OWS-E Operator Workstation Reference Manual

SR–3077 2.0

Cray Research, Inc.

Copyright © 1991, 1992 Cray Research, Inc. All Rights Reserved. This manual or parts thereof may not be reproduced in any form unless permitted by contract or by written permission of Cray Research, Inc.

SNMP software Copyright © 1988, 1989 by Carnegie Mellon University

Permission to use, copy, modify, and distribute Simple Network Management Protocol (SNMP) software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of CMU not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission.

CMU DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL CMU BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Autotasking, CRAY, Cray Ada, CRAY Y-MP, CRAY-1, HSX, SSD, UniChem, UNICOS, and X-MP EA are federally registered trademarks and CCI, CF77, CFT, CFT2, CFT77, COS, Cray Explorer, CRAY S-MP, CRAY X-MP, CRAY XMS, CRAY-2, Cray/REELlibrarian, CRInform, CRI/*Turbo*Kiva, CSIM, CVT, Delivering the power . . ., Docview, IOS, MPGS, OLNET, RQS, SEGLDR, SMARTE, SUPERCLUSTER, SUPERLINK, Trusted UNICOS, Y-MP, and Y-MP C90 are trademarks of Cray Research, Inc.

Ethernet is a trademark of Xerox Corporation. HP LaserJet III is a trademark of Hewlett Packard, Inc. HYPERchannel and NSC are trademarks of Network Systems Corporation. NeWS, NeWSprint, OpenWindows, SPARCstation, Sun, Sun-4, Sun Microsystems, Sun Workstation, SunView, XView, and X11/NeWS are trademarks of Sun Microsystems, Inc. Open Look and UNIX are trademarks of UNIX System Laboratories, Inc. Telex is a trademark of Memorex TELEX Corporation. X Window System is a trademark of Massachusetts Institute of Technology.

The UNICOS operating system is derived from the UNIX System Laboratories, Inc. UNIX System V operating system. UNICOS is also based in part on the Fourth Berkeley Software Distribution (BSD) under license from The Regents of the University of California.

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 Workstation Reference Manual SR–3077 2.0

This rewrite of the *OWS-E Operator Workstation Reference Manual* supports the 2.0 release of OWS-E. The changes include the following:

- The following man pages are new:
  - `configfile`(5) and `lapfile`(5) describe the OWS-E default configuration file and line-arbitration priority file
  - `owse_overview`(7) provides an overview of the OWS-E commands
  - `conv`(8) describes the command that converts a file from old `edump` file format to new `edump` file format
  - `craymon`(8) describes the command that monitors the mainframe status and sets the OWS-E screen background color
  - `dumpsys`(8) describes the command that takes a dump image of UNICOS
  - `eping`(8) describes the command that sends an echo packet to an IOP from the OWS-E
  - `lapdaemon`(8) describes the command that validates CRI `tty` lines for users
  - `fyadmin`(8), `fyformat`(8), and `fyroute`(8) describe the commands that control the `fy` driver, format the raw trace buffer information extracted from it, and set/display the driver's IP Interface Routing table
  - `snmpget`(1), `snmpgetnxt`(1), `snmpnetstat`(1), `snmpstatus`(1), `snmptest`(1), `snmptrap`(1), `snmptrapd`(1), `snmpwalk`(1), and `snmproute`(8) describe the various commands used with Simple Network Management Protocol (`SNMP`)
- The following pages have significant changes:
  - `ecrash`(8) has been updated to present a more uniform user interface and incorporate some features (variables and conditional execution) of a high-level language
  - `zip`(8) has been changed to apply to the `fy` driver
  - `eboot`(8), `ediag`(8), `edump`(8), and `ehalt`(8) all allow you to specify multiple clusters and IOPs in one command line
- The `-z` option has been eliminated, except for the `fyadmin`(8) command
- The following man pages have been deleted: `config`(3), `peek_cpu`(3), `peek_iop`(3), `cy`(4), `cz`(4), `bootall`(8), `booteiop`(8), `bootios`(8), `bootmux`(8), `cztool`(8), `echopkt`(8), `resetscreen`(8), `scyadmin`(8), `scytest`(8), `sysdump`(8), `systart`(8), and `ucon`(8).

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. These include the new `dumpsys`(8) command and changes to the options of several commands because of the incorporation of the `fy` driver.

This preface describes the scope of this manual and its audience, provides an overview of the interrelationships among some of the OWS-E scripts and commands, and lists conventions used in this manual, sources of more information, and ways you can send comments about this manual to Cray Research, Inc. (CRI).

ASSUMPTIONS

This guide was written for administrators of the OWS-E operator workstation. Readers should have at least 16 hours of 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.
`bold typewriter font`	In examples of interactive sessions, bold typewriter font denotes literal items entered by the user. Output is shown in nonbold typewriter font.

In this publication, *Cray Research*, *CRI*, and *Cray* refer to Cray Research, Inc. and/or its products. *CRAY Y-MP* always refers to a CRAY Y-MP mainframe with an I/O subsystem model E (IOS-E).

Most arguments require a leading 0 to specify octal interpretation (for example, 030). For arguments that assume octal, a leading 0 will not cause an error. Therefore, it is good practice to specify a leading 0 whenever you want octal interpretation.

The entries in this manual are based on a common format. The following list shows the order of sections in an entry and describes each section. Most entries contain only a subset of these sections.

NAME	Specifies the name of the entry and briefly states its function.
SYNOPSIS	Presents the syntax of the entry. The following conventions are used in this section: Brackets [ ] enclosing a command-line parameter indicate that the parameter is optional. When an argument or operand is shown as *name* or *file*, it always refers to a file name. Ellipses . . . indicate that the preceding command-line parameter may be repeated. A parameter beginning with a minus, plus, or equal sign (–, +, or =) is usually an option.
DESCRIPTION	Discusses the entry in detail.
NOTES	Points out items of particular importance.
CAUTIONS	Describes actions that can destroy data or produce undesired results.

WARNINGS	Describes actions that can harm people, equipment, or system software.
CONFIGURATION FILE PARAMETERS	Describes parameters from the OWS-E configuration file (by default, `/etc/configfile`) that are read by the command in question.
BUILT-IN COMMANDS	Describes subcommands that may be invoked from within a command.
ENVIRONMENT VARIABLES	Describes predefined shell variables that determine some of the characteristics of the shell or that affect the behavior of some programs, commands, or utilities.
RETURN VALUE	Describes possible error returns.
MESSAGES	Describes the informational, diagnostic, and error messages that may appear. Self-explanatory messages are not listed.
BUGS	Indicates known bugs and deficiencies.
EXAMPLES	Shows examples of usage.
FILES	Lists files that are either part of the entry or related to it.
SEE ALSO	Lists entries that contain related information and specifies the manual in which each entry appears.

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 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 named `ls` on-line by typing `man ls`.

Printed versions of the man pages are published in *OWS-E Operator Workstation Reference Manual,* publication SR–3077. Man pages are grouped into sections numbered. Each section contains entries of a particular type. Types of entries include user commands, administrator commands, and file formats.

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 and Publication**
1	SunOS user commands, found in *SunOS Reference Manual* (Vol. I)
	UNICOS User commands, found in *UNICOS User Commands Reference Manual,* publication SR–2011
	Simple Network Management Protocol (SNMP) user commands, found in *OWS-E Operator Workstation Reference Manual*, publication SR–3077
5	OWS-E file formats, found in *OWS-E Operator Workstation Reference Manual*, publication SR–3077
7	OWS-E topics, found in *OWS-E Operator Workstation Reference Manual*, publication SR–3077

8	OWS-E administrator commands, found in *OWS-E Operator Workstation Reference Manual*, publication SR–3077
	SunOS administrator commands, found in *SunOS Reference Manual* (Vol. III)
	UNICOS administrator commands, found in *UNICOS Administrator Commands Reference Manual*, publication 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 are related publications, listed 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 and Network Administration* (Vol. 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 user information	*UNICOS User Commands Reference Manual* (SR–2011) *UNICOS User Commands Ready Reference* (SQ–2056) *UNICOS Message Reference Manual* (SR–2200)
UNICOS 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). This document is CRAY RESEARCH PRIVATE. It can be distributed to non-CRI personnel only with approval of the appropriate Cray manager.

For a more detailed list of Sun Microsystem's, Inc., documentation, see the *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` (UNIXsystem 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 one of the following Internet addresses:

  `pubs3077@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

NAME

snmpget – Communicates with a network entity by using SNMP GET requests

SYNOPSIS

snmpget [-d] *host community variable-name* [*variable-name*]...

DESCRIPTION

The snmpget command is an SNMP application that uses the GET request to query for information on a network entity. You can specify one or more fully qualified object identifiers as arguments on the command line.

The snmpget command accepts the following arguments:

-d	Directs the application to dump input and output packets.
host	Specifies either a host name or an Internet address in dot notation.
community	Specifies the community name for the transaction with the remote system.
variable-name	Specifies the fully qualified object identifier to be retrieved by the snmpget request.

EXAMPLES

The following command retrieves the sysDescr.0 and sysUpTime.0 variables:

```
snmpget  dang.cray.com  criccn  mgmt.mib-2.system.sysDescr.0 \
mgmt.mib-2.system.sysUpTime.0
```

The output is as follows:

```
Name: mgmt.mib-2.system.sysDescr.0
OCTET STRING- (ascii):   Kinetics FastPath2

Name: mgmt.mib-2.system.sysUpTime.0
Timeticks: (2270351) 6:18:23
```

If the network entity encounters an error while processing the request packet, an error packet is returned and a message is shown, which helps to determine the error in the request. If other variables were in the request, the request is resent without the bad variable.

SEE ALSO

RFC 1155, RFC 1156, RFC 1157

NAME

snmpgetnxt – Communicates with a network entity by using SNMP GET NEXT requests

SYNOPSIS

snmpgetnxt [-d] *host community variable-name* [*variable-name*]. . .

DESCRIPTION

The snmpgetnxt command is an SNMP application that uses the GET NEXT request to query for information on a network entity. You can specify one or more object identifiers as arguments on the command line. For each one, the variable that is lexicographically "next" in the remote entity's Management Information Base (MIB) is returned.

The snmpgetnxt command accepts the following arguments:

-d	Directs the application to dump input and output packets.
host	Specifies either a host name or an Internet address in dot notation.
community	Specifies the community name for the transaction with the remote system.
variable-name	Specifies the fully qualified object identifier to be retrieved by the snmpgetnxt request.

EXAMPLES

The following command retrieves the sysDescr.0 and sysUpTime.0 variables:

```
snmpgetnxt  dang.cray.com  criccn  mgmt.mib-2.system.sysDescr.0 \
mgmt.mib-2.system.sysUpTime.0
```

The output is as follows:

```
Name: mgmt.mib-2.system.sysObjectID.0
OBJECT IDENTIFIER:   .iso.org.dod.internet.private.enterprises.34

Name: mgmt.mib-2.system.sysContact.0
OCTET STRING- (ascii): John Doe doe@cray.com
```

If the network entity encounters an error while processing the request packet, an error message is shown, which helps to determine the error in the request.

SEE ALSO

RFC 1155, RFC 1156, RFC 1157

NAME

`snmpnetstat` – Shows network status by using SNMP

SYNOPSIS

`snmpnetstat` *host community*
`snmpnetstat` *host community* `[-an]`
`snmpnetstat` *host community* `[-inrs]`
`snmpnetstat` *host community* `[-n] [-I` *interface*`]` *interval*
`snmpnetstat` *host community* `[-p` *protocol*`]`

DESCRIPTION

The `snmpnetstat` command symbolically displays the values of various network-related information retrieved from a remote system by using the SNMP protocol. There are several output formats, depending on the options for the information presented. The first form of the command displays a list of active sockets. The second form presents the values of other network-related information according to the option selected. Using the third form, with an *interval* specified, `snmpnetstat` continuously displays the information about packet traffic on the configured network interfaces. The fourth form displays statistics about the specified protocol.

The `snmpnetstat` command accepts the following arguments:

host	Specifies either a host name or an Internet address in dot notation.
community	Specifies the community name for the transaction with the remote system.
`-a`	With the default display, shows the state of all sockets; usually sockets used by server processes are not shown.
`-n`	Shows network addresses as numbers (usually `snmpnetstat` interprets addresses and attempts to display them symbolically). You can use this option with any of the display formats.
`-i`	Shows the state of all interfaces.
`-r`	Shows the routing tables. When `-s` is also present, shows routing statistics instead.
`-s`	Shows per-protocol statistics.
`-I` *interface*	Shows information about only the specified interface; used with the *interval* argument.
interval	Specifies interval (in seconds) through which packet traffic information is displayed.
`-p` *protocol*	Shows statistics about *protocol*, which is either a well-known name for a protocol or an alias for it. Some protocol names and aliases are listed in the `/etc/protocols` file. A null response typically means that there are no interesting numbers to report. The program complains if *protocol* is unknown or if no statistics routine for it exists.

For active sockets, the default display shows the local and remote addresses, protocol, and internal state of the protocol. If a socket's address specifies a network but no specific host address, address formats are of the form `host.port` or `network.port`. When known, the host and network addresses are displayed symbolically according to the `/etc/hosts` and `/etc/networks` databases, respectively. If a symbolic name for an address is unknown, or if the `-n` option is specified, the address is printed numerically, according to the address family. For more information about the Internet dot format, see `inet`(3). Unspecified or wildcard addresses and ports appear as `*`.

The interface display provides a table of cumulative statistics about packets transferred, errors, and collisions. The network addresses of the interface and the maximum transmission unit (MTU) are also displayed.

The routing table display indicates the available routes and their status. Each route consists of a destination host or network and a gateway to use when forwarding packets. The *flags* field shows the state of the route (U if up), whether the route is to a gateway (G), whether the route was created dynamically by a redirect (D), and whether the route was modified by a redirect (M). Direct routes are created for each interface attached to the local host; the gateway field for such entries shows the address of the outgoing interface. The interface entry indicates the network interface used for the route.

When you invoke `snmpnetstat` with an *interval* argument, it displays a running count of statistics related to network interfaces. This display consists of a column for the primary interface and a column summarizing information for all interfaces. Use the `-I` option to replace the primary interface with another interface. The first line of each screen of information contains a summary since the system was last rebooted. Subsequent lines of output show values accumulated over the preceding interval.

EXAMPLES

The following `snmpnetstat` commands produce network statistics:

```
snq1-% snmpnetstat localhost criccn -i
Name     Mtu   Network   Address         Ipkts Ierrs    Opkts Oerrs
hy0*     16432 none      none                0     0        0     0
hy1      16432 none      none           112544     0    87800     0
vme0*    16432 none      none                0     0        0     0
vme1*    16432 none      none                0     0        0     0
lsx0*    16432 none      none                0     0        0     0
hi0*     65528 none      none                0     0        0     0
hi1*     65528 none      none                0     0        0     0
unet0*   32880 none      none                0     0        0     0
lo0      65535 none      none            49528     0    49534     0

snq1-% snmpnetstat localhost criccn -I hy1
Name       Mtu   Network   Address         Ipkts Ierrs    Opkts Oerrs
hy1        16432 none      none           113178     0    88523     0

snq1-% snmpnetstat localhost criccn
Active Internet Connections
Proto Recv-Q Send-Q  Local Address           Foreign Address          (state)
tcp        0      0  *.1272                  *.*                      CLOSED
tcp        0      0  localhost.cray.c.1272   localhost.cray.c.sunrp   TIMEWAIT
tcp        0      0  snq1.cray.com.telnet    berserkly.cray.c.1518    ESTABLISHED
tcp        0      0  snq1.cray.com.telnet    cherry28.cray.co.1934    TIMEWAIT
tcp        0      0  snq1.cray.com.telnet    fir21.cray.com.1083      ESTABLISHED
tcp        0      0  snq1.cray.com.telnet    palm15.cray.com.1093     ESTABLISHED
tcp        0      0  snq1.cray.com.telnet    palm15.cray.com.1094     ESTABLISHED
tcp        0      0  snq1.cray.com.telnet    sumac15.cray.com.1256    ESTABLISHED
tcp        0      0  snq1.cray.com.telnet    sumac15.cray.com.1257    ESTABLISHED
tcp        0      0  snq1.cray.com.telnet    sumac15.cray.com.1258    ESTABLISHED
tcp        0      0  snq1.cray.com.telnet    hose.cray.com.2946       ESTABLISHED
tcp        0      0  snq1.cray.com.login     palm03.cray.com.1021     ESTABLISHED
tcp        0      0  snq1.cray.com.login     palm10.cray.com.1022     ESTABLISHED
tcp        0      0  snq1.cray.com.login     poplar17.cray.co.1021    ESTABLISHED
tcp        0      0  snq1.cray.com.809       aspen18.cray.com.980     TIMEWAIT
tcp        0      0  snq1.cray.com.815       cherry28.cray.co.shell   TIMEWAIT
```

SEE ALSO

`inet`(3) in the *Volume 2: UNICOS C Library Reference Manual*, publication SR–2080
`hosts`(5), `networks`(5), `protocols`(5), `services`(5) in the *UNICOS File Formats and Special Files Reference Manual*, publication SR–2014
RFC 1157

NAME

`snmpstatus` – Retrieves important information from a network entity by using SNMP requests

SYNOPSIS

`snmpstatus [-d]` *host community*

DESCRIPTION

The `snmpstatus` command is an SNMP application that retrieves several important statistics from a network entity.

The `snmpstatus` command accepts the following arguments:

`-d`	Directs the application to dump input and output packets.
host	Specifies either a host name or an Internet address in dot notation.
community	Specifies the community name for the transaction with the remote system. If you do not specify this argument, the community name defaults to `public`.

The information returned is as follows:

- The IP address of the entity.
- A textual description of the entity (`sysDescr.0`).
- The up time of the entity (`sysUpTime.0`).
- The sum of received packets on all interfaces (`ifInUCastPkts.* + ifInNUCastPkts.*`).
- The sum of transmitted packets on all interfaces (`ifOutUCastPkts.* + ifOutNUCastPkts.*`).
- The number of IP input packets (`ipInReceives.0`).
- The number of IP output packets (`ipOutRequests.0`).

EXAMPLES

The following `snmpstatus` command produces statistical information:

```
snmpstatus netdev-kbox.cc.cmu.edu public
```

The output is as follows:

```
[128.2.56.220]=>[Kinetics FastPath2] Up: 1 day, 4:43:31
IP recv/trans packets 262874/39867 |
IP recv/trans packets 31603/15805
```

The `snmpstatus` command also checks the operational status of all interfaces (`ifOperStatus.*`); if it finds any that are not running, it reports the interfaces as in the following example:

```
2 interfaces are down!
```

If the network entity encounters an error while processing the request packet, an error packet is returned and a message is shown, which helps to determine the error in the request. `snmpstatus` attempts to reform its request to eliminate the malformed variable, but this variable will then be missing from the displayed data.

SEE ALSO

RFC 1155, RFC 1156, RFC 1157

NAME

snmptest – Communicates with a network entity by using SNMP requests

SYNOPSIS

snmptest [-d] *host community*

DESCRIPTION

The snmptest command is a flexible SNMP application that can monitor and manage information on a network entity.

The snmptest command accepts the following arguments:

-d	Directs the application to dump input and output packets.
host	Specifies either a host name or an Internet address in dot notation.
community	Specifies the community name for the transaction with the remote system.

After invoking the program, a command-line interpreter begins to accept commands. It prompts with the following request:

```
Please enter the variable name:
```

You can enter one or more variable names, one per line. A blank line is a command to send a request for each of the variables (in a single packet) to the remote entity.

EXAMPLES

In the following snmptest command, the system.sysDescr.0 name is entered at the prompt:

```
snmptest netdev-kbox.cc.cmu.edu public
Please enter the variable name:  mgmt.mib-2.system.sysDescr.0
Please enter the variable name:
```

The following information about the request and reply packets is returned:

```
Name: system.sysDescr.0
OCTET STRING- (ascii):
```

On startup, the program defaults to sending a GET request packet. This can be changed to a GET NEXT request or a SET request by entering the $N or $S command, respectively. Entering $G returns you to the GET request mode.

The $D command toggles the dumping of each sent and received packet.

When in SET request mode, more information is requested by the prompt for each variable. The following prompt requests that you enter the type of the variable:

```
Please enter variable type [i|s|n]:
```

Enter i for an integer, s for an octet string, or n for a null value.

You are then prompted for a value, as follows:

```
Please enter new value:
```

If it is an integer value, enter the integer (in decimal). If it is a string, enter decimal numbers separated by white space, one per byte of the string. Again, enter a blank line at the prompt for the variable name to send the packet.

Entering `$Q` at the prompt quits the program.

SEE ALSO

RFC 1155, RFC 1156, RFC 1157

NAME

`snmptrap` – Sends an SNMP `TRAP` message to a host

SYNOPSIS

`snmptrap` *host community trap-type specific-type device-description* [`-a` *agent-addr*] [`-d`]

DESCRIPTION

The `snmptrap` command is an SNMP application that forms and sends an SNMP `TRAP` message to a host.

The `snmptrap` command accepts the following arguments:

host Specifies either a host name or an Internet address in dot notation.

community Specifies the community name for the transaction with the remote system.

trap-type Specifies the type of `TRAP` message being sent. Trap types are integers defined as follows:

`0` (Cold start)
The sending protocol entity is reinitializing itself such that the agent's configuration or the protocol entity implementation can be altered.

`1` (Warm start)
The sending protocol entity is reinitializing itself such that neither the agent configuration nor the protocol entity implementation is altered.

`2` (Link down)
The sending protocol entity recognizes a failure in one of the communication links represented in the agent's configuration.

`3` (Link up)
The sending protocol entity recognizes that one of the communication links represented in the agent's configuration has come up.

`4` (Authentication failure)
The sending protocol entity is the addressee of a protocol message that is not properly authenticated. While implementations of the SNMP must be able to generate this trap, they must also be able to suppress the emission of such traps through an implementation-specific mechanism.

`5` (EGP neighbor loss)
An exterior gateway protocol (EGP) neighbor for whom the sending protocol entity was an EGP peer was marked down and the peer relationship no longer remains.

`6` (Enterprise specific)
The sending protocol entity recognizes that some enterprise-specific event has occurred.

specific-type Identifies the particular trap that occurred.

device-description
Provides a textual description of the device sending this trap, which is used as the value of a `system.sysDescr.0` variable sent in the variable list of this trap message.

`-a` *agent-addr*
Changes the address from which the trap reports it is being sent; otherwise, the sending host's address is used. This argument is optional.

`-d` Directs the application to dump the input and output packets.

EXAMPLES

The following `snmptrap` command sends a cold start trap to the specified machine:

```
snmptrap nic.andrew.cmu.edu public 0 0
```

SEE ALSO

RFC 1155, RFC 1156, RFC 1157

NAME

snmptrapd – Receives and logs SNMP TRAP messages

SYNOPSIS

snmptrapd [-p] [-d]

DESCRIPTION

The snmptrapd command is an SNMP application that receives and logs SNMP TRAP messages sent to the SNMP-TRAP port (162) on the local machine.

The snmptrapd command accepts the following options:

-p Prints trap messages to the standard output; otherwise, it uses syslogd(8) to log messages. These syslog messages are sent with the level of LOG_WARNING and, if available (usually on BSD 4.3 systems), they are sent to the LOG_LOCAL0 facility.

Following is an example of a log message:

```
Sep 17 22:39:52 suffern snmptrapd: 128.2.13.41: Cold Start  \
       Trap (0) Uptime: 8 days, 0:35:46
```

-d Directs the application to dump input and output packets.

The snmptrapd command must be run as root so that UDP port 162 can be opened.

EXAMPLES

The following is an example of the use of snmptrapd. The snmpd daemon sends the coldstart trap (last line of the example) when it is started.

```
# snmptrapd -p &
# sdaemon -k snmpd
Stopping daemon: snmpd.
# sdaemon -s snmpd
Starting daemon: snmpd.
# 128.162.82.6: Cold Start Trap (0) Uptime: 0:00:00
```

SEE ALSO

syslogd(8) in the *UNICOS Administrator Commands Reference Manual*, publication SR–2022
RFC 1155, RFC 1156, RFC 1157

NAME

`snmpwalk`, `snmpwalka` – Communicates with a network entity by using SNMP `GET NEXT` requests

SYNOPSIS

`snmpwalk` *host community* [*variable-name*] [`-d`]

`snmpwalka` *host community* [*variable-name*] [`-d`]

DESCRIPTION

The `snmpwalk` command is an SNMP application that uses `GET NEXT` requests to query for a tree of information about a network entity. `snmpwalka` performs the same function asynchronously; it does not wait for a response from the agent before issuing another request.

The `snmpwalk` and `snmpwalka` commands accept the following arguments:

host Specifies either a host name or an Internet address in dot notation.

community Specifies the community name for the transaction with the remote system.

variable-name

Specifies the portion of the object identifier space that is searched, using `GET NEXT` requests. All variables in the subtree below the given variable are queried and their values presented to the user.

If the *variable-name* argument is not present, `snmpwalk` searches the whole Internet Management Information Base (MIB).

`-d` Directs the application to dump input and output packets.

EXAMPLES

The following example retrieves the mib-2 system variables:

```
snmpwalk   netdev-kbox.cc.cmu.edu public mgmt.mib-2.system
```

The output is as follows:

```
Name:  system.sysDescr.0
OCTET  STRING- (ascii):   Kinetics FastPath2

Name:       system.sysObjectID.0
OBJECT       IDENTIFIER: .iso.org.dod.internet.private.enterprises.\
            CMU.sysID.CMU-KIP

Name: system.sysUpTime.0
Timeticks: (2291082) 6:21:50
```

If the network entity encounters an error while processing the request packet, an error packet is returned and a message is shown, which helps to determine the error in the request.

If the tree search causes attempts to search beyond the end of the MIB, the following message is displayed:

```
End of MIB.
```

SEE ALSO

RFC 1155, RFC 1156, RFC 1157

NAME

`/etc/configfile` – Default OWS-E configuration file

DESCRIPTION

The system configuration file, `/etc/configfile`, contains the system parameter labels and their corresponding values used by the OWS-E software. This file is divided into two sections, as follows:

- Configurable parameters set during the installation process to site-specific values
- Configurable parameters set at release time

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 *OWS-E Operator Workstation Administrator's Guide*, CRI publication SG–3079.

The parameters, with brief descriptions, are listed here in alphabetical order; for more details, see *OWS-E Operator Workstation Administrator's Guide*, which lists parameters in the order in which they are found in `/etc/configfile`.

`ADUMPDIR`	Defines the path name of the directory holding the dump lock file mentioned in the descriptions of the `CPUPANIC` and `IOPHALT` parameters. Default: `/home/__HOSTNAME__/cri/bin/adm`
`AUTODUMP`	Defines the path name of the command that controls automatic dumping of the mainframe and the IOS-E. Default: `/home/__HOSTNAME__/cri/bin/autodump`
`BASEPORT`	Defines the starting port value used for the various operator interface software daemons. Default: `4370`
`BOOTFILE`	Defines the path name of the boot file. Default: `/var/logs/bootfile`
`CPUD`	Defines the path name of the command that gathers data and disperses CPU time statistics. Default: `/home/__HOSTNAME__/cri/bin/cpud`
`CPUD_HOSTNAME`	Specifies the name of the machine on which the CPU monitor, `cpud`(8), is running. Default: `__CPUDHOSTNAME__` token (replaced during installation)
`CPUPANIC`	Defines the path name of the `cpupanic`(8) script. Default: `/home/__HOSTNAME__/cri/bin/cpupanic`
`CRAYMON`	Defines the colors of the OWS-E that denote whether the mainframe is up (first color) or down (second color). The colors must consist of one word each, and they must be separated by a comma; spaces cannot appear within or between the two colors. Default: `SkyBlue,red`

D0FWA D0LWA D1FWA D1LWA D2FWA D2LWA D3FWA D3LWA	These parameters define the actual mainframe memory ranges to be dumped. Default: 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.
DEF_MFCHAN	Defines the mainframe channel number of the low-speed channel attached to the cluster that deadstarts the mainframe. Default: `020`
DEFAULTIDUMPDIR	Defines the default dump directory path in which the dump shell script is created. Default: `/var/dumps`
DEFAULTIKERNDIR	Defines the path name to the directory in which all of the IOS-E binary files are kept. Default: `/home/__HOSTNAME__/cri/os/ios`
DEFAULTIOP	Defines the default IOP through which the IOS-E is booted. Default: `0`
DEFAULTUKERNFILE	Defines the path name to the default UNICOS binary file. Default: `/home/__HOSTNAME__/cri/os/uts/unicos`
DEFAULTUPARAMFILE	Defines the path name to the default UNICOS parameter file. Default: `/home/__HOSTNAME__/cri/os/uts/param`
DIOPATH	Defines the path that the memory dump will take from the mainframe to the disk. Default: `__DUMPIO__` token (replaced during installation)
DLEN	Defines the length, in sectors, of the disk slice to which the memory will be dumped. Default: `__DUMPLEN__` token (replaced during installation)
DSTART	Defines the starting sector of the disk slice to which the memory will be dumped. Default: `__DSTARTBL__` token (replaced during installation)
DTYPE	Defines the type of the disk to which the mainframe memory will be dumped. Default: `__DUMPTYPE__` token (replaced during installation)
DUMP	Defines the name of the dump lock file mentioned in the descriptions of the `ADUMPDIR`, `CPUPANIC`, and `IOPHALT` parameters. Default: `dump.on`

`DUNIT`	Defines the default dump device unit that the `mfdump`(8) command uses when routing the `mfsysdmp` binary file to the mainframe before the dump. Default: `__DUMPUNIT__` (replaced during installation)
`EBOOT`	Defines the path name of the command that boots an IOP from the OWS-E. Default: `/home/__HOSTNAME__/cri/bin/eboot`
`ECON`	Defines the path name of the command that configures a MUXIOP-to-EIOP low-speed channel up or down. Default: `/home/__HOSTNAME__/cri/bin/econ`
`EDIAG`	Defines the path name of the command that boots deadstart diagnostic tests into a specified IOP. Default: `/home/__HOSTNAME__/cri/smarte/bin/ediag`
`ERRLOG`	Defines the path name of the error log file. Default: `/var/logs/errlog`
`ERRLOGD`	Defines the path name of the error logging daemon. Default: `/home/__HOSTNAME__/cri/bin/errlogd`
`HBEAT`	Defines the path name of the IOP monitor. Default: `/home/__HOSTNAME__/cri/bin/hbeat`
`HCON`	Defines the path name of the command that configures a MUXIOP high-speed channel up or down. Default: `/home/__HOSTNAME__/cri/bin/hcon`
`IOP_DIAGNOSTICS`	Controls whether diagnostic tests are run by `bootsys`(8) before booting the IOS-E. Valid values are `on`, `off`, and `only`. Default: `on`
`IOPDEBUG`	Defines the path name of a temporary file that the `ecrash`(8) utility uses during its processing. Default: `/home/__HOSTNAME__/cri/os/ios/iopdebug`
`IOPHALT`	Defines the path name of the `iophalt` script. Default: `/home/__HOSTNAME__/cri/bin/iophalt`
`IOPLOG`	Defines the path name of the IOP log file. Default: `/var/logs/ioplog`
`IOPSAVE`	Defines the path name of a temporary file used by the `edump`(8) utility during its processing. Default: `/home/__HOSTNAME__/cri/os/ios/iopsave`
`IOSCPATH`	Defines the path name of the I/O clear diagnostic test. Default: `/home/__HOSTNAME__/cri/os/ios/cleario`
`IOSDPATH`	Defines the path name of the IOP deadstart diagnostic test. Default: `/home/__HOSTNAME__/cri/os/ios/dsdiag`

`LAPFILE`	Specifies the location of the line-arbitration priority file used by `lapdaemon`(8). Default: `/etc/lapfile`
`M_MEMORY`	Defines the memory size of the mainframe to which the OWS-E is attached. Default: `__MEMORY__` token (replaced during installation)
`MAIL_CPUFAIL`	Defines the login name to which mail is sent if a CPU panics. Default: `cri`
`MAIL_IOPFAIL`	Defines the login name to which mail is sent if an IOP halts. Default: `cri`
`MAINFRAME`	Defines the type of mainframe to which the OWS-E is attached. Default: `__MAINFRAME__` token (replaced during installation)
`MFBOOT`	Defines the path name of the bootstrap loader program used by the `mfdump`(8) command. Default: `/home/__HOSTNAME__/cri/os/uts/mfboot`
`MFINIT`	Defines the path name of the command that runs a mainframe and IOS-E initialization and confidence test. Default: `/home/__HOSTNAME__/cri/bin/mfinit`
`MFIPATH`	Defines the path name of the diagnostic program used by the `mfinit`(8) command. Default: `/home/__HOSTNAME__/cri/os/uts/mfchkye`
`MFSTART`	Defines the path name of the command that starts the mainframe CPU from the OWS-E. Default: `/home/__HOSTNAME__/cri/bin/mfstart`
`MFSYSDMP`	Defines the path name of the CPU-resident program used by the `mfdump`(8) command. Default: `/home/__HOSTNAME__/cri/os/uts/mfsysdmp`
`MOTDPATH`	Specifies the path name of a text file that is displayed by the `bootsys`(8) comand prior to booting a system. For example, this file might be used to convey information about system reconfiguration. If this parameter is not defined or is commented out, no message is displayed. The file must be readable by the group `cri`. A commented-out example is given in `/etc/configfile`.
`RCPUD`	Defines the path name of the remote CPU request daemon. Default: `/home/__HOSTNAME__/cri/bin/rcpud`
`ROOTDIR`	Defines the base directory used by scripts to find the CRI commands that they execute during processing. Default: `/home/__HOSTNAME__/cri`
`SERIALNUMBER`	Defines the serial number of the CRI mainframe to which the OWS-E is attached. Default: `__SERIALNUMBER__` token (replaced during installation)

`SMDEMON`	Defines the path name of the daemon that monitors the OWS-E for SMARTE. Default: `/home/__HOSTNAME__/cri/smarte/bin/smdemon`
`SSD_MEMORY`	Defines the memory size of the SSD attached to the mainframe to which the OWS-E is attached. Default: `__SSD_MEMORY__` token (replaced during installation)
`SSTBACKUP`	Specifies the back-up `hbeat`(8) status table. Default: `/var/logs/sstbackup`
`UPDATESECS`	Defines (in seconds) the polling rate for the passive CPU monitors. Default: `5`

FILES

`/usr/openwin/lib/rgb.txt`	Default colors file

SEE ALSO

`getconfig`(8) for information about retrieving system parameter values from the system configuration file
OWS-E Operator Workstation Administrator's Guide, publication SG–3079, for more information about `/etc/configfile`

NAME

`/etc/lapfile` – Default line arbitration priority file

DESCRIPTION

The `lapfile` line arbitration priority file is used by the `lapdaemon`(8) program 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 them 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 in the following example:

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

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

To activate changes to `/etc/lapfile`, 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 terminate the PID with the following command line (where *lpid* is the `lapdaemon` PID number):

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

EXAMPLES

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

FILES

`/etc/configfile`	Default configuration file

SEE ALSO

`configfile`(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about `/etc/configfile`
`lapdaemon`(8) for information about the line arbitration priority daemon
`zip`(8) for information about the command that supplies the terminal interface to a running CPU

NAME

`/etc/owsepermfile` – Default OWS-E permissions file

DESCRIPTION

The `owsepermfile` permissions file lists those who can access the following OWS-E commands:

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

The `/etc/owsepermfile` file must contain an entry 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 using 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 using 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.

NOTES

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.

EXAMPLES

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.

SEE ALSO

`autodump`(8) for information about controlling the automatic dumping of the mainframe and the IOS-E
`bootsys`(8) for information about booting the IOS-E and the mainframe
`cpuhalt`(8) for information about issuing a CPU master clear to stop the mainframe
`craymon`(8) for information about monitoring the mainframe status and sets the OWS-E background color
`eboot`(8) for information about booting one or more IOPs from the OWS-E
`econ`(8) for information about configuring a MUXIOP-to-EIOP low-speed channel up or down
`ecrash`(8) for information about examining an IOS-E dump image or a running system
`edump`(8) for information about dumping IOS-E IOP local memory images to the OWS-E
`ehalt`(8) for information about halting one or more IOPs from the OWS-E
`emon`(8) for information about restarting the IOS-E error-logging, heartbeat, SMARTE, and CPU monitors
`eping`(8) for information about sending an echo packet to an IOP from the OWS-E
`errlogd`(8) for information about the IOS-E hardware error-logging daemon
`estat`(8) for information about checking IOP status
`dumpsys`(8) for information about taking a dump image of UNICOS
`hbeat`(8) for information about monitoring the IOS-E system
`hcon`(8) for information about configuring a MUXIOP high-speed channel up or down
`mfdump`(8) for information about dumping the mainframe memory and CPU registers to a CRI disk on the IOS-E
`mfinit`(8) for information about running a mainframe and IOS-E initialization and confidence test
`mfstart`(8) for information about starting the mainframe CPUs from the OWS-E
`peek`(8) for information about peeking (looking) at memory
`poke`(8) for information about poking (placing) a pattern into memory
`rcpud`(8) for information about processing service requests from the mainframe (IOS-E remote CPU daemon)
`zip`(8) for information about the program that acts as the terminal interface to a running CPU

NAME

owse_overview – An overview of the OWS-E commands

DESCRIPTION

The OWS-E commands may be grouped into the following audiences:

- Operators
- Adminstrators
- Analysts
- Other commands; that is, those not normally invoked manually

In the following sections, each command is listed under the audience category to which it most often applies; this does not imply that a command may not be used by someone in another category.

OPERATOR COMMANDS

The following commands are normally invoked by an operator:

bootsys(8)	Boots the IOS-E and the mainframe
craymon(8)	Monitors the mainframe status and sets the OWS-E background color (normally included in each operator's .xinitrc file)
dumpdly(8)	Performs an incremental (level-9) backup of the OWS-E file systems
dumpsys(8)	Takes a dump image of UNICOS
dumpwkly(8)	Performs a full (level-0) backup of the OWS-E file systems
graphs(8)	Displays CPU time statistics in graphic form
zip(8)	Acts as the terminal interface to a running CPU

ADMINISTRATOR COMMANDS

The following commands are normally invoked by an administrator:

autodump(8)	Controls the automatic dumping of the mainframe and the IOS-E
cpudump(8)	Forces a UNICOS dump (however, the function of the cpudump command has been made obsolete by the new dumpsys(8) command)
cpuhalt(8)	Issues a CPU master clear to stop the mainframe
eboot(8)	Boots one or more IOPs from the OWS-E
econ(8)	Configures a MUXIOP-to-EIOP low-speed channel up or down
edump(8)	Dumps IOS-E IOP local memory images to the OWS-E
ehalt(8)	Halts one or more IOPs from the OWS-E
emon(8)	Restarts the IOS-E error-logging, heartbeat, SMARTE, and CPU monitors
estat(8)	Checks IOP status
hcon(8)	Configures a MUXIOP high-speed channel up or down
mfdump(8)	Dumps the mainframe memory and CPU registers to a CRI disk on the IOS-E
mfinit(8)	Runs a mainframe and IOS-E initialization and confidence test
mfstart(8)	Starts the mainframe CPUs from the OWS-E
smdemon(8)	Monitors the OWS-E system for SMARTE

`smdstop`(8)	Terminates the SMARTE OWS-E system monitor daemon
`snmpget`(1)	Communicates with a network entity by using SNMP `GET` requests
`snmpgetnxt`(1)	Communicates with a network entity by using SNMP `GET NEXT` requests
`snmpnetstat`(1)	Shows network status by using SNMP
`snmproute(8)`	Performs route tracing with the Simple Network Management Protocol
`snmpstatus`(1)	Retrieves important information from a network entity
`snmptrap`(1)	Sends an SNMP `TRAP` message to a host
`snmptrapd`(1)	Receives and logs SNMP `TRAP` messages
`snmpwalk`(1)	Communicates with a network entity by using SNMP `GET NEXT` requests
`xsnmpmon`(8)	Invokes the SNMP network monitor

ANALYST COMMANDS

The following commands are normally invoked by an analyst:

`conv`(8)	Converts a file from old `edump` file format to new `edump` file format
`eping`(8)	Sends an echo packet to an IOP from the OWS-E
`ecrash`(8)	Examines an IOS-E dump image or a running system
`fyadmin`(8)	Controls the `fy` driver
`fyformat`(8)	Formats raw trace buffer information extracted from `fy` driver modules
`fyroute`(8)	Sets or displays the `fy` driver's IP Interface Routing table
`olnet`(8)	Detects and isolates network problems with the OLNET on-line diagnostic network communications tool
`peek`(8)	Peeks (looks) at memory
`poke`(8)	Pokes (places) a pattern into memory

COMMANDS USED BY OTHER COMMANDS

The following commands are normally invoked by another command:

`cpud`(8)	Gathers data and disperses CPU time statistics
`cpupanic`(8)	Takes a UNICOS panic dump
`ediag`(8)	Boots deadstart diagnostic tests (`dsdiag` or `cleario`) into one or more specified IOPs
`errlogd`(8)	IOS-E hardware error-logging daemon
`getconfig`(8)	Retrieves system parameter values from the system configuration file
`hbeat`(8)	Monitors the IOS-E system
`iophalt`(8)	Dumps an IOP in the event of an IOP failure
`lapdaemon`(8)	Validates CRI `tty` lines for users
`newlog`(8)	Creates new `errlog` and `ioplog` files while backing up the existing ones
`rcpud`(8)	Processes service requests from the mainframe (IOS-E remote CPU daemon)

NAME

`autodump` – Controls the automatic dumping of the mainframe and the IOS-E

SYNOPSIS

/home/*localhost*/cri/bin/autodump on
/home/*localhost*/cri/bin/autodump off

DESCRIPTION

The `autodump` script creates (`on`) or deletes (`off`) the file associated with the DUMP parameter in the file `/etc/configfile`. The existence of the file (determined by the values of `ADUMPDIR` and DUMP) controls the action of the mainframe and IOS-E automatic dump scripts, which are invoked by the `hbeat`(8) monitor. The `autodump` script is called by `bootsys`(8). (The `bootsys` script asks whether you want to enable automatic dumps; it then invokes `autodump` with the appropriate switch, depending upon your answer.)

The arguments to the `autodump` command are as follows:

`on`	Creates the dump file and enables automatic dumping of the mainframe and IOS-E.
`off`	Deletes the dump file and disables automatic dumping of the mainframe and IOS-E.

Permission to access this command is set in `/etc/owsepermfile` by the system administrator.

CONFIGURATION FILE PARAMETERS

The `autodump` command reads the following parameters from `/etc/configfile`:

`ADUMPDIR`	Defines the path name of the directory holding the dump lock file mentioned in the descriptions of the `CPUPANIC` and `IOPHALT` parameters. Default: /home/*localhost*/cri/bin/adm
`DUMP`	Defines the name of the dump lock file. Default: `dump.on`

ENVIRONMENT VARIABLES

`OWSECONFIG`	Specifies the system configuration file; by default, it is set to `/etc/configfile`

FILES

`/etc/configfile`	Default OWS-E configuration file
`/etc/owsepermfile`	Permissions file that contains a list of accounts and the commands they are allowed to access

SEE ALSO

`configfile`(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about `/etc/configfile`
`owsepermfile`(5) for information about the default OWS-E permission file
`bootsys`(8) for information about the `bootsys` command, which calls `autodump`
`hbeat`(8) for information about the heartbeat monitor

NAME

`bootsys` – Boots the IOS-E and the mainframe

SYNOPSIS

`/home/`*localhost*`/cri/bin/bootsys [-c` *cluster*`] [-d` *diags*`] [-D] [-f` *filesystem*`] [-F] [-i] [-n] [-p` *parameter*`] [-u` *binary*`] [-v] [-w]`

DESCRIPTION

The `bootsys` command boots the entire IOS-E and the mainframe. First it starts `hbeat`(8) to monitor for IOP halts and hangs, `errlogd`(8) to look for `HISP` errors, and `smdemon`(8) to monitor the OWS-E system for the System Maintenance and Remote Testing Environment (SMARTE). `bootsys` runs IOP boot-time diagnostic tests (unless specified otherwise), boots and configures each IOP, and runs a set of mainframe diagnostic tests by executing the `mfinit`(8) program. Finally, `bootsys` boots the mainframe by executing the `mfstart`(8) program and then, unless you specify the `-w` ("without `zip`") option, executes the `zip`(8) command to provide you with the UNICOS console. As appropriate, `bootsys` passes the parameters specified on the `bootsys` command line to the commands it executes as appropriate.

Normally, `bootsys` asks for confirmation before taking any potentially disruptive actions and asks you if you want to enable autodumps. You can force `bootsys` to **not** ask these questions by specifying the `-F` option.

When you execute `bootsys`, you may receive an OWS-E message-of-the-day statement. This statement is located in a file specified by the `MOTDPATH` parameter in the system configuration file (which by default is `/etc/configfile`).

The `bootsys` command retrieves the information it needs (such as the configuration of the IOS-E and the binaries to be loaded) from the UNICOS parameter file. For more information about this file, see *UNICOS System Administration*, publication SG–2113.

The arguments to `bootsys` are as follows:

`-c` *cluster* Specifies the cluster number to be used to control the mainframe master clear deadstart lines. The range of valid cluster numbers depends on the number of clusters in the IOS-E. The default is `0`.

`-d` *diags* Specifies whether IOP boot-time diagnostics are to be run. *diags* can be set to one of the following:

`on`	Runs diagnostics before an IOP is booted
`off`	Does not run diagnostics before an IOP is booted
`only`	Runs diagnostics but does not boot the IOP. (The mainframe diagnostics are not run, and the mainframe is not booted.)

If you do not specify this option, the default action is defined by the `IOP_DIAGNOSTICS` variable in the file `/etc/configfile`, which can be defined as either `IOP_DIAGNOSTICS on` or `IOP_DIAGNOSTICS off`. If this variable is not defined, `on` is the default.

`-D` Sets debug mode in the appropriate commands called by `bootsys` and sends the output to standard error.

`-f` *filesystem* Specifies the path name of a RAM file system image to be used when `mfstart`(8) boots UNICOS.

`-F` Forces `bootsys` not to ask for confirmation before booting the IOS-E and suppresses most informative messages. (Error messages will still appear.)

`-i`	Boots just the IOS-E. (The mainframe diagnostics are not run, and the mainframe is not booted.)
`-n`	(no execution) Goes through the steps of booting but does not actually boot the IOS-E or the mainframe. This is useful for catching syntax errors in the UNICOS parameter file.
`-p` *parameter*	Specifies the path name of the parameter file to use. If you do not specify the `-p` option, the default parameter file is specified by the `DEFAULTUPARAMFILE` parameter in `/etc/configfile`; by default, it is set to `/home/`*localhost*`/cri/os/uts/param`.
`-u` *binary*	Specifies the path name of the UNICOS binary file to use. If you do not specify the `-u` option, the default UNICOS binary file is specified by the `DEFAULTUKERNFILE` parameter in `/etc/configfile`; by default, it is set to `/home/`*localhost*`/cri/os/uts/unicos`.
`-v`	Sets verbose mode. This option forces the appropriate commands called by `bootsys` to print informative messages to standard error.
`-w`	(without `zip`) Prevents the execution of the `zip`(8) command at the end of the `bootsys` process.

Permission to access `bootsys` and the commands it calls -- `autodump`(8), `ediag`(8), `mfinit`(8), `mfstart`(8), and `zip`(8) -- is set in `/etc/owsepermfile` by the system administrator.

CONFIGURATION FILE PARAMETERS

The `bootsys` command reads the following parameters from `/etc/configfile`:

`AUTODUMP`	Defines the path name of the command that controls automatic dumping of the mainframe and the IOS-E. Default: `/home/`*localhost*`/cri/bin/autodump`
`DEFAULTUKERNFILE`	Defines the path name to the default UNICOS binary. Default: `/home/`*localhost*`/cri/os/uts/unicos`
`DEFAULTUPARAMFILE`	Defines the path name to the default UNICOS parameter file. Default: `/home/`*localhost*`/cri/os/uts/param`
`EDIAG`	Defines the path name of the command that boots deadstart diagnostics into a specified IOP. Default: `/home/`*localhost*`/cri/smarte/bin/ediag`
`ERRLOGD`	Defines the path name of the error logging daemon. Default: `/home/`*localhost*`/cri/bin/errlogd`
`HBEAT`	Defines the path name of the IOP monitor. Default: `/home/`*localhost*`/cri/bin/hbeat`
`IOP_DIAGNOSTICS`	Controls whether diagnostics are run by `bootsys`(8) before booting the IOS-E. Valid values are `on`, `off`, and `only`. Default: `on`
`IOPLOG`	Defines the path name of the IOP log file. Default: `/var/logs/ioplog`
`MFINIT`	Defines the path name of the command that runs a mainframe and IOS-E initialization and confidence test. Default: `/home/`*localhost*`/cri/bin/mfinit`

`MFSTART`	Defines the path name of the command that starts the mainframe CPU from the OWS-E. Default: `/home/`*localhost*`/cri/bin/mfstart`
`MAINFRAME`	Defines the type of mainframe to which the OWS-E is attached. Default: `__MAINFRAME__` token (replaced during installation)
`MOTDPATH`	Specifies the path name of a file that contains text that used to convey information, such as system reconfiguration. By default, this parameter is commented out in `/etc/configfile`.
`ROOTDIR`	Defines the base directory used by scripts to find the CRI commands that they execute during processing. Default: `/home/`*localhost*`/cri`
`SMDEMON`	Defines the path name of the daemon that monitors the OWS-E for SMARTE. Default: `/home/`*localhost*`/cri/smarte/bin/smdemon`

EXAMPLES

Example 1: The following sample `bootsys` session uses the `-n` option to check the validity of the UNICOS parameter file and the `-v` option to produce informative messages (`IOP_DIAGNOSTICS` is set to `off`):

```
ows1600% bootsys -nv
INFO: bootsys: no-execute mode - mainframe will not be booted.
INFO: bootsys: Analyzing the parameter file '/home/ows1600/cri/os/uts/param'.
Verifying root device accessibility.
Verifying swap device accessibility.
Diagnostics    : off
Boot cluster   : 0
Unicos kernel  : /home/ows1600/cri/os/uts/unicos
Boot cluster 0, iop 4 with /home/ows1600/cri/os/ios/iopmux
Boot cluster 0, iop 3 with /home/ows1600/cri/os/ios/eiop.dca1
Boot cluster 0, iop 2 with /home/ows1600/cri/os/ios/eiop.dca2
Boot cluster 0, iop 1 with /home/ows1600/cri/os/ios/eiop.bmx
Boot cluster 0, iop 0 with /home/ows1600/cri/os/ios/eiop.comm
HISP cluster 0, channel 010, mode c100d200, target mainframe
HISP cluster 0, channel 014, mode c100d200, target ssd
Boot cluster 1, iop 4 with /home/ows1600/cri/os/ios/iopmux
Boot cluster 1, iop 3 with /home/ows1600/cri/os/ios/eiop.hippi
Boot cluster 1, iop 2 with /home/ows1600/cri/os/ios/eiop.dca2
Boot cluster 1, iop 1 with /home/ows1600/cri/os/ios/eiop.dca2
Boot cluster 1, iop 0 with /home/ows1600/cri/os/ios/eiop.dca2
HISP cluster 1, channel 010, mode c100d200, target mainframe
HISP cluster 1, channel 014, mode c100d200, target ssd
ows1600%
```

Example 2: Suppose you wanted to boot the system using a UNICOS binary named `newunicos` and a parameter file named `newparam`, rather than the defaults for these files. If your home directory were `mydir`, you would enter the following:

```
bootsys -u ~mydir/newunicos  -p ~mydir/newparam
```

Example 3: Suppose you always want to execute the `-p` and `-v` arguments whenever you execute `bootsys`. You can use the `BOOTSYS_ARGS` to specify this, as follows:

```
setenv BOOTSYS_ARGS "-p ~mydir/myparam -v"        (C shell)
```

or

```
BOOTSYS_ARGS="-p ~mydir/myparam -v"               (Bourne shell)
export BOOTSYS_ARGS
```

ENVIRONMENT VARIABLES

`BOOTSYS_ARGS`	Specifies options and arguments that are always executed when you execute `bootsys`
`OWSECONFIG`	Specifies the system configuration file; by default, it is set to `/etc/configfile`

FILES

`/etc/configfile`	Default OWS-E configuration file
`/etc/owsepermfile`	Permissions file that contains a list of accounts and the commands they are allowed to access

SEE ALSO

`configfile`(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about `/etc/configfile`
`owsepermfile`(5) for information about the default OWS-E permission file
`autodump`(8) for information about controlling automatic dumping of the mainframe and the IOS-E
`eboot`(8) for information about booting the IOS-E from the OWS-E
`econ`(8) for information about configuring a MUXIOP-to-EIOP low-speed channel up or down
`errlogd`(8) for information about the IOS-E hardware error-logging daemon
`hbeat`(8) for information about monitoring the IOS-E system
`hcon`(8) for information about configuring a MUXIOP high-speed channel up or down
`mfinit`(8) for information about running a mainframe initialization and confidence test
`mfstart`(8) for information about starting the mainframe CPU from the OWS-E
`smdemon`(8) for information about the SMARTE daemon
`zip`(8) for information about the command that acts as the terminal interface to a running CPU
I/O Subsystem Model E (IOS-E) Guide, publication SD–2107, for illustrated descriptions of system deadstart. (This document contains information private to Cray Research, Inc. It can be distributed to non-CRI personnel only with approval of the appropriate Cray Research manager.)

NAME

conv – Converts a file from old edump file format to new edump file format

SYNOPSIS

/home/*localhost*/cri/bin/conv [-s *serial*] *oldformat newformat*

DESCRIPTION

The conv command reads a specified file (*oldformat*), converts it from the old edump file format into the new edump file format, and names the converted file *newformat*. The old file is not deleted.

The arguments to conv are as follows:

-s *serial*	Specifies the mainframe serial number for identification purposes
oldformat	Specifies the file to be converted
newformat	Specifies the name of the converted file

EXAMPLES

The following example converts a dump file named dump.2.10 from the old file format to the new, specifies the serial number, and names the new file dump.2.10.new:

```
conv -s 1600 dump.2.10 dump.2.10.new
```

SEE ALSO

ecrash(8) for information about using the IOS-E dump files
edump(8) for information about dumping the IOS-E
OWS-E Operator Workstation Administrator's Guide, publication SG–3079, for information about the file format

NAME

cpud – Gathers data and disperses CPU time statistics

SYNOPSIS

/home/*localhost*/cri/bin/cpud [-D]

DESCRIPTION

The cpud daemon gathers mainframe CPU statistics. It then listens on a socket (BASEPORT + 2 by default) for graphs(8) client connections, and sends out the gathered data to those clients connected.

The argument to cpud is as follows:

-D	Sets debug mode in the cpud program and sends the output to standard error.

First cpud determines the starting location in memory where the pws (processor working storage) kernel data structure resides. Then it looks at the pws structure in mainframe memory via the IOS-E.

This data is converted to Sun words (32 bits), and then written on the socket. This makes the data available to all graphs clients listening on that socket.

This command must be started manually when you are root. Permission to access this command is set in /etc/owsepermfile by the system administrator.

CONFIGURATION FILE PARAMETERS

The cpud command reads the following parameters from /etc/configfile:

BASEPORT	Defines the starting port value used for the various operator interface software daemons. Default: 4370

FILES

/etc/owsepermfile	Permissions file that contains a list of accounts and what commands they are allowed to access
/etc/configfile	Default configuration file where BASEPORT is defined

RETURN VALUES

If the cpud program completes successfully, a value of 0 is returned. If an error is encountered, a value of 1 is returned.

SEE ALSO

configfile(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about /etc/configfile
graphs(8) for information about displaying CPU time statistics

NAME

cpudump – Forces a UNICOS dump

SYNOPSIS

/home/*localhost*/cri/bin/cpudump

DESCRIPTION

The function of the cpudump command has been made obsolete by the new dumpsys(8) command. It is still provided with the OWS-E 2.0 release for backward compatibility, but it will be deleted in a future release.

The cpudump script allows you to force a dump of UNICOS. cpudump boots enough of the IOS-E to communicate with the dump device, then invokes mfdump to place a dump image on the disk slice specified by the mainframe dump configuration parameters DIOPATH, DLEN, DSTART, DTYPE, and DUNIT in /etc/configfile. The cpudump command logs its action in /var/logs/dumplog.

CONFIGURATION FILE PARAMETERS

The cpudump command reads the following parameters from /etc/configfile:

DIOPATH	Defines the path that the memory dump will take from the mainframe to the disk. Default: __DUMPIO__ token (replaced during installation)
DLEN	Defines the length, in sectors, of the disk slice to which the memory will be dumped. Default: __DUMPLEN__ token (replaced during installation)
DSTART	Defines the starting sector of the disk slice to which the memory will be dumped. Default: __DSTARTBL__ token (replaced during installation)
DTYPE	Defines the type of the disk to which the mainframe memory will be dumped. Default: __DUMPTYPE__ token (replaced during installation)
DUNIT	Defines the default dump device unit that the mfdump(8) command uses when routing the mfsysdmp binary file to the mainframe before the dump. Default: __DUMPUNIT__ (replaced during installation)

FILES

/etc/configfile	Default OWS-E configuration file
/var/logs/dumplog	Default UNICOS dump log file

SEE ALSO

configfile(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about /etc/configfile
cpupanic(8) for information about taking a UNICOS panic dump
dumpsys(8) for information about taking a dump image of mainframe memory
mfdump(8) for information about dumping the mainframe memory and CPU registers to a CRI disk on the IOS-E

NAME

`cpuhalt` – Issues a CPU master clear to stop the mainframe

SYNOPSIS

`/home/`*localhost*`/cri/bin/cpuhalt [-c` *cluster*`] [-D] [-F]`

DESCRIPTION

The `cpuhalt` command sends an S-packet to the MUXIOP in the designated cluster, requesting that the MUXIOP issue a CPU master clear. The MUXIOP must be running when this command is issued; otherwise, the command will time out. In normal operation, this command should only be used when UNICOS is in single-user mode and all caches have been flushed to disk.

The argument to `cpuhalt` is as follows:

`-c` *cluster*	Specifies the number of the IOS-E that controls the mainframe master clear and deadstart lines. The range of valid cluster numbers depends on the number of clusters in the IOS-E. The default is `0`.
`-D`	Sets debug mode in the `cpuhalt` program and sends the output to standard error.
`-F`	Forces `cpuhalt` not to ask for confirmation before issuing a master clear and suppresses most informative messages. (Error messages will still appear.)

Permission to access this command is set in `/etc/owsepermfile` by the system administrator.

CONFIGURATION FILE PARAMETERS

The `cpuhalt` command reads the following parameter from `/etc/configfile`:

`IOPLOG`	Defines the path name of the IOP log file. Default: `/var/logs/ioplog`

FILES

`/etc/owsepermfile`	Permissions file that contains a list of accounts and the commands they are allowed to access

SEE ALSO

`configfile`(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about `/etc/configfile`
`owsepermfile`(5) for information about the default OWS-E permission file

NAME

`cpupanic` – Takes a UNICOS panic dump

SYNOPSIS

`/home/`*localhost*`/cri/bin/cpupanic` *reason*

DESCRIPTION

When a UNICOS panic occurs, `cpupanic` boots the IOS-E and takes a dump image of the mainframe CPU by calling the `dumpsys(8)` command. `cpupanic` places this dump on the disk slice specified by the mainframe dump configuration parameters. `cpupanic` logs this action in `/var/logs/dumplog` and sends mail to the login name specified by the `MAIL_CPUFAIL` parameter in `/etc/configfile`. By default, this parameter is set to `cri`.

The `rcpud(8)` daemon calls `cpupanic`. Usually, `cpupanic` is used as part of a script. In the rare event that you enter it directly, you can enter a reason, as shown by *reason* in the SYNOPSIS line.

The argument to `cpupanic` is as follows:

reason	Specifies the reason `cpupanic` was entered. *reason* is truncated after 79 characters, not including quotation marks (a longer reason will not cause an error, but the 80th and succeeding characters will not be used). If you specify more than one reason, only the first one encountered is used.

The system administrator should modify the parameters listed in "CONFIGURATION FILE PARAMETERS" to site-specific values.

CONFIGURATION FILE PARAMETERS

The `cpupanic` command reads the following parameters from `/etc/configfile`:

`DEFAULTUPARAMFILE`	Defines the path name to the default UNICOS parameter file. Default: `/home/`*localhost*`/cri/os/uts/param`
`MAIL_CPUFAIL`	Defines the login name to which mail is sent if a CPU panics. Default: `cri`

FILES

`/etc/configfile`	Default OWS-E configuration file
`/var/logs/dumplog`	Default UNICOS panic log file
`/home/`*localhost*`/cri/os/uts/param`	Default UNICOS parameter file

SEE ALSO

`aliases(5)` for information about the SunOS file for `sendmail(8)`
`configfile(5)` and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about `/etc/configfile`
`cpudump(8)` for information about forcing a UNICOS dump
`dumpsys(8)` for information about the command that takes a dump image of UNICOS
`rcpud(8)` for information about the daemon that processes service requests from the mainframe and calls `cpupanic`
UNICOS System Administration, publication SG–2113.

NAME

craymon – Monitors the mainframe status and sets the OWS-E background color

SYNOPSIS

/home/*localhost*/cri/bin/craymon [-n] [-r *seconds*]

DESCRIPTION

The craymon command continually monitors the table produced by the hbeat(8) command to determine whether the mainframe is running and sets the background color of the OWS-E accordingly. As long as the mainframe is running properly, craymon sets the background of the OWS-E to the first color specified by the CRAYMON parameter in /etc/configfile; if the mainframe is not running, craymon sets the background to the second color specified by CRAYMON. By default, the background is set to SkyBlue when the mainframe is running and to red when it is not.

Normally, the operator should have the following line set in the .xinitrc file:

```
craymon &
```

This specifies that craymon will execute in the background for as long as the operator is logged in.

The arguments to craymon are as follows:

-n	(no-looping) Forces craymon to read the hbeat table just once, changing the background color as necessary. This option is useful if you suspect that your original craymon process has died and you want to determine the status of the mainframe.
-r *seconds*	(rate) Specifies the rate, in seconds, at which craymon will check the hbeat table. The default is 15.

NOTES

If you change the color values in /etc/configfile after invoking craymon, you must force craymon to reread the values by sending it a SIGUSR1 signal. To do this, you must determine the process ID number of the craymon program and then use the kill command to send the signal. For example, suppose that the ps command shows a process ID number of 239 for craymon; you could then enter the following to send a SIGUSR1 signal to craymon:

```
kill -SIGUSR1 239
```

CONFIGURATION FILE PARAMETERS

The craymon command reads the following parameters from /etc/configfile:

CRAYMON	Defines the colors of the OWS-E that denote whether the mainframe is up (first color) or down (second color). The colors must consist of one word each and they must be separated by a comma; spaces cannot appear within or between the two colors. Default: `SkyBlue,red`
IOPLOG	Defines the path name of the IOP log file. Default: `/var/logs/ioplog`

`SERIALNUMBER`	Defines the serial number of the CRI mainframe to which the OWS-E is attached. Default: `__SERIALNUMBER__` token (replaced during installation)

EXAMPLES

The following example specifies that `craymon` should read the `hbeat` table once every minute:

```
craymon -r 60
```

ENVIRONMENT VARIABLES

`OWSECONFIG`	Specifies the system configuration file; by default, it is set to `/etc/configfile`

FILES

`/usr/openwin/lib/rgb.txt`	Default colors file

SEE ALSO

`configfile`(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about `/etc/configfile`
`hbeat`(8) for information about monitoring the IOS-E system

NAME

`dumpdly` – Performs an incremental (level-9) backup of the OWS-E file systems

SYNOPSIS

`/home/`*localhost*`/cri/bin/dumpdly`

DESCRIPTION

The `dumpdly` ("dump daily") script calls the SunOS `dump`(8) utility with the various file systems configured on the OWS-E. You should use it every day to perform an incremental (level-9) backup of the system, which will produce a dump of all files that have changed since the previous lower-level backup. Every week, you should run the `dumpwkly` ("dump weekly") script to perform a full (level-0) backup of the complete system.

By using the `dumpdly` and `dumpwkly` scripts as directed, you will ensure valid backups as well as save on the actual time necessary to perform the backups. Use the `restore`(8) command to restore a dump to a system. (The `restore` command is part of SunOS.)

If you want to be certain that no files change while you are performing the dump, execute `dumpdly` in single-user mode on the OWS-E.

NOTES

The `dumpdly` script assumes the following file system configuration:

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

If the site's configuration does not match this, the system administrator must modify the `dumpdly` script to ensure valid backups.

SEE ALSO

`dumpwkly`(8) for information about the script that performs weekly level-0 dumps
`dump`(8) and `restore`(8) in Sun Microsystem *SunOS Reference Manual* for more details about these utilities and level-9 and level-0 backups
OWS-E Operator Workstation Administrator's Guide, publication SG–3079, for more information about backup procedures

NAME

dumpsys – Takes a dump image of UNICOS

SYNOPSIS

/home/*localhost*/cri/bin/dumpsys [-b *bootstrap*] [-c *cluster*] [-D] [-f *mfsysdmp*] [-F] [-i] [-I *iosdump*] [-n] [-p *paramfile*] [-v] [*reason*]

DESCRIPTION

The dumpsys command takes a dump image of UNICOS. dumpsys reads the UNICOS parameter file, boots the MUXIOP that has the deadstart line to the mainframe (which can be modified by the -c option), and stops the mainframe. It then boots the disks and MUXIOPs needed for the dump device (as specified in the UNICOS parameter file) and writes the mainframe system dump binary file. dumpsys then writes the exchange package into mainframe memory and starts the mainframe, polling for status returns from the dump binary file until it times out after 1 minute of inactivity or when a success or failure is encountered. Finally, it stops the mainframe and IOPs.

The arguments to dumpsys are as follows:

-b *bootstrap*	Specifies the full path name of the bootstrap loader program for the CPU. If you do not specify this option, the program used will be the one specified by MFBOOT in /etc/configfile; by default, this program is /home/*localhost*/cri/os/uts/mfboot.
-c *cluster*	Specifies the cluster through which the dump image will be taken. The default is 0.
-D	Sets debug mode in dumpsys and sends the output to standard error.
-f *mfsysdmp*	Specifies the mainframe system dump binary file. The default is specified by MFSYSDMP in /etc/configfile; at release, MFSYSDMP is set to /home/*localhost*/cri/os/uts/mfsysdmp.
-F	Forces dumpsys not to ask for confirmation before dumping the mainframe and suppresses most informative messages. (Error messages will still appear.)
-i	Boots the required portions of the IOS-E but does not write the mainframe system dump binary file nor the exchange package and does not start the mainframe. This option is useful when you want to use the mfdump(8) command after executing dumpsys. It is not logical to use the -f or -b options with this option.
-I *iosdump*	Specifies a name for an accompanying IOS-E dump file that resides on the OWS-E; this action would usually be performed with edump -e. This option will be useful with future releases of the UNICOS crash(8) command; UNICOS implementation is currently deferred.
-n	(no execution) Goes through the steps of dumping but does not actually dump the mainframe. This is useful for catching syntax errors.
-p *paramfile*	Specifies the parameter file that describes the system. The default for *parameter* is specified by the DEFAULTUPARAMFILE parameter in /etc/configfile; at release, DEFAULTUPARAMFILE is set to /home/*localhost*/cri/os/uts/param.
-v	Sets verbose mode. This option forces dumpsys to print informative messages to standard error.

reason	Specifies the reason for the dump; for example, "CPU Hung". Quotation marks are optional, even if the reason contains white space; however, you should use them to avoid problems with special characters. *reason* is truncated after 79 characters, not including quotation marks (a longer reason will not cause an error, but the 80th and succeeding character will not be used). Entering a halt code as a reason helps to distinguish dumps as dump files start to accumulate.

EXAMPLE

If you want to specify cluster 1 rather than cluster 0, a parameter file named `param.test`, and verify your syntax, 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, 2 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
```

CONFIGURATION FILE PARAMETERS

The `dumpsys` command reads the following parameters from `/etc/configfile`:

DEFAULTUPARAMFILE	Defines the path name to the default UNICOS parameter file. Default: `/home/`*localhost*`/cri/os/uts/param`
IOPLOG	Defines the path name of the IOP log file. Default: `/var/logs/ioplog`
MFBOOT	Defines the path name of the bootstrap loader program used by the `mfdump`(8) command. Default: `/home/`*localhost*`/cri/os/uts/mfboot`
MFSYSDMP	Defines the path name of the CPU-resident program used by the `mfdump`(8) command. Default: `/home/`*localhost*`/cri/os/uts/mfsysdmp`

ENVIRONMENT VARIABLES

`OWSECONFIG`	Specifies the system configuration file; by default, it is set to `/etc/configfile`

FILES

`etc/configfile`	Default OWS-E configuration file

SEE ALSO

`configfile`(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about `/etc/configfile`
`edump`(8) for information about dumping IOS-E IOP local memory images to the OWS-E
`mfdump`(8) for information about manually dumping the mainframe memory and CPU registers to a CRI disk on the IOS-E

NAME

dumpwkly – Performs a full (level-0) backup of the OWS-E file systems

SYNOPSIS

/home/*localhost*/cri/bin/dumpwkly

DESCRIPTION

The dumpwkly ("dump weekly") script calls the SunOS dump(8) utility along with the various file systems configured on the OWS-E. You should use it every week to perform a full (level-0) backup of the complete system. Every day, you should perform an incremental backup with the dumpdly ("dump daily") script to back up those files that have changed since the previous level-0 backup.

By using the dumpdly and dumpwkly scripts as directed, you will ensure valid backups as well as save on the actual time necessary to perform the backups. Use the restore(8) command to restore a dump to a system. (The restore command is part of SunOS.)

If you want to be certain that no files change while you are performing the dump, execute dumpwkly in single-user mode on the OWS-E.

NOTES

The dumpwkly script assumes the following file system configuration:

File System	**Mounted On**
/dev/sd0a	/
/dev/sd0d	/usr
/dev/sd0g	/home
/dev/sd0h	/home/*localhost*/cri
/dev/sd0f	/var

If the site's configuration does not match this, the system administrator must modify the dumpwkly script to ensure valid backups.

SEE ALSO

dumpdly(8) for information about the script to perform daily level-9 dumps
dump(8) and restore(8) in Sun Microsystem *SunOS Reference Manual* for more details about these utilities and level-9 and level-0 backups
OWS-E Operator Workstation Administrator's Guide, publication SG–3079, for more information about backup procedures

NAME

`eboot` – Boots one or more IOPs from the OWS-E

SYNOPSIS

`/home/`*localhost*`/cri/bin/eboot [-d] [-D] [-F] [-h] [-M] [-v]` *cluster* : *iop* :*file* [*cluster* : *iop* :*file*] ...

`/home/`*localhost*`/cri/bin/eboot [-d] [-D] [-F] [-h] [-M] [-v] -c` *cluster* `-i` *iop* `-f` *file*

DESCRIPTION

The `eboot` command loads a selected I/O processor (IOP) with a binary file (specified with the `-f` option) and deadstarts the IOP. `eboot` records its actions in the file specified by the `IOPLOG` variable in `/etc/configfile` (which at release is set to `/var/logs/logfile`).

Usually, you will use the `bootsys`(8) command to boot the system, rather than the `eboot` command. For normal system operation, use the `econ`(8) command to configure up a channel between an EIOP and the MUXIOP and use the `hcon`(8) command to configure up the MUXIOP high-speed channels.

Normally, `eboot` asks for confirmation before taking any action. You can force `eboot` to **not** ask for confirmation by specifying the `-F` option.

Permission to access this command is set in `/etc/owsepermfile` by the system administrator.

You must specify either *cluster* : *iop* :*file* or the `-c`, `-i`, and `-f` options. The arguments to the `eboot` command are as follows:

cluster : *iop* :*file* [*cluster* : *iop* :*file*] ...
: Specifies the cluster and IOP that should be booted with the specified binary file; at least one set of *cluster* : *iop* :*file* must be specified. You may specify more than one set of *cluster* : *iop* :*file* if you separate the sets with white space.

  The range of valid values for *cluster* consists of integers from `0` through `15`, depending upon the hardware configuration at you site. *cluster* may be a single integer, a list of integers separated by commas, or a range of integers separated by a hyphen.

  The range of valid values for *iop* consists of integers in the range `0` through `4`, depending upon the hardware configuration at your site. *iop* may be a single integer, a list of integers separated by commas, or a range of integers separated by a hyphen.

  The specified *file* may begin with a tilde character (~), which is expanded to the home directory of the login name that follows the tilde or, if not followed by a login name, to your home directory.

  For examples of legal syntax, see the "EXAMPLES" section.

`-d`
: Sets debug mode in the IOPs booted.

`-D`
: Sets debug mode in the `eboot` program and sends the output to standard error.

`-F`
: Forces `eboot` to not ask for confirmation before booting an IOP and suppresses most informative messages. (Error messages will still appear.)

`-h`
: Sets headerless load mode. Use this option when the file you specify does not have a header. If you do not specify this option, the `eboot` program strips the 64-byte `segldr`(1) header from the binary file; if there is no header, information will be lost. (By default, IOP binaries include a 64-byte `a.out` header.)

`-M`
: Issues the cluster master clear function. If booting the MUXIOP, a cluster master clear is issued to all IOPs in the cluster. **CAUTION**: this stops any other IOPs running in that cluster; therefore, you should specify the MUXIOP before other clusters.

`-v`	Sets verbose mode. This option forces `eboot` to print informative messages to standard error.
`-c` *cluster*	Specifies the IOS-E cluster number. The range of valid cluster numbers depends on the number of clusters in the IOS-E. If you specify this option, you must also specify `-i` and `-f`.
`-i` *iop*	Specifies the number of the IOP to be booted. *iop* can be an integer in the range `0` through `4` (`4` indicates the MUXIOP; `0` through `3` indicate EIOPs) or `mux` for the MUXIOP. If you specify this option, you must also specify `-c` and `-f`.
`-f` *file*	Specifies the path name of the binary file to boot. If you specify this option, you must also specify `-c` and `-i`.

Permission to access this command is set in `/etc/owsepermfile` by the system administrator.

EXAMPLES

All of the following example formats are legal:

```
eboot 0:1:/home/owse/cri/os/ios/bootit
eboot 0:1:/home/owse/cri/os/ios/boot.1 0:2:/home/owse/cri/os/ios/boot.2
eboot 0,5:0:/home/owse/cri/os/ios/bootit
eboot 1-15:0,4:~cri/os/ios/bootit
```

The following example loads and deadstarts the `/`*username*`/eiop` binary file into IOS 0, cluster 4, IOP 3:

```
    eboot -c 4 -i 3 -f /username/eiop
```

CONFIGURATION FILE PARAMETERS

The `eboot` command reads the following parameter from `/etc/configfile`:

`IOPLOG`	Defines the path name of the IOP log file. Default: `/var/logs/ioplog`

RETURN VALUES

When `eboot` executes properly, the return value is `0`. If an error occurs, the return value will be `1`. If the binary file cannot be found, the return value will be `2`.

FILES

`/etc/configfile`	Default OWS-E configuration file
`/etc/owsepermfile`	Permissions file that contains a list of the accounts that may access each command
`/var/logs/ioplog`	Default file that stores a history of IOPs that have been booted

SEE ALSO

`configfile`(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about `/etc/configfile`
`owsepermfile`(5) for information about the default OWS-E permission file
`bootsys`(8) for information about the command that boots the IOS-E and the mainframe
`econ`(8) for information about configuring the MUXIOP-to-EIOP low-speed channel
`hcon`(8) for information about configuring the MUXIOP high-speed channel
`segldr`(1) in *UNICOS User Commands Reference Manual*, publication SR–2011, for information about linking relocatable object modules to produce an executable program
I/O Subsystem Model E (IOS-E) Guide, publication SD–2107, for illustrated descriptions of system deadstart. (This document contains information private to Cray Research, Inc. It can be distributed to non-CRI personnel only with approval of the appropriate Cray Research manager.)

NAME

econ – Configures a MUXIOP-to-EIOP low-speed channel up or down

SYNOPSIS

/home/*localhost*/cri/bin/econ [-c *cluster*] [-d] [-D] *eiop*

DESCRIPTION

The econ command configures a MUXIOP-to-EIOP low-speed channel up by default. The channel is configured down if you specify the -d option. The MUXIOP of the designated cluster must be running.

The arguments to econ are as follows:

-c *cluster*	Specifies the cluster in which the MUXIOP resides; the range of valid cluster numbers depends on the number of clusters in the IOS-E. The default is 0.
-d	Configures the channel down. If you do not specify this option, econ configures the channel up.
-D	Sets the debug mode in the econ program and sends the output to standard error.
eiop	Specifies the EIOP number, which can be an integer in the range 0 through 3.

Log messages generated by this command are sent to the file specified by the IOPLOG parameter in /etc/configfile. By default, IOPLOG is set to /var/logs/ioplog. Error messages are written to standard error.

Permission to access this command is set in /etc/owsepermfile by the system administrator.

EXAMPLES

The following example configures down a MUXIOP low-speed channel to IOS 0, cluster 3, EIOP 2:

```
econ -d -c 3 2
```

CONFIGURATION FILE PARAMETERS

The econ command reads the following parameter from /etc/configfile:

IOPLOG	Defines the path name of the IOP log file. Default: /var/logs/ioplog

FILES

/etc/configfile	Default OWS-E configuration file in which IOPLOG is specified
/etc/owsepermfile	Permissions file that contains a list of accounts that may access each command
/var/logs/ioplog	Default IOP log file

SEE ALSO

configfile(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about /etc/configfile
owsepermfile(5) for information about the default OWS-E permission file
hcon(8) for information about configuring a MUXIOP high-speed channel up or down

NAME

`ecrash` – Examines an IOS-E dump image or a running system

SYNOPSIS

`/home/`*localhost*`/cri/bin/ecrash [-f` *file*`]` [*script*]

DESCRIPTION

The `ecrash` command is an interactive utility for examining either an IOS-E dump image file (off-line mode) or a running system (on-line mode). It allows you to examine registers, the exit stack, flags, local memory, and channel buffers. On running systems, it also includes an IOS-E debugger and allows you to examine CPU and SSD memory.

Note: `ecrash` commands can no longer be abbreviated. Also, you must enclose statements in quotation marks where noted.

The arguments to `ecrash` are as follows:

`-f` *file*	Specifies the dump file that you want to examine with `ecrash`. Specify the complete path name if the file is not in your current directory. If you do not specify this option, `ecrash` will assume that it is trying to process a running system and will proceed to look at the actual hardware. When you enter `ecrash`, the prompt reminds you of the cluster and IOP numbers being used; for example, the `c0-i0->` prompt reminds you that you are using cluster 0 and IOP 0.
script	Specifies an executable file that contains `ecrash` built-in commands to be executed. See "BUILT-IN COMMANDS."

Upon invocation, `ecrash` reads and executes `ecrash` built-in commands listed in the `$HOME/.ecrashrc` file (if it exists and is readable). Thereafter, `ecrash` reads commands from standard input or a script file. Use a new-line character to separate commands; you can use braces ({}) to group commands to form a single statement. `ecrash` supports a set of mathematical operators and simple program-flow statements such as `while` and `for` loops.

The `ecrash` program has some characteristics of a programming language and was modeled after the C language. The following sections describe the built-in commands, the use of expressions and registers, flow control, and I/O redirection.

Permission to access this command in on-line mode is set in `/etc/owsepermfile` by the system administrator.

BUILT-IN COMMANDS

The `ecrash` program contains the following built-in commands:

address	Displays the number of words specified by the `words` command or the `WORDS` variable, starting at *address* in memory.
+ [*expression*]	Moves forward *expression* locations in memory and displays the number of subsequent words specified by the `words` command or the `WORDS` variable. The default is 1 word.
– [*expression*]	Moves backward *expression* words in memory and displays the number of preceding words specified by the `words` command or the `WORDS` variable. The default is 1 word.

. + *expression*	Adds this offset (*expression*) to the beginning address of the currently displayed memory, then displays the subsequent number of words specified by the `words` command or the `WORDS` variable. If you are displaying CPU or channel buffer memory, *expression* is interpreted as words; if you are displaying local IOP memory, *expression* is interpreted as parcels.
`<RETURN>)`	Displays the number of subsequent words ("page forward") in memory specified by the `words` command or `WORDS` variable, starting from the last displayed address.
`<`	Displays the number of preceding words ("page back") in memory specified by the `words` command or `WORDS` variable, from the beginning address of the current memory display. This is valid only in on-line mode.
`!!`	Repeats the last command line.
`?`	Displays command summary information; this is the same as the `help` command.
`a`	Displays the A register. If you are examining a running system, this command is valid only when you are in the debugger on the IOS-E.
`b`	Displays the B register. If you are examining a running system, this command is valid only when you are in the debugger on the IOS-E.
`c`	Displays the C register. If you are examining a running system, this command is valid only when you are in the debugger on the IOS-E.
`calc` *expression*	Calculates and displays the specified expression in octal, hexadecimal, and signed and unsigned decimal forms.
`cb` [*expression*]	Sets the current memory type to the channel buffer accessed by channel *expression*. The default is channel `030`. See also `cpu`, `local`, and `ssd`.
`change` *type* [*expression*]	Changes memory, register, stack, and so on in an IOP if it is in the IOS-E debugger. If you specify *type* without expression, you will receive the current value for *type* and will be asked if you want to change it; if you specify *expression* with *type*, the value of *expression* will be immediately used. *type* can be set to one of the following: `base` (IOS-E base register); `a` (A register); `b` (B register); `c` (C register); `e` (E register); `r[`*expression*`]`, in which *expression* must evaluate to a number in the range `0` to `0177` (R register); `e[`*expression*`]`, in which *expression* must evaluate to a number in the range `0` to `037` (E stack). *type* can also be an expression itself, in which case the value of the expression is used as an address in the currently selected memory target. Note that you must literally enter the brackets shown in `e[`*expression*`]` and `r[`*expression*`]`.
`cluster` [*expression*]	Sets the current cluster number to *expression*. The legal range for *expression* is an integer from `0` through `15`; if *expression* evaluates to an invalid number, the next access attempt will report an error.
`cpu`	Displays CPU memory. See also `cb`, `local`, and `ssd`.
`debug off`	Exits the debugger.
`debug on`	Enters the debugger.
`display` [*type*]	Sets the display base type to either octal or hexadecimal; *type* can be set to either `oct` (octal) or `hex` (hexadecimal). The default is `oct`.
`e`	Displays the E pointer. When you are on a running system, this command is valid only when you are in the debugger on the IOS-E.

`e` *n*	Displays the E stack entry *n*; *n* can be set to an integer value from `0` through `040` (octal). When you are examining a running system, this command is valid only when you are in the debugger on the IOS-E.
`flags` [*expression*]	Displays the specified *expression* channel flags; the default is to print all channel flags. If you are on a running system, this command is valid only when you are in the debugger on the IOS-E.
`format` *type*	Sets the memory display format according to the *type* specified. *type* can be one of the following:

`bit`	Displays memory as a bit field
`byte`	Displays memory as bytes
`elan`	Displays local memory as IOS-E elan instructions
`parcel`	Displays memory as parcels
`word`	Displays memory as words

`help`	Prints a usage summary for on-line or off-line mode, depending upon the current mode. Because the output may be more than one screenful, you may want to pipe it through a pager such as the SunOS `more`(1) command. This command is the same as the `?` command.
`i`	Displays the Interrupt Enable flag. This command is valid only when you are in the debugger on the IOS-E or for a dump.
`include "`*file*`"`	Reads input from the specified file until the end-of-file character; you must enclose *file* in quotation marks. If the file cannot be opened, `ecrash` will issue an error message. You can have up to 10 nested `include` statements.

`issue` [*channel*] [*function*] [`A` *reg*]

Reads a channel state if only a channel number is specified, or reads a channel state and issues a channel function as follows (this command is valid only when you are in the debugger on the IOS-E):

channel	Specifies the channel number; *channel* can be set to an integer from `0` through `65536`.
function	Specifies the function; *function* can be set to an integer from `0` through `65536`.
`A` *register*	Specifies the contents of the accumulator register at the time the function is issued; *register* can be set to an integer from `0` through `65536`. `0` is the default.

The following example issues function 17 (octal) with accumulator register 3 to channel 20 (octal):

```
issue 20 17 3
```

`local`	Sets the current memory type to IOP local memory. See also `cb`, `cpu`, and `ssd`.

`od`
`od` *address*
`od` *address* `,` *amount*

`od` *address* `,` *amount* `,` *format*

Formats and displays the currently selected memory type.

address	Specifies the address to be dumped; *address* can be an expression. The default is `0`.

amount	Specifies the number of words to display; *amount* can be an expression. The default is the value to which the `words` command or `WORDS` variable is set. See "VARIABLES."
format	Specifies the format of the dump; the current format is the default.

`od -r` Formats and dumps all registers. You can use this option for dumps or for examining a running system in the debugger.

`p` Displays the program counter. You can use this option for dumps or on a running system in the debugger.

`processor` [*expression*]
Sets the IOP number to *expression*; *expression* can be set to an integer from `0` through `4` (`4` is the MUXIOP). The default is `0`. This is equivalent to setting the `IOP` variable as follows: `IOP=`*expression*

`quit` [*expression]* Quits the `ecrash` program. *expression* specifies the exit code; the default for *expression* is `0`.

`register` [*expression*]
Displays registers memory starting with *expression.* The number of registers displayed is controlled by the `WORDS` variable and the `words` command. *expression* can be set to an integer value from `00` through `0177` (octal). The default is `0`. You can use this option for dumps or on a running system in the debugger.

`set` [*address1*]

`set` [*address1*,*address2*]
Sets a single breakpoint at *address1* or a double breakpoint at *address1* and *address2*; *address1* and *address2* can be set to an integer from `0` through `65536`. On a single breakpoint, the IOS-E resets the breakpoint. On a double breakpoint, the breakpoint stays set. This command is valid only when you are in the debugger.

`ssd` Displays SSD memory. See also `cb`, `cpu`, and `local`.

`status` Shows the current status of `ecrash`. This command is mode-dependent. In off-line mode, it shows the following: the contents of the dumpfile being examined; the date and time of the dump; the serial number of the machine being dumped (if specified in the dump); and the reason, if any, for the dump. In on-line mode, it shows the following: which IOPs can be accessed; the IOP with which you are currently communicating; the memory target; the state of the debuggers, in which a minus sign (`-`) indicates that the debugger is loaded in that particular IOP, a plus sign (+) indicates that the debugger is loaded and entered, and no symbol indicates that the debugger is not loaded; and any breakpoints set.

`trace` [*expression*] Prints the last *expression* entries from the trace table of the current IOP. If you do not specify *expression*, the whole table will be printed. The table is printed in reverse chronological order, with the newest entries first.

`unset` [*address1*]

`unset` [`all`] Unsets the breakpoint for *address1* or all addresses. You must enter either *address1* or `all`; there is no default. This command is valid only when you are in the debugger on the IOS-E.

`words` [*expression*] Sets the number of words to display when using the memory display commands (*address*, +, -, +*n*, -*n*, `.`+*n*, >, <), and the number of registers to display when using the `register` command. If you do not specify *expression*, the default is `1`. This is equivalent to setting the `WORDS` variable as follows: `WORDS=`*expression.*

`x` Continues program execution from debugger.

`xp` *address*	Displays 16 words of memory as a CPU exchange package. You cannot use this command while you are looking at local memory; if you try to do so, you will get an error message.
# *comment*	Specifies a comment. Any text following a # is ignored up to the next new-line character. This allows comments to be added to script files.

VARIABLES

Variables in `ecrash` script files or in standard input lines must be in the following format:

name=expression

A variable name must begin with an alphabetic letter or underscore and must contain a unique sequence of letters, digits, or underscores; the name can consist of 19 or fewer characters. You cannot use a variable name that is the same as an `ecrash` built-in command name, or the same as a command name listed in the "FLOW CONTROL" or "I/O REDIRECTION" section. `ecrash` is case-sensitive; therefore both `lower` and `LOWER`, for example, are unique.

The value associated with a variable must follow an equal sign (=) and must consist of an octal (preceded by `0`), decimal, or hexadecimal (preceded by `0x`) integer.

A variable is declared when you assign an initial value to it, as in the following example:

```
c0-i0-> LOWER=0100
```

If you refer to a variable that has not been declared, `ecrash` will issue an error.

The following variables are predeclared; you can change the definitions of these variables by redeclaring them:

Variable	**Definition**
`CLUSTER`	Specifies the cluster to be examined with `ecrash`. The initial value is `0`.
`IOP`	Specifies the IOP to be examined `ecrash`. The initial value is `0`.
`WORDS`	Specifies the number of words of memory displayed by the memory formatting commands. The initial value is `8`.
`BASE`	Specifies the base value used when referencing registers. The initial value is `0`.

The following predeclared variables cannot be modified:

Variable	**Definition**
`OFFLINE`	Logically true if `ecrash` is running in off-line mode (that is, accessing a dump file).
`ONLINE`	Logically true if `ecrash` is running in on-line mode (that is, accessing a live system).

EXPRESSIONS AND REGISTERS

The `ecrash` program supports the following operators; *x* and *y* may be variables, numbers, registers, or other expressions, listed in order of precedence:

Operator	**Description**
(...)	Groups expressions
-*x*	Specifies a negative (twos complement) value of *x*
~*x*	Specifies the ones complement of *x*
@*x*	Specifies the value at address *x* in the currently selected memory target

`!`*x*	Specifies the logical inverse of *x*
x`*`*y*	Specifies *x* multiplied by *y*
x`/`*y*	Specifies *x* divided by *y* (integer division)
x`%`*y*	Specifies *x* modulus *y* (remainder)
x`+`*y*	Specifies *x* plus *y*
x`-`*y*	Specifies *x* minus *y*
x`<<`*y*	Specifies *x* left shifted *y* bits (zero filled)
x`>>`*y*	Specifies *x* right shifted *y* bits (zero filled)
x`<`*y*	Specifies a value of true if *x* is less than *y*
x`<=`*y*	Specifies a value of true if *x* is less than or equal to *y*
x`>`*y*	Specifies a value of true if *x* is greater than *y*
x`>=`*y*	Specifies a value of true if *x* is greater than or equal to *y*
x`==`*y*	Specifies a value of true if *x* is equal to *y*
x`!=`*y*	Specifies a value of true is *x* is not equal to *y*
x`&`*y*	Specifies bitwise `and` of *x* and *y*
x`^`*y*	Specifies bitwise `xor` of *x* and *y*
x|*y*	Specifies bitwise `or` of *x* and *y*
x`&&`*y*	Specifies logical `and` of *x* and *y*
x||*y*	Specifies logical `or` of *x* and *y*
.	Specifies the current address in the currently selected memory target

The `ecrash` program lets you use the value of registers inside expressions. They are referenced as follows:

Register	**Description**
`'a'`	Specifies the A register. You must enter the single quotation marks.
`'b'`	Specifies the B register. You must enter the single quotation marks.
`'c'`	Specifies the C register. You must enter the single quotation marks.
`'e'`	Specifies the E pointer. You must enter the single quotation marks.
`'intr'`	Specifies the interrupt flag. You must enter the single quotation marks.
`'base'`	Specifies the base register. You must enter the single quotation marks.
`'e[`*n*`]'`	Specifies exit stack entry *n* (*n* may be an expression). Note: you must literally enter the single quotation marks and brackets as well as value for *n*; the brackets in this case do not indicate optional information.
`'f[`*n*`]'`	Specifies the flags for channel *n* (*n* may be an expression). Note: you must literally enter single quotation marks and the brackets as well as value for *n*; the brackets in this case do not indicate optional information.
`'r[`*n*`]'`	Specifies register *n*, after applying `BASE` (*n* may be an expression). Note: you must literally enter single quotation marks and the brackets as well as value for *n*; the brackets in this case do not indicate optional information.
`'p'`	Specifies the instruction pointer. You must enter the single quotation marks.

FLOW CONTROL

The `ecrash` program supports constructions that alter the flow of execution, depending upon evaluation of conditions. The supported constructions are as follows:

`if (`*expression*`)` *statement*

If *expression* is true, (that is, it evaluates as nonzero) *statement* will be executed; if *expression* is false (that is, it evaluates to zero), continue.

`if (`*expression*`)` *statement1* `else` *statement2*

If *expression* is true (that is, it evaluates as nonzero), *statement1* will be executed; if *expression* is false (that is, it evaluates to zero), *statement2* will be executed. Because to the nature of the parser, `else` must be on the same line as the end of *statement1*; for example, if you enclose the statement block in braces, `else` must be on the same line as the closing brace for that statement.

`while (`*expression*`)` *statement*

As long as *expression* is true (that is, as long as it evaluates as nonzero), *statement* is executed. The `break` and `continue` commands allow program flow to be modified from within the loop (`break` exits it, `continue` jumps to the evaluation).

`for (`*expression1* `;` *expression2* `;` *expression3*`)` *statement*

```
expression1
while (expression2) {
        statement
        expression3
}
```

These constructions are equivalent. Any or all of the expressions may be absent, in which case they evaluate to true. In both constructions, *expression1* is evaluated once; then, as long as *expression2* is true, *statement* is executed and *expression3* is evaluated.

Example:

```
for (i=0; i<3; i=1+i) status
```

I/O REDIRECTION

The `ecrash` command allows you to include other source files as input and to redirect the output to files, programs, or pipes, as follows. *statement* can be a collection of one or more `ecrash` commands, and the quotation marks are required:

`include "`*file*`"` Reads input from the specified file until the end-of-file character; you must enclose *file* in quotation marks. If the file cannot be opened, `ecrash` will issue an error message. You can have up to 10 nested `include` statements.

statement > "*file*"	Redirects the output of *statement* to the specified file, restoring the output to standard output after execution of the statement has completed; you must enclose *file* in quotation marks. If *file* does not already exist, it will be created. If *file* already exists, its contents will be overwritten with the output from *statement* unless the `noclobber` environment variable is set; if `noclobber` is set, the existing file will not be overwritten and you will get a warning message.
statement >> "*file*"	Appends the output of *statement* to the specified file, restoring the output to standard output after execution of the statement has completed; you must enclose *file* in quotation marks. If *file* does not already exist, it will be created. If *file* already exists, the output is added to the end of the file; the previous contents will not be overwritten.
statement | "*unixcommand*"	Pipes the output of *statement* to the specified SunOS command (which may itself be a pipeline), restoring the output to standard output after execution of the statement has completed; you must enclose *unixcommand* in quotation marks. The command is executed in a subshell, so normal shell expansion can take place.
> "*file*"	Redirects the output of `ecrash` to the specified file until you end the `ecrash` session or enter > by itself; you must enclose *file* in quotation marks. If *file* does not already exist, it will be created. If *file* already exists, its contents will be overwritten with the output from *statement* unless the `noclobber` environment variable is set; if `noclobber` is set, the existing file will not be overwritten and you will get a warning message.
>> "*file*"	Appends the output of `ecrash` to the specified file until you end the `ecrash` session or enter > by itself; you must enclose *file* in quotation marks. If *file* does not already exist, it will be created. If *file* already exists, the output will be added to the end of the file; the previous contents will not be overwritten.
| "*unixcommand*"	Permanently pipes the output of `ecrash` to the specified OWS-E or Sun OS command, which may itself be a pipeline; you must enclose *unixcommand* in quotation marks. The command is executed in a subshell, so normal shell expansion can take place. To send output back to standard input, see > (below).
>	Restores the output of `ecrash` to standard output.

NOTES

The dump file format has changed. You can use the `conv`(8) program to convert old dumps to the new format.

Built-in commands may not be abbreviated.

EXAMPLES

The following examples show the declaration of a variable, an `ecrash` session using a dump file, an interactive session on a running system, the redirection of output, a search for strings, paging through a dump image, and displaying status in on-line mode.

Example 1: Using Variables

This example shows how you can use variables to specify ranges in an on-line `ecrash` session; what you type in is shown in bold:

```
c0-i0-> LOWER=0100
c0-i0-> UPPER=0200
c0-i0->       for (i = LOWER; i < UPPER; i = i + 1) {
       if (@i == 014) {
              od @(i+1),1,words
       }
}
c0-i0->
```

Example 2: Examining a Dump File

The following shows an example examination of a dump file. Comments are shown in roman font and preceded by a number sign (#).

```
machine$ ecrash -f newdump
c0-i0-> status
ecrash status:
       Processor: cluster 0, iop 0
       Memory   : local

Dump information :
       Mainframe: 1601
       Date     : 02/16/1992     Time: 13:02
       Reason   : IOP dumped by user
       Filename : dmp.02161302

Cluster 0, iop 0 :
       -registers
       -local memory (0000000 - 0177777)
Cluster 0, iop 1 :
       -registers
       -local memory (0000000 - 0177777)
Cluster 0, iop 2 :
       -registers
       -local memory (0000000 - 0177777)
Cluster 0, iop 3 :
       -registers
       -local memory (0000000 - 0177777)
Cluster 0, iop 4 :
       -registers
       -local memory (0000000 - 0177777)
Cluster 1, iop 0 :
       -registers
       -local memory (0000000 - 0177777)
Cluster 1, iop 1 :
       -registers
       -local memory (0000000 - 0177777)
Cluster 1, iop 2 :
       -registers
       -local memory (0000000 - 0177777)
Cluster 1, iop 3 :
       -registers
       -local memory (0000000 - 0177777)
Cluster 1, iop 4 :
       -registers
       -local memory (0000000 - 0177777)
```

#Print the last five trace entries in each IOP:

```
    c0-i0-> for (IOP = 0; IOP <= 4; IOP = IOP+1) {
               calc IOP
               trace 5
          }
    Oct - 0 Hex - 0 Signed dec - 0 Unsigned dec - 0
               rtc  000000 054257
              0125  000000
              0124  177000 000000
              0123  177000 000000 000002 000000
              0122  000002
    Oct - 1 Hex - 1 Signed dec - 1 Unsigned dec - 1
               rtc  000000 000002
              0173
              0172
              0002  000011
              0002  000010
    Oct - 2 Hex - 2 Signed dec - 2 Unsigned dec - 2
               rtc  000000 056327
              0411  056326 074172 120110 010015 000000
              0410  056326 066304 120110 140000 000000 000000
              0411  056326 066077 120110 140000 000000
              0407  056326 064564 120110 120110 000002 000000 000000
    Oct - 3 Hex - 3 Signed dec - 3 Unsigned dec - 3
               rtc  000000 000002
              0177  052771
              0176  037005
              0002  000011
              0002  000010
    Oct - 4 Hex - 4 Signed dec - 4 Unsigned dec - 4
               rtc  000000 066522
              0005  000010 000000 000002 177777
              0003  000030 000017 000130 000004 000004
               rtc  000000 063515
              0003  000034 000600 000144 000053 000015
```

# Examine low memory of IOP 0:

```
c0-i5-> processor 0
c0-i0-> format parcel
c0-i0-> local
c0-i0-> words 5
c0-i0-> 0
000000 010000 024000 075000 006777 062145 061165 063400 000000 ..(.z...debug...
000010 000000 000000 000000 000000 000000 140561 000000 000000 ...........q....
000020 064141 066164 000000 000000                             halt....
c0-i0-> <new-line>
000020                             072151 061553 000001 163322         tick....
000030 072162 060543 062400 000000 175764 002000 000000 000000 trace...........
000040 071565 061000 021762 000000 067157 026543 070165 000000 sub.#...no-cpu..
```

# Display the registers:

```
c0-i0-> register
Registers (BASE = 0000) :
0000    000000  004002  004062  000000  000000
```

# Use the built-in calculator to evaluate an octal value, shifted right by four places:

```
c0-i0-> calc 031056 >> 4
Oct - 1442 Hex - 322 Signed dec - 802 Unsigned dec - 802
```

# Locate the clock eyecatcher in low memory, using the SunOS `grep`(1) command:

```
c0-i0-> od 0,200,word | "grep tick"
000000020 0641413307200000000000 0721513066540000363322 halt....tick....
c0-i0-> od 0,200 | "grep tick"
0x000010 6861 6c74 0000 0000 7469 636b 0001 e6d2  halt....tick....
```

# Evaluate the result of using AND to compare the contents of local memory location 0x10 with 0xff:

```
c0-i0-> calc @0x10 & 0xff
Oct - 141 Hex - 61 Signed dec - 97 Unsigned dec - 97
```

# Exit with an exit code equal to the contents of local memory location 0x10 (only the lower 8 bits are significant):

```
c0-i0-> quit @0x10
```

# Show that the exit code was indeed 97 (decimal):

```
machine$ echo $?
97
machine$
```

Example 3: Examining a Running System

The commands used in the first example are also available when `ecrash` is used to examine a running system. However, many of the following commands are valid only when you are in the debugger (as previously noted).

```
machine% ecrash

c0-i0-> debug on
c0-i0-> a
A register     163116
c0-i0-> e
E pointer      000001
c0-i0-> e[2]
Exit Stack :
000                    113663  105360  121443  107541  010162  113663
010     000000  000000
c0-i0-> flags
Channel flags:
ch00 = NOT busy - NOT done      ch01 = NOT busy - NOT done
ch02 = NOT busy -     done      ch03 = NOT busy - NOT done
ch04 =     busy -     done      ch05 = NOT busy - NOT done
ch06 = NOT busy - NOT done      ch07 = NOT busy - NOT done
ch10 = NOT busy - NOT done      ch11 = NOT busy - NOT done
ch12 = NOT busy - NOT done      ch13 = NOT busy - NOT done
ch14 = NOT busy - NOT done      ch15 = NOT busy - NOT done
ch16 = NOT busy - NOT done      ch17 = NOT busy - NOT done
ch20 = NOT busy - NOT done      ch21 = NOT busy - NOT done
ch22 = NOT busy -     done      ch23 = NOT busy - NOT done
ch24 = NOT busy - NOT done      ch25 = NOT busy - NOT done
ch26 = NOT busy - NOT done      ch27 = NOT busy - NOT done
ch30 = NOT busy - NOT done      ch31 = NOT busy - NOT done
ch32 = NOT busy - NOT done      ch33 = NOT busy - NOT done
ch34 = NOT busy - NOT done      ch35 = NOT busy - NOT done
ch36 = NOT busy - NOT done      ch37 = NOT busy - NOT done
c0-i0-> issue 30
A register returned = 0, busy = 1, done = 0
c0-i0-> set 0
c0-i0-> set 100
c0-i0-> x
c0-i0-> unset 100
c0-i0-> unset 0
c0-i0-> quit
machine%
```

Example 4: Redirecting Output

As discussed earlier, you can redirect `ecrash` command output, as you do on a UNIX command line. For example, the following will store the first 200 parcels of the dump image in a file named `image`:

```
od 0,200 > "image"
```

Example 5: Searching for Strings

You can search for strings with `grep`(1). For example, the following command line searches for the ASCII string `halt`:

```
od 0,0177777 | "grep halt"
```

Example 6: Paging through a Dump Image

You can also use the SunOS `more`(1) command. For example, the following lets you page through the dump image by piping the `od` output to `more`:

```
od 0,0177777 | "more"
```

Example 7: Displaying Status in On-line Mode

The following example shows the status command in on-line mode. What the user enters is shown in bold:

```
ows1600% ecrash
c0-i0-> status
ecrash status:
        Processor: cluster 0, iop 0
        Memory   : local

        Available IOPs:
                Cluster  0: iops 0  1- 2+ 3  4-
                Cluster  1: iops 0  1  2  3  4

                Key:  - Debugger loaded
                      + Debugger loaded & entered

      Breakpoints: NONE
c0-i0->
```

This example shows that the debugger is not loaded for IOPs 0 and 3 in cluster 0 nor for any of the IOPs in cluster 1, that it is loaded for IOPs 1 and 4 of cluster 0, and that it is loaded and entered for IOP 2 of cluster 0.

FILES

`/etc/owsepermfile`	Permissions file that contains a list of accounts and the commands they are allowed to access
`$HOME/.ecrashrc`	`ecrash` resource file containing `ecrash` commands to be executed upon invocation of `ecrash`

SEE ALSO

`owsepermfile`(5) for information about the default OWS-E permission file
`conv`(8) for information about converting files from the old `edump` file format to the new file format
`edump`(8) for information about dumping the IOS-E to the OWS-E
`mfdump`(8) for information about generating a CPU dump
I/O Subsystem Model E (IOS-E) Guide, publication SD–2107, for additional information about and examples of `ecrash`. (This document contains information private to Cray Research, Inc. It can be distributed to non-CRI personnel only with approval of the appropriate Cray Research manager.)

NAME

`ediag` – Boots deadstart diagnostic tests (`dsdiag` or `clear io`) into one or more specified IOPs

SYNOPSIS

`/home/`*localhost*`/cri/smarte/bin/ediag` [-d] [-D] [-F] [-h] [-M] [-v] *cluster*:*iop*:*file* [*cluster*:*iop*:*file*] ...

`/home/`*localhost*`/cri/smarte/bin/ediag` [-d] [-D] [-F] [-h] [-M] [-v] `-c` *cluster* `-i` *iop* `-f` *deadstart*

DESCRIPTION

The `ediag` command boots deadstart diagnostic tests into a specified IOP. It updates the heartbeat table to show that the IOP is no longer running IOP system code. If the diagnostic test fails, error information will be reported. `ediag` is usually run by the `eboot`(8) command. Normally, `ediag` asks for confirmation before taking any action. You can force `ediag` **not** to ask for confirmation by specifying the `-F` option.

The `ediag` command will notify SMARTE, which is running on the MWS-E, if a deadstart diagnostic test fails.

You must specify either *cluster*:*iop*:*file* or the `-c` and `-i` options. The arguments to `ediag` are as follows:

cluster:*iop*:*file* [*cluster*:*iop*:*file*] ...
: Specifies the cluster and IOP that should have the specified diagnostic binary file booted in; at least one set of *cluster*:*iop*:*file* must be specified. You may specify more than one set of *cluster*:*iop*:*file* if you separate the sets with white space.

  The range of valid values for *cluster* consists of integers from `0` through `15`, depending upon the hardware configuration at your site. *cluster* may be a single integer, a list of integers separated by commas, or a range of integers separated by a hyphen.

  The range of valid values for *iop* are integers in the range `0` through `4`, depending upon the hardware configuration at your site. *iop* may be a single integer, a list of integers separated by commas, or a range of integers separated by a hyphen.

  The specified *file* may begin with a tilde character (~), which is expanded to the home directory of the login name that follows the tilde or, if not followed by a login name, to your home directory.

  For examples of legal syntax, see the "EXAMPLES" section.

`-d`
: Sets debug mode in the diagnostic program.

`-D`
: Sets debug mode in the `ediag` program and sends the output to standard error.

`-F`
: Forces `ediag` not to ask for user confirmation before booting the deadstart diagnostic test and suppresses most informative messages. (Error messages will still appear.)

`-h`
: Sets headerless load mode. Use this option when the file you specify does not have a header. If you do not specify this option, the `eboot` program strips the 64-byte `segldr`(1) header from the binary file; if there is no header, information will be lost. (By default, IOP binaries include a 64-byte `a.out` header.)

`-M`
: Issues the cluster master clear function. If booting the MUXIOP, a cluster master clear is issued to all IOPs in the cluster. **CAUTION**: this stops any other IOPs running in that cluster; therefore, you should specify the MUXIOP before other clusters.

`-v`
: Sets verbose mode, which forces `ediag` to print informative messages to standard error.

`-c` *cluster*	Specifies the cluster in which to run the diagnostic test. The range of valid cluster numbers depends on the number of clusters in the IOS-E. If you specify this option, you must also specify `-i` and `-f`.
`-i` *iop*	Specifies the number of the IOP into which the diagnostic test is to be booted. If you specify this option, you must also specify `-c` and `-f`. *iop* can be an integer in the range `0` through `4` (`4` indicates the MUXIOP; `0` through `3` indicate EIOPs).
`-f` *deadstart*	Specifies the file name of the deadstart diagnostic test (specify the complete path name if the file is not in the current directory). If you specify this option, you must also specify `-c` and `-i`. *deadstart* is defined by `IOSDPATH` in `/etc/configfile` and can be one of the following:

`/home/`*localhost*`/cri/os/ios/dsdiag`	(the default)
`/home/`*localhost*`/cri/os/ios/cleario`	

CONFIGURATION FILE PARAMETERS

The `ediag` command reads the following parameters from `/etc/configfile`:

`IOSCPATH`	Defines the path name of the I/O clear diagnostic test. Default: `/home/`*localhost*`/cri/os/ios/cleario`
`IOSDPATH`	Defines the path name of the IOP deadstart diagnostic test. Default: `/home/`*localhost*`/cri/os/ios/dsdiag`

EXAMPLES

All of the following example formats are legal:

```
ediag 0:1:/home/owse/cri/os/ios/dsdiag
ediag 0:1:~cri/os/ios/boot.1 0:2:~cri/os/ios/dsdiag.2
ediag 0,5:0:/home/owse/cri/os/ios/dsdiag
ediag 1-15:0,4:~cri/os/ios/dsdiag
```

If you wanted to use cluster 1, IOP 1 in debug mode, with the `cleario` deadstart diagnostic file, and in verbose mode, you could enter the following:

```
ediag -c 1 -i 1 -d -f 'getconfig "IOSCPATH"' -v
```

To boot a binary file named `dsdiag` into IOPs 1 and 2 of cluster 0, enter the following:

```
ediag 0:1,2:dsdiag
```

FILES

`/etc/configfile`	Default OWS-E configuration file
`/home/`*localhost*`/cri/os/ios/cleario`	Default `cleario` deadstart diagnostic file
`/home/`*localhost*`/cri/os/ios/dsdiag`	Default `dsdiag` deadstart diagnostic file

SEE ALSO

`eboot`(8) for information about booting the IOS-E
OWS-E Operator Workstation Administrator's Guide, publication SG–3079, for information about `/etc/configfile`
UNICOS 6.E Early Release Software On-line Diagnostic Technical Note, publication SPN-1022. (This technical note is Cray Research Proprietary; dissemination of this information to non-CRI personnel requires approval from the appropriate vice president and a nondisclosure agreement. Export of technical information in this category may require a Letter of Assurance.)

NAME

`edump` – Dumps IOS-E IOP local memory images to the OWS-E

SYNOPSIS

`/home/`*localhost*`/cri/bin/edump [-a] [-D] [-e] [-f` *pathname*`] [-F] [-p` *path*`] [-q] [-v]`
cluster`:`*iop*`[:`*reason*`] [`*cluster*`:`*iop*`[:`*reason*`] ] ...`

`/home/`*localhost*`/cri/bin/edump [-a] [-D] [-e] [-f` *pathname*`] [-F] [-p` *path*`] [-q] [-v]`
`-c` *cluster* `-i` *iop* `[`*reason*`]`

DESCRIPTION

The `edump` command dumps multiple IOPs of multiple clusters to the OWS-E. `edump` creates a dump image that can be processed with other utilities, such as `ecrash`(8). After `edump` is used, the IOPs that have been dumped must be restarted with the `eboot`(8) command or one of the boot scripts. Processors will usually be dumped with `edump` when they are hung.

> **CAUTION:** If you execute `edump` on a running system, the system will crash. Normally, `edump` asks for confirmation before taking any action. You can force `edump` **not** to ask for confirmation by specifying the `-F` option.

Permission to access this command is set in `/etc/owsepermfile` by the system administrator.

You must specify either *cluster*`:`*iop*`[:`*reason*`]` or the `-c` and `-i` options. The arguments to the `edump` command are as follows:

cluster`:`*iop*`[:`*reason*`] [`*cluster*`:`*iop*`[:`*reason*`]] ...`
Specifies the cluster and IOP that should be dumped; at least one set of *cluster* and *iop* must be specified, and they must be separated by a colon. You may specify more than one set of *cluster*`:`*iop*`[:`*reason*`]` if you separate them with white space.

The range of valid values for *cluster* consists of integers from `0` through `15`, depending upon the hardware configuration at you site. *cluster* may be a single integer, a list of integers separated by commas, or a range of integers separated by a hyphen.

The range of valid values for *iop* are integers in the range `0` through `4`, depending upon the hardware configuration at your site. *iop* may be a single integer, a list of integers separated by commas, or a range of integers separated by a hyphen.

If *reason* is specified, it must be preceded by a colon; quotation marks are required if the *reason* contains white space. *reason* is truncated after 79 characters, not including quotation marks (a longer *reason* will not cause an error, but the 80th and succeeding characters will not be used). When you specify more than one *reason*, only the first one encountered is used. If you do not specify *reason*, the string "`IOP dumped by user`" will be used.

The following formats are all legal:

```
edump 0:1
edump 0,5:0:"system is hung"
edump 0:1 1-15:0,4:"hung system"
```

`-a`
Specifies that all memory types of the selected IOP are to be dumped; that is, local memory, register memory, and all four channel buffers. (There are no channel buffers on the MUXIOPs.) This will result in large dump files because the channel buffers for each IOP are 2 Mbytes. The default is to dump 64 Kparcels of local memory and all registers.

`-D`	Sets the debug mode in the `edump` program and sends the output to standard error.
`-e`	Echoes the file name of the resulting dump file to standard output. This option is useful in work with shell scripts; it is inadvisable to use the `-v` and `-q` options with `-e`.
`-f` *pathname*	Specifies the full path name of the `IOPSAVE` program that gets the internal registers and channel buffers. The default path name is `/home/`*localhost*`/cri/os/ios/iopsave`.
`-F`	Forces `edump` not to ask for confirmation before halting the IOP and suppresses most informative messages. (Error messages will still appear.)
`-p` *path*	Specifies the path name of the directory in which the dump image should reside. The dump image has the name `dmp.`*mmddhhmm*, in which *mmddhhmm* specifies the month, date, hour, and minute; for example, if a dump image were created at 4:43 P.M. on February 2, the file containing it would be named `dmp.02021643`. If you do not enter the `-p` option, the default directory will be specified by the `DEFAULTIDUMPDIR` parameter in `/etc/configfile`; at release, this parameter is set to `/var/dumps`. If two or more dumps are taken within the same minute, the letters `a`, `b`, and so on, are added to the file names to make them unique. See EXAMPLES.
`-q`	(query) Asks you for memory types to dump. The default values are for all of local memory and all internal registers. The `-q` option allows users to get a smaller range of memory and/or channel buffer memory. The default is to dump 64 Kparcels of local memory and all registers.
`-v`	Sets verbose mode. This option forces `edump` to print informative messages to standard output.
`-c` *cluster*	Specifies the cluster that should be dumped. The range of valid cluster numbers is an integer from `0` through `15`, depending upon the number of clusters in your IOS-E. If you specify this option, you must also specify `-i`.
`-i` *iop*	Specifies the number of the IOP to be dumped. *iop* can be one or more integers in the range `0` through `4` (separated by commas), the word `mux`, or the word `all`. `all` dumps the complete cluster; `mux` is the same as `4`. If you specify this option, you must also specify `-c`.
reason	Specifies the reason for the dump; for example, `"Halt 24"`. (Although the quotation marks are not always required, you should use them to avoid problems with special characters.) The reason given may be up to 79 characters long. Although it is not required, adding this explanation in the command line is especially useful if you intend to use scripts for autoboots or autodumps. Entering a reason also helps to distinguish dumps as dump files start to accumulate. If you do not specify a reason, the string `"IOP dumped by user"` will be used by default.

CAUTION

If you execute `edump` on a running system, the system will crash.

NOTE

The format for the name of a dump image file was `dmp.`*mmdd*`.`*hhmm*, but with OWS-E 2.0 it is now `dmp.`*mmddhhmm* to avoid truncation problems on some systems. Also, the default for `-p` is now `/var/dumps`.

EXAMPLES

The following examples show dumping multiple IOPs and specifying the dump-image directory.

Example 1: Dumping Multiple IOPs

The following example dumps IOP 1 and IOP 3 from IOS 0, cluster 2, and prints informative messages to standard error:

```
edump -v 2:1,3:"system panic"
```

Example 2: Dumping a Range of IOPs and Clusters

The following example dumps IOP 1, IOP 2, and IOP 3 from IOS 0, clusters 2 through 6, and prints informative messages to standard error:

```
edump -v 2-6:1-3:"system panic"
```

Example 3: Dumping a Group of IOPs and Clusters

The following example dumps the following IOPs from IOS 0: IOP 1, IOP 2, and IOP 3 from cluster 2; IOP 0 from cluster 3; and all of the IOPs from clusters 4 and 5. The reason used is "`system panic`" because it is the first reason encountered (the second value for *reason* is ignored):

```
edump 2:1-3:"system panic" 3:0 4,5:0-4:"halt code 32"
```

Example 4: Specifying the Directory

The following example specifies that the dump image should be placed in `/var/temp/dumpdir` and creates the file `/var/temp/dumpdir/dmp.08160623`:

```
edump -c 0 -i 0 -p /var/temp/dumpdir  "Halt 24"
```

If you took another dump in less than 1 minute, the second file would be named `/var/temp/dumpdir/dmp.08160623.a`.

Example 5: edump in a Shell Script

The following example shows a script that dumps all of the IOPs from clusters 0 and 1 in IOS 0, using the reason of "`system panic`" and without asking for confirmation. If the script does not execute properly, you are given the message that the dump failed; if it executes properly, you are given the name of the dump file:

```
#!/bin/sh
#Dump all IOPs in clusters 0 and 1
FNAME='edump -e -F 0,1:0-4:"system panic"'
if [ $? != 0 ]
then
      echo "Dump of IOS failed"
      exit
fi
echo "Dump file name is $FNAME"
```

RETURN VALUES

The `edump` command returns a value of `0` when it completes successfully.

CONFIGURATION FILE PARAMETERS

The `edump` command reads the following parameters from `/etc/configfile`:

`DEFAULTIDUMPDIR`	Defines the default dump directory path in which the dump shell script is created. Default: `/var/dumps`
`IOPLOG`	Defines the path name of the IOP log file. Default: `/var/logs/ioplog`
`IOPSAVE`	Defines the path name of a temporary file used by the `edump`(8) utility during its processing. Default: `/home/__HOSTNAME__/cri/os/ios/iopsave`
`SERIALNUMBER`	Defines the serial number of the CRI mainframe to which the OWS-E is attached. Default: `__SERIALNUMBER__` token (replaced during installation)

ENVIRONMENT VARIABLES

`OWSECONFIG`	Specifies the system configuration file; by default, it is set to `/etc/configfile`

FILES

`/etc/configfile`	Default OWS-E configuration file
`/etc/owsepermfile`	Permissions file that contains a list of accounts and what commands they are allowed to access
`/home/`*localhost*`/cri/os/ios/iopsave`	Default path name of the `IOPSAVE` program
`/var/dumps`	Default dump directory

SEE ALSO

`configfile`(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about `/etc/configfile`
`owsepermfile`(5) for information about the default OWS-E permission file
`conv`(8) for information on how to convert files from old `edump` file format into new `edump` file format
`eboot`(8) for information on how to reboot the IOS-E from the OWS-E after a dump is taken
`ecrash`(8) for information about dump processing
`dumpsys`(8) or `mfdump`(8) for information about generating a mainframe dump

NAME

`ehalt` – Halts one or more IOPs from the OWS-E

SYNOPSIS

`/home/`*localhost*`/cri/bin/ehalt [-C] [-F] [-h` *code*`] [-M] [-v]` *cluster*`:`*iop*`[:`*reason*`]` `[`*cluster*`:`*iop*`[:`*reason*`]] ...`

`home/`*localhost*`/cri/bin/ehalt [-C] [-F] [-h` *code*`] [-M] [-v] -c` *cluster* `-i` *iop*

DESCRIPTION

The `ehalt` command stops an IOP from the OWS-E. `ehalt` enters a halt code (operator stop) into IOP local memory and issues a master clear function to stop the processor.

Normally, `ehalt` asks for confirmation before taking any action. You can force `ehalt` **not** to ask for confirmation by specifying the `-F` option.

If you execute the `ehalt` command directly, it will usually be because you want to get a dump image with the `edump`(8) command. When you have the dump image, you will probably want to reboot the IOP with the `eboot`(8) or `bootsys`(8) commands.

Permission to access this command is set in `/etc/owsepermfile` by the system administrator.

You must specify either *cluster*`:`*iop*`[:`*reason*`]` or the `-c` and `-i` options. The arguments to the `ehalt` command are as follows:

cluster`:`*iop*`[:`*reason*`]` `[`*cluster*`:`*iop*`][:`*reason*`] ] ...`

Specifies the cluster and IOP that should be halted; at least one set of *cluster* and *iop* must be specified, and they must be separated by a colon. You may specify more than one set of *cluster*`:`*iop*`[:`*reason*`]` if you separate them with white space. The implementation for *reason* is currently deferred; it will be used as input to a system status table in a future release.

The range of valid values for *cluster* consists of integers from `0` through `15`, depending upon the hardware configuration at you site. *cluster* may be a single integer, a list of integers separated by commas, or a range of integers separated by a hyphen.

The range of valid values for *iop* consists of integers in the range `0` through `4`, depending upon the hardware configuration at your site. *iop* may be a single integer, a list of integers separated by commas, or a range of integers separated by a hyphen.

FUTURE IMPLEMENTATION: If *reason* is specified, it must be preceded by a colon and, if it contains any white space, it must be enclosed in quotation marks. If you do not specify *reason*, the string `"IOP halted by user"` will be used unless the `-C` option is specified, in which case the default reason will be `"IOP halted by CPU"`.

All of the following formats are legal:

```
ehalt 0:1
ehalt 0,5:0:"system is hung"
ehalt 1-15:0,4:"hung system" 0:1
```

`-C`

Specifies that the halt was requested by the mainframe. You will not normally invoke this option manually.

`-F`	Forces `ehalt` not to ask for confirmation before halting the IOP and suppresses most informative messages. (Error messages will still appear.)
`-h` *code*	Specifies the halt code.
`-M`	Master clears and halts the cluster. Use this option only when you are halting the MUXIOP.
`-v`	Sets verbose mode. This option forces `ehalt` to print informative messages to standard error.
`-c` *cluster*	Specifies the cluster in which the IOP resides. The range of valid cluster numbers depends on the number of clusters in the IOS-E. If you specify this option, you must also specify `-i`.
`-i` *iop*	Specifies the number of the IOP to be stopped. *iop* can be an integer in the range `0` through `4` (4 indicates the MUXIOP). If you specify this option, you must also specify `-c`.

BUGS

The use of the *reason* argument is deferred until a future release.

EXAMPLES

The following example halts IOP 3 in cluster 0:

```
ehalt -c 0 -i 3
```

The following example halts every IOP on the system:

```
ehalt 'estat -a'
```

FILES

`/etc/owsepermfile`	Permissions file that contains a list of accounts and the commands they are allowed to access

SEE ALSO

`owsepermfile`(5) for information about the default OWS-E permission file
`eboot`(8) for information about booting the IOS-E
`edump`(8) for information about dumping the IOS-E
`estat`(8) for information about displaying the IOP status

NAME

emon – Restarts the IOS-E error-logging, heartbeat, SMARTE, and CPU monitors

SYNOPSIS

/home/*localhost*/cri/bin/emon

DESCRIPTION

The emon command restarts the error logging routine, errlogd(8), the IOS-E halt and hang monitor, hbeat(8), cpud(8), smdemon(8) SMARTE monitor, and rcpud(8) "remote CPU daemon" CPU monitor. Use this command to restart the various IOS-E monitoring processes on the OWS-E if they are down.

If you enter emon when the processes are already running, no damage will be done and you will not get an error message. You will get an error message only if emon is unable to start one of the monitors.

Permission to access this command is set in /etc/owsepermfile by the system administrator.

ENVIRONMENT VARIABLES

OWSECONFIG	Specifies the system configuration file; by default, it is set to /etc/configfile

FILES

/etc/owsepermfile	Permissions file that contains a list of accounts and the commands they are allowed to access

SEE ALSO

owsepermfile(5) for information about the default OWS-E permission file
errlogd(8) for information on the error-logging routine
hbeat(8) for information on the IOS-E halt and hang monitor
rcpud(8) for information on the CPU monitor
smdemon(8) for information about the SMARTE monitor

NAME

eping – Sends an echo packet to an IOP from the OWS-E

SYNOPSIS

/home/*localhost*/cri/bin/eping [-c *cluster*] [-i *iop*] [-l *length*] [-n *number*] [-p *start*] [-m *increment*] [-t] [-W]

DESCRIPTION

The eping command sends echo packets from the OWS-E to an IOP to verify the service workstation interface (SWI) connection. An 8-bit data pattern is checked for accuracy upon packet return.

The arguments to eping are as follows:

-c *cluster*	Specifies the cluster in which the IOP resides. The range of valid cluster numbers depends on the number of clusters in the IOS-E. The default is 0.
-i *iop*	Specifies the number of the IOP to which packets are to be sent. *iop* can be an integer in the range 0 through 4. The default is 0.
-l *length*	Specifies the packet length in parcels. *length* must be a positive integer: for service echo packets, *length* must be an even integer greater than or equal to 4 (64 is the maximum); for workstation interface (WIN) loopback transfers, the value must be an integer greater than or equal to 1 (64 is the maximum). The default is 4.
-n *number*	Specifies the number of echo packets to be sent. *number* must be a positive integer. The default is 1.
-p *start*	Specifies the initial byte. Each subsequent byte is incremented by the pattern increment specified by the -m option. The default is 1.
-m *increment*	Specifies the number value by which the pattern is incremented; the initial value is specified by the -p option. Each subsequent packet continues from where the previous one left off. The default is 1.
-t	Prints the amount of time (in milliseconds) that each packet required to complete a round trip.
-W	Specifies a loopback from the OWS-E to the WIN.

MESSAGES

If the connection is good, you will receive the following message:

```
cluster x iop y is alive.
```

If the connection is bad, you will receive the following message:

```
Packet n - pattern miscompare at byte x - expected y - received z
```

Permission to access this command is set in /etc/owsepermfile by the system administrator.

CONFIGURATION FILE PARAMETERS

The `eping` command reads the following parameter from `/etc/configfile`:

`IOPLOG`	Defines the path name of the IOP log file. Default: `/var/logs/ioplog`

EXAMPLES

Example 1: In the following example, `eping` will send 20 packets to IOP 3 in cluster 1:

```
ows1600$ eping -c 1 -i 3 -n 20
cluster 1 iop 3 is alive.
ows1600$
```

Example 2: In the following example, `eping` will send 1 packet to IOP 0 in cluster 0. The pattern will start with 3 and be incremented by 2:

```
eping -p 3 -m 2
```

That is, the pattern will be as follows:

```
 -------------------------------
| 3 | 5 | 7 | 9 |11 |13 |15 |17 |      <-- Pattern
|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |      <-- Byte number
 -------------------------------
```

FILES

`/etc/owsepermfile`	Permissions file that contains a list of accounts and the commands they are allowed to access
`/etc/configfile`	Default OWS-E configuration file

SEE ALSO

`configfile`(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about `/etc/configfile`
`owsepermfile`(5) for information about the default OWS-E permission file
`hcon`(8) and `econ`(8) for information about configuring the IOS-E MUXIOP channels
`peek`(8) for information about examining various types of memory in a CRI system

NAME

`errlogd` – IOS-E hardware error-logging daemon

SYNOPSIS

`/home/`*localhost*`/cri/bin/errlogd`

DESCRIPTION

The `errlogd` daemon receives error information from the IOS-E and logs it in the file specified by the ERRLOG parameter in `/etc/configfile`. By default, this file is `/var/logs/errlog`. file.

Every 60 seconds, `errlogd` also reads MUXIOP errors out of IOS-E local memory for high-speed channels 010–017, low-speed input channel 020, and low-speed output channel 021, and reads EIOP errors for channel-buffer channels 020 and 027, and low-speed input channel 022. These errors are recorded in circular buffers in the IOPs. If too many errors occur between reads of the buffer by `errlogd`, some information will be lost. `errlogd` logs a warning that specifies how many error reports were lost.

The `errlogd` command also sends all IOS channel errors to the SMARTE system running on the MWS-E maintenance workstation.

CONFIGURATION FILE PARAMETERS

The `errlogd` command reads the following parameters from `/etc/configfile`:

`ERRLOG`	Defines the path name of the error log file. Default: `/var/logs/errlog`

ENVIRONMENT VARIABLES

`OWSECONFIG`	Specifies the system configuration file; by default, it is set to `/etc/configfile`

FILES

`/etc/configfile`	Default OWS-E configuration
`/var/logs/errlog`	Default hardware error log file as defined by ERRLOG in `/etc/configfile`

SEE ALSO

`configfile`(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about `/etc/configfile`
`owsepermfile`(5) for information about the default OWS-E permission file
`emon`(8) for information about restarting the IOS-E error logging, heartbeat, and CPU monitors

NAME

`estat` – Checks IOP status

SYNOPSIS

`/home/`*localhost*`/cri/bin/estat [-a] [-c` *cluster*`] [-i` *iop*`] [-v]`

DESCRIPTION

The `estat` command checks the status of the IOPs in a given cluster. The returned status indicates whether or not an IOP is running. The state of each IOP is determined from the Shared Memory table used by `hbeat`(8), the IOP heartbeat monitor. The default action of this command checks all possible IOPs in all possible clusters in the Shared Memory table.

The arguments to `estat` are as follows:

`-a`	(abbreviated) Specifies abbreviated mode, which prints the output in the *cluster* : *iop* : *string* format used by various OWS-E commands. For an example, see EXAMPLES below.
`-c` *cluster*	Specifies the IOS-E cluster number. The range of valid cluster numbers depends on the number of clusters in the IOS-E. If no cluster is given, all clusters are checked.
`-i` *iop*	Specifies the number of the IOP to be checked. *iop* can be an integer in the range `0` through `4` (`4` indicates the MUXIOP). If no IOP number is given, all IOPs are checked.
`-v`	Sets verbose mode. This option forces `estat` to print informative messages to standard error.

Permission to access this command is set in `/etc/owsepermfile` by the system administrator.

NOTES

The `estat` command checks for 16 clusters, even if your site has a different configuration. See EXAMPLES for a sample output.

EXAMPLES

The following command line prints the status of all IOPs in all clusters

```
estat
```

If your site has 1 cluster and you execute `estat`, you could get the following output (user input is shown in bold):

```
ows1600% estat
INFO: estat: iop 0, cluster 0 running
INFO: estat: iop 1, cluster 0 running
INFO: estat: iop 2, cluster 0 running
INFO: estat: iop 3, cluster 0 running
INFO: estat: iop 4, cluster 0 running
INFO: estat: no iop in cluster 1 running
INFO: estat: no iop in cluster 2 running
INFO: estat: no iop in cluster 3 running
INFO: estat: no iop in cluster 4 running
INFO: estat: no iop in cluster 5 running
INFO: estat: no iop in cluster 6 running
INFO: estat: no iop in cluster 7 running
INFO: estat: no iop in cluster 8 running
INFO: estat: no iop in cluster 9 running
INFO: estat: no iop in cluster 10 running
INFO: estat: no iop in cluster 11 running
INFO: estat: no iop in cluster 12 running
INFO: estat: no iop in cluster 13 running
INFO: estat: no iop in cluster 14 running
INFO: estat: no iop in cluster 15 running
ows1600%
```

To display this same information in abbreviated mode, use the `-a` option:

```
ows1600% estat -a
0:0-4
```

The following command line prints the status of all IOPs in cluster 0:

```
estat -c0
```

The following command line prints the status of IOP 2 in cluster 6:

```
estat -c6 -i2
```

FILES

`/etc/owsepermfile`	Permissions file that contains a list of accounts and the commands they are allowed to access

SEE ALSO

`owsepermfile`(5) for information about the default OWS-E permission file
`hbeat`(8) and `emon`(8) for information about monitoring the IOS-E system

NAME

`fyadmin` – Controls the `fy` driver

SYNOPSIS

`/etc/fyadmin -b` *type* [`-a` *file* | `-o` *file*] [`-z` *device*]
`/etc/fyadmin -c` [`-a` *file* | `-o` *file*] [`-z` *device*]
`/etc/fyadmin -C` *mode* [`-a` *file* | `-o` *file*] [`-z` *device*]
`/etc/fyadmin -d` *display* [`-a` *file* | `-o` *file*] [`-z` *device*]
`/etc/fyadmin -D` *binary* [`-a` *file* | `-o` *file*] [`-z` *device*]
`/etc/fyadmin -e` *action* [`-a` *file* | `-o` *file*] [`-z` *device*]
`/etc/fyadmin -r` [`-a` *file* | `-o` *file*] [`-z` *device*]
`/etc/fyadmin -t` *type* [`-a` *file* | `-o` *file*] [`-z` *device*]

DESCRIPTION

The `fyadmin` command allows you to reset the hardware boards and software drivers and to download operational firmware to the FEI-4. Most of the options will be used by the analyst only; the administrator will commonly use the following options:

`-r`
`-D`*binary*
`-b i`

The `-b`, `-c`, `-C`, `-d`, `-D`, `-e`, `-r`, and `-t` options are all mutually exclusive, as are the `-a` and `-o` options. Several argument values may be abbreviated to the first few characters; characters that are not required are shown in brackets.

The arguments to `fyadmin` are as follows:

`-b`*type* Sends a mailbox command. *type* can be one of the following:

`d[isable_hbug]`	Disables the ability of the FEI-4 debugger (`hbug`) to communicate with the FEI-4 serial port. `hbug` is produced by Heurikon; see *FEI-4 Cray-VMEbus Front End Interface User's Manual*, Heurikon Corporation, 1991.
`e[nable_hbug]`	Enables the ability of the FEI-4 debugger (`hbug`) to communicate with the FEI-4 serial port.
`i[nvalidate_ram]`	Invalidates the firmware flag in the FEI-4 but does not do anything to the code. If you are not sure what the downloaded code is, use this option to invalidate the present code, issue the `-r` option to reset the hardware and software, and then use the `-D` option to download known code.

`-c` Clears out trace buffers in order to get a clean trace, and resets the pointer to the top of the trace.

`-C` *mode*
: Changes the FEI-4 low-speed (LOSP) channel mode of operation. *mode* can be one of the following:

`6_n[ormal]`	Configures the LOSP channel in 6-Mbyte mode.
`6_i[nloop]`	Configures the LOSP channel in 6-Mbyte mode, but enables internal hardware loopback (FEI-4 has an internal loopback path), using 6-Mbyte rules of operation.
`6_d[irect]`	Configures the LOSP channel so that it allows the `mailbox` commands to read from or write to the channel directly; this disables the FEI-4 channel FIFOs.
`12_d[evice]`	Configures the LOSP channel in 12-Mbyte mode to operate as the device. This is the normal configuration of the FEI-4 when connected to a CRI IOS-E 12-Mbyte channel.
`12_c[hannel]`	Configures the LOSP channel in 12-Mbyte mode to operate as the channel. You will use this mode if, for example, your OWS-E is connected to another Sun Workstation, in which case one must be configured as the channel and the other must be configured as the device.
`12_i[nloop]`	Configures the LOSP channel in 12-Mbyte mode for internal loopback. The input side is configured as the device, and the output side is configured as the channel.
`12_o_i[nloop]`	Configures the LOSP channel in 12-Mbyte mode for internal loopback. The input side is configured as the channel, and the output side is configured as the device.
`12_e[xloop]`	Configures the LOSP channel in 12-Mbyte mode for external loopback and sends a master clear signal so that if FOL-4 boxes are in the configuration, they will be reset to a known state. (Use of an FOL-4 box allows you to extend the LOSP channel to a distance greater than 50 feet.) The input side is configured as the device, and the output side is configured as the channel.
`12_o_e[xloop]`	Configures the LOSP channel in 12-Mbyte mode for external loopback and sends a master clear signal so that if FOL-4 boxes are in the configuration, they will be reset to a known state. (Use of an FOL-4 box allows you to extend the LOSP channel to a distance greater than 50 feet.) The input side is configured as the channel, and the output side is configured as the device.

`-d` *display*
: Displays the specified structure. *display* can be one of the following:

`t[race_control]`	Displays the `fy` driver's execution trace control structure, including the units (`0` through `3`) that are tracing events and the way the trace buffer is being used.
`q[descriptor]`	Displays the state of the unit's read-request, write-request, read-reply, and write-reply queues descriptor.

`fw`	Displays the location at which the FEI-4 processor is executing, either random-access memory (RAM) or read-only memory (ROM), and the state of the firmware. The state is used to indicate what the firmware is doing. For an explanation of the various states, see the `f4_h.h` header file. When the current state is ROM, one of the following is true: the system is waiting for code to be downloaded, the system has just started, an attempt to download code failed, or a reset has been issued. When RAM is displayed, the FEI-4 is running operational code and usually is waiting for work.
`fy`	Displays the `fy` driver control structures (`fy.c`).
`fyc`	Displays the `fy` driver character special interface structures (`fyc.c`).
`fyi`	Displays the `fy` driver Internet Protocol (IP) structures (`if_fyi.c`).
`c[ontrol_reg]`	Displays current settings of the FEI-4's LOSP control register.
`d[efault]`	Displays the FEI-4's default LOSP control register settings that were set when the code was built. After a reset, these are the settings that will be used.
`s[tatus-reg]`	Displays the FEI-4 LOSP status register.

`-D` *binary* Downloads the specified binary file to the FEI-4's RAM and tells the FEI-4 to begin execution. This option is normally used by scripts run at boot time; you will not normally invoke it manually.

`-e` *action* Modifies the execution trace control structure and determines whether the buffer area (allocated at compile time) will be overwritten. The default tracing actions are set in the `fy_trace.h` file at compile time. To change the unit and the mode of writing (`once` or `circular`), you must specify two separate `fyadmin` command lines. *action* can be set to one of the following values.

`0`	Enables tracing on unit 0 only.
`1`	Enables tracing on unit 1 only.
`2`	Enables tracing on unit 2 only.
`3`	Enables tracing on unit 3 only.
`a[ll]`	Enables tracing on units 0 through 3.
`n[one]`	Stops the tracing activity.
`o[nce]`	Fills the area of memory allocated for tracing once and then stops; it does not overwrite itself. Use this value when you know that a certain sequence of events cause a problem but you do not have control over other concurrent processes.
`c[ircular]`	Continuously performs traces and writes them to one area of memory, overwriting itself as needed.

`-r` Resets the `fy` driver and its `fyx` modules; it is similar to the reset performed with a system boot.

`-t` *type* Displays the execution trace buffer or error trace buffer (which contains only the significant events) specified by *type*. The `fy` driver is normally compiled so that all modules use one common execution trace buffer; however, it can also be compiled so that each module (`fy`, `fyi`, and `fyc`) has its own individual trace buffer.

`c[ommon]`	Dumps all information in the common execution trace buffers.
`e[rror]`	Dumps the error trace buffers; only the major events are recorded in the error trace buffer. For example, a major event might be the reconfiguring of a channel or the occurrence of a LOSP channel error.
`fy`	Displays all execution trace buffer information for the `fy` module. (If the system is not configured for individual trace buffers, you will get an error message.)
`fyc`	Displays all execution trace buffer information for the `fyc` module. (If the system is not configured for individual trace buffers, you will get an error message.)
`fyi`	Displays all execution trace buffer information for the `fyi` module. (If the system is not configured for individual trace buffers, you will get an error message.)
`f4`	Displays all FEI-4 execution trace buffer information (FEI-4 only).

`-a`*file* | `-o`*file* Appends (`-a`) or overwrites (`-o`) the output of the `fyadmin` command to *file*; if no such file exists, it will be created. You cannot specify both `-a` and `-o` on the same command line.

`-z` *device* Specifies the control device for the unit. *device* can be one of the following values: `/dev/fyctl0`, `/dev/fyctl1`, `/dev/fyctl2`, or `/dev/fyctl3`. The default is `/dev/fyctl0`.

EXAMPLES

Example 1: Downloading Firmware

The following example shows the command lines to download firmware to an FEI-4 and to verify that the FEI-4 is executing it:

```
fei-test% fyadmin -z /dev/fyctl1 -b i
fei-test% fyadmin -z /dev/fyctl1 -r
fei-test% fyadmin -z /dev/fyctl1 -D /etc/fei4.fw
fei-test% fyadmin -z /dev/fyctl1 -d fw
 FEI-4 is executing in RAM:
        fw state = a
```

Example 2: Displaying Low-speed Channel Configuration

The following example shows the command line that displays the current low-speed (LOSP) channel configuration for the FEI-4:

```
fei-test% fyadmin -z /dev/fyctl1 -d c

CURRENT LOSP Control Register settings
LOSP control reg = ac9c9e40
++++++++++++++++++++++++++++++++++++++++++++
IMODE          = 6N     OMODE         = 6N
IRESET~        = 1      ORESET~       = 1
IFRCBUFOK~     = 1      OCHANOK       = 1
IRDEL          = 2      EOPAR         = 1
LOSPCNTRST~    = 1      LOOPBACK      = 0
CMWDAT         = 0      CLRPERR~      = 1
LED1           = OFF    LED2          = OFF
LED3           = OFF    LED4          = ON
DELRDY         = 0      WDI           = 1
CPUMCLR        = 0      IOMCLR        = 0
DEADMP         = 0      RTCINT        = 0
CONINT         = 0      CDISCEN       = 1
CLRITINT~      = 1      VICRESET      = 0
BERREN~        = 0      ORDYDEL~      = 0
++++++++++++++++++++++++++++++++++++++++++++
```

Example 3: Changing the FEI-4 Low-speed Channel Configuration

The following example shows the command lines that change the FEI-4 low-speed (LOSP) channel configuration and display the new settings:

```
fei-test% fyadmin -z /dev/fyctl1 -C 12_d
fei-test% fyadmin -z /dev/fyctl1 -d c

CURRENT LOSP Control Register settings
LOSP control reg = ae9e9e40
++++++++++++++++++++++++++++++++++++++++++++
IMODE          = 12D    OMODE         = 12D           <------- NEW
SETTINGS
IRESET~        = 1      ORESET~       = 1
IFRCBUFOK~     = 1      OCHANOK       = 1
IRDEL          = 2      EOPAR         = 1
LOSPCNTRST~    = 1      LOOPBACK      = 0
CMWDAT         = 0      CLRPERR~      = 1
LED1           = OFF    LED2          = OFF
LED3           = OFF    LED4          = ON
DELRDY         = 0      WDI           = 1
CPUMCLR        = 0      IOMCLR        = 0
DEADMP         = 0      RTCINT        = 0
CONINT         = 0      CDISCEN       = 1
CLRITINT~      = 1      VICRESET      = 0
BERREN~        = 0      ORDYDEL~      = 0
++++++++++++++++++++++++++++++++++++++++++++
```

FILES

`/usr/include/sundev/f4_h.h`	Header file that explains FEI-4 firmware states that are displayed with `fyadmin -d fw`
`/usr/include/sundev/fy_errs.h`	Header file that explains `fy` driver errors detected
`/usr/include/sundev/fy_trace.h`	Header file that explains the default tracing actions

SEE ALSO

`fyformat`(8) for information about formatting raw trace buffer information into ASCII text

NAME

fyformat – Formats raw trace buffer information extracted from fy driver modules

SYNOPSIS

/etc/fyformat *file*

DESCRIPTION

The fy driver includes a set of common execution trace macros (fy_trace.h) to be used as a debugging aid. The fyadmin(8) program can be used to dump the raw trace information that any fy driver module (such as fyc) collects. The fyformat program takes the raw trace data and formats it into readable ASCII text.

file Specifies the dump file to be formatted by fyformat.

EXAMPLES

The following example shows two command lines that produce files with raw and formatted trace buffer output:

```
fei-test% fyadmin -z /dev/fyctl1 -t all -o rawdata
fei-test% fyformat rawdata > formatted_trace

fei-test% more rawdata
1770
29969a5e 86a01339 0 4
29969a5e 86a01391 ff66fd80 0
many more entries .....
ffffffff ffffffff ffffffff ffffffff
2996ad10 e2b011bc f81788c4 0
2996ad10 e2b011b4 fff3f728 f81788c4
2996ad10 e2b011b6 f8160804 0
2996ad10 e2b011b8 fff15218 15218

fei-test% more formatted_trace
Printing oldest execution trace entry first.
Feb 10 10:49:20 unit1 fy_wr_tib_done: entry, service 1 RIB     f81788c4 0
Feb 10 10:49:20 unit1 fy_service_wr_rib: entry                 fff3f728 f81788c4
Feb 10 10:49:20 unit1 fy_service_wr_rib: got the TIB request   f8160804 0
Feb 10 10:49:20 unit1 fy_service_wr_rib: addresses of data     fff15218 15218
```

SEE ALSO

fyadmin(8) for information about dumping the raw trace information for the fy driver

NAME

`fyroute` – Sets or displays the `fy` driver's IP Interface Routing table

SYNOPSIS

```
/etc/fyroute interface -d
/etc/fyroute interface -D
/etc/fyroute interface -s routingfile
```

DESCRIPTION

The `fyroute` command allows you to set or display the `fy` driver's Internet Protocol (IP) Interface Routing table. This table maps Internet addresses to physical hardware addresses.

The arguments to `fyroute` are as follows:

interface	Specifies the interface whose table will be manipulated. The legal values for *interface* are `fyi0`, `fyi1`, `fyi2`, and `fyi3`.
`-d`	Dumps the routing table in raw format (used for debugging routing code).
`-D`	Dumps the routing table in readable format.
`-s` *routingfile*	Sets the `fy` driver's IP routing table according to the information in *routingfile*. To specify standard input rather than a file, enter a hyphen (`-`).

ROUTING FILE FORMAT

The information in the routing file has the following format (fields marked "`0`" are unused):

verb host destination `0 0` [*mtu*] `;`

Any amount of white space (blanks or tabs) can be used between items, and comments are preceded by * or # in column 1. Each statement must end with a semicolon. The components of statement lines are as follows:

verb Specifies the type of connection. There are three valid values for *verb*:

Value	**Description**
`direct`	Specifies a point-to-point connection between *host* and *destination*.
`swloopback`	Specifies a software loop used for testing local software; *destination* must be the local interface address. (The module that actually loops the data around is the `if_fyi.c` module.)
`hwloopback`	Specifies a hardware loop used for testing hardware loopback paths; *destination* must be the local interface address.

host Specifies the Internet address. The value specified for *host* can be either a host name (such as `feitest-036`) or an Internet address specified in dot notation (such as `128.162.33.5`). You can use dot notation to specify an Internet address that is not listed in `/etc/hosts`.

destination Associates a logical path with an Internet address. The bottom 4 bits of *destination* specifies the logical path to use when sending data to the specified *host*. The value for *destination* must be specified as a hexadecimal number. (`0005` is usually the path for a CRI mainframe.)

mtu Specifies the maximum transfer unit (packet size), in bytes, that the IP layer will receive from the underlying layer; *mtu* must be specified as a decimal number. The default for *mtu* is 4144 bytes, and this is the maximum size. This value results from 4 Kbytes of data plus 48 bytes of header (32 bytes of IOS parameter block header plus 16 bytes of Cray `np.c` header).

Typically, you would first use `swloopback` to test the software connection, then `hwloopback` to test the hardware paths, and finally `direct` to test the connection to the CRI mainframe.

EXAMPLES

Suppose you have the following routing file named `/etc/fycf.owse`:

```
#
#       The first line is the local interface address
#
direct          feitest-036     0001    0    0  4144;
#
direct          sn1600-036      0005    0    0  4144;
swloopback      feitest-swlp    0001    0    0;
hwloopback      feitest-hwlp    0001    0    0;
```

You can modify the driver's current routing table to conform to the specifications in this file and then bring up the interface with the following `fyroute`(8) and SunOS `ifconfig`(8) command lines:

```
ows1600% fyroute fyi0 -s /etc/fycf.owse
ows1600% ifconfig fyi0 feitest-036 netmask 0xffffff00
```

The following example shows the use of the `-d` option:

```
ows1600% /etc/fyroute fyi0 -d
table set time: Mon Feb 17 13:29:46 1992
hash 0 key dfff1d05 flags 3
        dst 0005 ctl 0000 access 0000 mtu  4144
hash 1 key dfff1d0a flags b
        dst 0001 ctl 0000 access 0000 mtu  4144
hash 2 key dfff1d0b flags b
        dst 0001 ctl 0000 access 0000 mtu  4144
hash 3 key dfff1d01 flags 3
        dst 0001 ctl 0000 access 0000 mtu  4144
gate[0] = 0
```

The following example shows the use of the `-D` option:

```
ows1600% /etc/fyroute fyi0 -D
direct          sn1600-036      0005    0000    0000     4144;
swloopback      feitest-swlp    0001    0000    0000     4144;
hwloopback      feitest-hwlp    0001    0000    0000     4144;
direct          feitest-036     0001    0000    0000     4144;
```

FILES

`/etc/hosts`	OWS-E host name database
`/etc/fycf.owse`	Default routing file

SEE ALSO

`hosts`(5) for information about the SunOS host name database
`ifconfig`(8) for information about the SunOS command that configures the network interface parameters

NAME

`getconfig` – Retrieves system parameter values from the system configuration file

SYNOPSIS

`/bin/getconfig` *parameter*

DESCRIPTION

The `getconfig` program serves as a command interface to the `config` library routine. `config` scans the system configuration file, `/etc/configfile`, for a requested system parameter label and returns the associated system parameter value.

The argument to `getconfig` is as follows:

parameter	Specifies a system parameter label. *parameter* is one of the strings defined in `/etc/configfile`.

The `/etc/configfile` file contains such parameters as path names of the default UNICOS kernel and parameter files, IOS-E binary files, and the various definitions necessary to configure the system dump device. All machine-dependent system parameters are also found in this file.

Various system scripts use `getconfig` when a system parameter value is required.

The `config` routine checks the `OWSECONFIG` environment variable for an alternate path name to the configuration file.

FILES

`/etc/configfile`	Default OWS-E configuration file

RETURN VALUES

If the *parameter* label is found, the associated string is written to standard output and a status of `0` is returned; otherwise, a status of `1` is returned.

SEE ALSO

`configfile(5)` and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about `/etc/configfile`

NAME

`graphs` – Displays CPU time statistics in graphic form

SYNOPSIS

`/home/`*localhost*`/cri/bin/graphs [-h` *hostname*`] [-l` *graphlabel*`] [-p` *port*`]` *graphlist*

DESCRIPTION

The `graphs` command displays time statistics in graphic form. The program accepts data via a socket from `cpud`(8) and displays it in line-graph form. This application is based on XView, with the actual display contained within an XView base frame.

The arguments to `graphs` are as follows:

`-h` *hostname* Specifies the name of the host where the CPU daemon (`cpud`) is running. This argument is an Internet address of either ASCII format (such as `host.cray.com`) or numerical format (such as `192.9.30.126`). The default is `CPUD_HOSTNAME` in `/etc/configfile`. This option can be used to monitor mainframes not directly connected to the current OWS-E. If you specify the *hostname* as the name of the workstation connected to the mainframe of interest, the `graphs` client reads the socket of the `cpud` running on that OWS-E.

`-l` *graphlabel* Prefixes the default `System Monitors` base frame header with *graphlabel*`:` (for example, `ows1600: System Monitors`).

`-p` *port* Specifies the port on which `cpud` is listening. As released, this value is 4372. If the `BASEPORT` parameter is changed in the `/etc/configfile` configuration file, then this default will be the new `BASEPORT` value + 2.

graphlist Indicates the CPU aspects to be monitored. A single `graphs` client may display a maximum of 15 graphs at one time. The valid *graphlist* values are as follows:

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

n is the CPU number; the CPUs begin with number 0. If you have 8 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`.

The display consists of a canvas where each line graph is drawn, a title bar (consisting of *graphlabel*`: System Monitors` if *graphlabel* is specified, or `System Monitors` if it is not) and a short textual description beneath each displayed graph.

EXAMPLES

The following command line displays the average idle, user, system-wait, and system time over all CPUs for system sn1600:

```
graphs -h ows1600.cray.com -l sn1600 idle user sysw unix
```

RETURN VALUES

If the `graphs` program exits successfully, a value of `0` is returned. If there is an error, a value of `1` is returned.

BUGS

You cannot resize the base frame.

CONFIGURATION FILE PARAMETERS

The `graphs` command reads the following parameters from `/etc/configfile`:

`BASEPORT`	Defines the starting port value used for the various operator interface software daemons. Default: `4370`
`CPUD_HOSTNAME`	Specifies the name of the machine in which the CPU monitor, `cpud`(8), is running. Default: `__CPUDHOSTNAME__` token (replaced during installation)

FILES

`/etc/configfile`	Default OWS-E configuration file

SEE ALSO

`configfile`(5) for information about `/etc/configfile`
`cpud`(8) for more information about gathering data and dispersing CPU time statistics
OWS-E Operator Workstation Administrator's Guide, publication SG–3079, for more information about CPU monitors and `/etc/configfile`

NAME

`hbeat` – Monitors the IOS-E system

SYNOPSIS

`/home/`*localhost*`/cri/bin/hbeat [-r]`

DESCRIPTION

The `hbeat` command continually polls each booted IOP, looking for either halt codes or clocks that have stopped (which indicates a hung system). If either condition is detected, a message is sent to the console informing the operator and the condition is logged in `/var/logs/ioplog`. `hbeat` is automatically started when the first IOP is booted. When an IOP halt is detected, `hbeat` runs the `iophalt`(8) script.

The argument to `hbeat` is as follows:

`-r`	Restarts `hbeat`. `hbeat` reads `/var/logs/sstbackup` to find out which IOPs are currently running. It then initializes itself based on this file. Use this option if the `hbeat` process was killed or if the OWS-E has crashed.

The `hbeat` command also sends all IOP halt messages to the SMARTE system running on the MWS-E maintenance workstation.

Permission to access this command is set in `/etc/owsepermfile` by the system administrator.

CONFIGURATION FILE PARAMETERS

The `hbeat` command reads the following parameters from `/etc/configfile`:

`SSTBACKUP`	Specifies the back-up `hbeat`(8) status table. Default: `/var/logs/sstbackup`
`IOPLOG`	Defines the path name of the IOP log file. Default: `/var/logs/ioplog`
`IOPHALT`	Defines the path name of the `iophalt` script. Default: `/home/`*localhost*`/cri/bin/iophalt`

ENVIRONMENT VARIABLES

`OWSECONFIG`	Specifies the system configuration file; by default, it is set to `/etc/configfile`

FILES

`/etc/configfile`	Default OWS-E configuration file
`/etc/owsepermfile`	Permissions file that contains a list of accounts and the commands they are allowed to access
`/home/`*localhost*`/cri/bin/iophalt`	Default script that is run by `hbeat` when a halt is detected
`/var/logs/ioplog`	Default IOP log file
`/var/logs/sstbackup`	Default file that `hbeat` uses to contain the last known image of the `hbeat` table

SEE ALSO

`configfile`(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about `/etc/configfile`
`owsepermfile`(5) for information about the default OWS-E permission file
`emon`(8) for more information about restarting the IOS-E error logging, heartbeat, and CPU monitors
`errlogd`(8) for more information about the IOS-E hardware error logging daemon, which receives error information from the IOS-E and logs it in the `errlog` file
`iophalt`(8) for information about dumping an IOP in the event of an IOP failure
`rcpud`(8) for more information about the IOS-E remote CPU daemon, which processes service requests from the mainframe
I/O Subsystem Model E (IOS-E) Guide, publication SD–2107, for information on `iophalt`. (This document contains information private to Cray Research, Inc. It can be distributed to non-CRI personnel only with approval of the appropriate Cray Research manager.)

NAME

`hcon` – Configures a MUXIOP high-speed channel up or down

SYNOPSIS

`/home/`*localhost*`/cri/bin/hcon [-c` *cluster*`] [-d] [-D] [-m` *mode*`] [-t` *target*`]` *hisp*

DESCRIPTION

The `hcon` command configures up a MUXIOP high-speed channel by default. The channel is configured down if you specify the `-d` option. The MUXIOP of the designated cluster must be running.

The arguments to the `hcon` are as follows:

`-c` *cluster*	Specifies the number of the cluster in which the MUXIOP resides. The range of valid cluster numbers depends on the number of clusters in the IOS-E. The default is `0`.
`-d`	Configures the channel down. If you do not specify this option, `hcon` configures the channel up.
`-D`	Sets debug mode for the `hcon` program and sends the output to standard error.
`-m` *mode*	Specifies the channel mode; this option is not normally used, because the correct mode is chosen for you when you specify the `-t` option. *mode* can be one of the following values:

`c100d100`	100 Mbytes/s for control and 100 Mbytes/s for data. (This is the default when you specify `-t ssd-D`.)
`c100d200`	100 Mbytes/s for control and 200 Mybtes/s for data. (This is the default when you specify `-t ymp.`)
`c200d200`	200 Mbytes/s for control and 200 Mbytes/s for data. (This is the default when you specify `-t ssd` or `-t c90`.)

`-t` *target*	Sets target memory for the high-speed channel. *target* can be one of the following values:

`ymp`	CRAY Y-MP central memory (default)
`c90`	CRAY Y-MP C90 central memory
`ssd`	Model E SSD memory
`ssd-D`	Model D SSD memory

hisp	Specifies the high-speed channel number, which can be either `0` or `1`.

Log messages generated by this command are sent to the file specified by the `IOPLOG` parameter in `/etc/configfile`. By default, `IOPLOG` is set to `/var/logs/ioplog`. Error messages are written to standard error.

Permission to access this command is set in `/etc/owsepermfile` by the system administrator.

EXAMPLE

The following example configures MUXIOP high-speed channel number 1 up for IOS 0, cluster 3, using SSD-E memory:

```
hcon -c 3 -t ssd 1
```

CONFIGURATION FILE PARAMETERS

The command reads the following parameters from `/etc/configfile`:

`IOPLOG`	Defines the path name of the IOP log file. Default: `/var/logs/ioplog`

FILES

`/etc/configfile`	Default OWS-E configuration file
`/etc/owsepermfile`	Permissions file that contains a list of which accounts may access each command
`/var/logs/ioplog`	Default path name of the IOP log file

SEE ALSO

`configfile`(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about `/etc/configfile`
`owsepermfile`(5) for information about the default OWS-E permission file
`econ`(8) for information about configuring a MUXIOP-to-EIOP low-speed channel up or down

NAME

`iophalt` – Dumps an IOP in the event of an IOP failure

SYNOPSIS

`/home/`*localhost*`/cri/bin/iophalt` *cluster iop* [*message*]

DESCRIPTION

The `iophalt` script is usually invoked by `hbeat`(8); you should not need to invoke it directly.

When `hbeat` encounters an IOP failure, the `iophalt` script uses the `edump` command to take dump images of the IOP in question and the MUXIOP for that cluster. `edump` places the dump images in the file specified by the `DEFAULTIDUMPDIR` parameter in `/etc/configfile`; by default, this file is `/var/dumps`. `iophalt` places *message* in the file specified by the `IOPLOG` parameter in `/etc/configfile`; by default, this file is `/var/logs/ioplog`.

The arguments to `iophalt` are as follows:

cluster	Specifies the number of the cluster on which the IOP to be dumped is located. *cluster* can be an integer value in the range `0` through `7`; you may specify only one cluster. There is no default.
iop	Specifies the IOP (or multiple IOPs) to be dumped. *iop* can be an integer value in the range `0` through `4` (`4` indicates the MUXIOP), `mux`, or `all`; to specify more than one IOP, separate the IOP numbers with a comma (for example, `0,1,3`). There is no default.
message	Specifies the reason for taking a dump of the IOP. The message can be up to 80 characters in length; the string must be enclosed in quotation marks if it contains white space.

`iophalt` also sends mail to the destination specified by the `MAIL_IOPFAIL` parameter in `/etc/configfile`; usually, mail is sent to the system administrator. To specify more than one user, use the SunOS `aliases`(5) file.

The system administrator should modify the `DEFAULTIDUMPDIR` and `MAIL_IOPFAIL` configuration parameters to site-specific values.

CONFIGURATION FILE PARAMETERS

The `iophalt` command reads the following parameters from `/etc/configfile`:

`DEFAULTIDUMPDIR`	Defines the default dump directory path in which the dump shell script is created. Default: `/var/dumps`
`IOPLOG`	Defines the path name of the IOP log file. Default: `/var/logs/ioplog`
`MAIL_IOPFAIL`	Defines the login name to which mail is sent if an IOP halts. Default: `cri`

FILES

`/etc/configfile`	Default OWS-E configuration file
`/var/dumps`	Default dump file
`/var/logs/ioplog`	Default IOP log file

SEE ALSO

`aliases`(5) for information about the SunOS file for `sendmail`(8)
`configfile`(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about `/etc/configfile`
`bootsys`(8) for information about the command that boots the IOS-E and the mainframe
`edump`(8) for information about dumping IOP local memory images to the OWS-E
`hbeat`(8) for information about the heartbeat monitor, which calls `iophalt`

NAME

`lapdaemon` – Validates CRI `tty` lines for users

SYNOPSIS

`home/`*localhost*`/cri/bin/lapdaemon`

DESCRIPTION

The `lapdaemon` program is called by the `zip`(8) program when you try to open a `tty` line. If you do not specify a particular `tty` line, `lapdaemon` will find the first line available, beginning with line `1`; if lines `1` through `4` are all busy, it will then check the console line (line `0`). If you do not request a specific line and all lines are busy, the line held by a user with a lower permission than yours will be usurped if usurp mode has been toggled on. If you request a specific line and that line is busy, the line will be usurped from the present owner if he or she has a permission lower than yours and if usurp mode has been toggled on. The owner of a line is sent a message if the line he or she has is usurped.

`lapdaemon` 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`.

CONFIGURATION FILE PARAMETERS

The `lapdaemon` command reads the following parameter from `/etc/configfile`:

`LAPFILE`	Specifies the location of the line-arbitration priority file used by `lapdaemon`(8). Default: `/etc/lapfile`

ENVIRONMENT VARIABLES

`OWSECONFIG`	Specifies the system configuration file; by default, it is set to `/etc/configfile`

FILES

`/etc/configfile`	Default OWS-E configuration file
`/etc/lapfile`	Default line arbitration priority file

SEE ALSO

`configfile`(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about `/etc/configfile`
`lapfile`(5) for information about the line arbitration priority file
`zip`(8) for information about the command that supplies the terminal interface to a running

NAME

`mfdump` – Dumps the mainframe memory and CPU registers to a CRI disk on the IOS-E

SYNOPSIS

`/home/`*localhost*`/cri/bin/mfdump [-b` *bootstrap*`] [-c` *cluster*`] [-D] [-f` *file*`] [-F] [-i` *iop*`] [-q] [-v] [`*reason*`]`

DESCRIPTION

The `mfdump` command dumps the mainframe memory and CPU registers to a CRI disk on an IOS-E. The dumped binary resides on the preallocated dump slice and can be moved to a spot in the file system upon reboot of the mainframe. Cluster 0 is the cluster through which the CPU dump binary is to be routed.

The arguments to `mfdump` are as follows:

`-b` *bootstrap*	Specifies the full path name of the bootstrap loader program for the CPU. If you do not specify this option, the default program is the one specified by `MFBOOT` in `/etc/configfile`; by default, this program is `/home/`*localhost*`/cri/os/uts/mfboot`.
`-c` *cluster*	Specifies the IOP cluster to use for data transfer and deadstart functions. The range of valid cluster numbers depends on the number of clusters in the IOS-E. The default is `0`.
`-D`	Sets debug mode in the `mfdump` program and sends the output to standard error.
`-f` *file*	Specifies the path name of the CPU-resident system dump program. If you do not specify this option, the default program is the one specified by `MFSYSDMP` in `/etc/configfile`; by default, this program is `/home/`*localhost*`/cri/os/uts/mfsysdmp`.
`-F`	Forces a dump. If there is a dump on the default slice and you try to take another dump, you will get an error message. (The default slice is the area of memory defined by `DIOPATH`, which defines the channel and IOP cluster, `DUNIT`, which defines the unit of the disk drive, `DSTART`, which defines the starting address on the disk, and `DLEN`, which defines the length of the slice. The defaults for these parameters are set in `/etc/configfile`.) However, if you use the `-F` option, the current dump will overwrite the existing dump. You should use this option if the current dump is more important than the first dump.
`-i` *iop*	Specifies the number of the IOP through which to route. *iop* can be set to an integer from `0` through `4`; `4` specifies the MUXIOP. The default is `0`.
`-q`	Queries you for memory types to dump. Memory ranges and types can be changed from the defaults, which are set in the `mfdump` program.
`-v`	Sets verbose mode. This option forces `mfdump` to print informative messages to standard error.

reason Specifies the reason for the dump; for example, `"CPU Hung"`. Quotation marks are optional, even if the reason contains white space; however, you should use them to avoid problems with special characters. *reason* is truncated after 79 characters, not including quotation marks (a longer reason will not cause an error, but the 80th and succeeding character will not be used). If you do not add *reason* in the command line, you will be prompted for it later. Although it is not required, adding this explanation in the command line is especially useful if you intend to use scripts for autoboots or autodumps; if you do so, you will not have to wait for the system to prompt you for a reason. Entering a halt code as a reason also helps to distinguish dumps as dump files start to accumulate.

Permission to access this command is set in `/etc/owsepermfile` by the system administrator.

EXAMPLES

The `mfdump` command is found on the OWS-E. The path from the OWS-E to the mainframe is by way of cluster 0 and IOP 0. The path from the mainframe to the dump device is by way of the dump cluster and dump IOP; the defaults for these are set with the `DIOPATH` parameter in `/etc/configfile`. `DIOPATH` is a composite value made up of the dump device's cluster (the first digit), IOP (the second digit), and a channel number.

To route the binary into the mainframe, you must halt all of the EIOPs with the `ehalt` command and then reboot the default cluster's MUXIOP with the `eboot`(8) command. When you have done this, you must configure up the high-speed channel from the default cluster's MUXIOP to the mainframe with `hcon`. You can then reboot the EIOPs and configure the channel from the MUXIOP to the EIOPs using the `eboot`(8) and `econ`(8) commands. Finally, you can take the dump with the `mfdump` command.

The following example shows the procedure you could follow, as an alternative to `bootsys -i`, to route the dump binary through cluster 0, IOP 0, and place the dump image on the dump cluster and IOP (which in this case are cluster 0 and IOP 2):

```
ows1600% getconfig DIOPATH
00230                                   (Dump to cluster 0, IOP 2, channel 30)
ows1600% ehalt 0:0-4
ows1600% eboot -F 0:4:~cri/os/ios/iopmux      (Boots MUXIOP)
ows1600% hcon -c 0 -t ymp 0         (Configures connection between MUXIOP and CPU)
ows1600% eboot 0:0:~cri/os/ios/eiop.comm     (Boots EIOP for loading mainframe)
ows1600% eboot 0:2:~cri/os/ios/eiop.dcal     (Boots EIOP for disk to receive dump image)
ows1600% econ -c 0 0                (Configures up LOSP between)
ows1600% econ -c 0 2                (EIOPS and the MUXIOP)
ows1600% mfdump "CPU hung"          (Takes the dump)
```

CONFIGURATION FILE PARAMETERS

The command reads the following parameters from `/etc/configfile`:

`D0FWA`	
`D0LWA`	
`D1FWA`	
`D1LWA`	
`D2FWA`	
`D2LWA`	
`D3FWA`	
`D3LWA`	These parameters define the actual mainframe memory ranges to be dumped. Default: 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.
`DEFAULTUPARAMFILE`	Defines the path name to the default UNICOS parameter file. Default: `/home/`*localhost*`/cri/os/uts/param`
`DEF_MFCHAN`	Defines the mainframe channel number of the low-speed channel attached to the cluster that deadstarts the mainframe. Default: `020`
`DIOPATH`	Defines the path that the memory dump will take from the mainframe to the disk. Default: `__DUMPIO__` token (replaced during installation)
`DLEN`	Defines the length, in sectors, of the disk slice to which the memory will be dumped. Default: `__DUMPLEN__` token (replaced during installation)
`DSTART`	Defines the starting sector of the disk slice to which the memory will be dumped. Default: `__DSTARTBL__` token (replaced during installation)
`DTYPE`	Defines the type of the disk to which the mainframe memory will be dumped. Default: `__DUMPTYPE__` token (replaced during installation)
`DUNIT`	Defines the default dump device unit that the `mfdump`(8) command uses when routing the `mfsysdmp` binary file to the mainframe before the dump. Default: `__DUMPUNIT__` (replaced during installation)
`IOPLOG`	Defines the path name of the IOP log file. Default: `/var/logs/ioplog`
`MFBOOT`	Defines the path name of the bootstrap loader program used by the `mfdump`(8) command. Default: `/home/`*localhost*`/cri/os/uts/mfboot`

MFSYSDMP	Defines the path name of the CPU-resident program used by the mfdump(8) command. Default: /home/*localhost*/cri/os/uts/mfsysdmp
SSD_MEMORY	Defines the memory size of the SSD attached to the mainframe to which the OWS-E is attached. Default: __SSD_MEMORY__ token (replaced during installation)

ENVIRONMENT VARIABLES

OWSECONFIG	Specifies the system configuration file; by default, it is set to /etc/configfile

FILES

/etc/configfile	Default OWS-E configuration file
/etc/owsepermfile	Permissions file that contains a list of accounts and the commands they are allowed to access
/home/*localhost*/cri/os/uts/mfboot	Default bootstrap loader program
/home/*localhost*/cri/os/uts/mfsysdmp	Default CPU-resident system dump program

SEE ALSO

configfile(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about /etc/configfile
owsepermfile(5) for information about the default OWS-E permission file
cpudump(8) for information about the script that boots the IOS and forces a UNICOS dump
eboot(8) for information about booting the IOS-E from the OWS-E
econ(8) and hcon(8) for information about configuring the IOS-E MUXIOP channels
edump(8) for information about dumping IOP local memory images to OWS-E
ehalt(8) for information about halting the IOS-E
hcon(8) for information about configuring the high-speed channel on the MUXIOP

NAME

`mfinit` – Runs a mainframe and IOS-E initialization and confidence test

SYNOPSIS

`/home/`*localhost*`/cri/bin/mfinit` [`-c` *cluster*] [`-C` *cpus*] [`-D`] [`-f` *pathname*] [`-i` *iop*] [`-l` *clusters*] [`-n` *errors*]

DESCRIPTION

The `mfinit` command runs a confidence check, `mfchkye`, in the CPU. (The name `mfchkye` refers to "mainframe check on the CRAY Y-MP and IOS-E.") `mfinit` issues a master clear function through the MUXIOP, loads the test binary through a running EIOP, and issues a drop master clear function through the same MUXIOP. The test checks memory and registers in all CPUs and reports status in a response block in CPU memory. `mfinit` polls the response block and reports any failures to the operator.

The `mfinit` command allows you to set a maximum error count after which the program aborts.

The arguments to `mfinit` are as follows:

`-c` *cluster*	Specifies the number of the cluster to be used to control the master clear deadstart lines. The range of valid cluster numbers depends on the number of clusters in the IOS-E. The default is `0`.
`-C` *cpus*	Specifies the number of CPUs to test, beginning with CPU 0. The range of valid CPUs depends on the number of CPUs available at your site. (If you enter `4` and have 8 CPUs available, CPUs 0 through 3 will be tested.) If you do not enter the `-C` option, the default number of CPUs tested is the total number of CPUs for which your system is configured.
`-D`	Sets debug mode in the `mfinit` program and sends the output to standard error.
`-f` *pathname*	Specifies the full path name of the mainframe test to load. If you do not enter the `-f` option, the default is specified by the `MFIPATH` parameter in `/etc/configfile`; at release, `MFIPATH` is set to `/home/`*localhost*`/cri/os/uts/mfchkye`.
`-i` *iop*	Specifies the number of the EIOP to be used to load the mainframe. *iop* can be set to an integer in the range 0 through 3. If you do not enter the `-i` option, the default is `0`.
`-l` *clusters*	Specifies the number of cluster registers to test on the mainframe. The range of valid clusters is machine dependent. If you do not specify the `-l` option, the default is the hardware configuration for cluster registers.
`-n` *errors*	Specifies the maximum number of errors allowed before aborting the test. If you do not specify this option, the default is `16`.

The `mfinit` command also sends all CPU faults to the SMARTE system running on the MWS-E maintenance workstation.

Permission to access this command is set in `/etc/owsepermfile` by the system administrator.

EXAMPLES

The following examples show the typical use of `mfinit` and testing for specified CPUs.

Example 1: Typical Use

In most cases, you will enter `mfinit` without any options:

```
mfinit
```

Example 2: Testing Specified CPUs

Suppose you had 8 CPUs and you wanted to test the first 6 of them. You would enter the following:

```
mfinit -C 6
```

CONFIGURATION FILE PARAMETERS

The `mfinit` command reads the following parameters from `/etc/configfile`:

`MFIPATH`	Defines the path name of the diagnostic program used by the `mfinit` command. Default: `/home/`*localhost*`/cri/os/uts/mfchkye`

FILES

`/etc/configfile`	Default OWS-E configuration file
`/etc/owsepermfile`	Permissions file that contains a list of accounts and the commands they are allowed to access
`/home/`*localhost*`/cri/os/uts/mfchkye`	Default mainframe test file

SEE ALSO

`configfile`(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about `/etc/configfile`
`owsepermfile`(5) for information about the default OWS-E permission file
`bootsys`(8) for information about booting the IOS-E and the mainframe, using values from the UNICOS parameter file rather than scripts

NAME

`mfstart` – Starts the mainframe CPUs from the OWS-E

SYNOPSIS

`/home/`*localhost*`/cri/bin/mfstart` [`-c` *cluster*] [`-D`] [`-f` *filesystem*] [`-i` *iop*] [`-m` *mainframe*] [`-p` *parameter*] [`-u` *unicos*] [`-U` *unicos*] [`-v`]

DESCRIPTION

The `mfstart` command starts the mainframe CPUs from the OWS-E. `mfstart` issues a master clear function through the MUXIOP, loads the binary through a running EIOP, and issues a drop master clear function to the MUXIOP.

`mfstart` starts the `rcpud` daemon if it is not currently running.

The arguments to `mfstart` are as follows:

`-c` *cluster*	Specifies the number of the cluster to be used to control the master clear deadstart lines. The range of valid cluster numbers depends on the number of clusters in the IOS-E. The default is `0`.
`-D`	Sets debug mode in the `mfstart` program and sends the output to standard error.
`-f` *filesystem*	Specifies the full path name of an install file system to load on a new system.
`-i` *iop*	Specifies the number of the EIOP to be used to load the mainframe. *iop* can be set to an integer in the range `0` through `3`. The default is `0`.
`-m` *mainframe*	Specifies the mainframe channel number to which the deadstarting MUXIOP is connected. The default channel number is specified by the `DEF_MFCHAN` parameter in `/etc/configfile`; at release, this parameter is set to `020`.
`-p` *parameter*	Specifies the full path name of the parameter file to use. If you do not specify the `-p` option, the default parameter file is the one specified by the `DEFAULTUPARAMFILE` label in `/etc/configfile`; by default, this file is `/home/`*localhost*`/cri/os/uts/param`.
`-u` *unicos*	Specifies the full path name of the UNICOS binary file to use. If you do not specify the `-u` option, the default UNICOS binary file is the one specified by the `DEFAULTUKERNFILE` label in `/etc/configfile`; by default, this file is `/home/`*localhost*`/cri/os/uts/unicos`.
`-U` *unicos*	Specifies the full path name of the CPU-resident UNICOS binary file.
`-v`	Sets verbose mode. This option forces `mfstart` to print informative messages to standard error.

Permission to access this command is set in `/etc/owsepermfile` by the system administrator.

CONFIGURATION FILE PARAMETERS

The command reads the following parameters from `/etc/configfile`:

`DEFAULTUPARAMFILE`	Defines the path name to the default UNICOS parameter file. Default: `/home/`*localhost*`/cri/os/uts/param`
`DEFAULTUKERNFILE`	Defines the path name to the default UNICOS binary. Default: `/home/`*localhost*`/cri/os/uts/unicos`

`DEF_MFCHAN`	Defines the mainframe channel number of the low-speed channel attached to the cluster that deadstarts the mainframe. Default: `020`
`RCPUD`	Defines the path name of the remote CPU request daemon. Default: `/home/`*localhost*`/cri/bin/rcpud`

FILES

`/etc/configfile`	Default OWS-E configuration file
`/etc/owsepermfile`	Permissions file that contains a list of accounts and the commands they are allowed to access
`/home/`*localhost*`/cri/os/uts/param`	Default parameter file
`/home/`*localhost*`/cri/os/uts/unicos`	Default UNICOS binary file

SEE ALSO

`configfile`(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about `/etc/configfile`
`owsepermfile`(5) for information about the default OWS-E permission file
`bootsys`(8) for information about booting the IOS-E and the mainframe using values from the UNICOS parameter file, rather than scripts
`rcpud`(8) for information about the IOS-E remote CPU daemon

NAME

newlog – Creates new errlog and ioplog files while backing up the existing ones

SYNOPSIS

/home/*localhost*/cri/bin/newlog

DESCRIPTION

The newlog script backs up the log files defined by the ERRLOG and IOPLOG labels in /etc/configfile. For example, it moves the contents of the file errlog.2 to errlog.3, the contents of errlog.1 to errlog.2, and so on. For example, this allows you to keep logs for the previous four days if newlog is run once a day.

EXAMPLES

You might want to use newlog in a crontab file. For example, if you wanted to run newlog every day at midnight, you could have the following line in /var/spool/cron/crontabs/cri:

```
0 0 * * * /bin/sh /home/localhost/cri/bin/newlog
```

CONFIGURATION FILE PARAMETERS

The newlog command reads the following parameters from /etc/configfile:

ERRLOG	Defines the path name of the error logging daemon. Default: /home/*localhost*/cri/bin/errlogd
IOPLOG	Defines the path name of the IOP log file. Default: /var/logs/ioplog

FILES

/etc/configfile	Default OWS-E configuration file

SEE ALSO

crontab(1) and crontab(5) for information about the SunOS command and file used to run periodic jobs
configfile(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about /etc/configfile

NAME

olnet – Detects and isolates network problems with the OLNET on-line diagnostic network communications tool

SYNOPSIS

/home/*localhost*/cri/smarte/bin/olnet [*subcommands*]

DESCRIPTION

The olnet program detects and isolates network problems. For the OWS-E, olnet tests the network connection using the fy driver through the FEI, the low-speed channel, and the IOS channel adapter to the CRI mainframe.

You can use olnet in either interactive mode or command-line mode; the commands are the same. (For details about the commands, see *OLNET On-line Diagnostic Network Communications Program Maintenance Manual for UNICOS*, publication SMM–1021.)

Interactive Mode

When you enter olnet by itself, you will get the following main menu:

```
OLNET main menu
---------------
VT - Call the VME test.
H  - Help with OLNET.
QT - Quit OLNET.
```

You must enter one of these commands: VT, H, or QT. If you enter VT, olnet displays the VME test menu, which is as follows:

```
                VME TEST

Command                         Value
-------                         -----

PC - Pass count --------------> 1
MP - Messages/pass -----------> 10
AL - Associated data length --> 100
PT - Pattern type------------>  ADDRESS
MD - Message proper data -----> DISABLED
RA - Remote address(Hex) -----> undefined
TM - Test mode ---------------> Active mode
DV - Device path -------------> undefined



Local address information       Value
-------------------------       -----

Local address(hex) ------------> undefined
```

```
Execute/exit commands
---------------------

EX - Execute the current VME test mode.
H  - Helpful information.
RT - Return to the Main menu.
QT - Quit OLNET.
```

Command-line Mode

To use command-line mode, enter `olnet`, followed by the subcommands and arguments you want; separate each subcommand and argument from the next by a comma. The subcommands are those shown in the `olnet` menus. For example, the following command line enters `olnet`, executes the `VT` (change to VME test menu) command, and sets the device path to `/dev/fyc01` with the `DV` command:

```
olnet VT,DV,/dev/fyc01
```

At this point, `olnet` would display the following:

```
              VME TEST

Command                        Value
-------                        -----

PC - Pass count --------------> 1
MP - Messages pass -----------> 10
AL - Associated data length --> 100
PT - Pattern type------------>  ADDRESS
MD - Message proper data -----> DISABLED
RA - Remote address(Hex) -----> undefined
TM - Test mode ---------------> Active mode
DV - Device path -------------> /dev/fyc01


Local address information      Value
-------------------            -----

Local address(hex) ------------> undefined


Execute/exit commands
---------------------

EX - Execute the current VME test mode.
H  - Helpful information.
RT - Return to the Main menu.
QT - Quit OLNET.
```

Notice that `DV - Device path` is now set to `/dev/fyc01`.

SEE ALSO

OLNET On-line Diagnostic Network Communications Program Maintenance Manual for UNICOS, publication SMM–1021.
UNICOS 6.E Early Release Software On-line Diagnostic Technical Note, publication SPN–1022. (These manuals are Cray Research Proprietary; dissemination of this information to non-CRI personnel requires approval from the appropriate vice president and a nondisclosure agreement. Export of technical information in this category may require a Letter of Assurance.)

NAME

`peek` – Peeks (looks) at memory

SYNOPSIS

`/home/`*localhost*`/cri/bin/peek -t` *type* [`-b` *channel*] [`-c` *cluster*] [`-D`] [`-d` *display*] [`-f` *format*] [`-i` *iop*] [`-v`] *address*[`,` *size*]

DESCRIPTION

The `peek` command allows a user to read any memory type. The types are CPU memory, SSD memory, IOP local memory, and channel buffer memory. If CPU or SSD memory is to be read, a MUXIOP and an EIOP must be running. Channel buffer memory is accessible only through an EIOP.

The arguments to `peek` are as follows:

`-t` *type*	Specifies the memory type. *type* can be set to `cbuf` (channel buffer), `cpu`, `iop`, or `ssd`. You must specify this option; there is no default. If you specify `cbuf`, you must also specify the `-b` option.
`-b` *channel*	Specifies the channel buffer channel number when you specify `cbuf` for `-t`. *channel* can be set in the range `030` to `037`; to specify octal, you must use a leading 0. This option is only valid when you specify `-t cbuf`, and is required with `-t cbuf`.
`-c` *cluster*	Specifies the number of the IOS-E cluster to be used. The range of valid cluster numbers depends on the number of clusters in the IOS-E. The default is `0`.
`-D`	Sets debug mode in the `peek` program and sends the output to standard error.
`-d` *display*	Specifies the display type. *display* can be set to `hex` (hexadecimal) or `octal`. The default is `octal`.
`-f` *format*	Specifies the format of the display. *format* can be set to `word`, `parcel`, `byte`, `bit`, `elan` (IOP instruction format), or `trace` (trace buffer dump). The default depends upon what you specify for `-t`: `parcel` if you specify `iop` for `-t`; `word` if you specify `cbuf`, `cpu`, or `ssd` for `-t`.
`-i` *iop*	Specifies the number of the IOP to be used. *iop* can be set to an integer in the range `0` through `4`. The default is IOP 0 if you specify `cpu`, `iop`, or `ssd` for the `-t` option; however, if you specify `cbuf` for the `-t` option, you must specify this option (there is no default with `cbuf`).
`-v`	Sets verbose mode. This option forces `peek` to print informative messages.
address	Specifies the starting octal address. You must enter this operand; there is no default.
size	Specifies the number of units looked at from the starting address. For IOP memory, the unit is a parcel; otherwise, the unit is a word. The default is 1 unit. If you specify *size*, you must separate it from *address* with a comma (`,`).

Permission to access this command is set in `/etc/owsepermfile` by the system administrator.

FILES

`/etc/owsepermfile`	Permissions file that contains a list of accounts and the commands they are allowed to access

SEE ALSO

`owsepermfile`(5) for information about the default OWS-E permission file
`ecrash`(8) for more information about examining an IOS-E dump image or viewing a running system

NAME

`poke` – Pokes (places) a pattern into memory

SYNOPSIS

`/home/`*localhost*`/cri/bin/poke -p` *pattern* `-t` *type* [`-b` *channel*] [`-c` *cluster*] [`-D`] [`-i` *iop*] [`-v`] *address*[`,` *size*]

DESCRIPTION

The `poke` command allows a user to place a pattern into any memory type. Types are CPU memory, SSD or IOP local memory or channel buffer memory. To write to CPU or SSD memory, a MUXIOP and an EIOP must be running. Channel buffer memory is accessible only through an EIOP.

The arguments to `poke` are as follows:

`-p` *pattern*	Specifies an octal or hexadecimal pattern of up to 64 bits to set into memory (16 bits if IOP memory).
`-t` *type*	Specifies the memory type. *type* can be set to `cbuf` (channel buffer), `cpu`, `iop`, or `ssd`. You must specify this option; there is no default. If you specify `cbuf`, you must also specify the `-b` option.
`-b` *channel*	Specifies the channel buffer channel number when you specify `cbuf` for `-t`. *channel* can be set in the range `030` to `037`; to specify octal, you must use a leading 0. This option is only valid when you specify `-t cbuf`, and is required with `-t cbuf`.
`-c` *cluster*	Specifies the number of the IOS-E cluster to be used. The range of valid cluster numbers depends on the number of clusters in the IOS-E. The default is `0`.
`-D`	Sets debug mode for the `poke` program and sends the output to standard error.
`-i` *iop*	Specifies the number of the IOP to be used. *iop* can be set to an integer in the range `0` through `4`. The default is IOP 0 if you specify `cpu`, `iop`, or `ssd` for the `-t` option; however, if you specify `cbuf` for the `-t` option, you must specify this option (there is no default with `cbuf`).
`-v`	Sets verbose mode. This option forces `poke` to print informative messages to standard error.
address	Specifies the starting octal address. You must enter this operand; there is no default.
size	Specifies the number of units peeked from the starting address. For IOP memory, the unit is a parcel; otherwise, the unit is a word. The default is 1 unit. If you specify *size*, you must separate it from *address* with a comma (`,`).

Permission to access this command is set in `/etc/owsepermfile` by the system administrator.

FILES

`/etc/owsepermfile`	Permissions file that contains a list of accounts and what commands they are allowed to access

SEE ALSO

`owsepermfile`(5) for information about the default OWS-E permission file
`ecrash`(8) for more information about examining an IOS-E dump image or a running system

NAME

rcpud – Processes service requests from the mainframe (IOS-E remote CPU daemon)

SYNOPSIS

/home/*localhost*/cri/bin/rcpud

DESCRIPTION

The rcpud daemon processes service requests from the mainframe. Requests to perform services for the CPU are sent through O-packets from the mainframe. Services include the following: starting an IOP; halting an IOP; placing an IOP on-line (upping); placing an IOP off-line (downing); notifying the operator that an IOP is alive or has died; getting the CPU time and date; and notifying the operator of a CPU panic. Upon performing the remote service, rcpud returns status information to the mainframe through an O-packet. When UNICOS panics, rcpud runs the script /home/*localhost*/cri/bin/cpupanic, which can do such things as dumping and restarting the system.

The rcpud daemon also sends all CPU panic messages to the SMARTE system running on the MWS-E maintenance workstation.

CONFIGURATION FILE PARAMETERS

The rcpud command reads the following parameters from /etc/configfile:

CPUPANIC	Defines the path name of the cpupanic script. Default: /home/*localhost*/cri/bin/cpupanic

ENVIRONMENT VARIABLES

OWSECONFIG	Specifies the system configuration file; by default, it is set to /etc/configfile

FILES

/etc/configfile	Default OWS-E configuration file
/home/*localhost*/cri/bin/cpupanic	Script that dumps and restarts the system when UNICOS panics.

SEE ALSO

configfile(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about /etc/configfile
cpupanic(8) for information about the script rcpud calls to take a UNICOS panic dump
emon(8) for information about restarting the IOS-E error logging and heartbeat monitors
hbeat(8) for information about monitoring the IOS-E system

NAME

`smdemon` – Monitors the OWS-E system for SMARTE

SYNOPSIS

`/home/`*localhost*`/cri/smarte/bin/smdemon`

DESCRIPTION

The SMARTE OWS-E system monitor daemon, `smdemon`, is used by the System Maintenance and Remote Testing Environment (SMARTE) product as its interface to the OWS-E.

`smdemon` gathers UNICOS panic messages, IOS-E halt messages, static IOS-E configuration, and dynamic IOS-E configuration from the OWS-E.

SEE ALSO

`smdstop`(8) for information about the command to stop `smdemon`
System Maintenance and Remote Testing Environment (SMARTE) Guide, publication SPM–1017. (This document is Cray Research Proprietary; dissemination of this information to non-CRI personnel requires approval from the appropriate vice president and a nondisclosure agreement. Export of technical information in this category may require a Letter of Assurance. This manual is currently available in draft form.)

NAME

smdstop – Terminates the SMARTE OWS-E system monitor daemon

SYNOPSIS

/home/*localhost*/cri/smarte/bin/smdstop

DESCRIPTION

The smdstop command sends a software termination signal (SIGTERM) to the OWS-E system monitor daemon, smdemon(8).

SEE ALSO

signal(3V) for information about the Sun OS simplified software signal facilities and SIGTERM
smdemon(8) for information about the system monitor daemon
System Maintenance and Remote Testing Environment (SMARTE) Guide, publication SPM–1017. (This document is Cray Research Proprietary; dissemination of this information to non-CRI personnel requires approval from the appropriate vice president and a nondisclosure agreement. Export of technical information in this category may require a Letter of Assurance. This manual is currently available in draft form.)

NAME

`snmproute` – Performs route tracing with the Simple Network Management Protocol

SYNOPSIS

`/usr/ucb/snmproute` [`-c` *community*] [`-v`] *fromaddress toaddress*

DESCRIPTION

`snmproute` uses the Simple Network Management Protocol (SNMP) protocol to trace a route from a source location to a destination location. For this command to work correctly, it is important that all intermediate nodes support SNMP and the MIB-II (RFC 1213) variables.

The `snmproute` command accepts the following options:

`-c` *community*	Specifies community name used for SNMP packets. The default is the community name `public`.
`-v`	Sets up `snmproute` in a verbose mode. More information is displayed.
fromaddress	Specifies source address.
toaddress	Specifies destination address.

SEE ALSO

`snmpd`(8)

NAME

`xsnmpmon` – Invokes the SNMP network monitor

SYNOPSIS

(On OWS-E)
`/home/`*localhost*`/cri/bin/xsnmpmon [-d` *display* | `-display` *display*`] [-bd` *bordercolor*`]`
`[-bg` *backgroundcolor*`] [-fg` *foregroundcolor*`] [-fn` *font*`] [-lfn` *largefont*`] [-ib` *file*`] [-iconic]`

(On CRI mainframe)
`/usr/lib/monitor/xsnmpmon [-d` *display* | `-display` *display*`] [-bd` *bordercolor*`]`
`[-bg` *backgroundcolor*`] [-fg` *foregroundcolor*`] [-fn` *font*`] [-lfn` *largefont*`] [-ib` *file*`] [-iconic]`

DESCRIPTION

The `xsnmpmon` command invokes a Simple Network Management Protocol (SNMP) network monitor that uses the X Window System graphical user interface. This monitor allows you to obtain the status for any SNMP-compliant machine on your network. This monitor is normally run on the OWS-E, but it can be run on a CRI mainframe.

The arguments to `xsnmpmon` are as follows:

`-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`. You can also set this by 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`. You can also set this by 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`. You can also set this by 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 pixels by 13 pixels.
`-lfn` *largefont*	Specifies the large font, which is used for highlighting text. The default is 6 pixels by 13 pixels, bold.
`-ib` *file*	Specifies the icon bitmap file, which allows you to create your own icon.
`-iconic`	Starts `xsnmpmon` in iconic state.

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

Whenever the program requires you to input text, a dialog window containing a question appears at the top of your screen. To perform an action, 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.

At the top left corner of the main window is the `Setup` button, which 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`. Every window contain a `Quit` button that allows you to exit that function.

Following the `Setup` and `Quit` buttons is a block of 24 buttons. The first 16 buttons in the block represent the 16 SNMP MIB-II (Management Information Base) variable groups. Each of these groups 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. (See the SEE ALSO section.)

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 represent the status of the network; run a performance monitor; start up an operator notification window; and use the SunOS `telnet`(1) command to contact the selected SNMP agent (if you are running `xsnmpmon` on a CRI mainframe, use the UNICOS `telnet`(1B) command).

Beneath the block of buttons are lines of synopsis information. As indicated by the text, clicking the left mouse button in the main window increases the sampling rate, and clicking the middle mouse button decreases it.

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.

The monitor is capable of keeping a log file of all the activity that occurs on a daily basis. By default, the log file is named `xsnmpmon.log.`*mmddyy*, where *mmddyy* is the month, day, and year, respectively. At midnight on each day, `xsnmpmon` closes the current day's log file and automatically opens a new log file for the new day. You can change the default file name by resetting the `SNMP_LOGFILE` environment variable. If you do not want a log file, set `SNMP_LOGFILE` to the following:

```
/dev/null
```

The following subsections describe each of the buttons in the order in which they appear in the window.

`Setup` Button

The `Setup` button invokes a window containing several buttons that 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. The buttons in the `Setup` window, from left to right and top to bottom, are as follows:

Button	**Description**
`Save`	Permanently saves the current setup in the `xsnmpmon.rc` file (or the file specified by the `SNMP_RCFILE` environment variable) and exits from the Setup window.
`Colors`	Specifies the color of the windows and their contents. All of the current colors are listed at the bottom of the `Setup` window. When you click on `Colors`, you will be asked if 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.
`Telnet`	Specifies a shell command string to use to contact an agent using the `telnet`(1) command. For example, to open an `xterm` window with 80 columns and 60 rows, set jump scrolling, and have the `telnet` program run in the window, you would enter the following in the dialog window: `xterm -geom 80x60 -j -e telnet`
`Bell`	Turns on/off the bell sound when an error is reported. Each time you click this button, you toggle the state of the option.

`SC Log`	Turns on/off the logging of state changes. Each time you click on this button, you toggle the state of the option.
`Trap Log`	Turns on/off the logging of trap messages sent from agents to the monitor. Each time you click on this button, you toggle the state of the option.
`AutoSkip`	Turns on/off the ability to skip agents that are not SNMP reachable during network scans. When set to `YES`, only ICMP packets are sent until an SNMP trap is received. When set to `NO`, SNMP packets are sent. Each time you click on this button, you toggle the state of the option.
`IP Names`	Turns on/off the displaying of names for Internet Protocol (IP) addresses. Each time you click this button, you toggle the state of the option.
`MIB-II`	Turns on/off the use of MIB-II variables when examining SNMP agents; this is useful when you are connected to an agent that does not support MIB-II in order to avoid repeated error messages and for performance reasons. Each time you click this button, you toggle the state of the option.
`Auto`	Turns on/off the automatic map loading, scan starting, and pop-up visual symbol event notification; this is useful only if the configuration is saved. Each time you click on this button, you toggle the state of the option.
`Sampling`	Specifies the rate, in seconds, at which the MIB variables are sampled when the statistics windows are active.
`Polling`	Specifies the rate, in seconds, at which the agents are polled during the scanning process; that is, the amount of time the monitor waits between the time when it finishes scanning one agent and begins scanning the next. For example, if you have 10 agents and you specify `2`, it will take about 20 seconds to scan all of them. (1 second is the fastest rate allowed.)
`Cycle`	Specifies the time, in seconds, that the monitor waits between the time it finishes polling the last agent and the time it goes back to the first in the network map. For example, if you have 10 agents and you specify `2` for polling and `600` for cycle, every 10 minutes a cycle that takes about 20 seconds to complete will occur. (1 second is the fastest cycle allowed.)
`Agent`	Connects to a different SNMP agent. If you do not also specify a change for the community by using the `Community` button, the new agent is assumed to be in the same community as the previous agent.
`Comm'ty`	Specifies the community for an agent. Use this button when you change to a new agent that is in a community different from that of the previous agent.
`Net Dir`	Specifies the directory in which the network files are kept. When a network is created and retrieved for the scanning process, the files are assumed to be in the current directory. You can also specify the directory by setting the `SNMP_NETDIR` environment variable.

`Netfile`	Specifies the *networkfile*, which is the first file the monitor will attempt to load; this file represents the area at the highest layer. If you create a network consisting of multiple layers or areas, each area is contained in a file by itself; if the network file calls other files that represent subareas, the monitor will recursively load all of the files associated with it. For example, suppose you have CRI mainframes in computer room A and computer room B. You might therefore have three files to describe these areas: `site.net`, `roomA.net`, and `roomB.net`. If you specify `site.net` as your network file, the monitor will load `site.net`, `roomA.net`, and `roomB.net`. If you specify `roomA.net` as your network file, it will load only that file, and you will be able to monitor only the network described in that file. You can also specify the network file by setting the `SNMP_NETFILE` environment variable.
`Done`	Exits the setup mode and invokes the changes made for this `xsnmpmon` session only. The changes are not written to the `xsnmpmon.rc` file.

`Quit` Button

To exit from `xsnmpmon`, click the `Quit` button. Most of the subwindows have a `Quit` button in the top right corner; when clicked on, the function being performed stops and the window closes.

`System Info` Button

The `System Info` button provides general information about the system (agent) in question. The most important part of this window is the list of network interfaces and their current status. Also shown is the network to which each of the interfaces is attached. At the bottom of the window there is a line indicating how long the agent has been up. This field, along with the status of each of the interfaces, is dynamically updated. If the status of an interface should change, it would show immediately in this window. Information that is new to MIB-II is the information entitled contact, location, name, and services. This information is used to convey the whereabouts of the machine and who the responsible party is. The services available on the system are also provided.

`I/F Stats` Button

The `I/F Stats` button dynamically shows the state of each set of variables associated with each interface within an agent. Click the left or middle mouse buttons to scan through each of the interfaces.

`Media Stats` Button

This function has not been implemented yet. Statistics unique to different types of media such as Ethernet and FDDI (fiber distributed data interface) 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` Button

Dynamically displays the state of variables (statistics) associated with Internet Control Message Protocol (ICMP) packets that enter and leave the agent.

`IP Stats` Button

Dynamically displays the state of variables (statistics) associated with IP packets that enter and leave the agent.

`TCP Stats` Button

Dynamically displays the state of variables (statistics) associated with Transmission Control Protocol (TCP) packets that enter and leave the agent.

`UDP Stats` Button

Dynamically displays the state of variables (statistics) associated with User Datagram Protocol (UDP) packets that enter and leave the agent.

`EGP Stats` Button

Dynamically displays the state of variables (statistics) associated with Exterior Gateway Protocol (EGP) packets that enter and leave the agent.

SNMP Stats Button
Dynamically displays the state of variables (statistics) associated with SNMP packets that enter and leave the agent.

AT Table Button
Displays the `Address Translation Table` window, which shows the binding between the 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 is used (`hyroute`). 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 Button
This 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. 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, this table will replace the `Address Translation` table if there ever is a MIB-III. However, because they are both part of MIB-II, they are both supported.)

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 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 Button
Displays the state of all the TCP connections to 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 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 Button
Displays information about all of the agent's EGP neighbors.

FDDI SMT Button
Displays all of the station management parameters for a given station management (SMT) entity in an FDDI station.

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 Button
Displays all of the port parameters for a given port entity in an FDDI station.

FDDI Attach Button
Displays all of the attachment parameters for a given attachment entity in an FDDI station.

`Error Log` Button
Displays the error log. When the monitor makes an SNMP request to an agent, and the request terminates with abnormal status, the event is logged in the error log. Each error is date- and time-stamped before being placed into the log. The error log keeps the last 50 errors; when the 51st error occurs, it over-writes the first error logged. A marker (>) is placed next to the last error logged to help you quickly locate the newest error.

`Trap Log` Button
Displays all of the acknowledged traps as normal text and the unacknowledged traps as warning text. One of the features of the SNMP protocol is the concept of a *trap*. A trap is an unsolicited message that can be sent from an agent to a client (application) whenever a significant event occurs within the agent. For example, when the agent reboots or when an interface changes states within the agent, a trap message would be sent. Each time a trap message is received by the monitor, it is date- and time-stamped before being placed into the trap log. This display shows the last 50 traps received. Because a trap from an agent can be an event that indicates the need for human intervention, when the trap is received it is marked as unacknowledged. The unacknowledged trap will remain in a warning state until you select the entry by clicking the left mouse button on it, at which time it will become highlighted; you can select multiple traps. When you have a trap selected, 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 of the selected traps. If the trap log fills up with unacknowledged traps, new traps are sent to the log file but are not placed into the trap log.

`State Chg Log` Button
Displays the state change log. Whenever the status of an agent or one of its interfaces changes from one poll cycle to the next, an entry is made in the state change log. The color of the message as it appears in the window will be something other than the normal text color. It will stay this way until you acknowledge it. Select an entry by clicking the left mouse button on it, at which time it will become highlighted; you can select multiple entries. When you have selected the state change entry, 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 of the selected entries. When the state change log fills up, the network scan will be automatically stopped. (This button does not work on monochrome displays.)

`Create Net` Button
Invokes a tools that allows you to create the network files used in the scan process. When you click this button, a subwindow containing several buttons and a map area opens at the top of your screen. To create a network configuration, click the button that describes the object you wish to create, and move the mouse pointer down into the map portion of the window; at this point, the mouse pointer becomes a pencil. Click the left mouse button to create the object. In most cases, the program will ask for some basic information about the object being created (such as a name) by using a dialog window. When you enter the text and press <RETURN>, the dialog window disappears and the object appears in the map. You can place the object wherever you wish by moving it with the mouse. When the object is in the desired position, click the left mouse button to place the object there.

The buttons in the `Create Network Configuration` window, from left to right, are as follows:

Button	**Description**
`Area`	Creates and names an area, which provides a layered effect in the network map. An area can be thought of as a room, building, city, network, and so on. It is completely up to you to decide how to represent your network to the monitor. The figure for an area is a trapezoid. 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 an agent, which is any SNMP-manageable entity. That is, an agent can be a host, gateway, router, or any other box that has an SNMP agent running in it. An agent is represented by a rectangular figure. You are asked to enter the name of the agent in a dialog window. This name must be a name that appears in the `/etc/hosts` file. (If the program cannot resolve the IP address of the agent from the name given, you will be asked to enter the IP address.) Next, you are asked to enter the SNMP community string to be used when conversing with this agent. Finally, you should provide a description of the agent, such as "NSC IP router." 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 an interface. An *interface* connects an agent to a network. On the map, a line or set of line segments represents a physical network interface. To create an interface, the line representing the interface must touch the agent and the network to which the agent is connected. Because the interface can be drawn as a series of line segments, click the left mouse button to start the line and change its direction. To complete the line, click the middle or right mouse button. When your drawing is complete, the monitor automatically searches for the number of the interface; if it cannot find the number, the monitor asks you to enter the number in the dialog box. You can find the number of the interface by connecting to the agent and displaying the `System Information` window. The left-most column shows the interface numbers (or indexes).

`Bus`
: Creates and names a bus. A *bus* is a network topology in which all of the nodes connect to one wire. Examples of bus-type network media are Ethernet and HYPERchannel. A bus is represented by a line or set of line segments. Because the bus can be drawn as a series of line segments, click the left button to start the line and change its direction. 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 a ring. A *ring* is a network topology in which all of the nodes connect together in a closed loop. Examples of ring networks are FDDI and TOKEN Ring. A ring is represented on the map by an ellipse. 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.

`Link`
: Creates and names a link. A *link* is a network topology that involves a local and remote side, usually point-to-point in nature. T1, T3, and satellites are examples of link media. A jagged line (similar to a lightning bolt) represents a link on the map. Clicking the left mouse button starts and ends 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 to a network. To affix the connector on the map, click the left mouse button.

`Label`
: Creates a label, which is a text string that can be placed any where on the network picture. These are essentially comments and should be used to make notes as needed on the network drawing. Click the left mouse button to affix the label to the map.

`Delete`	Deletes objects on the map. To delete, click the `Delete` button, place the pencil cursor inside 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 the `Move` button, place the pencil cursor inside 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; if you do not want to move the object, click the right mouse button to cancel the operation.
`Load`	Loads an existing file in order to modify it, delete it, or copy to a different file name.
`Save`	Saves the map to a disk file for late retrieval.
`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.

`Net Status` Button

Displays a subwindow that contains the following buttons, from left to right:

Button	**Description**
`Start`	Starts the polling sequence from the beginning. Any previous status is cleared.
`Stop`	Stops the polling sequence, but retains the previously obtained status.
`Restart`	Restarts the polling sequence from where it left off (when the `Stop` button was clicked on).
`Load`	Loads a new network configuration.
`Route`	Performs a route trace. To show the route taken to connect the source agent to the destination agent and back again, click the left mouse button on the source and the right mouse button on the destination. 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 to perform a route trace between two specific interfaces for agents with multiple interfaces. When you have performed all of the route traces desired, click `Restart` to resume the network scan from where you left off (the same colors reappear), or click `Start` to start from the beginning.

A color legend is also located at the top of the window. Each agent is polled to determine its status and the status of the defined network interfaces. Although you may modify these colors, the following shows the defaults for each.

Agent status is represented by one of the following colors:

Color	**Description**
White	Untested
Blue	Scanning
Yellow	Reachable by ICMP but not responding to SNMP requests
Red	Not reachable or down
Green	Up

Interface status is represented by one of the following colors:

Color	**Description**
White	Untested
Blue	Scanning
Yellow	Unknown
Magenta	Testing
Orange	Misconfigured
Red	Down
Green	Up

Area status is represented by one of the following colors:

Color	**Description**
White	Untested
Blue	Scanning
Yellow	Problem
Green	OK or unknown

During the network scan process, `xsnmpmon` accesses two files: an action file (`xsnmpmon.act`) and an exception file (`xsnmpmon.xcp`). The `xsnmpmon.act` 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. Shell scripts can be tailored 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 do not support SNMP. This file can also be used to force 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.

During the polling sequence, each agent is sent an SNMP request to determine its status. If the monitor does not get an answer from the SNMP request, it then tries to reach the agent using an ICMP ECHO request. If this is unsuccessful, the agent's status is set to "unreachable." All agents that are set to unreachable are polled each polling cycle, unless the exception file is used (the exception file, `xsnmpmon.xcp`, 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.) If the ICMP request is successful, the status of the agent is set to "ICMP reachable" and that agent will not be polled again. This usually indicates that there is no SNMP agent within that agent or that it is down. When the SNMP agent becomes active, it should send a trap message to the monitor, at which time the agent will be placed back into the polling list.

If during the scan the state change log becomes full of unacknowledged state changes, the scan is stopped automatically. You must acknowledge the state changes and manually restart the scan by clicking on the `Restart` button.

The mouse buttons have actions assigned to them in the network status window also. If you place the cursor inside an agent and click the left mouse button, the monitor attempts to connect the monitor to that agent. (This has the same effect as using the `Agent` button in the `Setup` window). If successful, the main window is raised to the top of the window stack and you can use the other features of the monitor to view variables within the agent.

Use the middle mouse button to traverse the layers of the map. To go inside of an area (that is, to go down a layer), place the cursor inside the area figure and click the middle mouse button. To move up through the layers, place the cursor anywhere except inside an area figure and click the middle mouse button. If you place the cursor inside an agent or area and click the right mouse button, you will get a summary of that object. In the case of agents, the name, community, description, IP address, and list of interfaces will be shown. In the case of an area, the area's name and description is given as well as a summary of the status of all of the sub-areas, agents, and interfaces within that area is given.

When you click the `Route` button, use the left button to specify the source agent and the right button to specify the destination.

The status of an area is meant to represent the status of that entire area. An area is considered to have a problem if any agents within the area are unreachable or reachable only with ICMP or if any interfaces are down.

`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. ICMP turnaround time is the time it takes to get a minimum-size ICMP packet from the monitor station to the connected agent. This information can be helpful in solving problems related to network latency (such as TCP window size and other tuning issues).

`Notification` Button
Displays visual symbols that indicate when human intervention is required. As trap messages are received from the network, this window indicates that there are potential problems 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.

`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.

ENVIRONMENT

Your PATH must contain `/usr/bin/X11` in order to run `xsnmpmon`. Environment variables are checked only when the program is started. The following shell environment variables are used:

`MIBFILE`	Path to the text file used by the client created by Carnegie-Mellon University for the MIB specification. The default is `mib.txt` in the current working directory, or else `/etc/mib.txt`.
`SNMP_BADVARTYP`	When set to equal `ignore`, this tells `xsnmpmon` to ignore variable-type error messages. It is useful when you know that a particular agent is returning bad variable types, and you want to avoid the numerous variable-type error messages that it may send.
`SNMP_ACTIONFILE`	Path name of the action file. The default is `xsnmpmon.act` in the current working directory.
`SNMP_ASSISTFILE`	Path name of the assist file. The default is `xsnmpmon.ast` in the current working directory.

`SNMP_COLORPROG`	Path name of the color palate program. The default is `xnewsdemo.hv`. (When run on a CRI mainframe, this variable is set to `NULL` by default.)
`SNMP_EXCEPTIONFILE`	Path name of the exception file. The default is `xsnmpmon.xcp` in the current working directory.
`SNMP_LOGFILE`	Path name of the log file. The default is `xsnmpmon.log.`*mmddyy* (month, day, year) in the current working directory.
`SNMP_RCFILE`	Path name of the resource (`.rc`) file. The default is `xsnmpmon.rc` in the current working directory.

FILES

`xnewsdemo.hv`	Default color palate program (except when run on a CRI mainframe)
`xsnmpmon.log`	Default log file
`xsnmpmon.rc`	Default resource file
`/etc/hosts`	IP names and addresses of all the hosts on the network
`/etc/networks`	IP names and addresses of all the networks on the network
`/etc/services`	IP port numbers and protocols on the network

BUGS

The sizes of the windows are fixed within the program and cannot be resized by the window manager.

If you move agents or networks within the `Create Net` window, the interfaces to which they were connected are not moved.

If multiple `xsnmpmon` sessions are run on a single station, only the first invoked will receive traps.

EXAMPLES

One example of how the monitor may be used to detect and troubleshoot a network problem is as follows.

The operator invokes the program by entering `xsnmpmon` at a window and starts up the network scan function by clicking on the `Net Status` button. When the network scan function is running, the operator closes this window (leaving the scan running) and opens the `Notification` window by clicking on the `Notification` button. Then the operator iconifies the main window.

As the network scan continues, one of the interfaces of one of the agents changes states from up to down. When this is detected by the monitor, a number of things occur. First, a record of this event is indicated by a message in the logfile (and standard error). Next, the event is logged in the state change log and marked as unacknowledged. When this occurs, the state change event indicator in the event notification window changes from its "off" color (normally green) to its "on" color (normally red), which indicates to the operator that some intervention is required.

When the operator sees this indicator turn on, he or she opens the main window and clicks on the `State Chg Log` button to open the `State Change Log` window. A record of the event is indicated by the greater-than sign (">") and also the color of the message. The message reads as follows:

```
DD/MM/YY HH:MM:SS 128.162.25.3 (hostname) I/F 3 changed from UP to DOWN
```

This message tells the operator that interface number 3 in box 128.162.25.3 has gone down. At this point, the operator should contact the system administrator or network analyst.

The administrator would then click the `Net Status` button to bring up the `Network Status` window so that he or she could find the relevant box in the network diagram. Once found, the administrator would look at all of the other interfaces in that box to see if they are still up. Also, he or she may want to look at interfaces in other boxes connected to the same network to determine what state they are in.

If it looks like this is the only interface that is having a problem, the administrator may want to connect to that box and begin examining other information to see if the reason why this interface has gone down can be determined. By placing the pointer inside of the agent's icon and clicking on the left mouse button, the administrator directs the monitor to attempt to connect to that agent. Once connected, the administrator may open up the interface statistics window and click the left or middle mouse buttons until the window is displaying statistics for interface number 3. Once the proper set of statistics is being displayed, the first thing that the administrator should look at is the administrative status field to see if it says `up` or `down`. If it says `down`, the administrator now knows that the interface went down because someone has taken it down. If the administrative status says `up`, then the administrator knows that the interface was not intentionally taken down but has gone down for some other reason.

At this point, the administrator may want to use the `telnet` command to contact the box to see if he or she can restart the interface manually. To do this, the administrator clicks on the `Telnet` button and logs into the box.

At this point, the monitor's job has been completed; it has detected and notified the system staff that an interface out on the network has gone down and assisted them in determining that the interface either was or was not supposed to be down. When the administrator has corrected the problem, he or she should click the `State Chg Log` button to go back into the state change log and acknowledge the relevant log entry so as not to be alerted by it in the future.

SEE ALSO

`telnet`(1) or `telnet`(1B) for information about the SunOS or UNICOS command, respectively, for the user interface to a remote system using the TELNET protocol
`xterm`(1) for information about the SunOS command for the terminal emulator for the X Window System

OWS-E Operator Workstation Administrator's Guide, publication SG–3079, and *OWS-E Operator Workstation Operator's Guide*, publication SG–3078, for more information about CPU monitors.

The Simple Book, An Introduction to the Management of TCP/IP-based Internets by Marshall T. Rose. Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 1991.

The following Request for Comment (RFC) documentation:
SNMP over Ethernet, RFC 1089
Structure of Management Information, RFC 1155
Management Information Base (MIB-I), RFC 1156 (Obsoleted 1066)
SNMP Protocol, RFC 1157 (Obsoleted 1098)
Management Information Base II (MIB-II), RFC 1158
Bulk Table Retrieval using SNMP, RFC 1187

The following American National Standards Institute (ANSI) documentation:
Fiber Distributed Data Interface (FDDI) Token Ring Physical Layer Medium Dependent (PMD), X3.166–1990
Fiber Distributed Data Interface (FDDI) Token Ring Physical Layer Protocol (PHY), X3.148–1988
Fiber Distributed Data Interface (FDDI) Token Ring Media Access Control (MAC), X3.139–1987
FDDI Station Management, X3T9/90–X3T9.5/84–49 Rev 6.2 (draft)

Internet Draft FDDI Management Information Base (MIB)

NAME

`zip` – Acts as the terminal interface to a running CPU

SYNOPSIS

`/home/`*localhost*`/cri/bin/zip [-C] [-D]`
`/home/`*localhost*`/cri/bin/zip [-D]` [*line*]

DESCRIPTION

The `zip` program works as the terminal interface to a running CPU. You should use the `zip` program as a console and to talk to an IOP if the network is not up.

If you specify `zip` without `-C` or *line*, it will enter command mode, indicated by the `zip>` prompt. If you use *line*, `zip` will perform an `open` built-in command on that line. If you enter `-C`, `zip` will open line `0` (the UNICOS console line). After `zip` establishes a connection with the CPU, it enters input mode. Characters are sent to the CPU for processing and echoed back to the OWS-E.

The arguments to `zip` are as follows:

`-C`	Connects to the UNICOS console. This is equivalent to entering `0` for *line*; if you enter `-C`, you cannot specify a value for *line*.
`-D`	Specifies debug mode for the `zip` program and sends the output to standard error.
line	Specifies the line upon which `zip` performs an `open` built-in command. The value specified for *line* can be an integer from `0` through `4`; `0` specifies the UNICOS console, which is equivalent to entering the `-C` option (you cannot specify both `-C` and *line*).

BUILT-IN COMMANDS

The `zip` program contains the following built-in commands (you are required to enter only the first letter of each):

`c[lose]` Closes a connection to a device.

`d[ebug]` Enables or disables `zip` debug statements.

`e[scape]` Changes the escape sequence used in `zip`. When you enter this built-in command, you are prompted for a new escape character sequence.

> **NOTE:** The default escape sequence is `^]` (that is, press the `<CONTROL>` key and then the `]` key). Because this is the same as the `telnet` default escape sequence (`^]`), you cannot escape from `zip` without destroying your `telnet` session unless you change the `zip` sequence with `escape`.

`o[pen]` [*line*] Opens a connection to the specified line. *line* can be set to an integer; `0` specifies the UNICOS console. If you do not enter *line*, `zip` will find the first line available, beginning with line `1`; if all lines are busy, it will then check the console line (line `0`). If you do not request a specific line and all lines are busy, the line held by a user with a lower priority than yours will be usurped if usurp mode has been toggled on. If you request a specific line and that line is busy, the line will be usurped from the present owner if he or she has a priority lower than yours and if usurp mode has been toggled on. The location of the line-arbitration priority file is specified by the `LAPFILE` parameter in `/etc/configfile`; by default, `LAPFILE` is set to `/etc/lapfile`.

`q[uit]` Closes any existing connections and exits `zip`.

`r[oute]` *cluster*	Changes the route through an alternative cluster; the range of valid cluster numbers depends on the number of clusters in the IOS-E. If you do not enter a value for *cluster*, you will be prompted for one.
`s[tatus]`	Displays the line to which you are connected, the route, and the escape sequence.
`u[surp]` [*reason*]	Toggles usurp mode. If usurp mode is turned on when you execute the `open` built-in command, `zip` will usurp (take away) a `tty` line currently being used by someone else if you have a higher priority level (specified in the line-arbitration priority file) than that person. The location of the line-arbitration priority file is specified by the `LAPFILE` parameter in `/etc/configfile`; by default, `LAPFILE` is set to `/etc/lapfile`. The higher the number specified, the higher the priority of the user. By default, all users have a priority of 0; that is, no priority, which means that they cannot usurp a `tty` line. *reason* is the reason the line is being usurped; this reason is sent to the usurped user. Quotation marks are optional, even if the reason contains white space. *reason* is truncated after 79 characters, not including quotation marks (a longer reason will not cause an error, but the 80th and succeeding characters will not be used). If a `tty` line that you are using is usurped by another user, you can try to borrow it again when the usurper has finished with it.
`z`	Suspends a `zip` process. There can be only one process running in a device at a time; if you suspend a process with `z`, no one else can access the device.
`?`	Displays help for `zip`.

Permission to access this built-in command is set in `/etc/owsepermfile` by the system administrator.

EXAMPLES

The following example opens a connection through line 3 to the CPU:

```
zip 3
```

The following example shows a `zip` session (the bold text indicates user input).

```
machine% zip
zip>  status

NOT connected to mainframe
Routed via mux, cluster 0
Escape character is '^]'

zip> escape
new escape character: <control>^     ##Press the control key
Escape character is '^['
zip> open 2

Connected to mainframe via 2
Routed via mux, cluster 0
Escape character is '^['

login: ^[                            ##To escape back to zip
zip> close                           ##To close the connection
Connection closed
zip> status

NOT connected to mainframe
Routed via mux, cluster 0
Escape character is '^[^'

zip> quit
machine%
```

CONFIGURATION FILE PARAMETERS

The zip command reads the following parameters from /etc/configfile:

LAPFILE	Specifies the location of the line-arbitration priority file used by lapdaemon(8). Default: /etc/lapfile

FILES

/etc/owsepermfile	Permissions file that contains a list of accounts and the commands they are allowed to access
/etc/lapfile	Default line arbitration priority file
/etc/configfile	Default configuration file

SEE ALSO

telnet(1) for information about the SunOS command that invokes the user interface to a remote system, using the TELNET protocol
configfile(5) and *OWS-E Operator Workstation Administrator's Guide*, publication SG–3079, for information about /etc/configfile
owsepermfile(5) for information about the default OWS-E permission file
lapdaemon(8) for information about the line arbitration priority daemon

Index

FUNCTIONAL INDEX

Acts as the terminal interface to a running CPU	Acts as the terminal interface to a running CPU	zip(8)	127
An overview of the OWS-E commands	An overview of the OWS-E commands	owse_overview(7)	23
Arbitration file	Default line arbitration priority file	lapfile(5)	19
autodump	Controls the automatic dumping of the mainframe and the IOS-E	autodump(8)	25
Automatic dumping	Controls the automatic dumping of the mainframe and the IOS-E	autodump(8)	25
Backing up log files	Creates new errlog and ioplog files while backing up the existing ones	newlog(8)	105
Backups	Performs a full (level-0) backup of the OWS-E file systems	dumpwkly(8)	41
Backups	Performs an incremental (level-9) backup of the OWS-E file systems	dumpdly(8)	37
Boots deadstart diagnostic tests (dsdiag or cleario) into one or more specified IOPs	Boots deadstart diagnostic tests (dsdiag or cleario) into one or more specified IOPs	ediag(8)	61
Boots one or more IOPs from the OWS-E	Boots one or more IOPs from the OWS-E	eboot(8)	42
Boots the IOS-E and the mainframe	Boots the IOS-E and the mainframe	bootsys(8)	26
bootsys	Boots the IOS-E and the mainframe	bootsys(8)	26
Checks IOP status	Checks IOP status	estat(8)	74
Commands overview	An overview of the OWS-E commands	owse_overview(7)	23
Complete backups	Performs a full (level-0) backup of the OWS-E file systems	dumpwkly(8)	41
Configuration file	Default OWS-E configuration file	configfile(5)	14
Configures a MUXIOP high-speed channel up or down	Configures a MUXIOP high-speed channel up or down	hcon(8)	92
Configures a MUXIOP-to-EIOP low-speed channel up or down	Configures a MUXIOP-to-EIOP low-speed channel up or down	econ(8)	45
Console	Acts as the terminal interface to a running CPU	zip(8)	127
Controls the automatic dumping of the mainframe and the IOS-E	Controls the automatic dumping of the mainframe and the IOS-E	autodump(8)	25
Controls the fy driver	Controls the fy driver	fyadmin(8)	77
conv	Converts a file from old edump file format to new edump file format	conv(8)	30
Converts a file from old edump file format to new edump file format	Converts a file from old edump file format to new edump file format	conv(8)	30
CPU register dumping	Dumps the mainframe memory and CPU registers to a CRI disk on the IOS-E	mfdump(8)	97
CPU time statistics	Displays CPU time statistics in graphical form	graphs(8)	88
CPU time statistics	Gathers data and disperses CPU time statistics	cpud(8)	31
cpud	Gathers data and disperses CPU time statistics	cpud(8)	31
cpudump	Forces a UNICOS dump	cpudump(8)	32
cpuhalt	Issues a CPU master clear to stop the mainframe	cpuhalt(8)	33
cpupanic	Takes a UNICOS panic dump	cpupanic(8)	34
craymon	Monitors the mainframe status and sets the OWS-E background color	craymon(8)	35
Creates new errlog and ioplog files while backing up the existing ones	Creates new errlog and ioplog files while backing up the existing ones	newlog(8)	105
Daily backups	Performs an incremental (level-9) backup of the OWS-E file systems	dumpdly(8)	37
Deadstart	Boots one or more IOPs from the OWS-E	eboot(8)	42
Deadstart	Starts the mainframe CPUs from the OWS-E	mfstart(8)	103

Halts one or more IOPs from the OWS-E	Halts one or more IOPs from the OWS-E	ehalt(8)	68
Hardware error logging daemon	IOS-E hardware error-logging daemon	errlogd(8)	73
hbeat	Monitors the IOS-E system	hbeat(8)	90
hcon	Configures a MUXIOP high-speed channel up or down	hcon(8)	92
Heartbeat monitor (restarting)	Restarts the IOS-E error-logging, heartbeat, SMARTE, and CPU monitors	emon(8)	70
Incremental backups	Performs an incremental (level-9) backup of the OWS-E file systems	dumpdly(8)	37
Interface Routing table (setting)	Sets or displays the fy driver's IP Interface Routing table	fyroute(8)	84
Internet address mapping	Sets or displays the fy driver's IP Interface Routing table	fyroute(8)	84
Invokes the SNMP network monitor	Invokes the SNMP network monitor	xsnmpmon(8)	115
IOP booting	Boots one or more IOPs from the OWS-E	eboot(8)	42
IOP dump	Dumps an IOP in the event of an IOP failure	iophalt(8)	94
IOP failure	Dumps an IOP in the event of an IOP failure	iophalt(8)	94
IOP halting	Halts one or more IOPs from the OWS-E	ehalt(8)	68
iophalt	Dumps an IOP in the event of an IOP failure	iophalt(8)	94
IOS olebpw	Boots deadstart diagnostic tests (dsdiag or beario) into one or more specified IOPs	ediag(8)	61
IOS olelmr	Boots deadstart diagnostic tests (dsdiag or beario) into one or more specified IOPs	ediag(8)	61
IOS olelmw	Boots deadstart diagnostic tests (dsdiag or beario) into one or more specified IOPs	ediag(8)	61
IOS olemal	Boots deadstart diagnostic tests (dsdiag or beario) into one or more specified IOPs	ediag(8)	61
IOS oleoff	Boots deadstart diagnostic tests (dsdiag or beario) into one or more specified IOPs	ediag(8)	61
IOS oleorr	Boots deadstart diagnostic tests (dsdiag or beario) into one or more specified IOPs	ediag(8)	61
IOS oleorw	Boots deadstart diagnostic tests (dsdiag or beario) into one or more specified IOPs	ediag(8)	61
IOS status	Checks IOP status	estat(8)	74
IOS-E booting	Boots the IOS-E and the mainframe	bootsys(8)	26
IOS-E dumping (automatic)	Controls the automatic dumping of the mainframe and the IOS-E	autodump(8)	25
IOS-E error-logging monitor (restarting)	Restarts the IOS-E error-logging, heartbeat, SMARTE, and CPU monitors	emon(8)	70
IOS-E hardware error-logging daemon	IOS-E hardware error-logging daemon	errlogd(8)	73
IOS-E initialization and confidence test	Runs a mainframe and IOS-E initialization and confidence test	mfinit(8)	101
IOS-E local memory dumping	Dumps IOS-E IOP local memory images to the OWS-E	edump(8)	64
IOS-E monitor	Monitors the IOS-E system	hbeat(8)	90
Issues a CPU master clear to stop the mainframe	Issues a CPU master clear to stop the mainframe	cpuhalt(8)	33
lapdaemon	Validates CRI tty lines for users	lapdaemon(8)	96
Line arbitration file	Default line arbitration priority file	lapfile(5)	19
Line-arbitration priority daemon	Validates CRI tty lines for users	lapdaemon(8)	96
Looking at memory	Peeks (looks) at memory	peek(8)	109
Mainframe booting	Boots the IOS-E and the mainframe	bootsys(8)	26
Mainframe dumping (automatic)	Controls the automatic dumping of the mainframe and the IOS-E	autodump(8)	25
Mainframe initialization and confidence test	Runs a mainframe and IOS-E initialization and confidence test	mfinit(8)	101
Mainframe memory dumping	Dumps the mainframe memory and CPU registers to a CRI disk on the IOS-E	mfdump(8)	97
Mainframe starting	Starts the mainframe CPUs from the OWS-E	mfstart(8)	103
Mapping internet addresses	Sets or displays the fy driver's IP Interface Routing table	fyroute(8)	84

Reader's Comment Form

OWS-E Operator Workstation Reference Manual SR–3077 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

Reader's Comment Form

OWS-E Operator Workstation Reference Manual SR–3077 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