have a lab with three workstations and an office with one workstation. On the second floor, you have three workstations. The two floors are connected by a T3 link. You might want to create the following four net files:

- `bldg.net`, which represents the whole building, with areas for the first and second floors
- `firstfloor.net`, which represents the first floor, with a workstation and an area for the lab
- `lab.net`, which represents the lab on first floor, with three workstations
- `secondfloor.net`, which represents the second floor, with three workstations

Steps to create a network map
13.12.3.2

There are many ways to create network maps for this situation. One process you might follow is shown in the steps on the following pages.

1. Click on the `Create Net` button, as shown in Figure 15:

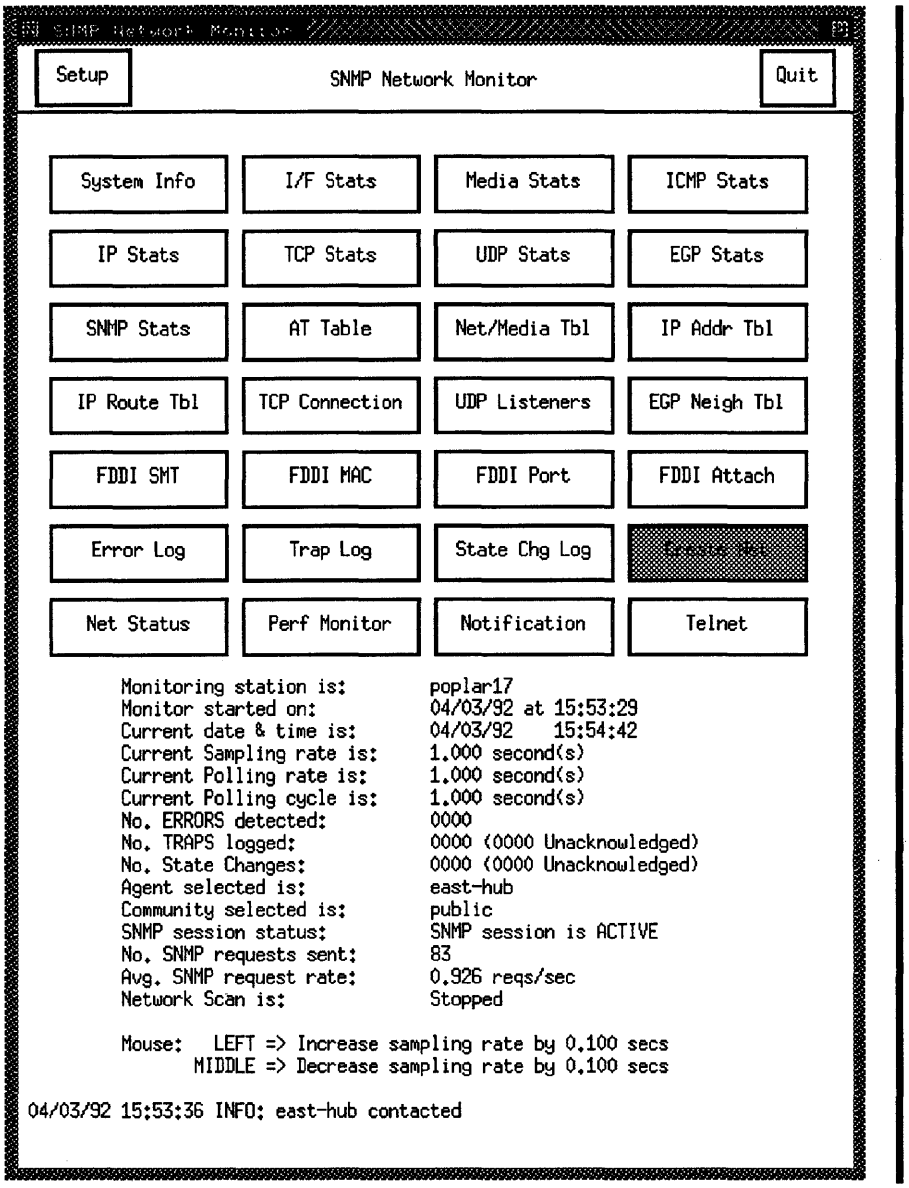


Figure 15. Displaying the `Create Network Configuration` window

2. Create a map for the building:

   a. To select the area function, click the left mouse button on the `Area` button, as shown in Figure 16.

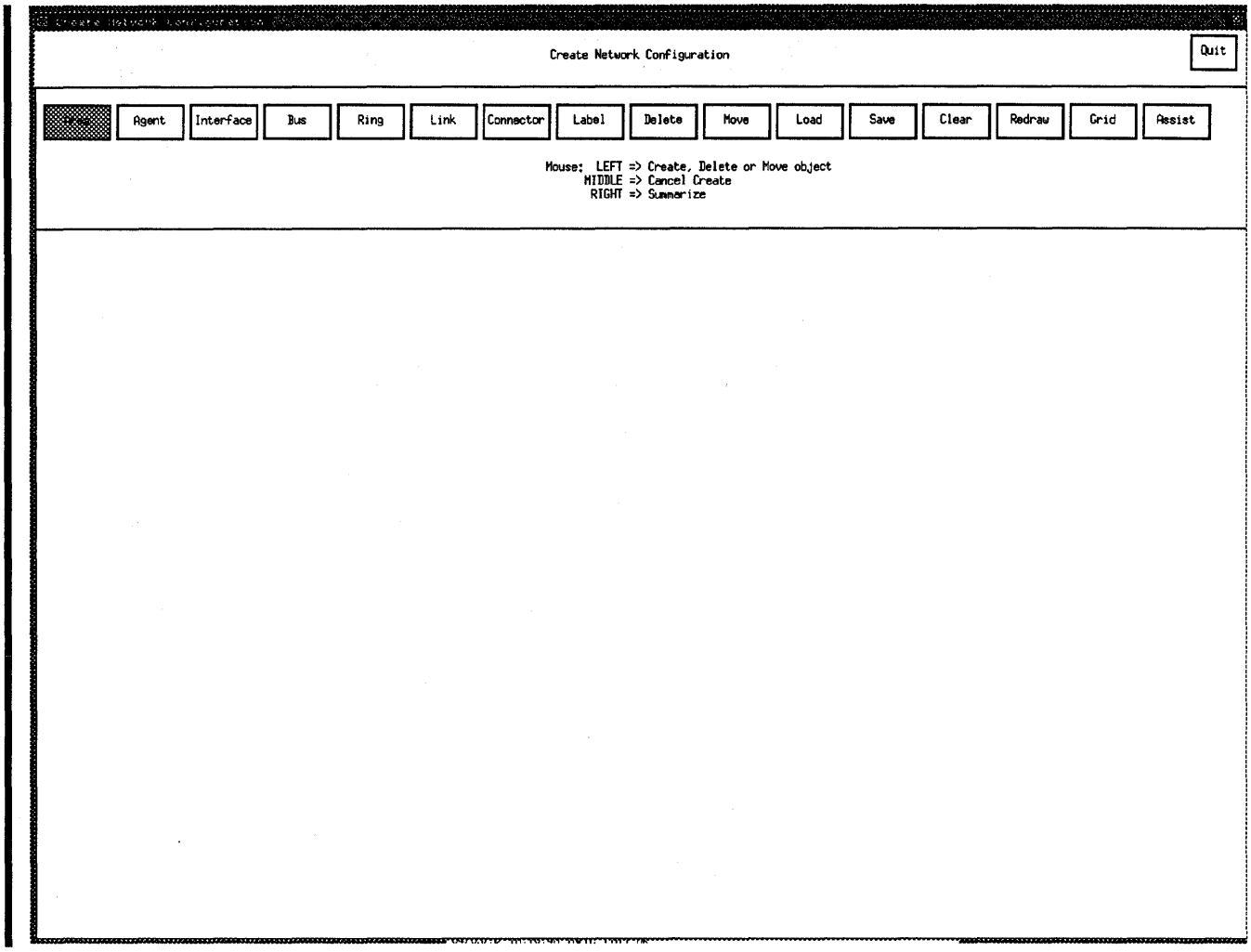


Figure 16. Selecting the `Area` button

b. To create the area trapezoid, move into the map area and click the left mouse button again. This invokes a dialog window, as shown in Figure 17; answer the questions.

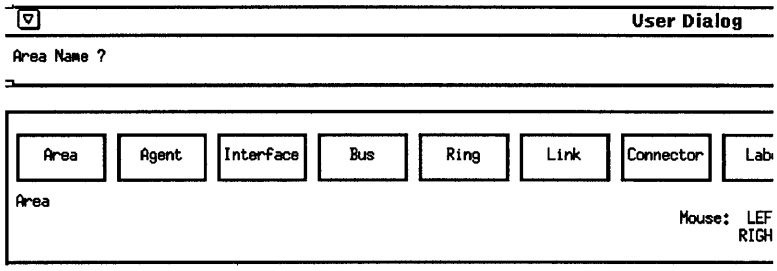


Figure 17. Dialog box

c. Place the mouse pointer back in the map area, position the trapezoid, and click the left mouse button to affix it to the map.

d. To create a second area, click the left mouse button in the map area and repeat steps b and c

e. Create a link with the `Link` button.

f. Click on the `Save` button and name the file `bldg.net`.

When you are finished, your map may look like the one in Figure 18, page 88.

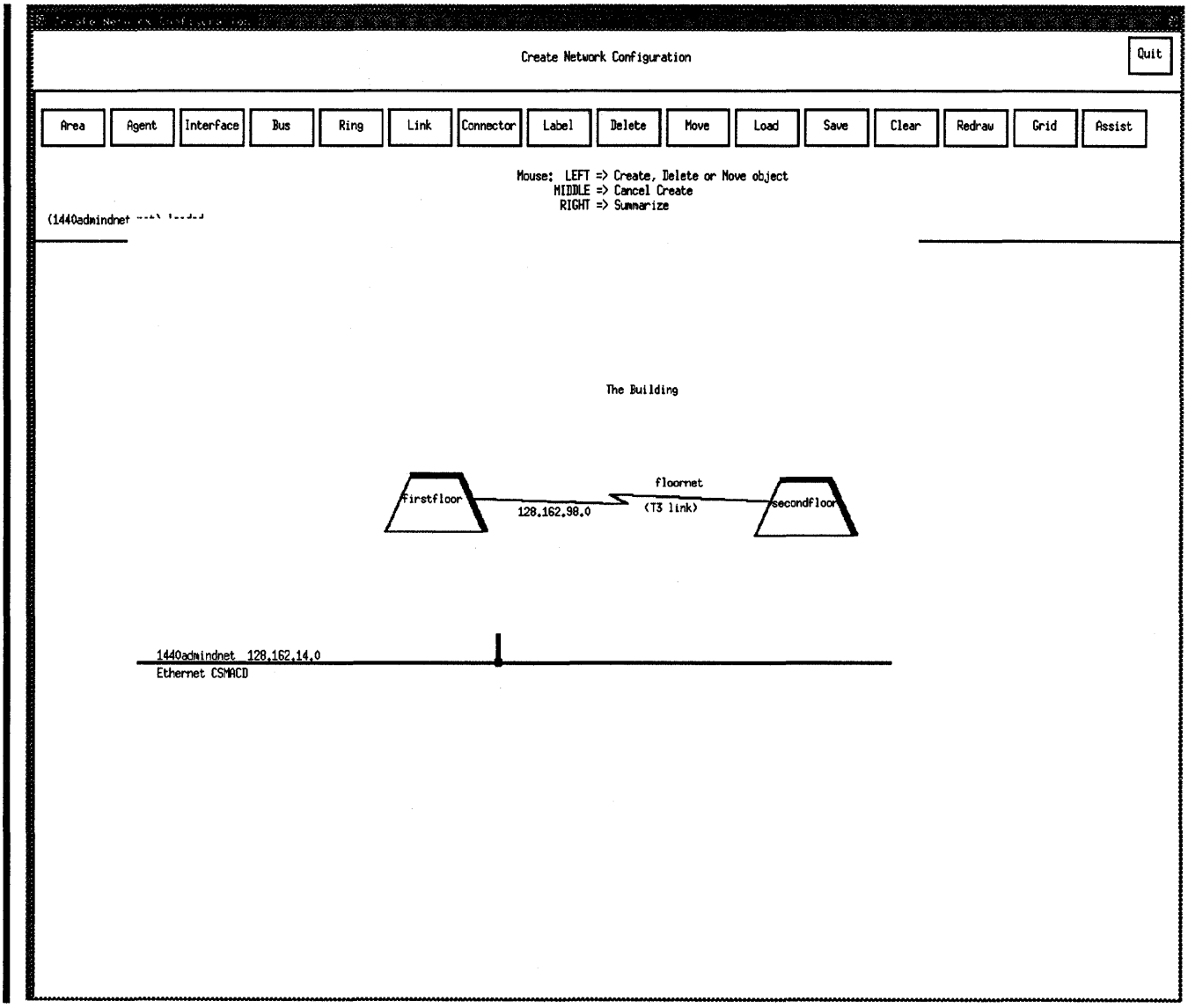


Figure 18. `bldg.net` map

3. Create a map for the first floor:

   a. Create an area with the `Area` button.

   b. Create an agent (for the workstation) with the `Agent` button.

   c. Create a network with the `Bus` button.

   d. Create an interface with the `Interface` button.

   e. Create a connector with the `Connector` button.

   f. Click on the `Save` button and name the file `firstfloor.net`.

   When you are finished, your map may look like the one in Figure 19.

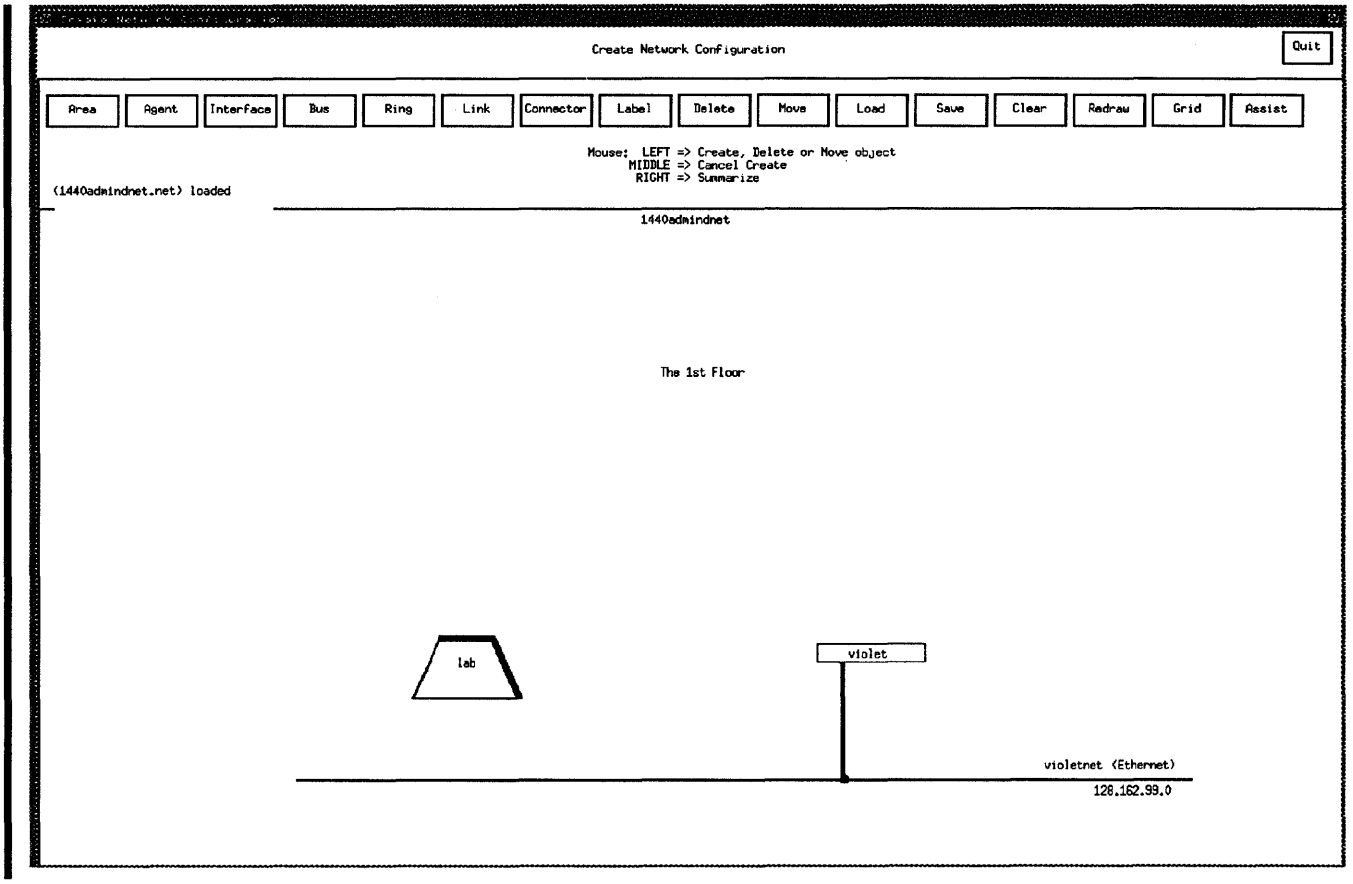


Figure 19. `firstfloor.net` map

4. Create a map for the lab:
   a. Create an agent with the `Agent` button; repeat for each workstation in the lab.
   b. Create a network with the `Bus` button.
   c. Create an interface with the `Interface` button; repeat for each workstation.
   d. Create a connector with the `Connector` button; repeat for each workstation.
   e. Click on the `Save` button and name the file `lab.net`.

   When you are finished, your map may look like the one in Figure 20.

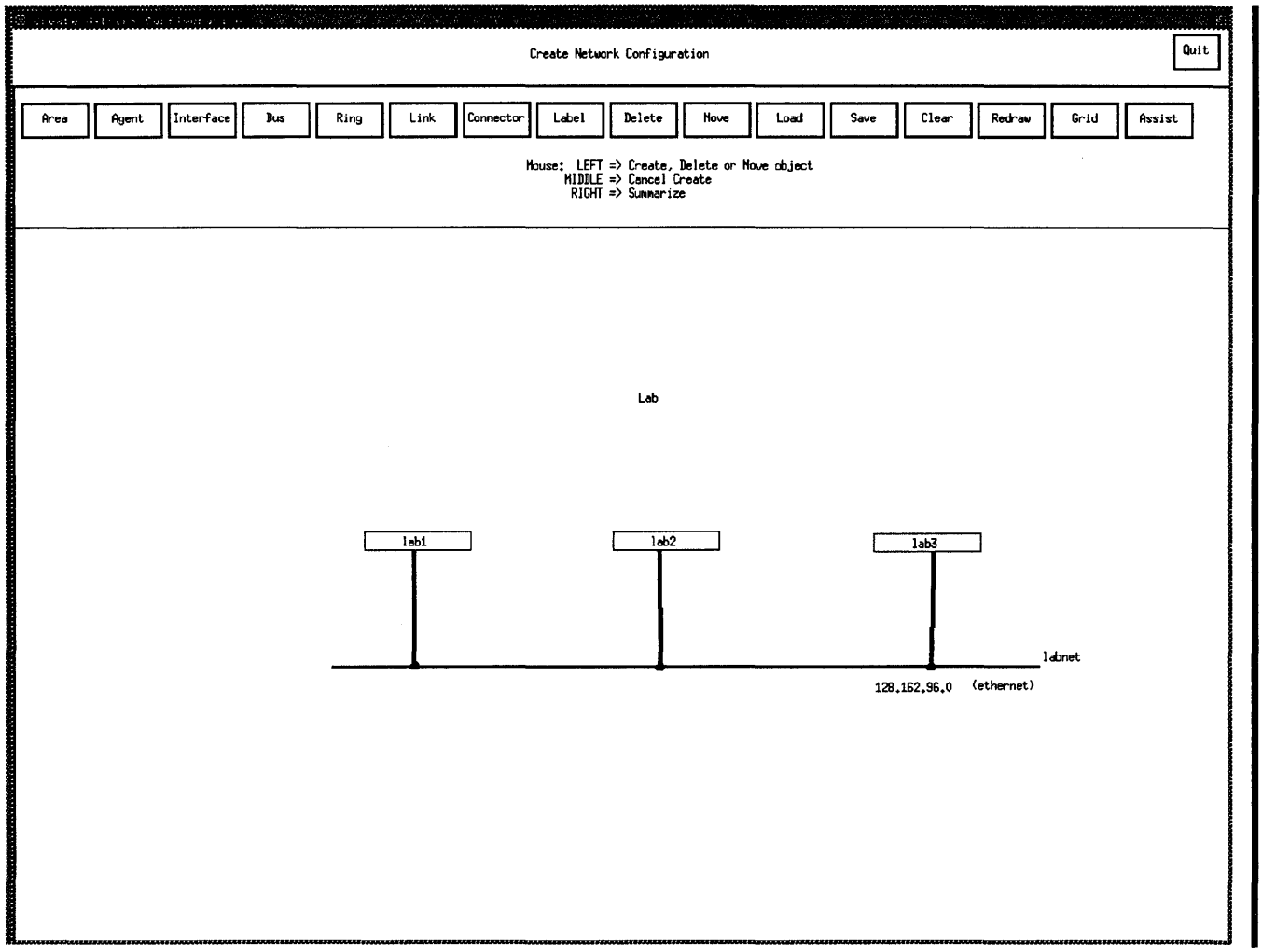


Figure 20. `lab.net` map

5. Create a map for the second floor:

   a. Create an agent with the `Agent` button; repeat for each workstation on the second floor.

   b. Create a network with the `Bus` button.

   c. Create an interface with the `Interface` button; repeat for each workstation.

   d. Create a connector with the `Connector` button; repeat for each workstation.

   e. Click on the `Save` button and name the file `secondfloor.net`.

   When you are finished, your map may look like the one in Figure 21.

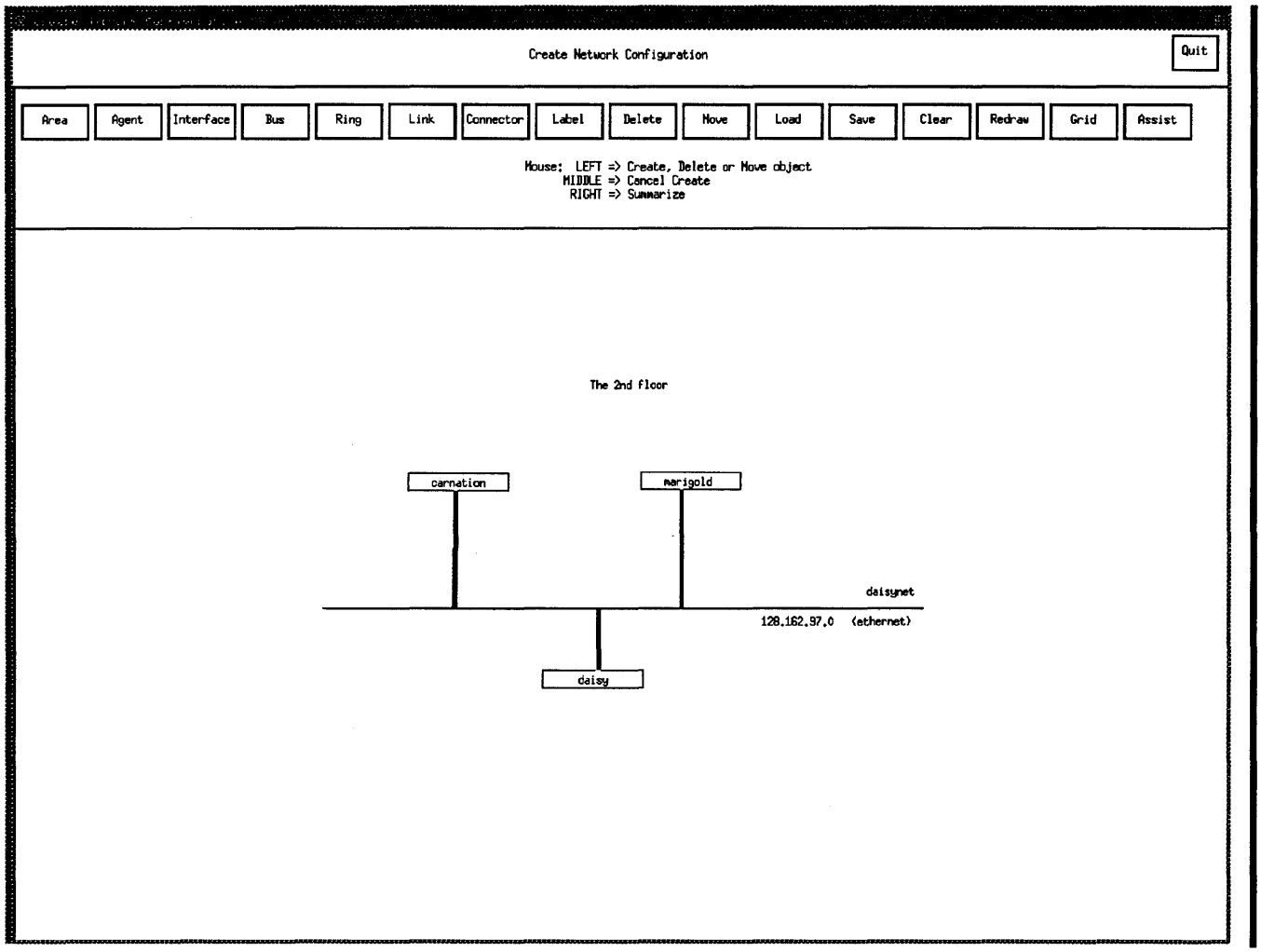


Figure 21. `secondfloor.net` map

Loading process
13.12.3.3

In this configuration, `bldg.net` loads three files: `firstfloor.net`, which loads `lab.net`, and `secondfloor.net`. Figure 22 shows the interrelationships among the files (maps) created in the steps described in the previous subsection.

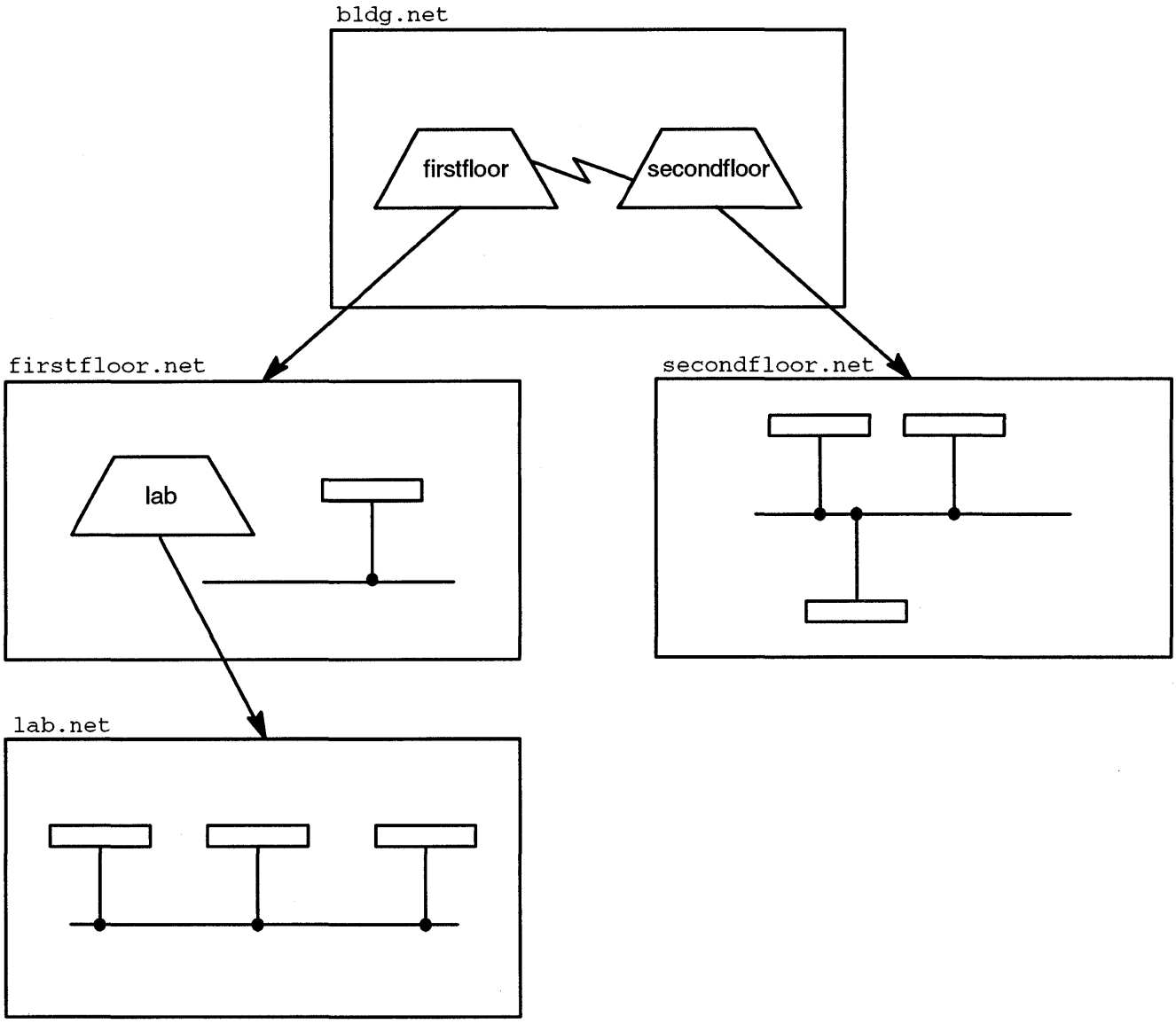


Figure 22. Interrelationships among the files (maps)

When the monitor loads a file, it first searches for an area within the file; when it encounters one, it loads the file named *area*`.net`. It then searches through *area*`.net` for another area; when it finds one, it loads that file and searches it for an area, and so on. When the monitor finally encounters a file without an area, it moves back one layer, searches for more areas, and then continues this moving-back process.

If you assign `bldg.net` to be your net file, the loading process for the monitor is as follows:

1. Loads `bldg.net`, searches for an area, and finds `firstfloor`.
2. Loads `firstfloor.net`, searches for an area, and finds `lab`.
3. Loads `lab.net` and searches for an area. Finding none, it reads the information about the three agents, the network, and the interfaces.
4. Returns to `firstfloor.net` and searches for an area. Finding none, it reads the information about the agent, the network, and the interface.
5. Returns to `bldg.net`, searches for an area, and finds `secondfloor`.
6. Loads `secondfloor.net` and searches for an area. Finding none, it reads the information about the three agents, the network, and the interfaces.
7. Returns to `bldg.net` and reads the information about the link.

If you assigned `firstfloor.net` as your network file, the monitor would load only `firstfloor.net` and `lab.net`.

Scanning with the `Net Status` button

13.13

The `Net Status` button allows you to start and stop scanning the network and graphically displays the status of the network, based on what it finds during the scan.

During the network scan process, `xsnmpmon`(8) accesses two files: an action file (`xsnmpmon.act`) and an exception file (`xsnmpmon.xcp`). The action file is consulted whenever a trap is received or a state change event occurs during a network scan. If an event occurs that is registered in the action file, a shell script (named in the action file) will be executed. The triggering events can range from quite general to very specific (see examples within the `xsnmpmon.act` file). The shell scripts can be written by the user to perform any necessary action, such as calling a pager or sending an electronic mail message to a network administrator.

The exception file is accessed from the `Network Status` (scan phase) window and can be used to prevent the polling of agents or interfaces that are known to have problems or that do not support SNMP. This file can also be used to force the polling of agents that do not support SNMP; normally, when the network scan detects an agent that it can reach by using ICMP but not SNMP, it stops polling that agent until an SNMP trap is received from it.

`Network Status` *window*

13.13.1

Figure 23 shows the `Network Status` window.

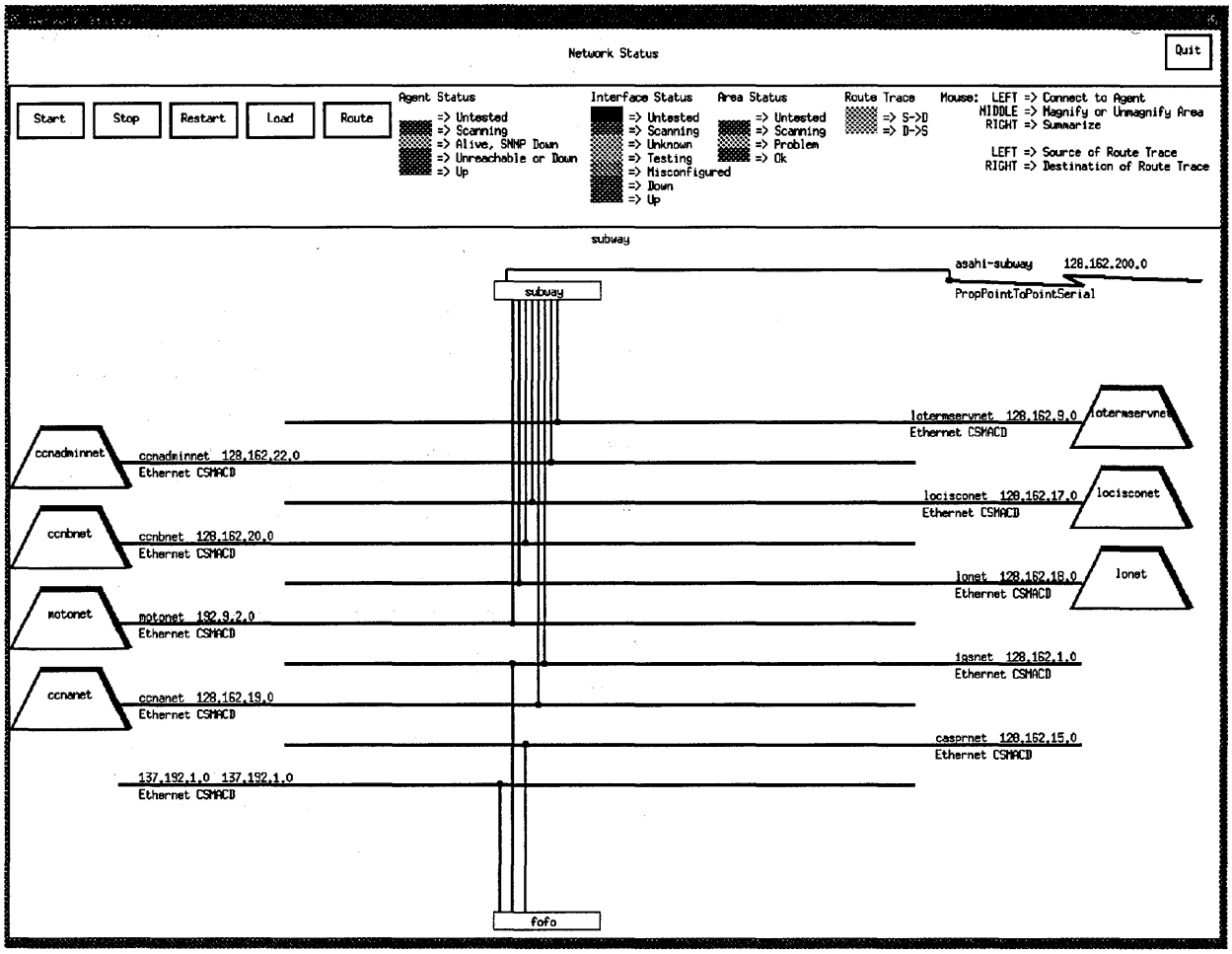


Figure 23. `Network Status` window

Buttons in the `Network Status` *window* 13.13.2

The `Network Status` window contains the following buttons:

Button	Description
`Start`	Starts the scan sequence from the beginning. Any previous status is cleared.
`Stop`	Stops the scan sequence, but retains the previously obtained status.
`Restart`	Restarts the polling sequence from where it stopped (when you clicked on the `Stop` button), restoring the saved status.
`Load`	Loads a new network configuration.
`Route`	Performs a route trace. To show the route taken between a source agent and a destination agent and back again, first stop the scan (if it is running), then click the left mouse button on the source and the right mouse button on the destination. The scan status colors disappear and the route from the source agent to the destination will appear in cyan (by default); the route from the destination back to the source will appear in tan (by default). You can also use this facility to perform a route trace between two specific interfaces for agents with multiple interfaces. When you have performed all route traces desired, either click on `Restart` to resume the network scan from where you stopped (the same colors reappear) or click on `Start` to start from the beginning.

Statuses 13.13.3

The `Network Status` window displays the map items in different colors according to their type and status. The colors are shown in a legend at the top of the window. The status for agents, interfaces, and areas are described in the following subsections.

If, during the scan, the state change log fills with unacknowledged state changes, the scan will stop automatically. You must acknowledge the state changes and manually restart the scan by clicking on the `Restart` button.

Agent status
13.13.3.1

An agent can have one of the following statuses at any time:

Status	Description
`Untested`	The scanning process has not yet tried to reach the agent.
`Scanning`	The agent is being interrogated by the monitor.
`Alive, SNMP Down`	The agent has responded to an ICMP `ECHO` request but not to an SNMP request. Either the SNMP entity within the agent is down, or the agent is not an SNMP-compliant device. When the SNMP agent becomes active, it should send a trap message to the monitor, at which time the agent will be placed back on the polling list.
`Unreachable or Down`	The agent is not responding to either ICMP or SNMP requests. The agent or the network to that agent is down, the OWS-E has no route to get to the agent, or the agent has no route to the OWS-E. An agent that is set to `Unreachable` is not polled again.
`Up`	The agent has responded to an SNMP request. The agent and the network are up.

Interface status
13.13.3.2

An interface can have one of the following statuses at any time:

Status	Description
`Untested`	The scanning process has not yet tried to reach the agent owning the interface.
`Scanning`	The interface is being interrogated by the monitor.

Status	Description
Unknown	An interface is Unknown if one of the following is true: if the state of an agent is Alive, SNMP Down or Unreachable, or if a garbled answer is received from the agent owning the interface.
Testing	The agent owning the interface received an SNMP request and has responded that it is in testing mode.
Misconfigured	The map's saved configuration for this interface does not match its current configuration.
Down	The agent owning the interface received an SNMP request and has responded that it is down.
Up	The agent owning the interface received an SNMP request and has responded that it is up.

Area status
13.13.3.3

The status of area is a summary of the agents and interfaces in that area. An area can have one of the following statuses at any time:

Status	Description
Untested	The scanning process has not yet tried to reach anything in the area.
Scanning	An agent or an interface in the area is being interrogated by the monitor.
Problem	The status of at least one agent is Alive, SNMP down or Unreachable or Down, or the status of at least one interface is Unknown, Testing, or Down.
OK	None of the underlying agents or areas are Misconfigured or Down.

Mouse button functions 13.13.4

The mouse buttons have particular functions in the `Network Status` window.

- When you are doing a route trace with the `Route` button, the mouse buttons have the following functions when clicked:

Button	Function
Left	Specifies the source agent or the source interface.
Right	Specifies the destination agent or the destination interface.

- During all other operations in the window, the mouse buttons have the following functions when clicked:

Button	Function
Left	Connects the monitor to the agent on which the mouse pointer is placed.
Middle	Moves up and down through the layers of the map. If you click the middle mouse button when the pointer is on an area, it will move down a layer. If you click the middle mouse button when the pointer is elsewhere on the map, it will move up a layer if possible.
Right	Pops up a window that contains summary information for the agent, interface, and network for the area or agent on which the mouse pointer is placed.

Displaying performance information with the `Perf Monitor` button

13.14

The `Perf Monitor` button displays a graphical representation of selected variables within the interface set of statistics and a separate graph of the ICMP turnaround time, which shows the time it takes to send an ICMP `ECHO` packet of minimum size from the OWS-E to the connected agent and receive a reply. This information can be helpful in solving problems related to network latency (such as TCP window size and other tuning issues).

The graphs start with a set of default limits, but the monitor dynamically adjusts the scale of the graphs based on the data it collects. The monitor allows the graphs to accumulate about 10% of the data before it first attempts to rescale the graphs. After this, the scale is readjusted, if necessary, during each sampling interval.

Using the `Notification` button

13.15

The `Notification` button displays visual symbols that indicate when human intervention is required. As trap messages are received from the network, the `Event Notification` window indicates that potential problems are occurring in the network by changing the color of the trap message light. Similarly, if an agent or an agent's interface changes states from one poll to the next, the state change light changes color. This is useful when the scan is started and the `Network Status` window is closed.

Figure 24 shows the `Event Notification` window with normal status, and Figure 25, page 103, shows the window with changed status.

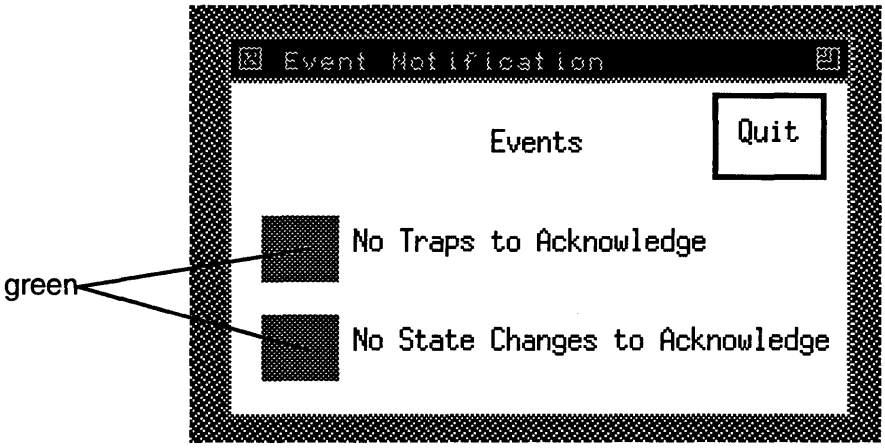


Figure 24. `Event Notification` window showing normal status

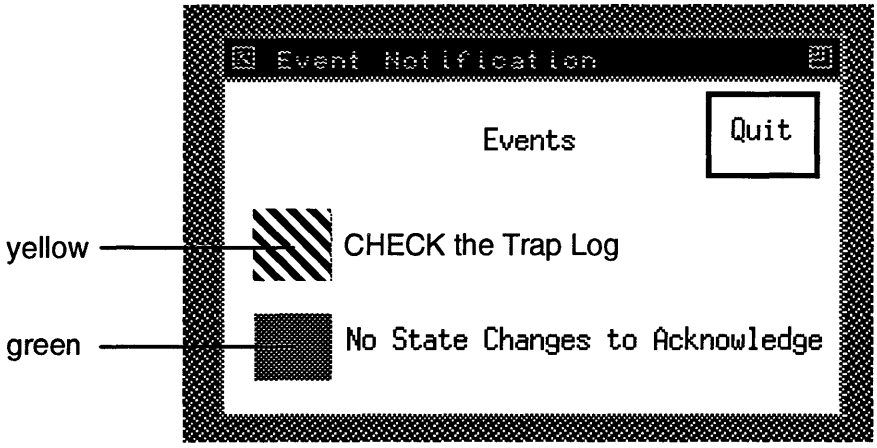


Figure 25. `Event Notification` window showing a change in status

Using the `Telnet` button
13.16

The `Telnet` button executes the shell command defined in the `.rc` file as the `telnet` string. Typically, clicking on this button causes a `telnet` session to be started with the specified agent.

Log files
13.17

The monitor is capable of keeping a log file of all activity that occurs on a daily basis. By default, the log file is named `xsnmpmon.log.`*mmddyy*, in which *mmddyy* is the month, day, and year. At midnight each day, `xsnmpmon`(8) closes the current day's log file and automatically opens a new log file for the new day. You can change the file name by setting the `SNMP_LOGFILE` environment variable.

If you do not want a log file, set the `SNMP_LOGFILE` environment variable to the following:

```
/dev/null
```

Index

Index

(1) section number, vii
(5) section number, vii
(7) section number, vii
(8) section number, vii

A

Accessing man pages, vi
Address Translation table, 76
ADUMPDIR parameter, 27
Agent, definition of in SNMP, 68
AnswerBook, 9
    documentation, viii
Assumptions for using this manual, v
AUTODUMP parameter, 28

B

Backing up and restoring OWS-E files, 1
Backplane connections, 8
BASEPORT parameter, 35
Binary, IOP route, 33
Bold typewriter font, vi
Boot, IOP default, 26
Booting the IOS-E and the mainframe, 1
Bootstrap loader program, 31
Brackets, v

C

CCA, definition, 5
CIN, 7
Clearing the OWS-E, 59
Client, definition of in SNMP, 68
Command directory parameter (ROOTDIR), 23
Commands
    See also Scripts
    autodump, 27, 28
    bootsys, 24, 25, 26, 27, 36
    cleario, 29
Commands *continued*
    conv, 45
    cpud, 28, 34, 35
    cpuhalt, 61
    cpupanic (script), 27
    craymon, 37
    dsdiag, 29
    dumpsys, 61
    eboot, 28
    econ, 29
    ecrash, 45
    ediag, 28
    edump, 45, 61
    errlogd, 27, 36
    fastboot, 57
    getconfig, 17
    graphs, 34, 63–66
    halt, 55
    hbeat, 26, 28, 35
    hcon, 29
    lapdaemon, 37, 43
    location in directory structure, 11
    man, vi
    mfdump, 25, 33
    mfstart, 29
    OWS-E (documentation for), vii
    rcpud, 28
    restricted, 40
    shutdown, 55, 57
    smdemon, 27
    SNMP (documentation for), vii
    SunOS (documentation for), vii
    sync, 55, 57
    UNICOS (documentation for), vii
    xsnmpmon, 67–104
Compact disk (CD), 5
Components
    hardware, 3–4
    software, 9
Configuration file. *See* System configuration file
Contacting CRI, x
Conventions in this manual, v
CPU monitors (graphs), 1, 63–66

CPUD_HOSTNAME parameter, 34
__CPUDHOSTNAME__ token, 22
CPUPANIC parameter, 25
Cray channel adaptor (CCA), 5
Cray serial number, 19
CRAY Y-MP, vi
CRAYMON parameter, 37
craymon(8), .xinitrc and, 14
crontab and the OWS-E backup scripts, 53
.cshrc file, 14, 15

D

DEF_MFCHAN parameter, 32
DEFAULTDUMPDIR parameter, 27
DEFAULTIKERNDIR parameter, 25
DEFAULTIOP parameter, 26
DEFAULTUKERNFILE parameter, 24
DEFAULTUPARAMFILE parameter, 24
Diagnostics
    cleario, 29
    dsdiag, 29
DIOPATH parameter, 32
Directory structure, 11
Disk
    drive, 3
    partitions, 10
    type, 19, 32
DLEN parameter, 33
D*n*FWA parameter, 33
D*n*LWA parameter, 33
DSTART parameter, 33
__DSTARTBL__ token, 20
DTYPE parameter, 32
Dump
    directory default, 27
    lock file directory default, 27
    lock file name default, 27
    slice, 20
DUMP parameter, 27
dumpdly script, 53
Dumping IOS-E and mainframe
    automatically, 1
    manually, 61–62
__DUMPIO__ token, 19
__DUMPLEN__ token, 20
__DUMPTYPE__ token, 19
__DUMPUNIT__ token, 20
dumpwkly script, 53
DUNIT parameter, 33

E

ECON parameter, 29
ecrash utility, holder file, 30
EDIAG parameter, 28
edump file format, 45–49
    file header structure, 47
    register section structure, 49
    section header structure, 48
edump utility, holder file, 30
EGP, 75, 77
EIOP, definition, 18
    See also IOP
Environment variables, 15
ERRLOG parameter, 36
ERRLOGD parameter, 27
Error log
    file path name, 36
    SNMP, 78
Error logging daemon, 27
/etc/configfile. *See* System configuration file
/etc/lapfile, 43
/etc/owsepermfile, 40
Event Notification window, 103

F

FDDI
    Attach, 78
    MAC, 77
    Port, 77
    SMT, 77
FEI. *See* Front-end interface (FEI)
File header structure, 47
Fonts, v–vi
Forced halt and reboot of OWS-E, 59
Front-end interface (FEI), 5, 7

G

getconfig(8) command, 17
Graphical interface, 13
graphs CPU monitor, 63–66
Group ID, 39

H

Halting the OWS-E. *See* Shutting down the OWS-E
Hardware environment
    CDs, 5
    components, 3–4
    connections, 5–8
    tapes, 5
HBEAT parameter, 28
HCON parameter, 29
__HOSTNAME__ token, 19
Hung OWS-E, 57

I

I/O clear diagnostic, 29
I/O subsystem model E (IOS-E)
    binaries directory, 25
    booting, 1
    connections, 5–8
    documentation, ix
    dumping
        automatically, 1
        manually, 61
    interaction with OWS-E, 7
ICMP, 75
Initialization files
    changed files, 14
    required files, 13
    variable settings, 15
Installation, documentation, viii
Introduction, 1–2
IOP
    deadstart diagnostic default, 29
    definition, 18
    log file path name, 36
    monitor, 28
IOP_DIAGNOSTICS parameter, 36
IOPDEBUG parameter, 30
IOPHALT parameter, 26
IOPLOG parameter, 36
IOPSAVE parameter, 30
IOS-E
    See also I/O subsystem model E (IOS-E)
    definition, 5
IOSCPATH parameter, 29
IOSDPATH parameter, 29
IP, 75
Italic font, v

K

Keys, vi

L

LAPFILE parameter, 37, 43
Length of slice, 33
Line arbitration priority file (/etc/lapfile),
  description of, 43–44
.login file, 14
Login name, 39
Low-speed channel, 5

M

M_MEMORY parameter, 34
Machine name, 19
MAIL_CPUFAIL parameter, 23
MAIL_IOPFAIL parameter, 23
Mainframe
    booting, 1
    dumping, 1
    memory ranges to be dumped, 33
    memory size, 21, 34
    serial number, 19, 26
    type, 20, 33
MAINFRAME parameter, 33
__MAINFRAME__ token, 20
Man page references, vi–vii
Management Information Base (MIB), 67
MANPATH environment variable, 15
Memory dump path, 32
Memory size
    mainframe, 21, 34
    SSD, 22, 34

__MEMORY__ token, 21
Messages, responding to, 1
MFBOOT parameter, 31
mfdump command, CPU-resident program, 31
mfinit diagnostic program, 32
MFINIT parameter, 29
MFIPATH parameter, 32
MFSTART parameter, 29
MFSYSDMP parameter, 31
MIB, 67
Monitors. *See* CPU monitors; Network monitor
Mouse button, in SNMP, 101
MUXIOP, definition, 18
    See also IOP
MWS-E, definition of, 1

N

Network monitor (xsnmpmon), 1
    agents, 68
    AT Table button, 76
    background information on SNMP, 67–68
    buttons, 71
    clients, 68
    Create Net button, 79–95
    creating a map for, 79
    EGP Neigh Tbl button, 77
    EGP Stats button, 75
    entering text, 71
    Error Log button, 78
    FDDI Attach button, 78–79
    FDDI MAC button, 77
    FDDI Port button, 77
    FDDI SMT button, 77
    I/F Stats button, 74
    ICMP Stats button, 75
    invoking, 69
    IP Addr Table button, 76
    IP Route button, 77
    IP Stats button, 75
    Log files, 104
    main window, 70
    Media Stats button, 74
    message line, 72
    MIB, 67
    Net Status button, 96–101
    Net/Media Table button, 76
    Notification button, 102–103
    overview, 67
    Perf Monitor button, 102
    RFC, 67
    scanning status of, 96
    Setup button, 72–73
    SNMP Stats button, 75–76
    State Chg Log button, 79
    synopsis information, 72
    System Info button, 74
    TCP Connection button, 77
    TCP Stats button, 75
    Telnet button, 104
    Trap Log button, 78
    UDP Listeners button, 77
    UDP Stats button, 75

O

olwm window manager, 13, 14
.openwin-init file, 13
.openwin-menu file, 13
OpenWindows, documentation, viii
Operator training, documentation, viii
Ordering publications, ix
OWS-E, man page manuals, vii
OWSECONFIG environment variable, 18

P

Parameters. *See* System configuration file
Partitions (disk), 10
Passwords, 39
Permissions file (/etc/owsepermfile), 40
Polling rate, passive CPU monitors, 35
Port value, 35
Priorities for line arbitration, 43–44
.profile file, 14, 15
Publications, ordering, ix

R

RCUD parameter, 28
Reader comments, x
Rebooting the OWS-E, 59
Register section structure, 49
Related publications, viii

Release level and .version, 11
Remote CPU request daemon, 28
Request for Comment (RFC), 67
Responding to messages, 1
RFC, 67
ROOTDIR parameter, 23

S

Scripts
    cpupanic, 27, 28, 51, 52
    dumpdly, 51, 53
    dumpwkly, 51, 53
    iophalt, 27, 28, 51, 52
    iophalt, 26
    overview, 51–52
    OWS-E backup scripts (dumpdly, dumpwkly), 53
Section header structure, 48
Security
    passwords, 39
    permissions file (/etc/owsepermfile), 40–41
    user and group identification, 39
SERIALNUMBER parameter, 26
__SERIALNUMBER__ token, 19
Service workstation interface (SWI), 5, 7
Shutting down the OWS-E
    clearing, 59
    if all else fails, 59
    normal circumstances, 55–56
    when the system is hung, 57–58
SMDEMON parameter, 27
SNMP
    See also Network monitor (xsnmpmon)
    commands, vii
Software overview, components, 9
SSD, memory size, 22, 34
SSD_MEMORY parameter, 34
__SSD_MEMORY__ token, 22
SSTBACKUP parameter, 35
Starting block, 33
Subnet mask, 76
SunOS
    commands, vii
    documentation, viii
SWI. *See* Service workstation interface (SWI)
System configuration file (/etc/configfile)
    ADUMPDIR parameter, 27
    AUTODUMP parameter, 28
    BASEPORT parameter, 35
    configurable parameters
        set at release time, 35–37
        set during installation, 23–34
    CPUD parameter, 28
    CPUD_HOSTNAME parameter, 34
    __CPUDHOSTNAME__ token, 22
    CPUPANIC parameter, 25
    CRAYMON parameter, 37
    DEF_MFCHAN parameter, 32
    DEFAULTDUMPDIR parameter, 27
    DEFAULTIKERNDIR parameter, 25
    DEFAULTIOP parameter, 26
    DEFAULTUKERNFILE parameter, 24
    DEFAULTUPARAMFILE parameter, 24
    DIOPATH parameter, 32
    DLEN parameter, 33
    D*n*FWA parameter, 33
    D*n*LWA parameter, 33
    DSTART parameter, 33
    __DSTARTBL__ token, 20
    DTYPE parameter, 32
    DUMP parameter, 27
    __DUMPIO__ token, 19
    __DUMPLEN__ token, 20
    __DUMPTYPE__ token, 19
    __DUMPUNIT__ token, 20
    DUNIT parameter, 33
    ECON parameter, 29
    EDIAG parameter, 28
    ERRLOG parameter, 36
    ERRLOGD parameter, 27
    HBEAT parameter, 28
    HCON parameter, 29
    __HOSTNAME__ token, 19
    IOP_DIAGNOSTICS parameter, 36
    IOPDEBUG parameter, 30
    IOPHALT parameter, 26
    IOPLOG parameter, 36
    IOPSAVE parameter, 30
    IOSCPATH parameter, 29
    IOSDPATH parameter, 29
    LAPFILE parameter, 37
    M_MEMORY parameter, 34
    MAIL_CPUFAIL parameter, 23
    MAIL_IOPFAIL parameter, 23
    MAINFRAME parameter, 33
    __MAINFRAME__ token, 20
    __MEMORY__ token, 21

MFBOOT parameter, 31
MFINIT parameter, 29
MFIPATH parameter, 32
MFSTART parameter, 29
MFSYSDMP parameter, 31
overview, 17
parameter order in configfile, 18
RCPUD parameter, 28
ROOTDIR parameter, 23
SERIALNUMBER parameter, 26
__SERIALNUMBER__ token, 19
SMDEMON parameter, 27
SSD_MEMORY parameter, 34
__SSD_MEMORY__ token, 22
SSTBACKUP parameter, 35
tokens, 18–22
UPDATESECS parameter, 35
System overview
directory structure, 11
disk partitions, 10

T

Tapes, 5
Tasks that can be performed with the OWS-E, 1
TCP/IP, 67, 75
Tokens, 18–22
See also System configuration file
Training, viii
Trap message, definition of in SNMP, 78
Typewriter font, v
Typographic conventions in this manual, v

U

UDP, 68, 75, 77
UNICOS
binary default, 24
commands, vii
documentation, viii
parameter file, 24
release level, v
UPDATESECS parameter, 35
User ID (login name), 39
Usurping tty lines and priorities in /etc/lapfile, 43

V

.version, 11

W

WIN, 7
Window manager (olwm), 13, 14

X

.Xdefaults file, 14
.xinitrc file, 14

Reader's Comment Form

OWS-E Operator Workstation Administrator's Guide SG–3079 2.0

Your reactions to this manual will help us provide you with better documentation. Please take a moment to complete the following items, and use the blank space for additional comments.

List the operating systems and programming languages you have used and the years of experience with each.

Your experience with Cray Research computer systems: _____0-1 year _____1-5 year _____5+years

How did you use this manual: _____in a class _____as a tutorial or introduction _____as a procedural guide _____as a reference _____for troubleshooting _____other

Please rate this manual on the following criteria:

	Excellent			Poor
Accuracy	4	3	2	1
Appropriateness (correct technical level)	4	3	2	1
Accessibility (ease of finding information)	4	3	2	1
Physical qualities (binding, printing, illustrations)	4	3	2	1
Terminology (correct, consistent, and clear)	4	3	2	1
Number of examples	4	3	2	1
Quality of examples	4	3	2	1
Index	4	3	2	1

Please use the space below for your comments about this manual. Please include general comments about the usefulness of this manual. If you have discovered inaccuracies or omissions, please specify the number of the page on which the problem occurred.

Name _____ Address _____
Title _____ City _____
Company _____ State/Country _____
Telephone _____ Zip code _____
Today's date _____ Electronic mail address _____

Cut along this line

Fold

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS MAIL PERMIT NO. 6184 ST. PAUL, MN

POSTAGE WILL BE PAID BY ADDRESSEE

CRAY RESEARCH, INC.

ATTN: Software Information Services
655 LONE OAK DR BLDG F
EAGAN MN 55121-9957
USA

Fold