UNICOS® Administrator Commands Reference Manual

SR–2022 10.0

Portions of this product may still be in development. The existence of those portions still in development is not a commitment of actual release or support by Cray Research, Inc. Cray Research, Inc. assumes no liability for any damages resulting from attempts to use any functionality or documentation not officially released and supported. If it is released, the final form and the time of official release and start of support is at the discretion of Cray Research, Inc.

New Features

This revision of the *UNICOS Administrator Commands Reference Manual*, Cray Research publication SR–2022, supports the 10.0 release of the UNICOS operating system.

The following command has been removed because it was specific to systems using the IOS model D:
`fdmp`(8)

Accounting commands. The socket accounting feature contains the new `csasocket`(8) command to process socket accounting data.

The following accounting commands have been updated:

`ckdacct`(8), `csaswitch`(8), `shutacct`(8), `startup`(8), `turndacct`(8)	Updated for the socket accounting feature, which tracks network usage from the perspective of sockets.
`csagcon`(8)	Has new `-E` and `-I` options that generate unconsolidated output. By default, the `csagcon`(8) command consolidates accounting data for session and `pacct` files, making per-process information no longer available. These new options generate unconsolidated output, enabling an administrator to select data for individual processes.

Array sessions feature commands. The array sessions feature supports the new `arrayd`(8) command.

File system quota commands. The `qudu`(8) and `quadmin`(8) commands were updated for the optional aggregate quota feature.

Network commands. The `named`(8) command was updated for the socket accounting feature, which tracks network usage from the perspective of sockets.

System activity monitoring (SAM) commands. The `-z` option of the `diskusg`(8) command was updated (if a user does not have a primary account ID, then the account ID of 0 is used and always reported).

Tape subsystem commands. The following tape subsystem commands are new:

`tpcore`(8)	Initiates an interactive monitor that provides continuous information about a running tape subsystem
`tpinit`(8)	Provides a means of initializing the tape subsystem
`xtpldr`(8)	Manages tape cartridges on autoloaders

The following tape subsystem commands contain changes:

`tpset`(8)	Provides information about overcommitted mount requests.
`tpconfig`(8)	Supports two new options. The `-n` option enables a user to disable automatic unloading of a volume when a tape that will be used repeatedly is released. The `-c` option supports a 6-digit number of GigaRing based systems.
`tpdaemon`(8)	The new `-b` command option instructs the tape daemon to bypass all configuration tasks and to use the existing tape configuration.
`tpdev`(8)	Uses the 6-digit number supported on GigaRing based systems in its updated display.
`tpgstat`(8)	Adds the `-a` option to provide status information for device groups that are reserved or active.

The tape subsystem command, `tpconvert`(8), is no longer supported.

Network monitor feature. The network monitor feature is no longer supported; so the following commands are not available: `dsa`(8), `dstrunk`(8), `dtnl130`(8), `dtu`(8), `dxmon`(8), `logn130`(8), `netstatd`(8), `v130`(8), `xdevmon`(8), `xdsa`(8), `xdsn130`(8), `xdstrunk`(8), `xdtu`(8), `xdxmon`(8), `xlogmon`(8), `xnetmon`(8), `xns`(8), `xnx`(8), `xdst3`(8), `xscc`(8), and `xsnmpmon`(8).

Record of Revision

Version	*Description*
2.0	September 1986 Original Printing.
3.0	June 1987 Revision to support the UNICOS 3.0 release.
4.0	July 1988 Revision to support the UNICOS 4.0 release.
5.0	March 1989 Revision to support the UNICOS 5.0 release.
6.0	January 1991 Revision to support the UNICOS 6.0 release.
7.0	September 1992 Revision to support the UNICOS 7.0 release.
8.0	January 1994 Revision to support the UNICOS 8.0 release.
9.0	September 1995 Revision to support the UNICOS 9.0 release.
10.0	November 1997 Revision to support the UNICOS 10.0 release.

Preface

This publication documents UNICOS release 10.0 running on Cray PVP systems. It provides descriptions of commands used in system administration of the Cray Research UNICOS operating system. It supplements the information contained in other manuals of the UNICOS documentation set.

This manual describes commands and shell procedures that are invoked directly by the super user or by command language procedures. It contains system maintenance procedures that generally reside in the `/etc` directory, which are searched by the command interpreter called the shell (see `sh`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011).

This is a reference manual for system administrators. Readers should have a working knowledge of either the UNICOS or the UNIX operating system.

Related publications

The following man page manuals contain additional information that may be helpful.

Note: For the UNICOS 10.0 release, man page reference manuals are not orderable in printed book form. Instead, they are available as printable PostScript files provided on the same DynaWeb CD as the rest of the supporting documents for this release. Individual man pages are still available online and can be accessed by using the `man`(1) command.

- *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011
- *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012
- *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014
- *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

The following ready references are available in printed form from the Distribution Center:

- *UNICOS User Commands Ready Reference*, Cray Research publication SQ–2056
- *UNICOS System Libraries Ready Reference*, Cray Research publication SQ–2147

- *UNICOS System Calls Ready Reference*, Cray Research publication SQ–2215
- *UNICOS Administrator Commands Ready Reference*, Cray Research publication SQ–2413

The following manuals contain additional information that may be helpful:

- *TCP/IP Network User's Guide*, Cray Research publication SG–2009
- *UNICOS Text Editors Primer*, Cray Research publication SG–2050
- *Tape Subsystem User's Guide*, Cray Research publication SG–2051
- *UNICOS vi Reference Card*, Cray Research publication SQ–2054
- *UNICOS ed Reference Card*, Cray Research publication SQ–2055
- *Scientific Libraries Reference Manual*, Cray Research publication SR–2081
- *Remote Procedure Call (RPC) Reference Manual*, Cray Research publication SR–2089
- *UNICOS Shells Ready Reference*, Cray Research publication SQ–2116
- *UNICOS Environment Variables Ready Reference*, Cray Research publication SQ–2117
- *Intrinsic Procedures Reference Manual*, Cray Research publication SR–2138
- *NQE Administration*, Cray Research publication SG–2150
- *General UNICOS System Administration*, Cray Research publication SG–2301
- *UNICOS Resource Administration*, Cray Research publication SG–2302
- *UNICOS Configuration Administrator's Guide*, Cray Research publication SG–2303
- *UNICOS Networking Facilities Administrator's Guide*, Cray Research publication SG–2304
- *Kerberos Administrator's Guide*, Cray Research publication SG–2306
- *Tape Subsystem Administration*, Cray Research publication SG–2307
- *CF90 Ready Reference*, Cray Research publication SQ–3900
- *CF90 Commands and Directives Reference Manual*, Cray Research publication SR–3901

- *Fortran Language Reference Manual, Volume 1*, Cray Research publication SR–3902
- *Fortran Language Reference Manual, Volume 2*, Cray Research publication SR–3903

Ordering Cray Research publications

The *User Publications Catalog*, Cray Research publication CP–0099, describes the availability and content of all Cray Research hardware and software documents that are available to customers. Cray Research customers who subscribe to the Cray Inform (CRInform) program can access this information on the CRInform system.

To order a document, either call the Distribution Center in Mendota Heights, Minnesota, at +1–612–683–5907, or send a facsimile of your request to fax number +1–612–452–0141. Cray Research employees may send electronic mail to `orderdsk` (UNIX system users).

Customers who subscribe to the CRInform program can order software release packages electronically by using the `Order Cray Software` option.

Customers outside of the United States and Canada should contact their local service organization for ordering and documentation information.

Conventions

The following conventions are used throughout this document:

Convention	Meaning
`command`	This fixed-space font denotes literal items such as commands, files, routines, path names, signals, messages, and programming language structures.
`manpage`(*x*)	Man page section identifiers appear in parentheses after man page names. The following list describes the identifiers:

1	User commands
1B	User commands ported from BSD
2	System calls

	3	Library routines, macros, and opdefs
	4	Devices (special files)
	4P	Protocols
	5	File formats
	7	Miscellaneous topics
	7D	DWB-related information
	8	Administrator commands

Some internal routines (for example, the `_assign_asgcmd_info`() routine) do not have man pages associated with them.

variable	Italic typeface denotes variable entries and words or concepts being defined.
`user input`	This bold, fixed-space font denotes literal items that the user enters in interactive sessions. Output is shown in nonbold, fixed-space font.
[ ]	Brackets enclose optional portions of a command or directive line.
...	Ellipses indicate that a preceding element can be repeated.

The following machine naming conventions may be used throughout this document:

Term	Definition
Cray PVP systems	All configurations of Cray parallel vector processing (PVP) systems.
Cray MPP systems	All configurations of the CRAY T3D series. The UNICOS operating system is not supported on CRAY T3E systems. CRAY T3E systems run the UNICOS/mk operating system.
All Cray Research systems	All configurations of Cray PVP and Cray MPP systems that support this release.

The default shell in the UNICOS and UNICOS/mk operating systems, referred to in Cray Research documentation as the *standard shell*, is a version of the Korn shell that conforms to the following standards:

- Institute of Electrical and Electronics Engineers (IEEE) Portable Operating System Interface (POSIX) Standard 1003.2–1992
- X/Open Portability Guide, Issue 4 (XPG4)

The UNICOS and UNICOS/mk operating systems also support the optional use of the C shell.

Cray UNICOS Version 10.0 is an X/Open Base 95 branded product.

Man page sections

The entries in this document 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.

Section heading	Description
NAME	Specifies the name of the entry and briefly states its function.
SYNOPSIS	Presents the syntax of the entry.
IMPLEMENTATION	Identifies the Cray Research systems to which the entry applies.
STANDARDS	Provides information about the portability of a utility or routine.
DESCRIPTION	Discusses the entry in detail.
NOTES	Presents 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.
ENVIRONMENT VARIABLES	Describes predefined shell variables that determine some characteristics of the shell or that affect the behavior of some programs, commands, or utilities.
RETURN VALUES	Describes possible return values that indicate a library or system call executed successfully, or identifies the error condition under which it failed.

EXIT STATUS	Describes possible exit status values that indicate whether the command or utility executed successfully.
MESSAGES	Describes informational, diagnostic, and error messages that may appear. Self-explanatory messages are not listed.
ERRORS	Documents error codes. Applies only to system calls.
FORTRAN EXTENSIONS	Describes how to call a system call from Fortran. Applies only to system calls.
BUGS	Indicates known bugs and deficiencies.
EXAMPLES	Shows examples of usage.
FILES	Lists files that are either part of the entry or are related to it.
SEE ALSO	Lists entries and publications that contain related information.

Reader comments

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

- Send us electronic mail at the following address:

  `publications@cray.com`

- Contact your customer service representative and ask that an SPR or PV be filed. If filing an SPR, use `PUBLICATIONS` for the group name, `PUBS` for the command, and `NO-LICENSE` for the release name.

- Call our Software Publications Group in Eagan, Minnesota, through the Customer Service Call Center, using either of the following numbers:

  1–800–950–2729 (toll free from the United States and Canada)

  +1–612–683–5600

- Send a facsimile of your comments to the attention of "Software Publications Group" in Eagan, Minnesota, at fax number +1–612–683–5599.

We value your comments and will respond to them promptly.

CONTENTS

NAME

`intro` – Introduces system maintenance commands, network maintenance and operation commands, and application programs that invoke shell procedures

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

This section describes, in alphabetical order, commands that are used mainly for system maintenance and administration purposes for all Cray Research systems.

The following terms identify UNICOS command components:

Command Component	Definition
Command	Name of an executable file.
Option	Command-line element indicated by a hyphen, followed by a letter.
Option-argument	Character string that supplies information for the preceding option.
Operand	Command-line element to be passed to the command; not associated with an option.

Items enclosed in square brackets, [ ], are optional. *White space* refers to any number of horizontal spaces or tabs.

For a more detailed description of conventions, see *UNICOS Command Conventions*, Cray Research publication CP–2058.

EXIT STATUS

On termination, each command returns 2 bytes of status; one supplied by the system and giving the cause for termination, and (in the case of "normal" termination) one supplied by the procedure (see `wait`(2) and `exit`(2)). The former byte is 0, indicating normal termination; the latter is usually 0, indicating successful execution, and nonzero indicating troubles such as erroneous parameters, bad or inaccessible data, or other inability to cope with the task at hand. It is called variously exit code, exit status, or return status, and is described only where special conventions are involved.

BUGS

Many commands do not use the aforementioned syntax.

SEE ALSO

`getopt`(1), `getopts`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`getopt`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

UNICOS Command Conventions, Cray Research publication CP–2058

NAME

acct – Overview of standard UNIX System V accounting commands

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The UNICOS operating system supports two accounting packages, Cray Research system accounting (CSA) and standard UNIX System V accounting. The standard UNIX accounting package is a set of C programs and shell scripts that provides methods for collecting resource use data per process, recording connect sessions, monitoring disk usage, and charging fees to specific logins. This man page describes the accounting commands used by either accounting system. The acctsh(8) man page describes the shell scripts used by either accounting system.

The UNICOS kernel performs process accounting. On termination of a process, one record per process is written to a file, usually /usr/adm/acct/day/pacct. The acctprc1 and acctprc2 commands (see acctprc(8)) summarize this data for charging purposes; the acctcms(8) command summarizes command usage, and the acctcom(1) command reports current process data.

Various programs handle connect-time accounting by writing records into the /etc/wtmp file, as described in utmp(5). The programs described on the acctcon(8) man page convert this file into login session and charging records, which can then be summarized by the acctmerg(8) command.

Process accounting, connect-time accounting, and any accounting records in the format described on the acct(5) man page can be merged and summarized into total accounting records by acctmerg (see the acct(5) for the tacct format). The prtacct(8) command formats or prints any tacct accounting records.

FILES

/etc/udb	User validation file that contains user control limits and contains login name to user ID conversions.
/etc/wtmp	Contains login and logoff history information.
/usr/adm/acct/day/pacct	Contains current process accounting information.
/usr/lib/acct	Contains most of the accounting commands listed in this manual.

SEE ALSO

acctcms(8), acctcon(8), acctdisk(8), acctdusg(8), acctmerg(8), accton(8), acctprc(8), acctsh(8), acctwtmp(8), csa(8), diskusg(8), dodisk(8), fwtmp(8), lastlogin(8), monacct(8), nulladm(8), prctmp(8), prdaily(8), prtacct(8), remove(8), runacct(8), shutacct(8), startup(8), turnacct(8)

acctcom(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

acct(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

acct(5), utmp(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

acctcms – Summarizes command usage from per-process accounting records

SYNOPSIS

/usr/lib/acct/acctcms [-a [[[-p] [-o]] [-e]]] [-c] [-j] [-n] [-s] [-S [-A]] *files*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The acctcms command reads one or more *files*, usually in the format described in acct(5). It adds all records for processes that executed identically named commands, and it sorts and writes them to the standard output, usually using an internal summary format. acctcms accepts the following options:

-a Prints output in ASCII rather than in the internal summary format. You can use the following options only with the -a option:

- -p Outputs a prime-time-only command summary.
- -o Outputs a nonprime-time-only (offshift) command summary.
- -e Outputs an extended report, printing additional fields. You can use the -e option only when the -p or -o options also are selected with the -a option.

  The default output produced with the -a option includes command name, number of times executed, total kcore-minutes, total CPU minutes, total real minutes, mean size (in K), mean CPU minutes per invocation, the amount of CPU resources (proportional to other processes) used, characters transferred, and blocks read and written, as in acctcom(1). Usually, the output is sorted by total kcore-minutes.

  When you specify both -p and -o with -a, acctcms produces a combination prime and nonprime time report.

  All output summaries indicate octal usage except number of times executed, CPU minutes, and real minutes, which are split into prime and nonprime.

-c Sorts by total CPU time rather than total kcore-minutes.

-j Combines all commands invoked only once under ‘‘***other’’.

-n Sorts by number of command invocations.

-s Indicates that any file names encountered hereafter are already in internal summary format.

-S Indicates that the Session record format is used on input.

-A Causes all jobs (even nonterminated sessions) to be considered. You must use this option with the -S option.

EXAMPLES

A typical sequence for performing daily command accounting and for maintaining a running total is as follows:

```
acctcms file . . . >today
cp total previoustotal
acctcms -s today previoustotal >total
acctcms -a -s today
```

SEE ALSO

acct(8), acctcon(8), acctmerg(8), acctprc(8), acctsh(8), fwtmp(8), runacct(8)

acctcom(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

acct(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

acct(5), utmp(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

acctcon1, acctcon2 – Performs connect-time accounting

SYNOPSIS

/usr/lib/acct/acctcon1 [-l *file*] [-o *file*] [-p] [-t]

/usr/lib/acct/acctcon2

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The acctcon1 command converts a sequence of login and logoff records read from its standard input to a sequence of records, one per login session. Usually, its input should be redirected from the /etc/wtmp file. Its output is ASCII, and it specifies device, user ID, login name, prime connect time (seconds), nonprime connect time (seconds), session starting time (numeric), and starting date and time.

The acctcon1 command maintains a list of lines cataloging which users are logged in. When it reaches the end of its input, it emits a session record for each line that still appears to be active. Usually, it assumes that its input is a current file; therefore, it uses the current time as the ending time for each session still in progress.

The acctcon1 command accepts the following options:

-l *file* Creates *file* to contain a summary of line usage that shows line name, number of minutes used, percentage of total elapsed time used, number of sessions charged, number of logins, and number of logoffs. This file helps track line usage, identify bad lines, and find software and hardware inconsistencies and errors. Hang-up, termination of login(1), and termination of the login shell each generate log-off records; therefore, the number of logoffs is often three to four times the number of sessions. See init(8) and utmp(5).

-o *file* Fills *file* with an overall record for the accounting period, giving starting time, ending time, number of reboots, and number of date changes.

-p Prints input only, showing line name, login name, and time (in both numeric and date/time formats).

-t Uses, rather then the current time, the last time found in its input as the ending time for active sessions. This ensures reasonable and repeatable numbers for noncurrent files.

acctcon2 expects as input a sequence of login session records and converts them into total accounting records (see tacct format in acct(5)).

BUGS

The line-usage report is confused by date changes. Use `wtmpfix` (see `fwtmp`(8)) to correct this situation.

EXAMPLES

Typically, these commands are used as follows (the `ctmp` file is created only for the use of `acctprc`(8) commands):

```
acctcon1 -t -l lineuse -o reboots <wtmp | sort  +1n  +2 >ctmp
acctcon2 <ctmp | acctmerg >ctacct
```

FILES

`/etc/wtmp`

SEE ALSO

`acct`(8), `acctcms`(8), `acctmerg`(8), `acctprc`(8), `acctsh`(8), `fwtmp`(8), `init`(8), `runacct`(8)

`acctcom`(1), `login`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`acct`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

`acct`(5), `utmp`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

acctdisk – Converts disk data to cacct or tacct format

SYNOPSIS

/usr/lib/acct/acctdisk [-a] [-A]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

By default, the acctdisk command reads standard input and converts records to tacct format, which it writes to standard output. (See acct(5) for the format.) Each input record contains a user ID, a login name, and the number of disk blocks allocated. The input file is generally the default ouput from the diskusg(8) command. You then can merge the tacct records with other tacct records by using the acctmerg(8) command.

The -a and -A options convert input records to cacct format, which is used by the Cray Research system accounting (CSA) feature. These records can be merged with other cacct records by using the csaaddc(8) command.

The acctdisk command accepts the following options:

-a Accepts as input the output produced by the diskusg(8) command specified with the -a option and produces output in cacct format.

-A Accepts as input the output produced by the diskusg(8) command specified with the -A output and produces output in cacct format.

EXAMPLES

The following example displays the conversion of diskusg(8) output to tacct format:

```
/usr/lib/acct/diskusg /dev/dsk/tmp | /usr/lib/acct/acctdisk > tacctfile
```

The following example displays the conversion of the contents of the diskdata file, the output of diskusg -a /dev/dsk/tmp, to cacct format:

```
/usr/lib/acct/acctdisk -a < diskdata > cacctfile
```

FILES

/etc/udb User validation file that contains user control limits; used for user information.

SEE ALSO

`acct`(8), `acctmerg`(8), `acctsh`(8), `csa`(8), `csaaddc`(8), `csaperiod`(8), `diskusg`(8), `runacct`(8)

`acct`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`acctdusg` – Computes and displays disk resource consumption by login

SYNOPSIS

`/usr/lib/acct/acctdusg` [`-p` *path*] [`-u` *file*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `acctdusg` command reads its standard input, usually from the `find / -print` command, and it computes disk resource consumption, including indirect blocks, by login. Output is written to standard output.

The `acctdusg` command accepts the following options:

`-p` *path* Specifies the path name of the user database (UDB) file. By default, the UDB file is defined as `/etc/udb`.

`-u` *file* Writes the names of files that have not been charged to anyone to the specified *file*. This information can help you identify users who are trying to avoid disk charges.

EXAMPLES

In the following example, `acctdusg` displays disk resource consumption by login:

```
find / -print | /usr/lib/acct/acctdusg > dusgdata
```

FILES

`/etc/udb` User validation file that contains user control limits; used for user information.

SEE ALSO

`acct`(8), `acctsh`(8)

`find`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`udb`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

acctmerg – Merges or adds total accounting files

SYNOPSIS

/usr/lib/acct/acctmerg [-a [-b] [-c] [-d] [-f] [-h] [-j] [-m] [-M] [-n] [-w] [-x] [-y]] [-i] [-p] [-s] [-t] [-v] [*files*]

/usr/lib/acct/acctmerg [-a [-b] [-c] [-d] [-f] [-h] [-j] [-m] [-M] [-n] [-w] [-x] [-y]] [-i] [-p] [-s] [-u] [-v] [*files*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The acctmerg command reads its standard input and additional files, all in the tacct format (see acct(5)) or an ASCII version thereof. It merges these inputs by adding records with keys (usually, user ID, login name, and account name) that are identical, and it expects the inputs to be sorted on those keys.

The acctmerg command accepts the following options:

-a Produces output in an ASCII version of tacct. The following options are valid only if you specify the -a option. They specify a subset of the total information and allow a flexible control of the resulting output format. If you specify any of the following options, each output line includes account name, user ID, and login name:

- -b Specifies system billing information (SBUs) and operational fees.
- -c Specifies CPU time memory integral and connect time in seconds.
- -d Specifies cumulative disk-block-usage count and number of disk samples taken.
- -f Provides full information about each entry. The output format consists of several lines per entry and provides all of the preceding information plus additional data about device-specific I/O (if available).
- -h Writes information header for all requested fields.
- -j Specifies number of processes and number of jobs.
- -m Specifies user CPU time breakdown for multiple CPUs running in parallel (multitasked processes only).
- -M (Cray MPP systems only) Specifies Cray MPP information.
- -n Splits data into prime and nonprime time data (two lines of output for each entry).
- -w Specifies I/O wait time and I/O wait-time memory integral in seconds.
- -x Specifies number of blocks transferred; real and logical I/O request counts.

-y Specifies number of SDS blocks transferred.

-i Specifies that input files are in an ASCII version of tacct.

-p Prints input without processing.

-s Summarizes by account name rather than user ID, login name, and account name (cannot be used with the -u option). Input that was created by using the acctprc2 -s option (see acctprc(8)) must be processed with this option.

-t Produces one record that totals all input.

-u Provides summary by user IDs rather than user ID, login name, and account name (cannot be used with the -s option).

-v Produces output in verbose ASCII format, with more precise notation for floating-point numbers.

NOTES

When using the -v or -a options, several fields within tacct records are printed out in a slightly different order than is defined by the tacct structure. Instead of printing ta_dc, ta_pc, and ta_sc, the order is switched to ta_pc, ta_sc, and ta_dc. If records are read in using the -i (ASCII input) option, the expected order of input differs from the tacct structure in exactly the same manner.

EXAMPLES

The following sequence is useful for making corrections to any file kept in this format:

```
acctmerg -v <file1 >file2
      edit file2 as desired . . .
acctmerg -i <file2 >file1
```

The following example merges three input files, ifile0, ifile1, and ifile2; the output file is ofile:

```
acctmerg ifile1 ifile2 < ifile0 > ofile
```

SEE ALSO

acct(8), acctcms(8), acctcon(8), acctprc(8), acctsh(8), fwtmp(8), runacct(8)

acctcom(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

acct(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

utmp(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

accton – Controls process accounting

SYNOPSIS

/usr/lib/acct/accton [*file*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The accton command specified without operands turns off process accounting. To turn on process accounting, execute the accton command with the following operand:

file Name of the *file* to which the kernel appends process accounting records. (See acct(2) and acct(5)).

If the specified *file* does not exist, accton creates it and properly sets the owner group and mode of the file.

The super user or a user who is in the group adm and has permission bit acct set in their user database (UDB) entry (see udbgen(8)) must invoke accton.

NOTES

The accton command is rarely invoked alone. Use the turnacct(8) command to enable and disable processing accounting.

Sites may allow users in the group adm who have the permission bit acct set in their UDB entries to run Cray Research system accounting (CSA). However, such users cannot run accounting after a super user has done so, because the group ID and permissions of the files will have changed. In this case, the csaperm(8) command must be executed to reset group IDs and permissions before nonsuper users can run accounting.

FILES

/etc/udb	User validation file that contains user control limits; used for user information.
/usr/lib/acct/day/pacct	Contains current process accounting information.

SEE ALSO

`acct`(8), `acctsh`(8), `csaperm`(8), `turnacct`(8), `udbgen`(8)

`acct`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

`acct`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

acctprc1, acctprc2 – Processes accounting

SYNOPSIS

```
/usr/lib/acct/acctprc1 [ctmp]
/usr/lib/acct/acctprc2 [-s]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The acctprc1 command reads input in the form described by acct(5), adds login names that correspond to user IDs, then writes per-process temporary data records (in ptmp format) to standard output.

The acctprc1 command accepts the following option:

ctmp If you specify the ctmp file, it is expected to contain a list of login sessions, in the form described in acctcon(8), sorted by user ID and login name. If you omit ctmp, it obtains login names from the user database file (/etc/udb). The information in ctmp helps it distinguish among different login names that share the same user ID.

The acctprc2 command reads records in the form written by acctprc1, summarizes them by user ID and name, then writes the sorted summaries to the standard output as total accounting records (tacct format).

The acctprc2 command accepts the following option:

-s Sorts the output by ascending account ID and by ascending user ID. If this output is processed later by acctmerg(8), you must use the acctmerg -s option.

Typically, these commands are used as follows:

```
acctprc1 ctmp </usr/adm/acct/day/pacct | acctprc2 >ptacct
```

BUGS

Although it is possible to distinguish among login names that share user IDs for commands run normally, it is difficult to do this for those commands run from cron(8), for example. You can do more precise conversion by faking login sessions on the console, using the acctwtmp(8) program.

The size of some structures in these two commands may be too large for small-memory machines. The dimensions for these structures are the A_SSIZE and A_USIZE variables, the maximum number of sessions and distinct login names per accounting run, respectively; they are defined in the /etc/config/acct_config file. To resize the structures to fit machine memory, decrease the values of these variables and rerun the commands.

FILES

`/etc/config/acct_config`	Contains configurable parameters.
`/etc/udb`	User validation file that contains user control limits and contains login names for system users.
`/usr/adm/acct/day/pacct`	Contains process accounting information.

SEE ALSO

`acct`(8), `acctcms`(8), `acctcon`(8), `acctmerg`(8), `acctsh`(8), `acctwtmp`(8), `cron`(8), `fwtmp`(8), `runacct`(8)

`acctcom`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`acct`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

`acct`(5), `utmp`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

acctsh – Overview of accounting shell scripts

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The UNICOS operating system supports two accounting packages, Cray Research system accounting (CSA) and standard UNIX System V accounting. Both packages consist of a set of C programs and shell scripts. Some shell scripts are used by both packages; others are unique to one package or the other.

Each shell script is described in detail on a separate man page in the *UNICOS Administrator Commands Reference Manual*, Cray Research publication SR–2022. The following list contains the accounting shell scripts and their descriptions:

Script	Description
acct2csa	Converts standard UNIX System V accounting tacct files to CSA cacct format.
ckdacct	Checks the size of the daemon accounting files.
ckpacct	Checks the size of the process accounting files.
csaperiod	Performs CSA periodic accounting.
csarun	Processes the CSA daily accounting files and generates reports.
dodisk	Performs disk accounting.
lastlogin	Determines the last date on which each user logged in.
monacct	Performs UNIX System V monthly accounting.
nulladm	Creates an empty file with mode 664 and both, owner and group set to adm.
prctmp	Prints the UNIX System V accounting login session file.
prdaily	Prints the UNIX System V accounting daily report.
prtacct	Prints the UNIX System V accounting total accounting (tacct) file.
remove	Removes the temporary UNIX System V accounting files.
runacct	Processes the UNIX System V accounting daily accounting.
shutacct	Turns off process and daemon accounting.
startup	Turns on system accounting and daemon accounting.
turnacct	Turns process accounting on and off or switches accounting files.
turndacct	Turns daemon accounting on and off or switches accounting files.

FILES

/etc/config/acct_config	Accounting configuration file
/etc/wtmp	Login and logoff summaries
/usr/adm/acct/day	Directory that contains the current process and daemon accounting files
/usr/adm/acct/day/fee	UNIX System V fee accumulator
/usr/adm/acct/day/nqacct*	CSA Network Queuing System (NQS) accounting files
/usr/adm/acct/day/pacct	Current process accounting file
/usr/adm/acct/day/pacct∗	Unprocessed process accounting files
/usr/adm/acct/day/tpacct*	CSA tape accounting files
/usr/adm/acct/nite	Working directory
/usr/lib/acct/ptecms.awk	UNIX System V accounting shell script that generates exceptional usage by command name
/usr/lib/acct/ptelus.awk	UNIX System V accounting shell script that generates exceptional usage by login ID
/usr/adm/acct/sum	Summary directory
/usr/adm/acct/sum/tacct∗	Standard UNIX System V accounting total accounting files

SEE ALSO

acct(8), acctcon(8), ckdacct(8), ckpacct(8), csa(8), csaperiod(8), csarun(8), dodisk(8), lastlogin(8), monacct(8), nulladm(8), prctmp(8), prdaily(8), prtacct(8), remove(8), runacct(8), shutacct(8), startup(8), turnacct(8), turndacct(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

acctwtmp – Creates a utmp(5) record

SYNOPSIS

/usr/lib/acct/acctwtmp "*reason*"

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The acctwtmp command sends a utmp(5) record to standard output. The record contains the current time and a character string that describes the specified *reason*. acctwtmp assigns a record type of ACCOUNTING (see utmp(5)).

The acctwtmp command accepts the following operand:

"*reason*" Must be a string that consists of 11 or fewer characters, numbers, $, or spaces contained in double quotation marks.

EXAMPLES

The following examples are suggestions for using acctwtmp in reboot and shutdown procedures, respectively:

```
/usr/lib/acct/acctwtmp "acct on" >> /etc/wtmp
/usr/lib/acct/acctwtmp "acct off" >> /etc/wtmp
```

FILES

/etc/wtmp Contains login and logoff information

SEE ALSO

acct(8), acctsh(8), fwtmp(8), runacct(8), shutacct(8), startup(8)

utmp(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

aircksconf – Prints and validates an AIR configuration file

SYNOPSIS

```
/usr/air/bin/aircksconf [-D #] [-a] filename
/usr/air/bin/aircksconf -v [-D #] [-a] filename
/usr/air/bin/aircksconf -p [-D #] [-a] filename
/usr/air/bin/aircksconf -s [-D #] [-a] filename
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The aircksconf command reads the file specified on the command line and verifies that it is a valid automated incident reporting (AIR) configuration file. You can use this command to debug configuration files before using them with the rest of the AIR system. By default, the file is verified and a summary of the data is displayed.

The aircksconf command accepts the following options:

-D # Debug mode. Takes a small integer argument that specifies the number of debugging messages to print (the larger the number, the more messages). The range of most commands is 0 through 20.

-a Suppresses access checks. Prevents access checking of files named in the configuration. Useful at early debugging stages when files do not yet exist.

-v Skip validation mode. Skips validation; attempts to reprint configuration data.

-p Pretty print mode. Useful for reformatting the configuration file.

-s Silent mode. Only error messages are displayed.

MESSAGES

Many messages can be generated from this program. Most messages pertain to configuration file errors and are easy to interpret.

BUGS

File access checking messages might state only that the file is "not accessible" when the real problem could be file execute permissions.

SEE ALSO

aird(8)

NAME

aird – Automated incident reporting (AIR) daemon

SYNOPSIS

aird [-d] [-C *number*] [-D *number*] [-L *number*] *config_file*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The aird daemon executes any configured monitoring functions. When the executed monitoring functions return, aird logs pertinent data into its binary log file. The report generators use this file as input, producing AIR reports on system and product availability.

When initiated, aird translates the contents of the specified configuration file into its internal processing worklist, which consists of the configured monitoring functions and their execution rates. aird also sets the specified return types in the environment. The monitoring functions use the environment variables as exit status values in order for the monitoring functions and aird to communicate.

If it receives a SIGHUP signal, aird stops executing the monitoring functions and rereads the configuration file. This capability allows for dynamic redefinition of aird's processing.

The aird daemon accepts the following options (the daily execution of aird is probably with no options):

-d Indicates that the process should not attempt to become a daemon. Use this option only for testing purposes.

-C *number* Specifies the time conversion rate factor. The specified integer value, *number*, indicates to aird how it should interpret the time specifications within the configuration file. The conversion factor is the number of seconds in a minute (the default is 60); by default, time in the configuration file is specified in minutes. You can compress AIR's internal time by lowering the conversion factor. For example, if you specify 1, time in the configuration file would be specified in units of seconds rather than minutes. Use this option only for testing purposes.

-D *number* Sets the debugging level. The *number* value is the number of debugging messages to print; possible values are integers in the range 1 through 20. A value of 0 sets debugging to off. As the number increases, so does the number of messages.

-L *number* Sets the logging level. Possible values are integers in the range 1 through 20. A value of 0 sets debugging to off. Higher numbers indicate more logging.

config_file AIR configuration file name.

BUGS

If the configuration file specified on the command line is not valid, the `aird` process terminates when it discovers errors in the file and ends the current session. Therefore, always run the `aircconf`(8) command on a new or edited configuration file before using that file.

EXAMPLES

In this example, the `aird` process is running in testing mode. The `-C` option, specified with a value of `1`, indicates that the unit of time in the configuration file should be interpreted as seconds. The `-d` option indicates that the program should not disconnect from the controlling terminal. The configuration file is `/etc/config_file`, and the messages are directed to the `/tmp/airlog` file.

```
aird -C 1 -d /etc/config_file >/tmp/airlog
```

SEE ALSO

`aircconf`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

airdchk – Ensures that the AIR daemon (aird(8)) is running (cron(8) script)

SYNOPSIS

```
/usr/air/bin/airdchk
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The airdchk script executes as a cron(8) job and verifies that the automated incident reporting (AIR) coordinator daemon (aird(8)) is running. The script uses the airexist(8) command to verify that the process aird(8) exists on the system.

If the return from the airexist(8) invocation indicates that the process no longer exists on the system, mail is sent to root indicating that the aird(8) process is not running.

SEE ALSO

aird(8), airexist(8), cron(8)

NAME

airdet – Generates detailed AIR reports based on aird(8) binary log file

SYNOPSIS

airdet [-b *time*] [-e *time*] [-p *product*] [-f *function*] [-n *time*] [-T *types*] [-O] [-m] [-h] [-t] [-l] [-D *number*] [-w] *file1* [*file2* ...]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The airdet command prints the contents of the automated incident reporting (AIR) daemon's (aird(8)) binary log file, excluding the configuration header records. The airprconf(8) command prints the configuration header records; airdet prints all other records.

The aird(8) process logs three types of records into the binary log file. *Event records* comprise the majority of the records and are logged when each monitoring function is returned. These records indicate the product monitored by the function, the function's name, its start and end times, and the status of the product.

The aird(8) process also logs its own heartbeat record at the specified rate, as defined in the configuration file. These records are used to determine the availability of the AIR system; that is, the percent of time during a specified interval that aird(8) was running.

If a monitoring function does not return in the length of time specified in the configuration file, the aird(8) process kills the function and logs a time-out record, which indicates abnormal termination of the function.

The airdet command accepts the following options, which are listed in the following two groups:

- Selection options that specify the records selected for display
- Information options that specify the information to display for each of the selected records

By default, this command selects all of the event, heartbeat, and time-out records from the binary log file and prints the product and function names and the exit status types.

Selection Options

-b *time* Specifies the sample start time. Time format is as follows:

"[*month/day*[*/year*]] *hour:min*[:*sec*]"

-e *time* Specifies the sample end time. Time format is as follows:

"[*month/day*[*/year*]] *hour:min*[:*sec*]"

-p *product* Prints detailed records of only the specified product. An example of the many products provided, including user configurable products, is the Network Queuing System (NQS).

`-f` *function* Prints detailed records of only the specified function (for example, `existence`).

`-n` *time* Prints detailed records if their elapsed time is greater than the specified time.

`-T` *types* Prints detailed records of only the specified types (for example, `PROD_UNAVAILABLE`).

`-O` OR listed types to the `-T` option; AND is the default.

Information Options

`-m` Prints detailed message text.

`-h` Prints headers.

`-t` Prints time stamps.

`-l` Prints elapsed times.

`-D` *number* Sets debugging level. The *number* value is the number of debugging messages to print. Possible values are integers in the range 1 through 20. A value of `0` sets debugging to off. As the number increases, so does the number of messages.

`-w` Prints return value bits (in octal).

file1 [*file2* ...] `aird`(8) binary log files to use as input. One input file is required. You can specify others, separating the file names with spaces.

NOTES

If the date is not included in the time specifications for the `-b` and `-e` options, `airdet` uses today's date. Be careful when using the `-b` and `-e` options without date specifications if you are looking at binary log files no longer directly logged to by `aird`(8). It is possible that no records will be selected because the contents of the binary log file were logged before the current day. For example, if a command line were executed on April 15, 1991, on a binary log file from April 11, no records would be selected, because the file would contain time stamps from only April 11.

The *time* argument to the `-b` and `-e` options is keyed off the function ending times.

The `*** New configuration read in. ***` string is printed whenever the report generator reads in a configuration header record. This string indicates that the `aird`(8) process has reread the AIR configuration file and reinitialized its internal processing work list.

EXAMPLES

The following examples show how you can use the command-line options to look at an `aird`(8) binary log file. All the examples are derived from the same binary log file.

Example 1: The following example shows the default selection of all records and the default information displayed for each record:

```
% airdet -h /usr/spool/air/logs/blog

Product     Function     Type of Message
Name        Name
----------- ------------ ----------------
kernel      response     PROD_AVAILABLE
kernel      response     PROD_AVAILABLE
kernel      response     PROD_AVAILABLE
kernel      response     PROD_AVAILABLE
tcp         smailexist   PROD_AVAILABLE
tcp         ntpdexist    PROD_AVAILABLE
tcp         lpdexist     PROD_AVAILABLE
tcp         namedexist   PROD_AVAILABLE
tcp         netexist     PROD_AVAILABLE
nqs         qfexist      PROD_AVAILABLE
tapes       existence    PROD_AVAILABLE
nqs         existence    PROD_AVAILABLE
msgdaemon   existence    PROD_AVAILABLE
nqs         netexist     PROD_AVAILABLE
tcp         existence    PROD_AVAILABLE
tcp         gatedexist   PROD_UNAVAILABLE
tapes       avrexist     PROD_AVAILABLE
tcp         snmpdexist   PROD_AVAILABLE
kernel      existence    PROD_AVAILABLE
kernel      response     PROD_AVAILABLE
kernel      response     PROD_AVAILABLE
kernel      response     PROD_AVAILABLE
kernel      response     PROD_AVAILABLE
kernel      response     PROD_AVAILABLE
```

Example 2: The following example shows the selection of records pertaining to specific tests of specific products:

```
% airdet -h -p kernel -f response /usr/spool/air/logs/blog

Product     Function     Type of Message
Name        Name
----------- ------------ ----------------
kernel      response     PROD_AVAILABLE
kernel      response     PROD_AVAILABLE
kernel      response     PROD_AVAILABLE
kernel      response     PROD_AVAILABLE
kernel      response     PROD_AVAILABLE
kernel      response     PROD_AVAILABLE
kernel      response     PROD_AVAILABLE
kernel      response     PROD_AVAILABLE
kernel      response     PROD_AVAILABLE
```

Example 3: The following example shows the selection of records with specific return types and the text display option:

```
% airdet -hm -p tcp -T PROD_UNAVAILABLE /usr/spool/air/logs/blog

Product     Function      Type of Message   Message
Name        Name                            Text
----------- ------------- ----------------- ---------------------------
tcp         gatedexist    PROD_UNAVAILABLE  Test Failed
```

Example 4: The following example shows the selection of records whose elapsed time exceeds the specified time and the elapsed time display option:

```
% airdet -hl -p nqs -T PROD_AVAILABLE -n 1 /usr/spool/air/logs/blog

Elapsed    Product     Function      Type of Message
Time       Name        Name
---------- ----------- ------------- ----------------
00:02:10   nqs         netexist      PROD_AVAILABLE
```

Example 5: The following example shows the selection of records whose ending time comes after the specified time and the time-stamp display option:

```
% airdet -ht -b "4/11/91 10:50" -p kernel /usr/spool/air/logs/blog

Start                End                  Product  Function  Type of
Time                 Time                 Name     Name      Message
-------------------- -------------------- -------  --------  ----------
Apr 11 10:50:37 1991 Apr 11 10:50:47 1991 kernel   response  PROD_AVAIL
Apr 11 10:51:37 1991 Apr 11 10:51:47 1991 kernel   response  PROD_AVAIL
Apr 11 10:52:37 1991 Apr 11 10:52:47 1991 kernel   response  PROD_AVAIL
Apr 11 10:53:37 1991 Apr 11 10:53:47 1991 kernel   response  PROD_AVAIL
```

Example 6: The following example shows the selection of records whose ending time comes within the specified range of time and the time-stamp display option:

```
% airdet -ht -b "10:50" -e "10:53" -p kernel /usr/spool/air/logs/blog

Start                End                  Product  Function  Type of
Time                 Time                 Name     Name      Message
-------------------- -------------------- -------- --------  ----------
Apr 11 10:50:37 1991 Apr 11 10:50:47 1991 kernel   response  PROD_AVAIL
Apr 11 10:51:37 1991 Apr 11 10:51:47 1991 kernel   response  PROD_AVAIL
Apr 11 10:52:37 1991 Apr 11 10:52:47 1991 kernel   response  PROD_AVAIL
```

SEE ALSO

aird(8), airprconf(8), airsum(8), airtsum(8)

NAME

airexist – Searches the process table in kernel memory for a process matching the command-line requirements

SYNOPSIS

/usr/air/bin/airexist [-a *acid*] [-j *jid*] [-p *pid*] [-u *uid*] [-J] [-P *ppid*] [-e] [-v] [-c *corefile*] [-n *namelist*] *process_name*

/usr/air/bin/airexist [-a *acid*] [-j *jid*] [-p *pid*] [-u *uid*] [-J] [-P *ppid*] [-I] [-v] [-c *corefile*] [-n *namelist*] *process_name*

/usr/air/bin/airexist [-a *acid*] [-j *jid*] [-p *pid*] [-u *uid*] [-J] [-P *ppid*] [-M] [-v] [-c *corefile*] [-n *namelist*] *process_name*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The airexist command searches in the kernel memory process table for a process matching the command-line specifications. If options are not specified, airexist immediately returns a no match value.

The airexist command accepts the following options.

Selection Criteria

-a *acid* Makes a match only if *process_name* has account ID (*acid*).

-j *jid* Makes a match only if *process_name* has job ID (*jid*).

-p *pid* Makes a match only if *process_name* has process ID (*pid*).

-u *uid* Makes a match only if *process_name* has user ID (*uid*).

-J Makes a match only if *process_name* is in its own job.

-P *ppid* Makes a match only if *process_name* has parent process ID (*ppid*).

Exit Status Criteria

(Note that the -e, -I, and -M options are mutually exclusive.)

-e Uses PASSED and FAILED environment variables for status.

-I Sets the exit status to the number of matches found.

-M If a match is not made, sets exit status to 0. If one match is made, sets exit status to 1. If more than one match is made, sets exit status to 2.

-v Prints an ASCII result string to standard output.

Searching Files

`-c` *corefile* Uses the *corefile* file in place of `/dev/mem`.

`-n` *namelist* Takes *namelist* as the name of an alternative system *namelist* file in place of `/unicos`.

process_name Name of the process to match.

FILES

`/unicos`	System namelist
`/dev/mem`	Memory
`/etc/passwd`	Supplies UID information

SEE ALSO

`ps`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

NAME

airping – Measures round-trip delays and packet loss across network paths, using the specified protocol

SYNOPSIS

/usr/air/bin/airping [-d] [-i] [-r] [-t] [-u] [-v] [-l *length*] [-n *num*] *host* [*length* [*num*]]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The airping command sends data across the specified protocol and determines the state of the network connection based upon the packet loss value. Available protocols are the Internet Control Message Protocol (ICMP), the Internet Transmission Control Protocol (TCP), and the Internet User Datagram Protocol (UDP).

The airping command accepts the following options and operands:

-d Enables the recording of socket debugging information.

-i Specifies the use of ICMP protocol; this is the default protocol.

-r Enables routing bypass for outgoing messages.

-t Specifies the use of TCP protocol.

-u Specifies the use of UDP protocol.

-v Sets verbose mode.

-l *length* Sets the data length for the packets.

-n *num* Sets the number of packets to send. The default is infinite.

host [*length* [*num*]]
Specifies the name of the host to be used as the end network node, the packet data length, and the number of packets to send.

SEE ALSO

ping(8)

icmp(4P), tcp(4P), udp(4P) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

airprconf – Prints AIR configuration file contents from configuration headers in the aird(8) binary log file

SYNOPSIS

airprconf [-a] [-D *number*] [-b *time*] [-e *time*] [-P] *file1* [*file2* ...]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The airprconf command prints the contents of the automated incident reporting (AIR) configuration file read in by the aird(8) daemon at the initiation of its processing or on reception of the SIGHUP signal. The aird(8) daemon logs this information as a configuration header in its binary log file after it reads and translates the contents of the configuration file.

When the -a option is specified, each of the configuration headers in the specified binary log files is processed, and the subsequent configuration file contents are printed. By default, airprconf prints only the last configuration header record in the specified binary log file.

The airprconf command accepts the following options:

-a Prints all configuration headers found. Overrides the default, which is to print only the last header, if any currently exist in the file.

-D *number* Sets the debugging level. The *number* value is the number of debugging messages to print; possible values are integers in the range 1 through 20. A value of 0 sets debugging to off. As the number increases, so does the number of messages.

-b *time* Prints all configuration headers found after the specified *time*. Time format is as follows:

"[*month/day*[/*year*]] *hour:min*[:*sec*]"

-e *time* Prints all configuration headers found before the specified *time*. By default, only the last configuration header is printed. Time format is as follows:

"[*month/day*[/*year*]] *hour:min*[:*sec*]"

-P Prints the configuration data in easy-to-read format as it appears in the configuration file.

file1 [*file2* ...] aird binary log files to use as input. Separate multiple file names, if any, with spaces.

EXAMPLES

Example 1: The following example prints version information and simple configuration information in the default format:

```
%  airprconf /usr/spool/air/logs/blog
Type count     2
Message count  4
Product count  2
Function count 5
TYPE 0 Tag: FAILED
TYPE 1 Tag: PASSED
PRODUCT: tcp
Message 0 Prod: tcp Tag: PASSED Text: Test Passed
Message 1 Prod: tcp Tag: FAILED Text: Test Failed
FUNCTION: existence
FUNCTION: function
PRODUCT: tapes
Message 2 Prod: tapes Tag: FAILED Text: Test Failed
Message 3 Prod: tapes Tag: PASSED Text: Test Passed
FUNCTION: existence
FUNCTION: response
FUNCTION: function
```

Example 2: The configuration information printed in the following example is the same as that printed in the previous example, but the format is like that of the configuration file:

```
%  airprconf -P /usr/spool/air/logs/blog
#
# Start of Configuration File Generated by airprconf on Tue Apr 16
# 10:16:02 1991
#
CONFIG  kernel_test_version
        #
        #       Define Coordinator logfile
        #
        COORD_LOG       /usr/spool/air/logs/coord.log
        #
        #       Define Coordinator Heart Beat
        #
        COORD_HBEAT     10
        #
        #       Define Coordinator DEBUG Level (-D)
        #
        COORD_DEBUG     0
        #
        #       Define Coordinator Test Initiation directory (-h)
        #
        COORD_TESTDIR   /usr/air/test
        #
        #       Define Coordinator Binary output file (-f)
        #
        COORD_BLOG      /usr/spool/air/logs/blog
        #
        #       Define Coordinator ASCII Logging Level (-L)
        #
        COORD_LOGLEV    0
        #
        #       Define TYPES
        #
        TYPE    FAILED  PROD_UNAVAILABLE
        TYPE    PASSED  PROD_AVAILABLE
        #
        #       Define product tcp
        #
        PRODUCT tcp     ON
                MESSAGE PASSED  Test Passed
                MESSAGE FAILED  Test Failed
                #
                #       Define Function existence of Product tcp
                #
```

```
        FUNCTION        existence      ON
                RATE    5
                EXECUTE /usr/air/test/tcp/tcp.exist
                LOGFILE NONE
                TIMEOUT NONE
                RETURN  PASSED  0       NONE
                RETURN  FAILED  1       NONE
        ENDFUNCTION     existence
        #
        #       Define Function function of Product tcp
        #
        FUNCTION        function      ON
                RATE    10
                EXECUTE /usr/air/test/tcp/tcp.funct
                LOGFILE NONE
                TIMEOUT NONE
                RETURN  FAILED  1       NONE
                RETURN  PASSED  0       NONE
        ENDFUNCTION     function
ENDPRODUCT      tcp
#
#       Define product tapes
#
PRODUCT tapes   ON
        MESSAGE FAILED  Test Failed
        MESSAGE PASSED  Test Passed
        #
        #       Define Function existence of Product tapes
        #
        FUNCTION        existence      ON
                RATE    5
                EXECUTE /usr/air/test/tapes/tape.exist
                LOGFILE NONE
                TIMEOUT NONE
                RETURN  PASSED  0       NONE
                RETURN  FAILED  1       NONE
        ENDFUNCTION     existence
        #
        #       Define Function response of Product tapes
        #
        FUNCTION        response      ON
                RATE    5
                EXECUTE /usr/air/test/tapes/tape.exist
                LOGFILE NONE
```

```
                        TIMEOUT NONE
                        RETURN  PASSED  0       NONE
                        RETURN  FAILED  1       NONE
                ENDFUNCTION     response
                #
                #       Define Function function of Product tapes
                #
                FUNCTION        function      ON
                        RATE    10
                        EXECUTE /usr/air/test/tapes/tape.funct
                        LOGFILE NONE
                        TIMEOUT NONE
                        RETURN  FAILED  1       NONE
                        RETURN  PASSED  0       NONE
                ENDFUNCTION     function
        ENDPRODUCT      tapes
ENDCONFIG       kernel_test_version
#
# End of Configuration File Generated by airprconf on Tue Apr 16
# 10:16:02 1991
#
```

SEE ALSO

airckconf(8), aird(8), airdet(8), airsum(8), airtsum(8)

NAME

airrep – Produces AIR activity reports

SYNOPSIS

/usr/air/bin/airrep [-o *file*] *logfiles*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The airrep shell script produces generic activity reports for the automated incident reporting (AIR) product set by invoking the airsum(8), airtsum(8), and airdet(8) report generator commands. The time period over which airrep reports is specified in the binary log file arguments.

By default, airrep invokes the following report generator command lines:

```
airsum -hBA logfiles
airtsum -rhaS logfiles
airdet -T PROD_UNAVAILABLE -hmt logfiles
```

The airsum(8) shell script prints the product availability summary and breakdown information, airtsum(8) prints the monitoring function's information, including the return type breakdowns, and airdet(8) prints the records indicating an unavailable product in the monitored product set. For further information on available options, see the man pages for each report generator.

The airrep shell script accepts the following option and operand:

-o *file* Redirects standard output to the specified file.

logfiles aird(8) binary log files to be used as input.

SEE ALSO

aird(8), airdet(8), airsum(8), airtsum(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

airsum – Generates availability summary reports based on aird(8) binary log file

SYNOPSIS

airsum [-b *time*] [-e *time*] [-a [*key*]] [-u [*key*]] [-h] [-E] [-B] [-l] [-m] [-s] [-t] [-A] [-D *number*] [-L] [-M] [-S] [-T] *file1* [*file2* ...]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The airsum command collects data from the automated incident reporting (AIR) daemon's (aird(8)) binary log file and prints summary statistics on the availability of the monitored products.

Three main forms of statistical presentation are available:

- Periodic breakdown of availability
- Statistical summary of the periodic breakdowns
- Overall availability summary

The final pass at data collection results in a series of availability structures for each product. Each element in this chain indicates the state of the product and the time during which the product was in that state. From this chain, the periodic breakdown summaries and statistics are generated.

The product's state is determined by the results of all of the configured functions; for a product to be determined available, all configured functions must return an available status.

By default, this command prints the names of the monitored products, the total time during which each product was available and unavailable, and the relative and real percentages of time available and unavailable.

The relative percentage column presents the percentage of time that the aird(8) process was running during which each of the products was available. The real percentage column represents the percentage of time each of the products was available during the entire specified interval.

The airsum command accepts the following selection and printing options.

Selection Options

-b *time* Specifies the sample start time. Time format is as follows:

"[*month/day*[/*year*]] *hour:min*[:*sec*]"

-e *time* Specifies the sample end time. Time format is as follows:

"[*month/day*[/*year*]] *hour:min*[:*sec*]"

`-a` [*key*] Specifies the availability of the type key. By default, this key is defined as the `PROD_AVAILABLE` type. Use this option to key a different function return type for the availability determinations.

`-u` [*key*] Specifies the unavailability of the type key. By default, this key is defined as the `PROD_UNAVAILABLE` type. Use this option to key a different function return type for the availability determinations.

Printing Options

`-h` Prints headers.

`-E` Prints a summary of each configuration read. The default is to print only the total summary.

`-B` Prints the elements of the availability breakdown chain. Each element in the chain indicates either a state change, from available to unavailable or vice versa, or a new configuration.

`-l` Prints the longest time period available.

`-m` Prints the average time period available.

`-s` Prints the shortest time period available.

`-t` Prints the total time available.

`-A` Prints the breakdown by period summary report (same as specifying the options `-lmstLMST`).

`-D` *number* Sets the debugging level. The *number* value is the number of debugging messages to print; possible values are integers in the range 1 through 20. A value of `0` sets debugging to off. As the number increases, so does the number of messages.

`-L` Prints the longest time period unavailable.

`-M` Prints the average time period unavailable.

`-S` Prints the shortest time period unavailable.

`-T` Prints the total time unavailable.

file1 [*file2* ...] `aird`(8) binary log files to use as input. One input file is required. You can specify others, separating the file names with spaces.

NOTES

If the date is not included in the time specifications for the `-b` and `-e` options, `airsum` uses today's date. Be careful when using the `-b` and `-e` options without date specifications if you are looking at binary log files no longer directly logged to by `aird`(8). It is possible that no records will be selected because the contents of the binary log file were logged before the current day. For example, if a command line were executed on April 15, 1991, on a binary log file from April 11, no records would be selected because the file would contain time stamps from only April 11.

The *time* argument to the `-b` and `-e` options is keyed off the function ending times.

Multiple log file arguments' contents must be ordered exclusively by time.

EXAMPLES

Example 1: The following example shows the default availability summary information displayed:

```
% airsum -h /usr/spool/air/logs/blog

*** Total Availability Summary ***

Summary Information

Product    Total Time     Total Time     Rel. Perc. Real Perc.
Name       Available      Unavailable    Available  Available
---------- -------------- -------------- ---------- ----------
airdaemon      6:14:41:39       10:20:00        100         93
kernel         6:14:40:30       00:00:00         99         93
tapes          6:06:04:03       06:46:40         94         88
tcp              00:00:00     6:12:50:22          0          0
nqs              07:38:02     6:05:07:19          4          4
-------------------------------------------------------------
```

Example 2: The following example shows an availability summary for each configuration rather than the default action of only the final and grand total summary:

```
% airsum -hE /usr/spool/air/logs/blog

Summary Information

Product    Total Time     Total Time     Rel. Perc. Real Perc.
Name       Available      Unavailable    Available  Available
---------- -------------- -------------- ---------- ----------
airdaemon        02:28:38       00:00:00        100        100
kernel           02:28:38       00:00:00        100        100
tapes            02:27:28       00:00:00         99         99
tcp              00:00:00       02:27:28          0          0
nqs              02:27:28       00:00:00         99         99
-------------------------------------------------------------
```

```
*** New configuration read in. ***

Summary Information

Product    Total Time     Total Time     Rel. Perc. Real Perc.
Name       Available      Unavailable    Available  Available
---------- -------------- -------------- ---------- ----------
airdaemon        00:08:10       00:00:00        100        100
kernel           00:08:10       00:00:00        100        100
tcp              00:00:00       00:04:00          0          0
nqs              00:04:00       00:00:00         48         48
tapes            00:04:00       00:00:00         48         48
------------------------------------------------------------

*** New configuration read in. ***
*** Total Availability Summary ***

Summary Information

Product    Total Time     Total Time     Rel. Perc. Real Perc.
Name       Available      Unavailable    Available  Available
---------- -------------- -------------- ---------- ----------
airdaemon      6:14:41:39       10:20:00        100         93
kernel         6:14:40:30       00:00:00         99         93
tapes          6:06:04:03       06:46:40         94         88
tcp              00:00:00     6:12:50:22          0          0
nqs              07:38:02     6:05:07:19          4          4
------------------------------------------------------------
```

Example 3: The following example shows the additional periodic breakdown of product availability that can be requested:

```
%  airsum -hB /usr/spool/air/logs/blog

*** Total Availability Summary ***


Product Availability Breakdown

Product    Product          From                 Until
Name       Status           Time                 Time
---------- ---------------- -------------------- --------------------
airdaemon  PROD_AVAILABLE   Apr 11 08:15:09 1991 Apr 11 10:43:47 1991
           PROD_AVAILABLE   Apr 11 10:45:37 1991 Apr 11 10:53:47 1991
           PROD_AVAILABLE   Apr 11 10:55:37 1991 Apr 11 11:03:47 1991
kernel     PROD_AVAILABLE   Apr 11 08:15:09 1991 Apr 11 10:43:47 1991
           PROD_AVAILABLE   Apr 11 10:45:37 1991 Apr 11 10:53:47 1991
           PROD_AVAILABLE   Apr 11 10:55:37 1991 Apr 11 11:03:47 1991
tapes      PROD_AVAILABLE   Apr 11 08:15:09 1991 Apr 11 10:42:37 1991
           PROD_AVAILABLE   Apr 11 10:45:37 1991 Apr 11 10:49:37 1991
           PROD_AVAILABLE   Apr 11 10:55:37 1991 Apr 11 10:59:37 1991
           PROD_UNAVAILABLE Apr 11 23:54:19 1991 Apr 12 05:59:19 1991
           PROD_AVAILABLE   Apr 12 07:41:08 1991 Apr 12 08:44:01 1991
           PROD_AVAILABLE   Apr 12 09:19:34 1991 Apr 12 16:58:28 1991
           PROD_UNAVAILABLE Apr 16 07:07:20 1991 Apr 16 07:12:20 1991
           PROD_AVAILABLE   Apr 16 07:12:20 1991 Apr 16 07:17:20 1991
           PROD_AVAILABLE   Apr 16 07:17:20 1991 Apr 16 07:37:20 1991
           PROD_UNAVAILABLE Apr 16 07:37:20 1991 Apr 16 07:47:20 1991
           PROD_AVAILABLE   Apr 16 07:47:20 1991 Apr 16 08:12:20 1991
           PROD_AVAILABLE   Apr 16 09:05:10 1991 Apr 16 11:07:30 1991
           PROD_UNAVAILABLE Apr 18 07:29:37 1991 Apr 18 07:51:17 1991
           PROD_UNAVAILABLE Apr 18 07:51:17 1991 Apr 18 07:56:17 1991
           PROD_AVAILABLE   Apr 18 07:56:17 1991 Apr 18 08:46:17 1991
           PROD_AVAILABLE   Apr 18 08:58:37 1991 Apr 18 09:23:11 1991
tcp        PROD_UNAVAILABLE Apr 11 08:15:09 1991 Apr 11 10:42:37 1991
           PROD_UNAVAILABLE Apr 11 10:45:37 1991 Apr 11 10:49:37 1991
           PROD_UNAVAILABLE Apr 11 10:55:37 1991 Apr 11 10:59:37 1991
nqs        PROD_AVAILABLE   Apr 11 08:15:09 1991 Apr 11 10:42:37 1991
           PROD_AVAILABLE   Apr 11 10:45:37 1991 Apr 11 10:49:37 1991
           PROD_AVAILABLE   Apr 11 10:55:37 1991 Apr 11 10:59:37 1991
           PROD_UNAVAILABLE Apr 11 13:11:15 1991 Apr 11 14:05:30 1991
           PROD_UNAVAILABLE Apr 11 14:19:41 1991 Apr 12 05:59:19 1991
           PROD_AVAILABLE   Apr 17 09:10:45 1991 Apr 17 13:10:45 1991
           PROD_UNAVAILABLE Apr 17 13:10:45 1991 Apr 17 17:00:45 1991
           PROD_UNAVAILABLE Apr 17 17:11:45 1991 Apr 17 17:56:14 1991
```

```
Summary Information

Product     Total Time      Total Time      Rel. Perc. Real Perc.
Name        Available       Unavailable     Available  Available
----------- --------------- --------------- ---------- ----------
airdaemon       6:14:49:51        10:20:00        100         93
kernel          6:14:48:42        00:00:00         99         93
tapes           6:06:14:03        06:46:40         94         88
tcp               00:00:00     6:13:00:22          0          0
nqs               07:23:28     6:05:31:53          4          4
-------------------------------------------------------------
```

Example 4: The following example shows the availability statistics that can be displayed for each product:

```
% airsum -hA /usr/spool/air/logs/blog

*** Total Availability Summary ***

Summary Information

Product           Total Time      Total Time      Rel. Perc. Real Perc.
Name              Available       Unavailable     Available  Available
----------------- --------------- --------------- ---------- ----------
airdaemon             6:00:11:41        08:15:07        100         94
kernel                6:00:10:32        00:00:00         99         94
tapes                 5:16:15:29        06:20:00         94         89
tcp                     00:00:00     5:22:35:08          0          0
nqs                     07:23:28     5:15:06:39          5          4
-------------------------------------------------------------------
```

```
Summary Information for Periods

Product    Total Time  Total Time  Shortest Per. Shortest Per.
Name       Available   Unavailable Available     Unavailable
---------- ----------- ----------- ------------- -------------
airdaemon   6:00:11:41    00:00:00      17:21:57      00:00:00
kernel      6:00:10:32    00:00:00      17:21:57      00:00:00
tapes       5:16:15:29    06:20:00      17:20:47      06:05:00
tcp           00:00:00  5:22:35:08      00:00:00      17:20:24
nqs           07:23:28  5:15:06:39      04:00:00      16:21:25

Product    Longest Per.   Longest  Per.  Average Per.   Average Per.
Name       Available      Unavailable    Available      Unavailable
---------- -------------- -------------- -------------- --------------
airdaemon      6:00:11:41       00:00:00     6:00:11:41       00:00:00
kernel         6:00:10:32       00:00:00       17:21:57       00:00:00
tapes          5:16:15:29       06:20:00       17:20:47       06:05:00
tcp              00:00:00     5:22:35:08       00:00:00       17:20:24
nqs              07:23:28     5:15:06:39       04:00:00       16:21:25

*** Total Availability Summary ***

Summary Information

Product    Total Time     Total Time     Rel. Perc. Real Perc.
Name       Available      Unavailable    Available  Available
---------- -------------- -------------- ---------- ----------
airdaemon      6:15:03:06       10:20:00        100         93
kernel         6:15:01:42       00:00:00         99         93
tapes          6:06:29:05       06:46:40         94         88
tcp              00:00:00     6:13:15:24          0          0
nqs              07:23:28     6:05:46:55          4          4
-------------------------------------------------------------
```

```
Summary Information for Periods

Product    Total Time  Total Time  Longest Per.  Longest Per.
Name       Available   Unavailable Available     Unavailable
---------- ----------- ----------- ------------- --------------
airdaemon   6:15:03:06    00:00:00      17:21:57       00:00:00
kernel      6:15:01:42    00:00:00      17:21:57       00:00:00
tapes       6:06:29:05    06:46:40      17:20:47       06:05:00
tcp           00:00:00  6:13:15:24      00:00:00       17:20:24
nqs           07:23:28  6:05:46:55      04:00:00       16:21:25


Product    Shortest Per. Shortest Per.  Average Per.   Average Per.
Name       Available     Unavailable    Available      Unavailable
---------- ------------- -------------- -------------- --------------
airdaemon       00:04:10       00:00:00       03:58:34       00:00:00
kernel          00:04:10       00:00:00       03:58:32       00:00:00
tapes           00:04:00       00:05:00       03:29:58       01:21:20
tcp             00:00:00       00:04:00       00:00:00       03:55:53
nqs             00:04:00       00:00:00       00:27:43       05:09:53
```

Example 5: The following example shows the selection of certain fields for display from the availability statistics for each product in the previous example:

```
% airsum -hlLsS /usr/spool/air/logs/blog

*** Total Availability Summary ***

Summary Information

Product    Total Time     Total Time     Rel. Perc. Real Perc.
Name       Available      Unavailable    Available  Available
---------- -------------- -------------- ---------- ----------
airdaemon      6:15:41:54       10:20:00        100         93
kernel         6:15:40:45       00:00:00         99         93
tapes          6:07:04:03       06:46:40         94         88
tcp              00:00:00     6:13:50:22          0          0
nqs              07:23:28     6:06:21:53          4          4
-------------------------------------------------------------
```

```
Summary Information for Periods

Product    Longest Per.   Longest Per.   Shortest Per.  Shortest Per.
Name       Available      Unavailable    Available      Unavailable
---------- -------------- -------------- -------------- --------------
airdaemon        17:21:57       00:00:00       00:04:10       00:00:00
kernel           17:21:57       00:00:00       00:04:10       00:00:00
tapes            17:20:47       06:05:00       00:04:00       00:05:00
tcp              00:00:00       17:20:24       00:00:00       00:04:00
nqs              04:00:00       16:21:25       00:04:00       00:00:00
```

Example 6: The following example shows the specification of different keys for availability determination (the default is PROD_AVAILABLE and PROD_UNAVAILABLE). In this example, they have been switched.

```
% airsum -h -aPROD_UNAVAILABLE -uPROD_AVAILABLE /usr/spool/air/logs/blog

*** Total Availability Summary ***

Summary Information

Product    Total Time     Total Time     Rel. Perc. Real Perc.
Name       Available      Unavailable    Available  Available
---------- -------------- -------------- ---------- ----------
airdaemon      6:16:00:56       10:20:00        100         93
kernel           00:00:00     6:15:59:47          0          0
tapes            06:25:00     6:07:45:43          4          3
tcp              00:00:00     6:14:10:22          0          0
nqs              11:54:36     6:02:10:45          7          6
-------------------------------------------------------------
```

Example 7: The following example shows the selection of a summary sample time that begins with the specified time:

```
% airsum -h -b "4/17/91 10:00" /usr/spool/air/logs/blog

*** Total Availability Summary ***

Summary Information

Product    Total Time     Total Time     Rel. Perc. Real Perc.
Name       Available      Unavailable    Available  Available
---------- -------------- -------------- ---------- ----------
airdaemon        23:41:18       02:41:15        100         89
tapes            23:14:53       00:20:00         98         88
tcp              00:00:00       23:34:53          0          0
nqs              00:00:00       23:34:53          0          0
kernel           23:45:11       00:00:00        100         90
------------------------------------------------------------
```

Example 8: The following example shows the selection of a summary sample that comes within the specified range of time:

```
% airsum -h -b "9:00" -e "10:00" /usr/spool/air/logs/blog

*** Total Availability Summary ***

Summary Information

Product    Total Time     Total Time     Rel. Perc. Real Perc.
Name       Available      Unavailable    Available  Available
---------- -------------- -------------- ---------- ----------
airdaemon        00:50:50       00:00:01        100         99
kernel           00:50:51       00:00:00        100         99
nqs              00:00:00       00:50:01          0          0
tcp              00:00:00       00:50:01          0          0
tapes            00:50:01       00:00:00         98         98
------------------------------------------------------------
```

SEE ALSO

aird(8), airdet(8), airprconf(8), airtsum(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

airtsum – Generates detailed AIR reports based on aird(8) binary log file

SYNOPSIS

/usr/air/bin/airtsum [-b *time*] [-e *time*] [-h] [-E] [-r] [-S] [-l] [-s] [-a] [-p] [-c] [-D *number*] *file1* [*file2* ...]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The airtsum command gathers data from the event records in the automated incident reporting (AIR) daemon's (aird(8)) binary log file and prints summary statistics on the monitoring functions executed by the AIR daemon. This information is useful when you are analyzing the reports generated by the airsum(8) command. A clearer indication of the testing activity for each product gives credibility to the statistics reported by airsum(8).

This information is also useful in targetting functions that may be incorrectly configured for the system. For example, you may be executing the test for the network daemon component of the Network Queuing System (NQS) while not having that component configured on your system. The function would always return a failed status and, as a result, NQS would always be reported as unavailable. In this case, you could disable that function in the configuration file and obtain more accurate statistics on the availability of NQS.

The default action for this command is to print the name of the functions that were executed for each product, the number of times each function was executed, and the total time the product was tested by each function.

The airtsum command accepts the following options and operands:

-b *time* Specifies the sample start time. Time format is as follows:

"[*month/day*[/*year*]] *hour:min*[:*sec*]"

-e *time* Specifies the sample end time. Time format is as follows:

"[*month/day*[/*year*]] *hour:min*[:*sec*]"

-h Prints headers.

-E Prints a summary of each configuration read. The default is to print only the total summary.

-r Prints a breakdown of return values per function. This information includes a list of return types and the number of times that each type was returned for each function.

-S Prints the beginning and ending times the products were tested by each function.

-l Prints the longest time interval between function executions.

`-s` Prints the shortest time interval between function executions.

`-a` Prints the average time interval between function executions.

`-p` Prints the percentage of time the products were tested by each function.

`-c` Prints the configured time interval between test execution for each function as defined in the configuration header.

`-D` *number* Sets the debugging level. The *number* value is the number of debugging messages to print; possible values are integers in the range 1 through 20. A value of `0` sets debugging to off. As the number increases, so does the number of messages.

file1 [*file2* ...] `aird` binary log files to use as input. One input file is required. You can specify others, separating the file names with spaces.

NOTES

If the date is not included in the time specifications for the `-b` and `-e` options, `airtsum` uses today's date. Be careful when using the `-b` and `-e` options without date specifications if you are looking at binary log files no longer directly logged to by `aird`(8). It is possible that no records will be selected because the contents of the binary log file were logged before the current day. For example, if a command line were executed on April 15, 1991, on a binary log file from April 11, no records would be selected, because the file would contain time stamps from only April 11.

The *time* argument to the `-b` and `-e` options is keyed off the function ending times.

EXAMPLES

Example 1: The following example shows the default testing summary information displayed:

```
%  airtsum -h /usr/spool/air/logs/blog

*** Total Test Summary ***

Function Summary Information

Product    Function      Total    Total Time
Name       Name          Executed Tested
---------- ------------- -------- --------------
kernel     response          9312     7:00:37:41
           existence         1835     7:00:30:34
tapes      avrexist          1840     7:00:30:32
           existence         1840     7:00:30:32
           response           906     7:00:25:32
tcp        namedexist        1840     7:00:30:32
           gatedexist        1840     7:00:30:32
           ntpdexist         1840     7:00:30:31
           snmpdexist        1840     7:00:30:32
           smailexist        1840     7:00:30:32
           existence         1840     7:00:30:32
           lpdexist          1840     7:00:30:32
           function           907     7:00:25:31
nqs        qfexist           1840     7:00:30:32
           netexist          1840     7:00:30:31
           existence         1840     7:00:30:32
           response           907     7:00:25:31
           function           600     6:21:32:54
           slexist            295     6:21:17:38
```

Example 2: The following example shows a testing summary for each configuration rather than the default action of the final and grand total summary:

```
% airtsum -hE /usr/spool/air/logs/blog

*** Sample Test Summary ***

Function Summary Information

Product    Function      Total    Total Time
Name       Name          Executed Tested
---------- ------------- -------- --------------
kernel     response            11       00:10:10
           existence            2       00:05:04
tapes      avrexist             2       00:05:00
           existence            2       00:05:00
           response             1       00:00:00
tcp        namedexist           2       00:05:00
           gatedexist           2       00:05:00
           ntpdexist            2       00:05:00
           snmpdexist           2       00:05:00
           smailexist           2       00:05:00
           existence            2       00:05:00
           lpdexist             2       00:05:00
           function             1       00:00:00
nqs        qfexist              2       00:05:00
           netexist             2       00:05:00
           existence            2       00:05:00
           response             1       00:00:00

*** Sample Test Summary ***

Function Summary Information

Product    Function      Total    Total Time
Name       Name          Executed Tested
---------- ------------- -------- --------------
kernel     response             9       00:08:10
           existence            1       00:00:03
tcp        smailexist           1       00:00:00
           ntpdexist            1       00:00:00
           lpdexist             1       00:00:00
           namedexist           1       00:00:00
           existence            1       00:00:00
```

```
            gatedexist           1       00:00:00
            snmpdexist           1       00:00:00
nqs         qfexist              1       00:00:00
            existence            1       00:00:00
            netexist             1       00:00:00
tapes       existence            1       00:00:00
            avrexist             1       00:00:00


*** Sample Test Summary ***
.
.
.

*** Total Test Summary ***


Function Summary Information

Product     Function      Total    Total Time
Name        Name          Executed Tested
----------- ------------- -------- --------------
kernel      response          9317     7:00:42:41
            existence         1836     7:00:35:36
tapes       avrexist          1841     7:00:35:33
            existence         1841     7:00:35:32
            response           906     7:00:25:32
tcp         namedexist        1841     7:00:35:32
            gatedexist        1841     7:00:35:32
            ntpdexist         1841     7:00:35:32
            snmpdexist        1841     7:00:35:32
            smailexist        1841     7:00:35:32
            existence         1841     7:00:35:32
            lpdexist          1841     7:00:35:32
            function           907     7:00:25:31
nqs         qfexist           1841     7:00:35:32
            netexist          1841     7:00:35:33
            existence         1841     7:00:35:32
            response           907     7:00:25:31
            function           600     6:21:32:54
            slexist            295     6:21:17:38
```

Example 3: The following example shows the return type breakdown for each function:

```
% airtsum -hr /usr/spool/air/logs/blog

*** Total Test Summary ***

Function Summary Information

Product    Function    Total    Return          Number   Total Time
Name       Name        Executed Type            Returned Tested
---------- ----------- -------- -------------- -------- --------------
kernel     response        9321 PROD_AVAILABLE     9321     7:00:46:41
           existence       1837 PROD_AVAILABLE     1837     7:00:40:33
tapes      avrexist        1842 PROD_AVAILABLE     1764     7:00:40:31
                                PROD_UNAVAILABLE     78
           existence       1842 PROD_AVAILABLE     1838     7:00:40:31
                                PROD_UNAVAILABLE      4
           response         907 PROD_AVAILABLE      904     7:00:35:31
                                PROD_UNAVAILABLE      3
tcp        namedexist      1842 PROD_AVAILABLE     1842     7:00:40:31
           gatedexist      1842 PROD_UNAVAILABLE   1842     7:00:40:31
           ntpdexist       1842 PROD_AVAILABLE     1842     7:00:40:31
           snmpdexist      1842 PROD_AVAILABLE     1842     7:00:40:31
           smailexist      1842 PROD_AVAILABLE     1842     7:00:40:31
           existence       1842 PROD_AVAILABLE     1842     7:00:40:31
           lpdexist        1842 PROD_AVAILABLE     1842     7:00:40:31
           function         908 PROD_UNAVAILABLE    908     7:00:35:31
nqs        qfexist         1842 PROD_AVAILABLE     1742     7:00:40:31
                                PROD_UNAVAILABLE    100
           netexist        1842 PROD_AVAILABLE     1780     7:00:40:31
                                PROD_UNAVAILABLE     62
           existence       1842 PROD_AVAILABLE     1743     7:00:40:31
                                PROD_UNAVAILABLE     99
           response         908 PROD_AVAILABLE      859     7:00:35:31
                                PROD_UNAVAILABLE     49
           function         600 PROD_UNAVAILABLE    600     6:21:32:54
           slexist          295 PROD_UNAVAILABLE    295     6:21:17:38
```

Example 4: The following example shows the range of times during which the functions are monitoring their products:

```
% airtsum -hS /usr/spool/air/logs/blog

*** Total Test Summary ***

Function Summary Information

Product   Function   Total  Total Time  From             Until
Name      Name       Exec.  Tested      Time             Time
--------- ---------- ------ ----------- ---------------- ---------------
kernel    response     9324  7:00:49:41 Apr 11 10:33:37  Apr 18 11:23:18
          existence    1838  7:00:45:34 Apr 11 10:37:37  Apr 18 11:23:11
tapes     avrexist     1843  7:00:45:31 Apr 11 10:37:37  Apr 18 11:23:08
          existence    1843  7:00:45:31 Apr 11 10:37:37  Apr 18 11:23:08
          response      907  7:00:35:31 Apr 11 10:42:37  Apr 18 11:18:08
tcp       namedexist   1843  7:00:45:31 Apr 11 10:37:37  Apr 18 11:23:08
          gatedexist   1843  7:00:45:31 Apr 11 10:37:37  Apr 18 11:23:08
          ntpdexist    1843  7:00:45:31 Apr 11 10:37:37  Apr 18 11:23:08
          snmpdexist   1843  7:00:45:31 Apr 11 10:37:37  Apr 18 11:23:08
          smailexist   1843  7:00:45:31 Apr 11 10:37:37  Apr 18 11:23:08
          existence    1843  7:00:45:31 Apr 11 10:37:37  Apr 18 11:23:08
          lpdexist     1843  7:00:45:31 Apr 11 10:37:37  Apr 18 11:23:08
          function      908  7:00:35:31 Apr 11 10:42:37  Apr 18 11:18:08
nqs       qfexist      1843  7:00:45:31 Apr 11 10:37:37  Apr 18 11:23:08
          netexist     1843  7:00:45:31 Apr 11 10:37:37  Apr 18 11:23:08
          existence    1843  7:00:45:31 Apr 11 10:37:37  Apr 18 11:23:08
          response      908  7:00:35:31 Apr 11 10:42:37  Apr 18 11:18:08
          function      601  6:21:47:54 Apr 11 13:35:30  Apr 18 11:23:24
          slexist       295  6:21:17:38 Apr 11 13:50:30  Apr 18 11:08:08
```

Example 5: The following example shows some of the interval statistics that can be displayed for each function:

```
% airtsum -hlsc /usr/spool/air/logs/blog

*** Total Test Summary ***

Function Summary Information

Product   Function   Total  Total Time    Long      Short     Configured
Name      Name       Exec.  Tested        Interval  Interval  Interval
--------- ---------- ------ ------------- --------- --------- ----------
kernel    response     9329    7:00:54:41  00:01:54  00:00:01  00:01:00
          existence    1839    7:00:50:33  00:05:20  00:00:11  00:05:00
tapes     avrexist     1844    7:00:50:31  00:05:22  00:04:10  00:05:00
          existence    1844    7:00:50:31  00:05:14  00:04:10  00:05:00
          response      908    7:00:45:31  00:19:55  00:09:10  00:10:00
tcp       namedexist   1844    7:00:50:31  00:05:13  00:03:48  00:05:00
          gatedexist   1844    7:00:50:31  00:05:22  00:04:10  00:05:00
          ntpdexist    1844    7:00:50:31  00:05:22  00:03:48  00:05:00
          snmpdexist   1844    7:00:50:31  00:05:22  00:04:10  00:05:00
          smailexist   1844    7:00:50:31  00:05:14  00:04:10  00:05:00
          existence    1844    7:00:50:31  00:05:22  00:04:10  00:05:00
          lpdexist     1844    7:00:50:31  00:05:22  00:03:48  00:05:00
          function      909    7:00:45:31  00:10:10  00:07:45  00:10:00
nqs       qfexist      1844    7:00:50:31  00:05:22  00:03:48  00:05:00
          netexist     1844    7:00:50:31  00:05:22  00:02:45  00:05:00
          existence    1844    7:00:50:31  00:05:22  00:04:10  00:05:00
          response      909    7:00:45:31  00:10:10  00:09:10  00:10:00
          function      601    6:21:47:54  00:15:00  00:13:49  00:15:00
          slexist       295    6:21:17:38  00:30:01  00:29:10  00:30:00
```

Example 6: The following example shows more of the interval statistics that can be displayed for each function:

```
% airtsum -hap /usr/spool/air/logs/blog

*** Total Test Summary ***

Function Summary Information

Product    Function      Total    Total Time     Percent  Average
Name       Name          Executed Tested         Time     Interval
---------- ------------- -------- -------------- -------- -------------
kernel     response          9343     7:01:08:42       98      00:00:49
           existence         1841     7:01:00:35       98      00:04:50
tapes      avrexist          1846     7:01:00:31       98      00:04:53
           existence         1846     7:01:00:31       98      00:04:53
           response           909     7:00:55:31       98      00:09:43
tcp        namedexist        1846     7:01:00:31       98      00:04:53
           gatedexist        1846     7:01:00:31       98      00:04:53
           ntpdexist         1846     7:01:00:31       98      00:04:53
           snmpdexist        1846     7:01:00:31       98      00:04:53
           smailexist        1846     7:01:00:31       98      00:04:53
           existence         1846     7:01:00:31       98      00:04:53
           lpdexist          1846     7:01:00:31       98      00:04:53
           function           910     7:00:55:31       98      00:09:42
nqs        qfexist           1846     7:01:00:31       98      00:04:53
           netexist          1846     7:01:00:31       98      00:04:53
           existence         1846     7:01:00:31       98      00:04:53
           response           910     7:00:55:31       98      00:09:42
           function           602     6:22:02:53       96      00:14:08
           slexist            296     6:21:47:39       96      00:27:27
```

Example 7: The following example shows the selection of a summary sample time that begins with the specified time:

```
% airtsum -h -b "4/17/91 10:00" /usr/spool/air/logs/blog

*** Total Test Summary ***

Function Summary Information

Product          Function         Total    Total Time
Name             Name             Executed Tested
---------------- ---------------- -------- --------------
tapes            existence             281     1:02:07:23
                 avrexist              281     1:02:07:24
                 response              140     1:02:02:23
tcp              snmpdexist            281     1:02:07:23
                 namedexist            281     1:02:07:23
                 smailexist            281     1:02:07:23
                 existence             281     1:02:07:23
                 lpdexist              281     1:02:07:23
                 ntpdexist             281     1:02:07:23
                 gatedexist            281     1:02:07:23
                 function              140     1:02:02:23
nqs              existence             281     1:02:07:24
                 netexist              281     1:02:07:23
                 qfexist               281     1:02:07:23
                 response              140     1:02:02:23
                 function               92     1:01:57:39
                 slexist                46     1:01:42:23
kernel           existence             280     1:02:07:26
                 response             1420     1:02:10:32
```

Example 8: The following example shows the selection of a summary sample that comes within the specified range of time:

```
% airtsum -h -b "9:00" -e "10:00" /usr/spool/air/logs/blog

*** Total Test Summary ***

Function Summary Information

Product          Function         Total    Total Time
Name             Name             Executed Tested
---------------- ---------------- -------- --------------
kernel           response               52       00:50:50
                 existence              10       00:45:03
nqs              netexist               10       00:45:01
                 qfexist                10       00:45:01
                 existence              10       00:45:01
                 response                5       00:40:01
                 function                3       00:30:16
                 slexist                 1       00:00:00
msgdaemon        existence              10       00:45:01
                 response                5       00:40:01
tcp              existence              10       00:45:01
                 lpdexist               10       00:45:01
                 ntpdexist              10       00:45:01
                 snmpdexist             10       00:45:01
                 namedexist             10       00:45:01
                 gatedexist             10       00:45:01
                 smailexist             10       00:45:01
                 function                5       00:40:01
nqs-superlink    slexist                10       00:45:01
tapes            avrexist               10       00:45:01
                 existence              10       00:45:01
                 response                5       00:40:01
```

SEE ALSO

aird(8), airdet(8), airprconf(8), airsum(8)

NAME

arp – Displays address resolution display and control

SYNOPSIS

```
arp hostname
arp -a
arp -d hostname
arp -s hostname ether_addr [temp] [pub]
arp -f filename
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The arp program displays and modifies the Internet-to-Ethernet address translation tables that the address resolution protocol (ARP) uses. See the arp(4P) man page for a description of the translation tables.

With no flags specified, the program displays the current ARP entry for *hostname*. You can verify the host by name or by number, using Internet dot notation.

The arp command accepts the following options:

`-a` Displays the current ARP entries by reading the table via sysctl(3C).

`-d` *hostname* Deletes an entry for the host called *hostname*.

`-s` *hostname ether_addr* [temp] [pub]
Creates an ARP entry for the host called *hostname* with the Ethernet address *ether_addr*. The Ethernet address is specified as 6 hexadecimal bytes separated by colons. The entry is permanent unless you specify the temp argument in the command. If you specify the pub argument, the entry will be published (for example, this system acts as an ARP server, responding to requests for *hostname* even though the host address is not its own).

`-f` *filename* Causes the file named *filename* to be read and multiple entries to be set in the ARP tables. Entries in the file should have the following form:

hostname ether_addr [temp] [pub]

hostname Displays the current ARP entry for the specified host name or number.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

SEE ALSO

`inet`(3C) to manipulate internet addresses

`ifconfig`(8) to configure network interface parameters

`privtext`(1) for information about getting the privilege text of a file in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`arp`(4P) for information on the address resolution protocol in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

arrayd – Array services daemon

SYNOPSIS

arrayd [-c] [-f *filename* ...] [-m *number*] [-n] [-nf] [-nm] [-p *number*] [-qf] [-qm] [-sm] [-v...]

IMPLEMENTATION

IRIX and UNICOS systems

DESCRIPTION

The arrayd command invokes the array services daemon. The daemon performs several different tasks related to the use of an array of two or more machines, including the following:

- Allocating global array session handles
- Forwarding array commands to all of the machines in an array
- Maintaining a database of the current array configuration and providing that information to other commands and programs
- Determining which processes belong to a particular array session and providing that information to other commands and programs

The arrayd command itself has several command line options. The valid options include the following:

-c
: Causes arrayd to read any configuration files and then exit immediately, sending any errors to stderr rather than syslog (which is the usual behavior). This is primarily of use for checking the validity of new configuration files. This is the same as the -qf option.

-f *filename*
: Specifies the name of a single configuration file. This option may be specified more than once, in which case the files will be processed in the order in which they are specified. One reason to have multiple configuration files is to allow all of the machines in an array to use a single file for array entries (accessed, for example, through the network file system (NFS)) and still maintain private configuration files for local options and/or security information. The format of an arrayd configuration file is described in arrayd.conf(5). If no configuration files are specified, then /usr/lib/array/arrayd.conf and /usr/lib/array/arrayd.auth will both be used (in that order).

-m *number*
: Sets the machine identifier used by the array services daemon for generating global array session handles to *number*. UNICOS may also use this value when generating array session handles. *number* must be a value between 0 and 32767. It will override any IDENT setting in the LOCAL section of any configuration file.

`-n` Ordinarily, `arrayd` will automatically *daemonize* itself; that is, it will disassociate itself from the current terminal and place itself in the background. Specifying this option causes `arrayd` to run in the foreground on the current terminal. This is mostly useful for testing purposes.

`-nf` Specifies that no configuration files are to be read. This is most useful with options like `-sm` that cause `arrayd` to quit after performing tasks that do not require configuration information. `-nf` will override any `-f` options.

`-nm` Specifies that the system machine ID should not be set. This is used to override a `LOCAL OPTIONS SETMACHID` statement in the configuration file.

`-p` *number* Specifies the port on which the array services daemon should listen for requests. It will override any `PORT` setting in the `LOCAL` section of any configuration file.

`-qf` Directs `arrayd` to quit after parsing the configuration file(s). This is the same as the `-c` option.

`-qm` Directs `arrayd` to quit after setting the system machine ID. This causes `arrayd` to exit as soon as it has set the system machine identifier. This may be useful in cases where a nondefault system machine identifier is desired, but none of the other array services provided by `arrayd` are needed. This can also be used to change the machine identifier on a system that is already running another copy of `arrayd`; in this case, kernel-generated array session handles will use the new machine identifier, while those generated by `arrayd` will continue to use the original machine identifier.

`-sm` Some versions of UNICOS permit setting a system machine identifier, which is used by the kernel for generating global array session handles. If the current system has this facility, and `-sm` is specified, `arrayd` will set the machine ID to the value specified by a `LOCAL IDENT` statement in the configuration file or on the command line by using the `-m` option.

`-v` Specifies the verbose mode: the daemon runs in the foreground (as with the `-n` option) and sends any error messages, plus some additional messages, to `stderr` rather than `syslog`. Specifying this option more than once, or specifying more than one `v` (for example `vvv`), causes additional debugging information to be generated.

NOTES

The `arrayd` command can be set up to run automatically at system initialization time by editing `/etc/config/daemons` to turn on the array feature. To do this, modify the `arrayd` line to read `YES` instead of `NO`.

WARNINGS

An /etc/hosts.equiv file is created when the UNICOS system is built; a localhost line, which enables array services and the message passing interface (MPI) to run, is automatically put in this file. If you have an existing /etc/hosts.equiv file on your system, you need to manually add the localhost line to your file.

SEE ALSO

arrayd.conf(5)

array_services(7), array_sessions(7)

NAME

`atmadmin` – Configures and displays ATM administration statistics

SYNOPSIS

`/etc/atmadmin` [`-u` *unit_number*] [`-aCdhqsSt`] [`-c` *param*] [`-i` *VCI*] [`-I` *VCI*]

IMPLEMENTATION

Cray Research systems with IOS model E and HIPPI channel

CRAY J90 series

CRAY EL series

Note: This command is available only for systems that do not have GigaRing technology.

DESCRIPTION

The `atmadmin` command configures and displays statistics for the Cray Research Asynchronous Transfer Mode (ATM) driver.

The `atmadmin` command accepts the following options:

`-u` *unit_number* Specifies the unit number to use for issuing the command. For example, you would use 0 for interface `atm0`.

`-a` Displays the current settings on the OC3 SONET user network interface (SUNI) chip. This command does not function properly with the transparent asynchronous transmitter/receiver interface (TAXI) ATM cards.

`-C` Clears the IOS trace buffer.

`-d` Toggles IOS tracing on or off.

`-h` Displays the hardware heartbeat and system flags. This can be used to determine if the hardware is still functioning.

`-q` Displays IOS queue information.

`-s` Displays the statistics kept on the IOS.

`-S` Displays the statistics kept on the ATM hardware.

`-t` Dumps an IOS trace.

`-c` *param* Changes the OC3 configuration on the ATM card. This command does not function properly with the TAXI ATM cards. *param* can take one of the following forms:

- Provides register, value, mask values in the form `register:value:mask`. For example, 3A:02:FF sets register 0x3a to value 0x02 with a setting mask of 0xff.
- Sets the registers by using one of the following key words:

`sonet`	Sets up SUNI chip to run in synchronous optical network (SONET) mode
`sdh`	Sets up SUNI chip to run in synchronous digital hierarchy (SDH) mode
`internal`	Uses internal timing for cell transmission
`external`	Uses external timing for cell transmission
`loopback`	Enables internal loopback
`wire`	Disables internal loopback
`fscramble`	Enables frame scrambling
`nfscramble`	Disables frame scrambling
`sscramble`	Enables stream scrambling
`nsscramble`	Disables stream scrambling
`idle`	Uses idle cell insertion
`unassigned`	Uses unassigned cell insertion

`-i` *VCI* Activates an input Virtual Channel Identifier (VCI) on the ATM card. Usually, all of the input VCIs are activated when the device is configured up. Not recommended for normal use.

`-I` *VCI* Deactivates an input VCI on the ATM card. Not recommended for normal use.

SEE ALSO

`atmarp`(8)

NAME

atmarp – Configures and displays an ATM ARP table

SYNOPSIS

/etc/atmarp -a [-v]

/etc/atmarp -s *host ifc aal vpi vci* [*qos*]

/etc/atmarp -d *host*

IMPLEMENTATION

Cray Research systems with IOS model E and HIPPI channel

CRAY J90 series

CRAY EL series

DESCRIPTION

The atmarp command configures and displays the Asynchronous Transfer Mode (ATM) address resolution protocol (ARP) table on Cray Research systems that have a Native ATM connection or BBG ATM connection.

The atmarp command accepts the following options:

-a Displays the current ATM ARP table entries.

-v Produces a verbose display of the table entries.

-s Sets a permanent ATM ARP table entry. The following are the parameters:

host The IP host name of the remote host. If it is an alias, it must be in the /etc/hosts file.

ifc Interface name (for example, atm0, atm1, bbg0:atm1 ...) that this system uses to reach the remote host. The name is as it appears in the output of the netstat(1B) command with the -i option.

aal The ATM adaption layer (AAL) to be used by this Permanent Virtual Circuit (PVC). This value is based on ATM standards. Specify this number in decimal form.

vpi The Virtual Path Identifier (VPI). The VPI is placed into each ATM cell header so that the cell can be routed through the ATM network. Specify this number in decimal form.

vci The Virtual Channel Identifier (VCI). The VCI is placed into each ATM cell header so that the cell can be routed through the ATM network. This number should be between 32 and 1023. Consult your local network administrator when determining the VCI. Specify this number in decimal form.

qos The quality of service. This is the peak data rate (expressed in kilobits per second), at which this host delivers ATM cells to the remote host through the ATM interface. The default value of 0 causes the peak rate control feature to be disabled when sending to this remote host, thus allowing unlimited bandwidth. Specify this number in decimal form.

`-d` *host* Deletes an ATM ARP table entry associated with the host specified.

SEE ALSO

`netstat`(1B) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`bbg`(4) (Available only online)

NAME

automount – Mounts NFS or NFS version 3 file systems automatically

SYNOPSIS

/etc/automount [-f *mapfile*] [-m] [-n] [-tl *duration*] [-tm *interval*] [-tw *interval*] [-v] [-M *tmp_dir*] [-T] [*directory* [*mapname*] [-*mount-options*]]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The automount daemon automatically and transparently mounts an network file system (NFS) whenever a file or directory within that system is opened. automount creates a daemon (using fork(2), which appears to the kernel to be an NFS server; this daemon intercepts lookups on the specified *directory* and uses the map contained in *mapname* to determine the server, exported file system, and appropriate mount options for a given file system. The specified map must be a file on the local system. *directory* is a full path name starting with a /.

When supplied, -*mount-options* consists of a leading - and a comma-separated list of mount(8) options; if mount options are specified in the map, however, those in the map take precedence.

When the file system is mounted, members of the *directory* are made available by use of a symbolic link to the real mount point within a temporary directory. If *directory* does not exist, the daemon creates it; it is removed automatically when the daemon exits.

The automount daemon accepts the following options:

-f *mapfile*	Specifies the automount map file.
-m	Suppresses the initialization of *directory-mapname* pairs listed in the auto.master network information service (NIS) database.
-n	Disables dynamic mounts. With this option, references made through the automount daemon succeed only when the target file system was mounted previously. This option can be used to prevent NFS servers from cross-mounting each other.
-tl *duration*	Specifies the number of seconds that a looked-up name remains cached when not in use. The default is 5 minutes.
-tm *interval*	Specifies the number of seconds between attempts to mount a file system. The default is 30 seconds.
-tw *interval*	Specifies the number of seconds between attempts to dismount file systems that have exceeded their cached times. The default is 60 seconds.
-v	Specifies verbose mode. automount reports what it is doing on standard output.
-M *tmp_dir*	Specifies the directory under which the real NFS mounts occur.

`-T`	Specifies trace mode. Expands each NFS call. You must redirect the output of this option to a file.
directory	Specifies the name of the directory on which lookups are performed.
mapname	Specifies the name of the map that contains the mount options.
-mount-options	Specifies a list of mount options.

Maps

An `automount` map is composed of a list of mappings, with one mapping per line. Each mapping is composed of the following fields:

basename [*-mount-options*] *location* [...]

The *basename* field is the name of a subdirectory within the *directory* specified in the `automount` command line or in a master map specified by the `-f` option; it is not a relative path name. The *location* field consists of an entry of the following form:

host `:`*directory*[`:`*subdir*]

The *host* field is the name of the host from which to mount the file system, *directory* is the path name of the directory to mount, and *subdir* (when supplied) is the name of a subdirectory to which the symbolic link is made. You can use this argument to prevent duplicate mounts when multiple directories in the same remote file system are accessed.

You can continue a mapping across line breaks by using a \ as the last character before the new-line character. Comments begin with # and end at the subsequent new-line character.

If more than one *location* is supplied, there is no guarantee as to which location will be used; the first location to respond to the mount request gets mounted. The *mount-options* field can be used to supply options to the `mount`(8) command for the mounted file system.

Since the automounter maps do not have the ability to specify a file system type, specifying '`NFS3`' on either the '*mount-options*' command line or in an automounter map file will cause the automounter to mount a NFS version 3 file system type instead of an NFS file system type. If your system is not licensed for ONC+™ this *mount-option* is ignored.

Special Maps

Currently, two special maps are available. The `-hosts` map specifies mounts of all exported file systems from any host. For example, if the following `automount` command is already in effect, a reference to `/net/hermes/usr` will initiate an automatic mount of all file systems from `hermes` that `automount` can mount:

```
automount -m /net -hosts
```

References to a directory under `/net/hermes` refer to the corresponding directory on `hermes`. The `-passwd` map uses the `passwd`(5) database in its attempt to locate the home directory of a user. For example, assume that the following `automount` command is already in effect:

```
automount -m /homes -passwd
```

If the home directory shown in the password entry for the user *username* has the form */dir/server/username*, and *server* matches the host system on which that directory resides, the result of references to files in `/homes/`*username* is that the file system containing that directory is mounted if necessary, and all such references refer to that user's home directory.

BUGS

Shell file name expansion does not apply to objects not currently mounted or cached. For example, in the preceding example, the `ls /net/*` command might not list `hermes` as a subdirectory of `/net`.

EXAMPLES

The following example provides `automount` access to the exported file systems of any host in the `/etc/hosts` file, by prefixing the path name with `/net/`*hostname*`/ :`, as follows:

```
tutorial# automount  -m  /net  -hosts
```

You can then perform any directory operation on the `/net` subdirectory, as in the following example, using `ls`(1):

```
tutorial% ls /net/hermes/usr/src
```

FILES

`/tmp_mnt`	Directory under which file systems are mounted dynamically by default; you can specify this directory by using the `-M` option.

SEE ALSO

`mount`(8)

NAME

bb – Creates relative bad block file from ASCII flaw table files

SYNOPSIS

/etc/bb [-g] [-p] *special* [*aft_files*]

/etc/bb [-g] [-p] -D *device* -C *cylinder* -H *head* -S *sector special*

IMPLEMENTATION

Cray PVP systems (except CRAY J90 series and CRAY EL series)

DESCRIPTION

The bb command has two forms. The first format takes input from one or more ASCII flaw tables (aft(5)) files and writes to the standard output a relative bad block table for the given special file, *special*. By default, bb references files in the /etc/aft directory. These files are named for the physical devices they represent and are usually created by using the ift(8) command. If no *aft_files* are listed, bb assumes standard device names. If *aft_files* are listed, they must be presented in the order in which the logical device is constructed. It is strongly recommended that you use the default names.

In its alternate form, you can use bb to locate a relative bad block on the special file, *special*, given the physical device, cylinder, head, and sector by using command-line option.

The *special* file must be a character or block special disk type device. Use the output of bb as input to the mkfs(8) and mkdmp(8) commands.

The bb command accepts the following options:

-g Lists good blocks, which are areas between the bad blocks.

-p Physical mode. Prints physical block numbers (sectors).

In its alternate form, bb requires the following information:

-D *device* Target device.

-C *cylinder* Target cylinder.

-H *head* Target head.

-S *sector* Target sector.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Allowed to specify any file.

`sysadm` Allowed to specify any file, subject to security label restrictions on the file's path. Shell redirected I/O is subject to security label restrictions.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to specify any file.

EXAMPLES

Example 1: An ASCII flaw table (`aft`) file is created using the `ift` command:

```
ift /dev/ift/0134 >/etc/aft/0134
```

Example 2: A typical use of the `bb` command is to initialize the dump device:

```
bb /dev/dsk/dump | mkdmp -b /dev/dsk/dump
```

Example 3: In its alternate form, `bb` is used to locate the logical block number of a given physical disk address:

```
bb -D 0230.0 -C 100 -S1 -H1 /dev/dsk/root.a
```

FILES

`/etc/aft/*`

SEE ALSO

`ift`(8), `mkdmp`(8), `mkfs`(8)

`aft`(5), `dsk`(4), `pdd`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

bds – Bulk data server

SYNOPSIS

bds [*options*] ...

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

bds is a user level server, implemented as an enhancement to NFS, that provides direct I/O capabilities over the network. bds can be thought of as a remote DMA server: read/write requests are passed from client to the server; the server performs the request on the client's behalf. bds has no inherent block size limitation (other than the server DMA limit), and does not pass data through the file system caches on either the server or the client.

bds is normally started out of the startup scripts at boot time.

The bds command accepts the following options:

-*debug* Turns on debugging (sent to stderr).

-*devnull* Discards data written to the BDS server. Used for network debugging and performance testing only.

-*devzero* Generates zero filled data for read requests. Used for network debugging and performance testing only.

-*log* Logs open and close requests to stderr. close requests print out performance statistics.

-*touch* Touches the data after each I/O operation. Simulates local file system overhead when used with debugging and performance testing options.

CAUTIONS

bds binds to a fixed port number (port 2050). If the server is killed while there is an active connection, the port iwll remain "busy" until TCP determines there are no stray packets coming. Kill client applications before restarting the server.

BUGS

bds does not update file attributes during read/write operations. Applications that depend on immediate attribute updates after reads/writes may display unexpected behavior.

SEE ALSO

mount(8)

NAME

`bmap` – Identifies a block on a given file system

SYNOPSIS

```
/etc/bmap bno fsdev
/etc/bmap -p pbno pdev
/etc/bmap -p [-d] cyl trk sec pdev
```

IMPLEMENTATION

Cray PVP systems except CRAY J90 series and CRAY EL series

DESCRIPTION

The `bmap` command discovers the current use of a given block on a given file system. This is useful when an uncorrectable disk error occurs on a block and it needs to be put in a flaw table.

If a system administrator knows to what file a block belongs, the administrator can inform the file owner of the loss or restore it from backups.

You can specify the following operands on the command line. If a number given for the *bno*, *pbno*, *cyl*, *trk*, or *sec* operand begins with 0, it is assumed to be an octal number.

bno	Logical block number relative to file system.
fsdev	Block special device name containing file system.
`-p`	Indicates physical block is being specified.
`-d`	Indicates that disk relative sector is being given.
pbno	Physical block (512 words) offset on device.
pdev	Physical device name.
cyl	Cylinder.
trk	Track
sec	Sector

The specified block can be any of the following types:

Type	Description
`boot block`	Unused
`super block`	File system super block
`copy N of super block`	Copy N of file system super block
`dynamic super block`	File system dynamic block
`shared filesystem block`	(For 9 systems) Shared file system block (SFS only)

`inode region map block`	File system inode bit map block
`an inode block for inodes P.R.N - M`	File system block that contains inodes in partition P, region R, numbers N through M
`filesystem bit map block`	File system block that contains bit map of available space
`free block`	Unassigned file system blocks
`inode`	Inode for the named file
`data block`	Data block for the named file
`indirect data block`	Block of address extents for the named file
`ACL block`	Access control list (ACL) block for the named file
`bad block`	Flawed block belonging to inode 0
`PAL block`	Privilege assignment list (PAL) block for the named file

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	**Action**
`system`, `secadm`	Allowed to use this command.
`sysadm`	Allowed to use this command. Shell redirected output is subject to security label restrictions.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

In the following example, block 15,944 is the ACL block for the file `test.out`. The output from the command is as follows:

```
%  bmap 15944 /dev/dsk/scr100

#
block 15944 is ACL block for file ./test.out
physical address:  49-A1-22 cylinder 01270 track 03 sector 032 block 234008
(After adding spiral offset:  49-A1-22 cylinder 01270 track 03 sector 032)
#
```

SEE ALSO

`dmap`(8), `fsmap`(8)

General UNICOS System Administration, Cray Research publication SG–2301

NAME

brc, bcheckrc, rc, rc.mid, rc.pre, rc.pst – Invokes system initialization shell scripts

SYNOPSIS

```
/etc/bcheckrc
/etc/brc
/etc/rc
/etc/rc.mid
/etc/rc.pre
/etc/rc.pst
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

These shell scripts are executed by init(8) using entries in /etc/inittab when changing the system out of single-user mode.

The brc and bcheckrc shell scripts execute only the first time the system is changed out of single-user state (that is, after a reboot). The rc script can be executed whenever init(8) changes the run-level of the system. Usually, init(8) executes rc when changing from single-user mode to run-level 2, the usual multiuser run-level.

As released by Cray Research, the bcheckrc, brc, and rc scripts are not intended to be modified directly. The actions of these scripts can be controlled by setting environment variables in the /etc/config/rcoptions file. Additionally, the rc script calls several user-customizable subsidiary scripts, to allow for local configuration actions that fall outside of the usual actions performed by rc. (If you find it necessary to modify the bcheckrc, brc, or rc scripts to accomplish your necessary system startup actions, notify Cray Research so that the modification can be considered for inclusion in future released versions of these scripts.)

The bcheckrc script performs necessary system checks before the rc script can be executed. The bcheckrc script will set the system date and time as controlled by the RC_DATE environment variable. The bcheckrc script will check file system consistency by calling the mfsck(8) utility. The RC_FSCK environment variable controls whether the mfsck -u option will be used for this check. The bcheckrc script should not be used to start processes, because those processes will be killed and not restarted during the subsequent system shutdown or startup.

The brc script is intended for use in initializing hardware devices; it also copies raw core dump files to a regular UNICOS file in a separate file system, by executing the coredd(8) shell script. The brc script should not be used to start processes, because those processes will be killed and not restarted during the subsequent system shutdown or startup.

The rc shell script performs the following functions:

- Reads the environment variables in the /etc/config/rcoptions file to determine which actions it should perform (see the ENVIRONMENT VARIABLES section).
- Saves the existing contents of the startup output log file (specified by the RC_LOG environment variable; default is /etc/rc.log) to the same log file name with a suffix of .PREF (for example, /etc/rc.log.PREV).
- Logs the time of system startup and the version of rc performing the startup.
- Starts the file system error log daemon fslogd(8).
- Executes the local script rc.pre to perform any local initialization configured by the system administrator. (At this point in system startup, no file systems, including the /usr file system, have been mounted. Thus, any initialization performed in the rc.pre script must **not** attempt to access any files, commands, or directories that are not on the root file system. This includes the commands in /usr/bin and /usr/ucb. For initialization that occurs after the file systems have been mounted, see rc.mid below.)
- When coming from single-user mode, unmounts all currently-mounted file systems (to clear the way for the later file system mounts).
- Mounts the /tmp file system, using the device specified by the TMPDEV environment variable. If no device is specified by the TMPDEV environment variable, the rc script will use the device specified by the first entry for /tmp in the /etc/fstab file. As controlled by the RC_MKTMP environment variable, the rc script may first reinitialize the /tmp file system by using the mkfs(8) command.
- Mounts the /usr file system, using the device specified by the USRDEV environment variable. If no device is specified by the USRDEV environment variable, the rc script will use the device specified by the first entry for /usr in the /etc/fstab file.
- Mounts the /usr/tmp file system, using the device specified by the USRTMPDEV environment variable. If no device is specified by the USRTMPDEV environment variable, the rc script will use the device specified by the first entry for /usr/tmp in the /etc/fstab file. As controlled by the RC_MKTMP environment variable, the rc script may first reinitialize the /usr/tmp file system by using the mkfs(8) command.
- Mounts the user file systems specified in the /etc/fstab file and as controlled by the CRI_RC for the entries in that file.
- Removes world access to the /usr/src directory.
- Mounts the /proc file system.
- Adjusts the SECDED processing parameters to reduce system overhead from memory errors.
- Initializes the logical device cache by executing the ldcache(8) command with input from the /etc/config/ldchlist file.
- Initializes security by starting the security log daemon, setting security wildcard files, and initializing the network access lists.

- Preserves any interrupted vi(1) or ex(1) sessions.
- Executes the local script rc.mid to perform any local initialization configured by the system administrator. (This is the first local initialization script that is called after the user file systems have been mounted.)
- Performs administrative cleanup by saving mail from any system dump that was collected, removing account and mail queue lock files, and managing /usr/adm/sulog, the cron(8) log file (location specified by the RC_CRONLOGDIR environment variable; default is /usr/lib/cron), and the NQS log file (specified by the RC_NQSLOGFILE environment variable; default is /usr/spool/nqs/log).
- Initializes the /etc/wtmp file.
- Starts system accounting as controlled by the RC_ACCT environment variable.
- Starts the system activity daemon sadc(8) as controlled by the RC_SADC environment variable.
- Starts the system daemons in the SYS1 by using the sdaemon(8) utility.
- Initializes system networking by calling the netstart(8) script, as controlled by the RC_NET environment variable.
- Starts the system daemons in the SYS2 by using the sdaemon(8) utility.
- Executes the local script rc.pst to perform any local initialization configured by the system administrator.
- Logs the time of multiuser startup in the /etc/boot.log file.
- Logs successful completion of system startup.

The rc script can be used for several run-level states. Use the who(1) command to get the run-level information.

The rc.pre, rc.mid, and rc.pst scripts are called by the rc script at various points to allow for local startup initialization that can not be accomplished through the existing mechanisms in the rc script. The rc script passes the name of the startup output log file to each of the rc.pre, rc.mid, and rc.pst scripts, so that these scripts can then explicitly append any necessary output to the log file at the discretion of the person responsible for maintaining those scripts.

ENVIRONMENT VARIABLES

Most of the actions performed by the various system startup scripts are controlled by environment variables that are read in from the /etc/config/rcoptions file. Typically, the following values for a controlling environment variable specify whether or not the script will perform the action:

Value	Description
YES	Perform the action
NO	Do not perform the action
ASK	Prompt the operator for whether to perform the action

Additionally, an action can be tailored to specific run-levels by specifying a *decision string* with the following format:

runlevels=*action*[:*runlevels*=*action* ...]

runlevels is a concatenation of the run-level characters for which the specified *action* will be used. The *runlevels*= portion can be omitted to specify all run-levels. For example, the following decision string specifies that the action will be performed for run-levels 2 and 3, the operator should be prompted for run-levels 4 and 5, and the action should **not** be performed for any other run-levels:

```
23=YES:45=ASK:NO
```

Various other environment variables are used not to control an action, but to specify a device name, log file name, and so on.

The startup scripts use the following environment variables:

Script	Description
DUMPFS	Specifies the device name of the file system to be used for copying system core dumps. The default value is core.
DUMPMPT	Specifies the mount point to be used when copying system core dumps.
DUMPDIR	Specifies the directory in which the dump directory will be created when copying system core dumps.
RC_ACCT	Controls whether the rc script will start system accounting.
RC_CONTERR	Controls whether the rc script will continue system startup despite errors.
RC_CRONLOGDIR	Specifies the directory where the cron(8) log file is stored. The default value is /usr/lib/cron.
RC_DATE	Controls whether the bcheckrc script sets the system date and time.
RC_FSCK	Controls whether the bcheckrc script uses the mfsck -u option for file system checks.
RC_LDCH	Controls whether the rc script will activate the logical device cache using the ldcache(8) command.
RC_LOG	Specifies the file in which the rc script will save startup output. The default value is /etc/rc.log. If the value is null, all startup output will be sent to the system console (/dev/console).
RC_MKTMP	Controls whether the rc script will initialize the /tmp file system (using the mkfs(8) command).
RC_MKUTMP	Controls whether the rc script will initialize the /usr/tmp file system (using the mkfs(8) command).

RC_NET	Controls whether the rc script will call the netstart(8) script to start the networks.
RC_NQSLOG	Controls whether the rc script will clear the NQS log file.
RC_NQSLOGFILE	Specifies the name of the NQS log file. The default value is /usr/spool/nqs/log.
RC_SADC	Controls whether the rc script will start the system activity daemon sadc(8).
RC_SECHIGH	Specifies the upper security level used when initializing the /tmp or /usr/tmp file systems. The default value is 0. On systems that use nonzero security labels, this value should be set to 16, or if syshigh/syslow security labels are used, this value should be set to syshigh.
RC_SECLOW	Specifies the lower security level used when initializing the /tmp or /usr/tmp file systems. The default value is 0. On systems that use syshigh/syslow security labels, this value should be set to syslow.
RC_SECMASK	Specifies the security compartment mask used when initializing the /tmp or /usr/tmp file systems. The default value is 0. On systems that use nonzero security labels, this value should be set to 0777777777777777777 (low-order 63 bits).
RC_USRMNT	Controls whether the rc script will mount the user file systems.
TMPDEV	Specifies the device name of the file system that the rc script will mount on /tmp.
TMPOPTS	Specifies additional options for the mkfs(8) command that will be used to initialize the /tmp file system. This can be used to specify file system quotas for /tmp.
USRDEV	Specifies the device name of the file system that the rc script will mount on /usr.
USRTMPDEV	Specifies the device name of the file system that the rc script will mount on /usr/tmp.
USRTMPOPTS	Specifies additional options for the mkfs(8) command that will be used to initialize the /usr/tmp file system. This may be used to specify file system quotas for /usr/tmp.

FILES

/etc/boot.log	Logs the times of successful multiuser startups.
/etc/config/ldchlist	Initialization file for logical device cache (ldcache(8)).
/etc/config/rcoptions	Contains environment variables for configuring startup actions.

`/etc/fstab`	File system mount entries. Used by the `rc` script to determine the `/tmp` and `/usr/tmp` file systems, if not specified by environment variables.
`/etc/rc.log`	Default log file for system startup output and information.

SEE ALSO

`coredd`(8), `cron`(8), `fslogd`(8), `init`(8), `ldcache`(8), `mfsck`(8), `mkfs`(8), `netstart`(8), `sadc`(8), `sdaemon`(8), `shutdown`(8)

`who`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`fstab`(5), `inittab`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

captoinfo – Converts a termcap description into a terminfo(5) description

SYNOPSIS

/usr/bin/captoinfo [-v...] [-1] [-w *width*] *file* ...

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The captoinfo command looks in *file* for termcap descriptions. For each description that is found, an equivalent terminfo(5) description is written to standard output, along with any comments found. A description that is expressed as relative to another description (as specified in the termcap tc= field) will be reduced to the minimum superset before being output.

The captoinfo command accepts the following options:

-v... Prints tracing information on standard error as the program runs. Specifying an additional -v option will cause more detailed information to be printed.

-1 Causes the fields to print, one to a line. Otherwise, the fields will be printed several to a line with a maximum width of 60 characters.

-w *width* Changes the output to *width* characters.

file ... If no *file* is specified, the TERMCAP environment variable is used for the file name or entry. If TERMCAP is a full path name to a file, only the terminal whose name is specified in the TERM environment variable is extracted from that file. If the TERMCAP environment variable is not set, the /etc/termcap file is read.

NOTES

Certain termcap defaults are assumed to be true. For example, the bell character (terminfo bel) is assumed to be ^G. The line-feed capability (termcap nl) is assumed to be the same for both cursor_down and scroll_forward (terminfo cud1 and ind, respectively). Padding information is assumed to belong at the end of the string.

The algorithm used to expand parameterized information for termcap fields such as cursor_position (termcap cm, terminfo cup) will sometimes produce a string which, though technically correct, may not be optimal. In particular, the rarely used termcap operation %n will produce strings that are especially long. Most occurrences of these nonoptimal strings will be flagged with a warning message and may need to be recoded by hand.

The short two-letter name at the beginning of the list of names in a `termcap` entry, a hold-over from an earlier version of the UNIX system, has been removed.

WARNINGS

The `captoinfo` command should be used to convert `termcap` entries to `terminfo`(5) entries because the `termcap` database (from earlier versions of UNIX System V) may not be supplied in future releases.

FILES

`/usr/lib/terminfo/?/*`	Compiled terminal description database

SEE ALSO

`infocmp`(8), `tic`(8)

`tput`(1), `tset`(1B) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`curses`(3) (available only online)

`term`(5), `terminfo`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

chargefee – Charges a fee to a user

SYNOPSIS

/usr/lib/acct/chargefee *login-name account-name fee*

/usr/lib/acct/chargefee -d *login-name account-name fee*

/usr/lib/acct/chargefee -D -d *login-name fee*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The chargefee command charges a fee to a user. chargefee writes a record to /usr/adm/acct/day/fee; the runacct(8) main accounting shell procedure or the csaperiod(8) command merges this record with other accounting records.

The chargefee command accepts the following options:

-d	Outputs the fee records in cacct format. When using Cray Research system accounting (CSA), you should specify this option.
-D	Uses the user's default account ID.
login-name	The user name from the /etc/udb file.
account-name	The account ID field from the /etc/udb file.

SEE ALSO

csaperiod(8), runacct(8)

NAME

chmem – Changes the system's notion of physical memory size

SYNOPSIS

Current:

```
/etc/chmem -s
/etc/chmem -m reserve|rsv [-s]
/etc/chmem -m release|rls [-s]
```

Obsolescent (may not be supported in future releases):

```
/etc/chmem [[+]inc[m]]
/etc/chmem [[+]inc[M]]
/etc/chmem [[-]inc[m]]
/etc/chmem [[-]inc[M]]
```

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

As of the UNICOS 8.3 release, the functionality of the chmem command changed. The older functionality, however, is still supported. This man page lists the current command usage under the heading "Current," and the older usage under the heading "Obsolescent."

Current

The chmem command displays the sizes of physical memory areas and dynamically configures maintenance memory at run time. Dynamic maintenance memory management is supported on CRAY T90 systems only. The chmem command accepts the following options:

`-s` Displays the sizes of the currently configured memory areas that comprise physical memory. If a memory area (as described above) is not currently configured, it is not displayed.

If the `-m` option is specified, the displayed sizes reflect the memory status after the operation designated by the `-m` option has been performed.

`-m reserve|rsv`
Reserves maintenance memory by reducing the overall size of configured physical memory by the amount required for diagnostics. The required space comes from user memory.

`-m release|rls`
Releases maintenance memory by restoring the overall size of configured physical memory. The space used for maintenance memory is returned back to user memory.

Physical memory can contain the following areas (the first two always exist; the last three are configuration-dependent and may not exist):

Memory Areas	Description
kernel memory	Three separate areas that are not necessarily contiguous: Space allocated at build time. Space allocated at boot time. Space allocated at run time.
user memory	Area where user processes reside.
ramdisk	Memory resident disk.
guest memory	Area where guest operating systems reside (mutually exclusive from downed memory).
downed memory	Area that was formerly used by diagnostics. This area is an artifact of the archaic form of `chmem` (mutually exclusive from guest memory).
maintenance memory	Area used by diagnostics.

Obsolescent

The `chmem` command is the administrator command interface to the `chmem`(2) system call. When invoked without the size specification argument (*inc*), `chmem` displays the value of the system's current notion of physical memory.

inc The *inc* operand can be preceded immediately by a `+` or `-` indicating a relative adjustment; otherwise, *inc* is taken as absolute size. The *inc* operand can be immediately followed by an `m` or `M`, indicating that *inc* is being expressed in Mwords.

Only an appropriately authorized user can adjust the system's current notion of physical memory.

NOTES

Current

To use the `chmem` command, a user must have read permission for `/dev/mem`.

Obsolescent

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`	Allowed to use this command.
`sysadm`	Allowed to use this command. Shell redirected I/O is subject to security label restrictions.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

Obsolescent

The following command line reduces the system's notion of physical memory by 8 Mwords:

```
chmem -8m
```

The following command line increases the system's notion of physical memory by 4 Mwords:

```
chmem +4m
```

SEE ALSO

chmem(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

NAME

chroot – Changes the root directory and executes a command

SYNOPSIS

/bin/chroot *newroot command*

IMPLEMENTATION

All Cray Research systems

STANDARDS

POSIX, XPG4

DESCRIPTION

The chroot command executes *command* relative to *newroot*. The meaning of any initial slashes (/) in path names is changed to *newroot* for a command and any of its children. Furthermore, the initial working directory is *newroot*.

The following command line creates the x file relative to the original root, rather than the new one:

```
chroot newroot command >x
```

Only an appropriately authorized user can use this command.

The new root path name is always relative to the current root: even if a chroot is currently in effect, the *newroot* operand is relative to the current root of the running process.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Allowed to use this command.
sysadm	Allowed to use this command. File access and shell redirected I/O is subject to security label restrictions.

If the PRIV_SU configuration option is enabled, the super user or a user with the CHROOT permbit is allowed to use this command.

CAUTIONS

Use extreme caution when referencing special files in the new root file system, because files above the newly defined root in the file structure will become inaccessible.

SEE ALSO

udbgen(8)

chdir(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

NAME

ckdacct – Checks the size of daemon accounting files

SYNOPSIS

/usr/lib/acct/ckdacct [-n *blocks*] *daemons*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The ckdacct command checks the size of the daemon accounting files and checks the amount of free space on the file system containing the /usr/adm/acct directory. If the size of the daemon accounting file exceeds 500 blocks (default) or exceeds the specified number of blocks, ckdacct starts a new accounting file by invoking the turndacct(8) command with the switch operand.

ckdacct also ensures that the ACCT_FS file system contains at least MIN_BLKS free blocks. If there is not this much free space, ckdacct turns off accounting for the specified daemons by invoking turndacct(8) with the off operand. ckdacct calls turndacct(8) with the on operand to re-enable daemon accounting when at least MIN_BLKS free blocks are available.

ACCT_FS is a parameter which defines the file system on which /usr/adm/acct resides and is defined in the accounting configuration file /etc/config/acct_config. The MIN_BLKS parameter also is defined in this file.

This feature is sensitive to the frequency at which ckdacct is executed. You should run ckpacct(8) periodically by using the cron(8) command.

The ckdacct command accepts the following option, argument, and operand:

-n *blocks* Specifies the maximum size (in blocks) to which the daemon accounting files can grow before they are switched. The default is 500 blocks.

daemons Specifies a list of daemons for which the accounting file sizes are checked. Daemon names are separated by white space. Valid daemon names are nqs, tape and socket.

In the released template of the accounting configuration file, /etc/config/acct_config, ACCT_FS is set to /usr. If this is not correct for your system, you must define ACCT_FS properly in /etc/config/acct_config.

EXAMPLES

The following example is a suggested entry for the /usr/spool/cron/crontabs/root file so that cron(8) automatically runs ckdacct on the hour:

```
0   *   *   *   *   /usr/lib/acct/ckdacct nqs tape
```

FILES

`/etc/config/acct_config`	Accounting configuration file
`/usr/adm/acct/day`	Directory that contains current daemon accounting files
`/usr/spool/cron/crontabs/root`	Root `crontab`(1) file

SEE ALSO

`acctsh`(8), `cron`(8), `csa`(8), `turndacct`(8)

`crontab`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

ckpacct – Checks the size of the process accounting file

SYNOPSIS

/usr/lib/acct/ckpacct [*blocks*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The ckpacct script checks the size of the process accounting file /usr/adm/acct/day/pacct and checks the amount of free space on the file system containing the /usr/adm/acct directory. If the size of the pacct file exceeds 500 blocks (default) or exceeds the specified number of blocks, ckpacct starts a new accounting file by invoking the turnacct(8) command with the switch operand.

ckpacct also ensures that the ACCT_FS file system contains at least MIN_BLKS free blocks. If there is not this much free space, ckpacct turns off process accounting by invoking turnacct(8) with the off operand. ckpacct calls turnacct(8) with the on operand to re-enable process accounting when at least MIN_BLKS free blocks are available.

ACCT_FS is a parameter which defines the file system on which /usr/adm/acct resides and is defined in the accounting configuration file /etc/config/acct_config. The MIN_BLKS parameter also is defined there.

This feature is sensitive to the frequency at which ckpacct is executed. You should run ckpacct periodically using the cron(8).

The ckpacct script accepts the following operand:

blocks Specifies the maximum size (in blocks) to which the process accounting file can grow before it is switched. The default is 500 blocks.

In the released template of the accounting configuration file, /etc/config/acct_config, ACCT_FS is set to /usr. If this is not correct for your system, you must define ACCT_FS properly in /etc/config/acct_config.

EXAMPLES

The following example is a suggested entry for the /usr/spool/cron/crontabs/root file so that cron(8) automatically runs ckpacct on the hour:

```
0   *   *   *   *   /usr/lib/acct/ckpacct
```

FILES

`/etc/config/acct_config`	Accounting configuration file
`/usr/adm/acct/day/pacct*`	Process accounting files
`/usr/spool/cron/crontabs/root`	Root `crontab`(1) file

SEE ALSO

`acctsh`(8), `cron`(8), `csa`(8), `turnacct`(8)

`crontab`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

cleantmp – Deletes job temporary directories

SYNOPSIS

```
/etc/cleantmp username path
/etc/cleantmp  all
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The cleantmp command accepts the following options and operands:

username path When specified with a *username* and *path*, deletes job temporary directories and temporary directories created by tmpdir(1). cleantmp is called by init(8) and the Network Queuing System (NQS).

all When specified from single-user mode, deletes job temporary directories and temporary directories created by tmpdir(1) that were not deleted normally due to a system crash or another problem.

NOTES

The cleantmp command should not be used in multiuser mode.

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

MESSAGES

The cleantmp command prints its error messages on the system console.

FILES

${TMPDIR}/.tmpdir	File that contains list of temporary directories.
${TMPDIR}/.tmpdir[a-z]	On a UNICOS system using multilevel security labels, the files created by tmpdir(1) have names ending in a letter a through z. This naming convention enables a user to successfully change security levels and/or compartments (up to a maximum of 26 times) and still access the temporary directory feature.

SEE ALSO

`init`(8)

`tmpdir`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

NAME

cll – Lists or resets the login failure attempts field in user database (UDB)

SYNOPSIS

```
/etc/cll -r user
/etc/cll -R
/etc/cll -l user
/etc/cll -L
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The cll command is used by an appropriately authorized user to list or reset the failed login attempts for one user or all users. The number of failed login attempts is recorded in the logfails field of the user's entry in the user database (UDB).

The cll command accepts the following options (at least one of these options must be specified on a command line):

-r *user* Resets the login failed attempts field for the specified *user*. This option can be used alone or with the -R option.

-R Resets the login failed attempts field for all users. This option can be used alone or with the -r option.

-l *user* Lists the login failed attempts field for the specified *user*. This option can be used alone or with the -L option.

-L Lists the login failed attempts field for all users. This option can be used alone or with the -l option.

To lock out a user, the following conditions must be met:

- The MAXLOGS parameter must be greater than 0.
- The DISABLE_ACCT parameter must be enabled.
- The DISABLE_TIME parameter must be greater than 0.

The MAXLOGS, DISABLE_ACCT, and DISABLE_TIME parameters are defined in the uts/cf.SN/config.h file.

If these conditions are met and logfails equals or exceeds the value defined by MAXLOGS, the user is locked out of the system until the logfails field is reset by an appropriately authorized user or the number of seconds specified by the DISABLE_TIME parameter has elapsed.

NOTES

If this command is installed with a privilege assignment list (PAL), a user who is assigned the following privilege text upon execution of this command is allowed to perform the action shown:

Privilege Text	Action
`exec`	Allowed to use this command.

If this command is installed with a PAL, a user with one of the following active categories is allowed to perform the action shown:

Active Category	Action
`system`, `secadm`	Allowed to use this command.

If `PRIV_SU` is enabled, the super user is allowed to use this command.

FILES

`/etc/udb`	User database file
`/etc/udb.public`	Public user database file
`/etc/udb_2/udb.index`	Public extension index file
`/etc/udb_2/udb.priva`	Private field extension file
`/etc/udb_2/udb.pubva`	Public field extension file
`uts/cf.SN/config.h`	Kernel parameter definition file

SEE ALSO

`udbgen`(8)

`login`(1), `privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`udb`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

cm – Moves a spindle in IOS model E disk configuration

SYNOPSIS

cm [-D *ddir*] [-P *pdir*] [-a *value*] [-d] [-r] [-V] [-v] *from to*

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

DESCRIPTION

The cm command (configuration move) moves the configuration of slices or partitions from physical address *from* to physical address *to*. Additional options allow you to specify an ASCII summary of the physical address, move the data, and produce debug listings.

The program recognizes two directories: /dev/dsk, the location of mountable names, and /dev/pdd, the location of the physical disk devices. The scan of the disk farm begins in the /dev/dsk directory. The scan progresses recursively down the logical device definitions to the physical device definitions, usually in /dev/pdd, that contain the channel and unit numbers. This algorithm ignores physical devices that are not a part of a mountable logical device.

The cm command accepts the following operands, which are required:

from, *to* Physical addresses. In most cases, these are numeric character strings that are converted to internal values by using standard conventions for numeric conversion (0 x 40 = 0100 = 64). For DD-60 devices, a unit number may follow a period -- 0232.3.

To save the spindle configuration summary in an ASCII readable file, the *from* or *to* values can be file names.

from and *to* values of - correspond to stdin or stdout.

The cm command accepts the following options:

-D *ddir* Replaces the /dev/dsk as the name of the directory where the device scan begins recursively processing devices. This option is intended for debug purposes.

-P *pdir* Replaces /dev/pdd as the directory name where the physical devices are placed when a mknod is performed. Note that the physical devices are found by following logical device specifications from /dev/dsk. This option is intended for debug purposes.

-a *value* The -a option accepts two values: i and o. If the value is i, the *from* argument names an ASCII input file that contains the configuration to be installed at the *to* address. If the value is o (or not-i), the *to* argument names a file that will become the ASCII output description of the file.

`-d` Indicates that data should be moved. If the two arguments to the `cm` program are physical disk addresses, the movement of data will be transparent to the user when the file systems are remounted. If a version of the `-a` option is in effect, the data associated with `/dev/pdd/name_1` will be copied to/from `name_1` in the current directory. The data copy routine was designed to deal with failing input partitions; it will seek over areas where the read fails with EIO.

`-r` Removes the configuration at the *from* address. If the `-a` option, in either variety has not been specified, the removal of the *from* configuration is unavoidable. This flag is meaningful only if `-a o` is specified.

`-V` Indicates verbose no-operation.

`-v` Indicates verbose operation. Each `mknod`, `rename`, `unlink`, and `dd` is announced before execution.

NOTES

Use the following command to survey the situation:

```
cm -ao 0nnn.k -
```

Work carefully. This utility has the potential to deconfigure everything in the disk farm.

EXAMPLES

Example 1: Removes all devices at address 0204.5:

```
cm -r -a o 0204.5 /dev/null
```

Example 2: Replaces address 0206 with 0210:

```
cm -d 0206 0210
```

Example 3: Uses scratch files in the current directory to hold information while the device at 0206.6 is being replaced:

```
cm -a o -d 0206.6 savit
# pause for disk replacement
# pause for new flaws on 0206.6
cm -a i -d savit 0206.6
```

Example 4: Changes spindle 0234.7 with 0236.6, using scratch files in the current directory to hold the data:

```
cm -rdao 0234.7 xxfile
cm -rd   0236.6 0234.7
cm -dai  xxfile 0236.6
```

NAME

coredd – Automatically copies raw core dump files to a regular UNICOS file in a separate file system

SYNOPSIS

/etc/coredd

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The coredd shell script is executed by brc(8) when changing the system out of single-user mode (see inittab(5)). Dump file system (Dumpfs), dump mount point (Dumpmpt), and dump directory (Dumpdir) are passed to coredd from brc. They are set by using the configure system -> special system device definitions -> menu in the install tool. The default block device is /dev/dsk/core; the default file system and dump directory is /mnt. The Dumpfs dump file system is mounted on the Dumpmpt dump mount point. coredd calls the cpdmp(8) command, which is executed to determine whether a dump exists. If a dump exists, coredd performs the following operations:

- Creates a subdirectory in the /Dumpmpt/Dumpdir/ directory to contain the system dump information. The name of this subdirectory is a time stamp of when the dump was moved, in the format /Dumpmpt/Dumpdir/*mmddhhmm* (for example, /<mnt>/10120823).
- Calls cpdmp to copy the dump to the /Dumpmpt/Dumpdir/*mmddhhmm* subdirectory in a file called dump.
- Copies the version of the UNICOS operating system (that is, the kernel) that was running at the time of the crash. This is put in the /Dumpmpt/Dumpdir/*mmddhhmm* subdirectory in a file called unicos.
- Copies the version of crash(8) to be used with the dump; this is put in the /Dumpmpt/Dumpdir/*mmddhhmm* subdirectory in a file called crash.

If an error is detected in processing the dump, a message is sent to the console asking for operator intervention to move the dump, unicos, and crash files after the system is up in multiuser mode.

MESSAGES

The coredd script has been changed to mount /dev/dsk/core onto /mnt instead of /core. At boot time, this causes a warning message, as follows:

```
<core> mounted as <mnt>
```

This message should be ignored.

SEE ALSO

`brc`(8), `cpdmp`(8), `crash`(8)

`dd`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`inittab`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

cpdmp – Processes one or more IOS SYSDUMP areas

SYNOPSIS

/etc/cpdmp [-n] [-c] [-l *logfile*] [-i *rawdump*] [-o *dumpfile*]

IMPLEMENTATION

Cray PVP systems with IOS model E

DESCRIPTION

The cpdmp command processes one or more I/O subsystem (IOS) SYSDUMP areas, combining them into one system dump file. It is executed by the coredd(8) script when changing the system out of single-user mode (see inittab(5)).

The system dump contains a header that details the memories and ranges that were dumped. cpdmp reformats the dump and header to a format recognized by crash(8).

The cpdmp command accepts the following options:

-n	Does not copy system dump; sets exit code to reflect SYSDUMP status.
-c	Clears the dump copied flag; dump is not moved.
-l *logfile*	Specifies the log file name. The default is /etc/dump.log.
-i *rawdump*	Specifies the raw dump device. The default is /dev/pdd/dump.
-o *dumpfile*	Specifies the system dump file name. The default is /core.sys.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm, sysops	Allowed to specify any file.

If the PRIV_SU configuration option is enabled, the super user is allowed to specify any file.

EXIT STATUS

The cpdmp command generates the following exit codes:

Exit Code	Description
2	No SYSDUMP area to process.
1	Fatal error encountered; unable to process input or generate system dump file.
0	SYSDUMP successfully processed or available for processing (-n option).

FILES

`/dev/pdd/dump`	Device from which to copy the dump image
`/core.sys`	File to which dump image is written
`/etc/dump.log`	File containing time, memories, and ranges for each dump

SEE ALSO

`coredd`(8), `crash`(8)

`inittab`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

`mfsysdmp`(7) (online only)

NAME

cpset – Installs object files in binary directories

SYNOPSIS

/usr/bin/cpset [-o] [-n] *object destination* [*mode* [*owner* [*group*]]]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The cpset command installs the specified *object* file at the given *destination*. If *destination* already exists, it is removed before *object* is installed. If *destination* is a directory, *object* is installed as a file named *destination*/*object*. The *mode*, *owner*, and *group* of the destination file can be specified on the command line. If this data is omitted, two results are possible:

1. If you are an appropriately authorized user, the following defaults are provided:

   mode = 0755
   owner = bin
   group = bin

2. Otherwise, the default owner and group of the destination file are yours.

The cpset command accepts the following options:

-o Forces cpset to move *object* to OLD*object* in the *destination* directory before installing the new *object*.

-n Tells cpset to handle files that are "text busy" (ETXTBSY) by linking the files to OLD*object* in the destination directory before installing the new object. If the OLD*object* is also busy, cpset iteratively increments the name to OL2*object*, continuing to increment until a file name that is not busy is encountered.

The cpset command uses the /usr/src/destinations file to determine the final destination of a file. The *destination* file contains pairs of path names separated by spaces or tabs. The first name is the official destination (for example, /bin/echo). The second name is the new destination. For example, if echo is moved from /bin to /usr/bin, the entry in /usr/src/destinations would be as follows:

```
/bin/echo      /usr/bin/echo
```

When the actual installation occurs, cpset verifies that the old path name does not exist. If a file exists at that location, cpset issues a warning and continues. This file does not exist on a distribution tape; it is used by sites to track local command movement. The procedures used to build the source must define the official locations of the source.

If you are using `cpset` to move system binaries while generating a system (for example, while programming in a shell script or makefile that you have written), you should move to `$ROOT/`*directory/file* rather than to */directory/file*.

NOTES

`cpset` does not set access control lists (ACLs). ACLs must be set independently by using the `spset`(1) command.

If this command is installed with a privilege assignment list (PAL), a user who is assigned the following privilege text upon execution of this command is allowed to perform the action shown:

Privilege Text	Action
`id`	`cpset` attempts to set file owner, group, and mode to `bin:bin:0755`.

If this command is installed with a PAL, a user with one of the following active categories is allowed to perform the action shown:

Active Category	Action
`system`, `secadm`	Allowed to set file owner, group, and file mode to `bin:bin:0755`.

If the `PRIV_SU` configuration option is enabled, the super user or the user `bin` who belongs to the `bin` group is allowed to set the file owner, group, and mode to `bin:bin:0755`. If the user's effective ID is less than 100, `cpset` attempts to set the file owner, group, and mode to `bin:bin:0755.`

EXAMPLES

The following examples have the same effect (assuming that the user is appropriately authorized). The `echo` file is copied into `/bin` and is given access permissions of `0755`, owner ID of `bin`, and group ID of `bin`.

```
cpset echo /bin 0755 bin bin
cpset echo /bin
cpset echo /bin/echo
```

SEE ALSO

`make`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

NAME

cpu – Selects, dedicates, and changes mode bits per CPUs

SYNOPSIS

```
/etc/cpu mask | -n newmask [-D] [-r] [-m mode] command [arguments]
/etc/cpu [mask | -n newmask] [-D] [-r] [-m mode] -p pids
/etc/cpu mask | -n newmask -d | -u
/etc/cpu -d
/etc/cpu -t ticks
/etc/cpu mask | -n newmask -s | -S
/etc/cpu mask | -n newmask [-r] [-m mode] [-p pids] [command [arguments]]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The cpu command allows you to select one or more processors in which to run some designated processes.

The processors are chosen according to *mask* (obsolescent version) or -n *newmask* (current version).

The new style string consists of a comma-separated list (-n 0,3,4, ... *number-of-CPUs*) and/or a range designation (-n 0-5,10-14, ...).

Support for the old style is preserved for compatibility. The old style should be a string that contains letters (a through z) or numbers (0 through *7*). Of course, entries are limited to the number of CPUs on your system.

When the *command* operand is specified, cpu runs *command* (according to the specified arguments) on the chosen processors. When the -p option is specified, cpu runs each of the processes specified by *pid* on the chosen processors.

On Cray PVP systems, the *mask* argument is silently limited to the available processors. If all processors chosen are unavailable, the process is allowed to run on any processor.

The cpu command accepts the following option and operand on all Cray Research systems:

-n *newmask*
: Selects processors. The *newmask* string consists of a comma-separated list (-n 0,3,4, ... *number-of-CPUs*) and/or a range designation (-n 0-5,10-14 ...).

-p *pids*
: Specifies one or more process IDs with a comma-separated list or a spaces-separated list in quotes.

The cpu command accepts the following options and operands on CRAY Y-MP systems:

`-D` Dedicates the specified CPUs to the command or processes. The CPUs are freed when the process completes.

If used without a mask, any CPUs previously dedicated to the processes will no longer be dedicated to those processes.

`-d` If used with a mask, disables the CPUs specified by mask. The system does not let the last available CPU be disabled.

If used without a mask, displays the down CPUs. This option works on all Cray Research systems.

`-t` *ticks* Changes the system tick rate to the specified number of *ticks* per second. Valid values are 1 to 1000.

`-s` Disables scalar cache for CPUs specified by *mask* or *newmask*.

`-S` Enables scalar cache for CPUs specified by *mask* or *newmask*.

`-u` Enables the CPUs specified by mask.

`-r` Reports specific mode bits enabled or disabled.

`-m` *mode* Changes specific mode bits. Acceptable modes are as follows:

`monon` Monitor mode on (super user only).

`monoff` Monitor mode off (super user only).

`bdmon` Bidirectional memory on.

`bdmoff` Bidirectional memory off.

`emaon` Enables the EMA mode bit in the exchange package, which enables 24/32 bit address mode and the corresponding instructions.

`emaoff` Disables the EMA mode bit in the exchange package.

`avlon` Second vector logical on.

`avloff` Second vector logical off.

`fpeon` Floating-point interrupts on.

`fpeoff` Floating-point interrupts off.

`oreon` Operand range errors on.

`oreoff` Operand range errors off.

`icmon` Interrupt on correctable memory errors on.

`icmoff` Interrupt on correctable memory errors off.

`iumon` Interrupt on uncorrectable memory errors on.

`iumoff` Interrupt on uncorrectable memory errors off.

`immon` Interrupt on monitor mode on.

`immoff` Interrupt on monitor mode off.

`rpeon` Interrupt on register parity error mode on.

`rpeoff` Interrupt on register parity error mode off.

`scon` (CRAY T90 and CRAY J90 series) Scalar cache enabled/only runs on CPUs with cache

`scoff` (CRAY T90 and CRAY J90 series) Scalar cache disabled

The following modes are valid on CRAY T90 systems with IEEE floating point CPUs only:

`xion` IEEE floating point exceptional input interrupt on.

`xioff` IEEE floating point exceptional input interrupt off.

`nxon` IEEE floating point inexact interrupt on.

`nxoff` IEEE floating point inexact interrupt off.

`unfon` IEEE floating point underflow interrupt on.

`unfoff` IEEE floating point underflow interrupt off.

`ovfon` IEEE floating point overflow interrupt on.

`ovfoff` IEEE floating point overflow interrupt off.

`dvion` IEEE floating point divide by zero interrupt on.

`dvioff` IEEE floating point divide by zero interrupt off.

`nvion` IEEE floating point invalid interrupt on.

`nvioff` IEEE floating point invalid interrupt off.

`rm0on` IEEE rounding mode 0 on.

`rm0off` IEEE rounding mode 0 off.

`rm1on` IEEE rounding mode 1 on.

`rm1off` IEEE rounding mode 1 off.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	**Action**
`system`, `secadm`	Allowed to use this command.
`sysadm`	Allowed to use this command. Shell redirected I/O and access to specified processes are subject to security label restrictions.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

On IEEE CRAY T90 systems, the fpeon and fpeoff modes may be used to set and clear the IEEE floating point mode flags. The fpeon mode will turn on a default set of the IEEE interrupt modes (ovfon, dvion and nvion). The fpeoff mode will turn off all IEEE interrupt modes. Specifying any of the IEEE modes along with fpeon or fpeoff is an error.

The IEEE rounding mode specifications are as follows:

rm0off rm1off	Round to nearest
rm0on rm1off	Round up
rm0off rm1on	Round to zero
rm0on rm1on	Round down

MESSAGES

cpu sends messages about a *pid* that is not valid and about a mask character that is not valid.

EXAMPLES

Example 1: The following example downs CPUs 0, 2, and 3:

```
cpu -n 0,2,3 -d
```

Example 2: This command undedicates all CPUs from process 53536:

```
cpu -D -p 53536
```

SEE ALSO

cpselect(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

cpu(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

`crash` – Examines system core images

SYNOPSIS

`/etc/crash` [`-g` *name*] [`-s`] [*core_filename* [*namelist_filename*]]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `crash` command is an interactive utility for examining an operating system core image. It has facilities for interpreting and formatting the various control structures in the system and certain miscellaneous functions that are useful when perusing a dump.

The `crash` utility accepts the following options and operands

`-g` *name* Specifies the guest system name to operate on. A memory bias (guest load address) will be automatically added to all `crash` memory requests This option should be used only when a guest is active.

`-s` (Internal use only) Specifies whether the program is being executed from the command line or whether it has been called by `crash` on the host system. This parameter is used internally by `crash` and should not be used on the command line.

core_filename Specifies where the system image can be found. The default value of *core_filename* is `/dev/mem`, which lets you use `crash` without an operand to examine an active system. If you specify the system image file, it is assumed to be a system core dump and the default process is set to be that of the process active in the kernel at the time of the crash. This is determined by a value stored in a fixed location by the dump mechanism.

namelist_filename Specifies the binary matching *core_filename*. Its default value is `/unicos`.

Input to `crash` is typically of the following form:

command [*options*] [*slot numbers to be printed*] [`>` *file*] [`|` *command*]

If you specify no specific structure elements, all valid entries will be used. The following example would print the entire process table in standard format:

```
proc
```

When allowed, *options* modify the format of the printout. For example, the following command line prints process table slots 12, 15, and 3 in a long format:

```
proc - 12 15 3
```

Format data structure entries assume a decimal slot number (zero-based). Those commands that perform I/O with addresses assume an octal slot number.

The following subcommands are available for systems configured with Cray Research Distributed Computing Environment (DCE) Distributed File Service (DFS): `aggr`, `ccall`, `cct`, `ch`, `chtable`, `cm_conn`, `cm_serv`, `dcache`, `dfsmisc`, `dfsstat`, `fid`, `fshost`, `scache`, `sct`, `tkc`, `tkm`, `tkset`, `tpq`, and `volume`.

The `crash` command accepts the following commands:

`?` Prints synopsis of commands

`!` Escapes to shell

`q` Exits from `crash`

`.` Repeats the last command

`address` *table_name slot_number*
Aliases: `addr`
Specifies the address of the given table entry number. The address can then be used with the `od` subcommand.

`aggr` [`-`|`--`|`---`|`----`] [`-g`] [`-l`] [*addr*]
Alias: `ag`
(DCE DFS only) Displays aggregate entries (`struct aggr`) in the `ag_root` list.

`-`, `--`, `---`, `----`
Specifies verbose level; displays additional information in the output. Increasing the number of dashes (the `-` character) increases the amount of information that is displayed.

`-g` Produces a command and address in `grep`(1) format that can be used to display a specific structure.

`-l` Displays information about the lock structures contained in the structure(s) being displayed.

[*addr*] Limits the display to the data structure at the specified address. By default, all known data structures are displayed.

`aio` [*slot_numbers*]
Displays the specified `aio` (asynchronous I/O) entries. By default, all entries are displayed.

`bfreelist`
Displays the buffer cache free list.

`bfreemt`
Displays the buffer cache empty list.

`buf` [*buffer_headers*]
Aliases: `hdr`, `bufhdr`
Formats the specified system buffer headers.

`buffer` [*format*] [*buffers*]
Alias: `b`
Prints the data in the specified system buffer according to *format*. If you omit *format*, the previous *format* is used. Valid formats include `decimal`, `octal`, `hex`, `character`, `byte`, `directory`, `inode`, `och`, `parcel`, `instruction`, and `write`. The `write` format creates a file in the current directory (see the FILES section) containing the buffer data.

`callout`
Aliases: `calls`, `call`, `c`, `timeout`, `time`, `tout`
Prints all entries in the callout table.

`ccall` [`-`|`--`|`---`|`----`] [`-g`] [*addr*]
Alias: `ag`
(DCE DFS only) Displays the list of Kernel RPC client call handle entries.

`-`, `--`, `---`, `----`
Specifies verbose level; displays additional information in the output. Increasing the number of dashes (the `-` character) increases the amount of information that is displayed.

`-g` Produces a command and address in `grep`(1) format that can be used to display a specific structure.

[*addr*] Limits the display to the data structure at the specified address. By default, all known data structures are displayed.

`cct` [`-`|`--`|`---`|`----`] [`-g`] [*addr*]
(DCE DFS only) Displays the list of Kernel RPC client connection table entries.

`-`, `--`, `---`, `----`
Specifies verbose level; displays additional information in the output. Increasing the number of dashes (the `-` character) increases the amount of information that is displayed.

`-g` Produces a command and address in `grep`(1) format that can be used to display a specific structure.

[*addr*] Limits the display to the data structure at the specified address. By default, all known data structures are displayed.

`ch` [`-`] *slot* | *address* [*slot* | *addresses* ...]
Formats the DFS client handle. If the `-` option is specified, a long-form report is generated that includes the NFS mount information structure and the remote procedure call (RPC) client structure.

`chtable` [`-`]
Aliases: `chtab`, `cht`
(DCE DFS only) Formats the DFS client handle table. If the `-` option is specified, a long-form report is generated that includes the NFS mount information structure and the RPC client structure.

`cku_private` *address* [*address* ...]
Aliases: `cku_priv`, `ckup`
Formats the NFS `cku_private` structures.

`clkar` Prints the `clock.c` 'ticks' distribution table.

`clusters`
Aliases: `cl`
Displays mainframe cluster information. This command is meaningful only when a guest is active.

`cm_conn` [`-`|`--`|`---`|`----`] [`-g`] [`-l`] [`-s`] [*addr*]
Alias: `cmc`
(DCE DFS only) Displays the Cache Manager connection list entries found in the Cache Manager server list.

`-`, `--`, `---`, `----`
Specifies verbose level; displays additional information in the output. Increasing the number of dashes (the `-` character) increases the amount of information that is displayed.

`-g` Produces a command and address in `grep`(1) format that can be used to display a specific structure.

`-l` Displays information about the lock structures contained in the structure(s) being displayed.

`-s` Displays information about Cache Manager servers as well as Cache Manager connection list entries.

[*addr*] Limits the display to the data structure at the specified address. By default, all known data structures are displayed.

`cm_serv` [`-`|`--`|`---`|`----`] [`-c`] [`-g`] [`-l`] [*addr*]
Alias: `cms`
(DCE DFS only) Displays information from the list of Cache Manager server structures (`cm_server`).

`-`, `--`, `---`, `----`
Specifies verbose level; displays additional information in the output. Increasing the number of dashes (the `-` character) increases the amount of information that is displayed.

`-c` Displays information about Cache Manager connection list entries as well as Cache Manager servers.

`-g` Produces a command and address in `grep`(1) format that can be used to display a specific structure.

`-l` Displays information about the lock structures contained in the structure(s) being displayed.

[*addr*] Limits the display to the data structure at the specified address. By default, all known data structures are displayed.

`codecmp`
Aliases: `ccmp`, `code`, `cc`
Compares the code sections of *namelist_filename* to the same addresses in the dump. This option does not work for a kernel that was compressed using the `kcompress`(8) command. You must first decompress the kernel.

`core` *proc_table_slot_number* [*filename*]

Aliases: `co`, `corefile`

Creates a file that contains the memory image of a given process. If you specify *filename*, it writes the image to that file; otherwise, `core` writes the image to a file named `core`. The core image cannot be created if the process is swapped.

`cpu` Aliases: `cp`, `cpus`

Displays information regarding CPU activity.

`dcache` [`-`|`--`|`---`|`----`] [`-g`] [`-f` *file*] [*addr*]

Alias: `dc`

(DCE DFS only) Displays information about Cache Manager `dcache` structures.

`-`, `--`, `---`, `----`

Specifies verbose level; displays additional information in the output. Increasing the number of dashes (the `-` character) increases the amount of information that is displayed.

`-g` Produces a command and address in `grep`(1) format that can be used to display a specific structure.

`-f` *file* Dump contents of the `CacheItems` file. The *file* argument specifies the path to this file. This option cannot be used with the *addr* argument.

[*addr*] Limits the display to the data structure at the specified address. By default, all known data structures are displayed. This argument cannot be used with the `-f` option.

`dfsmisc -z` [`-`|`--`|`---`|`----`] [`-g`] [*addr*]

Alias: `dfs`

(DCE DFS only) Displays all items in the `zlc` (zero link count) list. If the `-z` argument is not specified, the `dfsmisc` command does nothing.

`-z` Displays all items in the `zlc` list. This option is required.

`-`, `--`, `---`, `----`

Specifies verbose level; displays additional information in the output. Increasing the number of dashes (the `-` character) increases the amount of information that is displayed.

`-g` Produces a command and address in `grep`(1) format that can be used to display a specific structure.

[*addr*] Limits the display to the data structure at the specified address. By default, all known data structures are displayed.

`dfsstat` [`-c`] [`-g`] [`-p`] [`-r`]

(DCE DFS only) Displays DFS counters. If no options are specified, `dfsstat` repeats the previous option(s) or displays Protocol Exporter statistics (`-p` option) if no previous invocation was made.

`-c` Displays Cache Manager statistics

`-g` Produces a command and address in `grep`(1) format that can be used to display a specific structure.

`-p` Displays Protocol Exporter statistics. This option is the default.

`-r` Displays RPC (remote procedure call) statistics

`dhbuf` Displays device hash headers.

`dnlc` [`-s`]
Prints a formatted list of the directory name lookup cache. If the `-s` option is specified, the lookup cache usage statistics are printed.

`ds` [*list of addresses*]
Finds the closest data symbol to the specified addresses.

`err` Displays the error log.

`eslice` [`-`] [*major_numbers*]
Displays IOS model E slice information. By default, a list of active major slices is displayed. The `-` option adds minor slice information. Specify a list of major slice numbers to limit the display to those major slices. When major slice numbers are specified, minor slice information for those slices is also displayed.

`fid` [`-`|`--`|`---`|`----`] [`-f`] [`-g`] [`-h`] [`-l`] [`-t`] [*addr*]
(DCE DFS only) Displays entries from the Token Manager file ID hash table. With no options, `fid` executes as if the `-f` option was specified.

`-`, `--`, `---`, `----`
Specifies verbose level; displays additional information in the output. Increasing the number of dashes (the `-` character) increases the amount of information that is displayed.

`-f` Displays entries in the file ID hash table (`fidhash`). This option is the default.

`-g` Produces a command and address in `grep`(1) format that can be used to display a specific structure.

`-h` Displays `fshost` (file system host) information.

`-l` Displays information about the lock structures contained in the structure(s) being displayed.

`-t` Displays Token Manager (`tkm`) information.

[*addr*] Limits the display to the data structure at the specified address. By default, all known data structures are displayed.

`file` [*file_table_entries*]
Aliases: `files`, `f`
Formats the specified file table.

`flock` [`-s`]
Alias: `fl`
Prints file locking information; `-s` prints a summary.

`fshost` [`-`|`--`|`---`|`----`] [`-g`] [`-l`] [*addr*]
(DCE DFS only) Displays information about all file system hosts (`fshosts`) found in either the `fshs_priHostID` table or the `fshs_secHostID` table.

`-`, `--`, `---`, `----`
Specifies verbose level; displays additional information in the output. Increasing the number of dashes (the `-` character) increases the amount of information that is displayed.

`-g` Produces a command and address in `grep`(1) format that can be used to display a specific structure.

`-l` Displays information about the lock structures contained in the structure(s) being displayed.

[*addr*] Limits the display to the data structure at the specified address. By default, all known data structures are displayed.

`gch` [*kernel_id*]
Formats and prints the guest channel (`gch`) table for all guests or for the guest with the specified kernel ID.

`gcom` Formats and prints the guest communications (`gcom`) table.

`gcx` Formats and prints the guest context (`gcx`) table.

`gpf` Formats and prints the guest performance (`gpf`) table for all guests.

`gpi` Formats and prints the guest pseudo interrupt (`gpi`) table.

`gpq` [*kernel_id*]
Formats and prints the guest packet queue (`gpq`) table for all guests or for the guest with the specified kernel ID.

`grt` Formats and prints the guest global resource (`grt`) table.

`gsn` [*count*]
Formats and prints the guest snapshot buffer and dumps the specified number of entries (32 by default) from the guest snapshot trace buffer in reverse chronological order.

`guest` [*kernel_name*]
Sets the dump memory bias to the specified guest system (by default, the first active guest system). All subsequent `crash` commands are relevant to the new memory bias. If a `crash` binary (associated by name) is available, the user will be asked if they wish to use that binary to examine the guest portion of the dump, as shown in the following example:

```
# ./crash dump unicos
> guest
Use ./crash_galeg1 (y|n)? > y
 Running ./crash_galeg1 -s -g galeg1 dump ./unicos_galeg1
>
```

`hblk` [*hblk_slot_numbers* | *addresses*]
Displays information about hash blocks (`hblks`) in the buffer cache. For example, `hblk 1 3 7` displays information about the blocks in slots 1, 3, and 7 of the hash blocks (`hblk`) table. Any *address* specified must be decimal.

`hbuf` [`-`] [`-c`] [`-l`] [`-s`] [*hbuf_slot_number* | *addresses*]
Displays information about hash header entries in the buffer cache. Any *address* specified must be decimal.

- `-` (same as `-l` option) Displays chains of `hblks` hashed to headers in addition to the default display.
- `-c` Displays only chains of `hblks` hashed to headers.
- `-l` Displays chains of `hblks` hashed to headers in addition to the default display.
- `-s` Displays hash header usage statistics.

`help` [*command*]
Aliases: `?`, `h`
If you specify `help` alone, prints a synopsis of all of the commands. If you specify *command*, `help` provides a more in-depth description of the specified command.

`host` Sets the dump memory bias to host memory (zero (0)). All subsequent `crash` commands are relevant to the new memory bias.

`inode` [`-`] [*inode_table_entries*]
Aliases: `ino`, `i`
The `inode` command has been disabled. Use the `vnode` command or the `nc1inode` command.

`iobuf` [`-`] [*iobuf_table_entries*]
Alias: `ddutab`
Prints the Physical Device table.

`jobs` [`-`] [*job_table_entries*]
Alias: `jtab`, `sess`
Prints the System Job (Session) Table. Without options, `jobs` generates a one-line description of all jobs (sessions). The `-` option produces a long listing of the `session` structure contents. Additional fields display the IPC usage: messages, semaphores, and shared memory (CRAY T90 systems only).

`kbds` Displays Kernel-Based Device Simulator information.

`kfp` [`-a`|`-k`|`-u`] [*argument_pointer stack_frame_pointer stack_base*]

Aliases: `r5`, `fp`

Prints the program's idea of the start of the current stack frame if you do not specify an argument, or sets the frame pointer to the supplied values. Register B01 contains the argument pointer, and B02 contains the stack pointer when a process is in the kernel. The `registers` command can be used to obtain the values of B01 and B02 from a dump. If *stack_base* is not present, the kernel stack is assumed. The `user` command can be used to get the address of the saved stack in the user structure and B01 and B02 for the three register save areas. Usually, the first save area contains the B01 and B02 registers that should be used. (The `stack` command defaults to these values.)

`-a` Specifies an alternate LAT table to use for address mapping. Selects absolute, one-for-one mapping.

`-k` Selects the kernel LAT table to use for address mapping.

`-u` Selects the user LAT table to use for address mapping.

`kftrace` [`1`] [`a`] [`c`] [`f`] [`k`] [`n`] [`t`]

Displays kernel flowtrace information. By default, `kftrace` sorts the output inversely by total time in function (the `t` argument).

`1` Excludes functions called only once

`a` Sorts output by average time in function

`c` Displays times in microseconds (default is raw clock periods)

`f` Sorts output inversely by calling frequency

`k` Stores information internally, so the same data can be displayed several ways

`n` Sorts output by name

`t` Sorts output inversely by total time in function (default)

`lat -d` [*addressing mode*]

Redefines the default LAT table and default mode. With no arguments, the `-d` option displays the default LAT table and mode.

addressing
Specifies type of addressing: `absolute` (all logical-to-physical address mapping is one-to-one), `kernel`, or `user`.

mode Specifies mode as any combination of `r` (read), `w` (write), or `x` (execute).

`lat` [`-k`|`-u`]

Displays the current kernel (`-k`) and/or user (`-u`) LAT tables.

`lat -k`|`-u` [`-t` *memory_type*] [`-m` *machine_type*] *XP_addr*

Defines the current kernel (`-k`) or user (`-u`) LAT table based on the specified exchange package (*XP_addr*).

`-m` *machine_type*
Selects the machine type to help specify the exchange package.

`-t` *memory_type*
Selects memory type to help specify the exchange package.

`lat -k|-u` [*entry*: *mode ba la pb*]
Redefines the current kernel (`-k`) or user (`-u`) LAT table.

entry Specifies the LAT entry (such as `lat0` or `lat1`).

mode Specifies the mode as any combination of `r` (read), `w` (write), and `x` (execute).

ba Specifies the logical base address.

la Specifies the logical limit address.

pb Specifies the physical bias, which equals the physical base address minus the logical base address.

`lat -k|-u` [*entry*: *clear*]
Clears the current kernel (`-k`) or user (`-u`) LAT table entry. The *entry* argument specifies the LAT entry (such as `lat0` or `lat1`).

`ldch`
Alias: `ldcache`
Prints a summary of the logical device cache buffers.

`ldchage` [`-`]
Displays a summary of logical device cache (ldcache) aging information. The `-` argument displays aging information for each cache block.

`ldmap` [*ldmap_table_entries*]
Alias: `ddmaps`
Prints the logical device maps.

`leb` Displays the kernel multi-threading lock violations logged in the lockrule error buffer.

`lnode` [`-`] [*lnode_table_entries*]
Aliases: `lno`, `l`
Prints the lnode table entries.

`loadem`
Forces a process to appear as if in core. This is useful when looking at a process that has been swapped, but its memory space has not yet been reused.

`map` [`-`] [`l`] [*map_names* | *map_addresses*]
Displays the specified `map` structures. Values for *map_names* are `bmrmap`, `coremap`, `execmap`, `mcache`, `sdsmap`, `swapmap`, and `all`. If *map_addresses* is specified, the data beginning at that address are displayed as if it were a `map`. By default, all maps are displayed.

`-` Includes a list of allocated and free areas.

`l` Includes an octal dump of the map words.

`mdw` Prints the memory descriptor words from the dump header.

`mem` [`-`[*any_character*]]
Prints a map of processes and shared text segments in memory. On CRAY T90 systems, this command also displays shared memory segments. The `-` option also displays the `sched` control structure. If the `-` option is followed by any character (for example, `-z`) the `Nic` (nice value) and `Pri` (priority) files in the output are replaced by `Skip`, which represents the line number within `sched.c` where the process was skipped over for selection.

`mme` [`-a`] [`-r`] [*CPU_numbers*]
(C90 and T90 systems only) Displays memory error status save area entries for the specified CPUs. By default, non-null entries for all CPUs are displayed.

`-a` Forces display of null entries.

`-r` Specifies raw mode; displays the `PORT` and `READ MODE` or `DESTINATION` fields as octal values, rather than as a descriptive (mnemonic) interpretation. In addition, the `MEMORY ERROR ADDRESS` field is displayed as a single octal value, rather than being separated into section, subsection, bank group, and bank (C90 systems only).

`mount` [`-`] [`-`] [*slot_number*]
Aliases: `mnt`, `m`
Formats the mount table. If the `-` option is specified, a long-form report is generated that includes the `vfs` entry. The `--` option generates a long-form report for every table entry.

`msgq` [`-`]
Displays the IPC message queue structures. With no options, information on the `msqid_ds` structures is displayed. The `-` option displays additional information.

`mthold` [`-`]
Alias: `mth`
Displays information for CPUs that are currently holding on a kernel multi-threading lock and for CPUs that have panicked trying to unlock an already unlocked multi-threading lock. The `-` option displays additional information about each lock.

`mtlocks` [`-`] [`sem`*nn* | `SEM`*nn* | *address* | *symbol*]
Alias: `mtlock`, `mtl`, `mt`
Displays information for kernel multi-threading locks. By default, information about a fixed list of locks is displayed. The `-` option displays additional information about each lock. Individual `SEMLOCKS` can be displayed using the option `sem`*nn*; *nn* specifies the controlling hardware semaphore number (decimal). Individual `MEMLOCKS` can be displayed using their addresses using the *address* option. In the `mtlocks` display, an `S` character in the `Status` field means that the controlling semaphore bit for a `SEMLOCK` was set; an `L` character means that the memory lock word was set.

`mux` Displays information in the MIOP tables, the tables controlling I/O to Model-E clusters, and the associated channel tables.

`nc1inode` [`-`] [`-`] [*slot_number*]
Alias: `nc`
Formats the `nc1inode` table. If the `-` option is specified, a long-form report is generated that includes the vnode and the attribute structure. The `--` option generates a long-form report for every table entry.

`nfsmi` *address* [*address...*]
Prints the specified NFS mount info structure(s).

`nm` *list of symbols*
Prints symbol value and type as found in *namelist_filename*.

`noprint`
Aliases: `nopr`, `np`
Turns off output to the terminal.

`od` [`-a`|`-k`|`-u`] [`-m` *mode*] [`-t` *memory_type*] [*address*|*symbol*] [*count*] [*format*]
Aliases: `dump`, `rd`
Dumps *count* data values starting at *address* (or *symbol*) according to *format*. Use the `-a`, `-k`, and `-u` options to specify other than the default LAT table for address mapping.

`-a` Specifies absolute (one-to-one) address mapping of the LAT table.

`-k` Specifies the kernel LAT table.

`-m` *mode*
Specifies the address mode as any combination of `r` (read), `w` (write), or `x` (execute).

`-t` *memory_type*
Displays the data from those portions of *core_filename* defined by *memory_type* memory descriptor words (`mdws`). Any valid `mdw` may be specified (use the `mdw` subcommand to display the mdws in *core_filename*). The initial default is `mem`. Once *memory_type* is specified, it remains the default for subsequent `od` subcommands until overridden. This option cannot be used on a running system.

`-u` Specifies the user LAT table.

address Specifies address; octal word addresses by default. *address* can be followed by one of the following:

`p` Denotes parcel address.

`a`,`b`,`c`,`d` Denotes a word address plus parcel offset.

`B` Denotes byte address.

format Specifies format. Allowed formats are `och`, `octal`, `decimal`, `hex`, `character`, `byte`, `parcel`, and `instruction` (abbreviated as `I` or `i`). The default for *format* is `och` (`octal` plus `character`).

`packet` *queue_name* | *octal addresses*
Aliases: `pack`, `pkt`, `pk`
Formats the packet queues. The possible choices for *queue_name* are: `ios`, `iosin`, `iosout`, and `ssd`. Queues are dumped in reverse order, most recent first, with unprocessed packets followed by old packets. When displaying a running system, the information may be inconsistent or erroneous. If a list of addresses is specified, `crash` tries to format what it finds at each address as an IOS packet.

`pbuf` [*pbuf_header_table_slot_number*]
Aliases: `pbufhdr`, `phdr`
Formats the system physical buffer headers.

`pddtab` [`-l`] [`-`] [*slot_number*]
Alias: `pdd`
(Cray PVP systems with IOS model E only) Prints the contents of `pddtab` for defined devices.

`-` Provides a long listing of contents for all devices.

`-l` Provides a very long listing for all devices.

slot_number Provides details for specified slot.

For Cray PVP systems with IOS model E, this command formats and displays the contents of the `pddtab` table for the physical disks in the system dumped.

`pktdi` [`-d` | `-i`] | [*device* [*device* ...]]
Formats the IPI-3/IPI packet driver traces. If no options are specified, a list of the devices and associated trace buffer pointers is displayed.

`-` Formats traces for all devices.

`-d` Formats traces for each IPI-3 device.

`-i` Formats traces for all IOP devices and the `reqt` device.

device Formats traces for the specified device or devices.

`pktdk` [`-d` | `-i`] | [*device* [*device* ...]]
Formats the IPI-3/HIPPI packet driver traces. If no options are specified, a list of the devices and associated trace buffer pointers is displayed.

`-` Formats traces for all devices.

`-d` Formats traces for each IPI-3 device.

`-i` Formats traces for all IOP devices and the `reqt` device.

device Formats traces for the specified device or devices.

`pp` Prints the process management hash tables and active process links.

`print` Alias: `pr`
Turns on output to the terminal.

`prnode` [`-`] [`-`] [*slot_number*]

Alias: `prn`

Formats the `prnode` table. If the `-` option is specified, a long-form report is generated that includes the vnode and the attribute structure. The `--` option generates a long-form report for each table entry.

`proc` [`- | -r | -l | -w`] [*process_table_entries*]

Aliases: `ps`, `p`

Formats the process table. One of the following options can be specified:

`-` Generates a longer listing.

`-r` Displays only processes that can be run.

`-l` Displays additional information about each process.

`-w` Displays the event field symbolically, if possible.

`pws` [*CPU_numbers*]

Displays the `pws` structures for the specified CPUs. If *CPU_numbers* is not specified, `pws` displays the `pws` structures for all CPUs. Included in the `pws` structure are the addresses of the `unix`, `user`, and `diag` exchange packages for the CPU.

`redirect` [`?`] [`+`] [*file*]

Aliases: `redir`, `>`, `>>`

Sends a copy of all output to the specified file. Specify `redirect` with no *file* argument to stop sending output to the file. `redirect ?` prints the current state of redirection. `redirect +` or `>>` appends the output to the specified file, *file*. There must be a space between `>` or `>>` and the *file*.

`registers` *CPU registers*

Aliases: `register`, `regs`, `reg`

Prints the B, T, and V registers from a dump. This command works only from a saved core file; it does not work on a running system. The register list can be individual registers, such as B01, V123, T23, or it can be a register type followed by an asterisk, as shown in the following example:

```
reg cpu3 B01 B02 V*
```

If you do not specify a CPU number, the default is the last legal CPU used on a `registers` command.

`resinfo` [*resinfo_table_entries*]

Prints the `resinfo` structures.

`rnode` [`-`] *slot* | *address* [*slot* | *address* ...]

Alias: `rn`

Formats an NFS rnode. If the `-` option is specified, a long-form report is produced that includes the vnode and the attribute structure. If *slot* is specified, a report is produced for that rnode. If *address* is specified, it is interpreted as an rnode address.

`rnodetab` [`-`]

Alias: `rnt`

Formats the NFS rnode table. If the `-` option is specified, a long-form report is generated that includes the vnode and attribute structure for each rnode.

`rpe` [`-a`] [`-r`] [`-s`] [`-u`] [*CPU_numbers*]

(Y-MP, C90 and T90 systems only) Displays the register parity error (`rpe`) status save area entries for the specified CPUs. (For each configured CPU, there are two slots in the `rpe` save area; one for the status of the most recent `rpe` that occurred while in system mode, and one for the most recent `rpe` in user mode.) By default, `rpe` displays the non-null system and user mode entries for all CPUs.

`-a` Forces display of null entries.

`-r` Specifies raw mode; displays the `RPE ERROR BITS` field in octal. By default, the descriptive (mnemonic) interpretation is displayed.

`-s` Displays system mode entries.

`-u` Displays user mode entries.

`rslogdump` > *filename*

Alias: `rslog`

(Secure kernel with buffered security logs records only; that is, records queued for read by the security log daemon) Dumps to *filename* all security log records that have not been buffered by the security log daemon. The data is dumped in a raw format; therefore, redirection is necessary. The resulting file can then be examined by using the `reduce`(8) command.

`scache` [`-`|`--`|`---`|`----`] [`-a`] [`-g`] [`-l`] [`-v`] [*addr*]

Alias: `sc`

(DCE DFS only) Displays Cache Manager `scache` entries.

`-`, `--`, `---`, `----`

Specifies verbose level; displays additional information in the output. Increasing the number of dashes (the `-` character) increases the amount of information that is displayed.

`-a` Displays the `vattr` attribute structure using the format of the `nc1inode` subcommand.

`-g` Produces a command and address in `grep`(1) format that can be used to display a specific structure.

`-l` Displays information about the lock structures contained in the structure(s) being displayed.

`-v` Displays the vnode using the format of the `nc1inode` subcommand.

[*addr*] Limits the display to the data structure at the specified address. By default, all known data structures are displayed.

`sct` [`-`|`--`|`---`|`----`] [`-g`] [*addr*]

(DCE DFS only) Displays information about Kernel RPC `sct` structures.

-, --, ---, ----
Specifies verbose level; displays additional information in the output. Increasing the number of dashes (the - character) increases the amount of information that is displayed.

-g Produces a command and address in grep(1) format that can be used to display a specific structure.

[*addr*] Limits the display to the data structure at the specified address. By default, all known data structures are displayed.

sema [-]
Displays the IPC semaphore structures. With no options, information on the semid_ds structures is displayed. The - option displays additional information.

semundo [-]
Alias: semu
Displays the IPC semaphore undo structures. With no options, this command displays information on the undo structures and the free and in-use chains. The - option displays information on the semaphore undo values.

sess [-] [*job_table_entries*]
Alias: jtab, jobs
Displays the System Job (Session) Table. Without options, sess generates a one-line description of all sessions (jobs). The - option produces a long listing of the session structure contents. Additional fields display the IPC usage: messages, semaphores, and shared memory (CRAY T90 systems only).

shm [-]
(CRAY T90 systems only) Displays information about the shared memory segments that exist on the system. The - option displays the configuration values.

shrc Displays the fair-share scheduler shrconst structure. This information is also available with the shradmin -v command; see the shradmin(8) man page for more information.

siginfo *proc_table_entries*
Alias: sig
Prints signal information for the specified processes.

sizeof *structure_names*
Prints the size of the specified structures.

slice [-l] [-m] [-p] [-r] [-s] [-S] [*minor number*]
Alias: sli
(Cray PVP systems with IOS model E only) Prints eslice definition for defined devices.

-l Provides dev_ldd (logical devices).

-m Provides dev_mdd (mirrored devices).

-p Provides dev_pdd (physical devices).

`-r` Provides `dev_rdd` (RAM devices).

`-s` Provides `dev_sdd` (striped devices).

`-S` Provides `dev_ssdd` (SSD).

minor number
Provides details for the specified *minor number*.

Example: `slice -l 3` prints `dev_ldd` minor number 3.

This command (without options) provides a long listing of the defined slices (`eslice` for all the recognized types of devices in the system (physical: `dev_pdd`, logical: `dev_pdd`, mirrored: `dev_mdd`, SSD: `dev_ssdd`, and striped: `dev_sdd`). The information printed is for IOS-model-E based systems because they are the only systems that contain defined `eslice` structures. Physical slices are printed as stand-alone definitions whereas the logical slices are printed with the physical slices that define the device space.

`slogstat`
Alias: `slog`
(Secure kernel with logging enabled only) Formats the `slg` structure. Displays various information about the security log status (such as the size, status, location, and offset).

`slot` *table_name address*
Gives the slot number in the given table, *table_name*, for the specified *address*. You can then use the slot number to print the table entry. For example, if word address 112457 is in `proc` table entry 25, the following returns slot number 25:

```
slot proc 112457
```

The following command formats this `proc` table entry:

```
proc 25
```

`slr` Displays the Shared Lock Region on systems running the Shared File System (SFS).

`smp` Displays SMP-1 and SMP-2 hardware semaphore device tables.

`ssd` Displays SSD configuration and statistics.

`ssddtab`
Alias: `ssdd`
Displays the SSD device tables.

`stack` [`-`] *process_table_entries*
Aliases: `stk`, `s`, `kernel`, `k`
Formats a dump of the kernel stack of a process from the `ublock` of the process. If the process was executing at the time of the dump, the stack pointers are obtained automatically from the saved registers for that CPU. If the process was not in the kernel, the stack trace may be garbage. The `-` option produces more extensive formatting of each stack frame. For explicit control of the starting stack frame, use the `registers` command to get the B01 and B02 registers, the `kfp` command to set these values, and the `trace` command with the `-r` option to print the stack. When looking at a dump taken from a different type of machine than the current one, the line numbers printed in the trace may be wrong.

`stat` Prints certain statistics found in the dump. These include the panic string (if a panic occurred), time of deadstart, and the CPU and process that were last in the kernel.

`swapmap` [`-`[*any_character*]]
Prints the swap file allocation map. The `-` option also displays the `sched` control structure. If the `-` option is followed by any character (for example, `-z`) the `Nic` (nice value) and `Pri` (priority) files in the output are replaced by `Skip`, which represents the line number within `sched.c` where the process was skipped over for selection.

`swapper`
Prints the `sched control structure.`

`swapq` [`-`[*any_character*]]
Prints a list of swapped processes that are eligible for swap-in. The `-` option also displays the `sched` control structure. If the `-` option is followed by any character (for example, `-z`) the `Nic` (nice value) and `Pri` (priority) files in the output are replaced by `Skip`, which represents the line number within `sched.c` where the process was skipped over for selection.

`svc_data` *address* [*address* ...]
Aliases: `svc_d`, `svcd`
Formats the NFS `svc_data` structures.

`svc_xprt` *address* [*address* ...]
Aliases: `svc_x`, `svcx`
Formats the RPC `SVCXPRT` structures.

`sysent`
Prints the system call timings found in the `sysent` structure within the kernel.

`sysint`
Prints the system interrupt timings found in the `sysint` structure within the kernel.

`tabinfo` *address*
Aliases: `tabinit`, `tab`
Sets for `crash` where the `tabinfo` structure resides to address.

`text` [*text_table_entries*]
Aliases: `txt`, `x`
Formats the text table.

`tkc` [`-`|`--`|`---`|`----`] [`-g`] [`-l`] [*addr*]
(DCE DFS only) Displays information about Token Cache (`tkc`) structures.

`-`, `--`, `---`, `----`
Specifies verbose level; displays additional information in the output. Increasing the number of dashes (the `-` character) increases the amount of information that is displayed.

`-g` Produces a command and address in `grep`(1) format that can be used to display a specific structure.

`-l` Displays information about the lock structures contained in the structure(s) being displayed.

[*addr*] Limits the display to the data structure at the specified address. By default, all known data structures are displayed.

`tkm` [`-`|`--`|`---`|`----`] [`-f`] [`-g`] [`-h`] [`-l`] [`-t`] [*addr*]
(DCE DFS only) Displays information from the Token Manager (`tkm`) list. If no options are specified, `tkm` behaves as if the `-t` option is selected.

`-`, `--`, `---`, `----`
Specifies verbose level; displays additional information in the output. Increasing the number of dashes (the `-` character) increases the amount of information that is displayed.

`-f` Displays entries in the Token Manager file ID hash table (`fidhash`).

`-g` Produces a command and address in `grep`(1) format that can be used to display a specific structure.

`-h` Displays `fshost` (file system host) information.

`-l` Displays information about the lock structures contained in the structure(s) being displayed.

`-t` Displays Token Manager (`tkm`) information. This is the default option.

[*addr*] Limits the display to the data structure at the specified address. By default, all known data structures are displayed.

`tkset` [`-`|`--`|`---`|`----`] [`-g`] [`-l`] [*addr*]
(DCE DFS only) Displays information about Token Sets (`tkset`).

`-`, `--`, `---`, `----`
Specifies verbose level; displays additional information in the output. Increasing the number of dashes (the `-` character) increases the amount of information that is displayed.

`-g` Produces a command and address in `grep`(1) format that can be used to display a specific structure.

`-l` Displays information about the lock structures contained in the structure(s) being displayed.

[*addr*] Limits the display to the data structure at the specified address. By default, all known data structures are displayed.

`tpq` [*addr*]

(DCE DFS only) Displays information about Thread Pool Queues (`tpq`).

[*addr*] Specifies the address returned from the `tpq_Init()` routine.

The following thread handles are available:

- `cm_auxThreadPoolHandle`
- `cm_threadPoolHandle`
- `tkm_threadPoolHandle`

The following example shows the use of the `od` subcommand to obtain the address for the thread handle `cm_threadPoolHandle`. This address is then used as the *addr* argument for `tpq`.

```
> od cm_threadPoolHandle
2645314: 000000000000117661644
> tpq 011766164S
```

`tps` [*device1*] [*device2*] . . .

Prints tape device structures. With no arguments, `tps` displays the tape I/O structures for all tape devices in the system. When a device name is specified, `tps` displays the tape structures associated with that device. Using a `-` character instead of *device1* prints out tape structures for all tape devices in the system.

`tpt` [*device1*] [*device2*] . . .

Prints kernel level tape device traces. `tpt` called without any arguments prints out a table containing the device name (as seen in the `tpstat`(1) display), index (physical device name), and the start, middle and end trace pointers for each device in the the tape table. `tpt` called with a device name prints out traces for that device. `tpt` called with `-` instead of *device1* dumps out traces for all tape devices in the system.

`trace` [`-`] [`-r`]

Aliases: `t`, `ytrace`, `ytra`, `ytr`, `yt`

Generates a kernel stack trace. The trace begins at the saved stack frame pointer in `kfp`. A `-` followed by `-r` does more intricate formatting of the stack frames. The line numbers printed can be incorrect if the type of the machine from which the dump was taken does not agree with the type of the machine for which `crash` was built. The `ytrace` command forces the stack to be formatted in Y-mode.

`ts` *list of symbols*

Prints symbol value and type as found in *namelist_filename*.

`tty` [*type*] [`-`] [*tty_table_entries*]
Aliases: `term`, `pty`
Prints the `tty` structures. The *type* argument determines which structure will be used (such as `tty` or `pty`). The default for *type* is `tty`. However, after you have specified *type*, the last value specified is used. The `-` option prints additional information, including the `stty`(1) options for the given line.

`uio` [*slot_numbers*]
Displays the specified `uio` entries. By default, all `uio` entries are displayed.

`unloadem`
Forces a process to appear as if swapped.

`user` [*process_table_entries*]
Aliases: `uarea`, `u_area`, `u`
Prints the user structure of the specified process as determined by the information contained in the process table entry. If no entry number is specified, the information from the last executing process will be printed. Swapped processes produce an error message.

`utc` [`-`] [*major_numbers*]
Displays UTC (Universal Time Clock) device tables.

`utrace` [`-`] [`+`] [*count*]
Alias: `ut`
Dumps *count* entries from the kernel trace buffer in reverse chronological order (latest first). If you specify the `-`, `utrace` prints the address of each `utrace` entry. The `+` option causes the dump of the `utrace` entries to continue from the next trace buffer entry. The default *count* is 32 entries.

`var` Aliases: `tunables`, `tunable`, `tune`, `v`
Prints the system parameters that can be tuned.

`vfs` Formats the `vfs` and print in chain order.

`vhisp` Displays `vhisp` configuration and statistics.

`vnode` [*addr*]
Prints a single-line description of all vnodes. Vnodes are housed inside the dependent inodes. The slot numbers on this report refer to the dependent inode tables. If *addr* is specified, it is interpreted as a vnode address. If the address is a vnode, `crash` determines the file system and the dependent inode.

`volume` [`-`|`--`|`---`|`----`] [`-g`] [`-l`] [*addr*]
Alias: `vol`
(DCE DFS only) Displays information about known DFS volumes.

`-`, `--`, `---`, `----`
Specifies verbose level; displays additional information in the output. Increasing the number of dashes (the `-` character) increases the amount of information that is displayed.

`-g` Produces a command and address in `grep`(1) format that can be used to display a specific structure.

`-l` Displays information about the lock structures contained in the structure(s) being displayed.

[*addr*] Limits the display to the data structure at the specified address. By default, all known data structures are displayed.

`xp` [`-m` *machine_type*] [`-t` *memory_type*] *address*

Alias: `exch`

Formats the data at *address*, assuming that it is an exchange package.

`-m` *machine_type*

Format the data as if the hardware were *machine_type*. Values for *machine_type* are `y`, `ymp`, `y-ea`, `ymp-ea`, `craymp`, `crayymp`, `c90`, `crayc90`, `cray-j90`, `crayts`, `cray-ts`, `T90`, `t90`, `t16`, `t32`, `T90I`, `t90i`, `crayts-ieee`, `cray-ts-ieee`, `T90-ieee`, `t90-ieee`, `t4-ieee`, `t16-ieee`, and `t32-ieee`. The default is the machine type of the dumped system.

`-t` *memory_type*

Displays the data from those portions of *core_filename* defined by *memory_type* memory descriptor words (`mdws`). Any valid `mdw` may be specified; however, only the `mem` and `xp` types contain exchange packages (use the `mdw` subcommand to display the mdws in *core_filename*). The initial default is `mem`. Once *memory_type* is specified, it remains the default for subsequent `xp` subcommands until overridden. This option cannot be used on a running system.

`xpa` [`-m` *machine_type*]

Displays the exchange package areas for all active (host and guest) kernels. If the `-m` option is specified, the exchange packet is formatted for the hardware specified by *machine_type*. Values for *machine_type* are `y`, `ymp`, `y-ea`, `ymp-ea`, `craymp`, `crayymp`, `c90`, `crayc90`, `cray-j90`, `crayts`, `cray-ts`, `T90`, `t90`, `t16`, `t32`, `T90I`, `t90i`, `crayts-ieee`, `cray-ts-ieee`, `T90-ieee`, `t90-ieee`, `t4-ieee`, `t16-ieee`, and `t32-ieee`.

`xsinfo` [`-`]

Alias: `xsi`

Prints the Multiplexed (MPX) Scheduler information table. If the `-` option is used, data for swap partitions 0 through `SWAP_PARTS` (defined in `sys/swap.h`) is displayed. Otherwise, the display is limited to partitions 0 through `swapper.swp_nparts - 1`. The `-` option also forces the display when MPX scheduling is not configurable (that is, `swapper.swp_nparts = 1`).

There are built-in aliases for many of the *formats*, as well as those listed for the commands. Some of the aliases are as follows:

Format	Alias
byte	b
character	char, c
parcel	par, p
instruction	instr, ins, I
decimal	dec, e
directory	direct, dir, d
hexadecimal	hexadec, hex, h, x
inode	ino, i
longdec	ld, D
longoct	lo, O
octal	oct, o
pddtab	pdd
slice	sli
write	w

NOTES

Because most flags are abbreviated and have little meaning to an uninitiated user, a source listing of the system header files is useful while using `crash`.

BUGS

Stack tracing of the current process on a running system does not work.

FILES

`/usr/include/sys/*.h`	Header files for table and structure information
`/dev/mem`	Default system image file
`buf.#`	Files created that contain buffer data

SEE ALSO

`kcompress`(8), `mount`(8), `reduce`(8)

`ps`(1), `sh`(1), `stty`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

General UNICOS System Administration, Cray Research publication SG–2301

NAME

cron – Clock daemon

SYNOPSIS

/etc/cron [-m *limit*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The cron program executes commands at specified dates and times.

The cron program accepts the following option:

-m *limit* Sets the upper *limit* on number of jobs that can be run at once. (Allows you to specify a value for MAXRUN.) Default is 25.
NOTE: The limit on number of jobs that can be run at once is also subject to job queue limits (see queuedefs(5)).

Commands that are executed on a regular basis can be specified according to instructions found in crontab files; users can submit their own crontab file by using crontab(1). Commands that are executed only once can be submitted using the at(1) command. Because cron never exits, it should be executed only once. This is best done by running cron from the initialization process through the /etc/rc file.

The cron program examines crontab files and at(1) command files only during process initialization and when a file changes through the crontab(1) command. This reduces the overhead of checking for new or changed files at regularly scheduled intervals.

NOTES

There are factors that cause cron to not immediately process changes made to a crontab file through the crontab(1) command. These factors include long-running cron jobs and system overhead.

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Allowed to start the cron daemon.

If the PRIV_SU configuration option is enabled, the super user is allowed to start the cron daemon.

MESSAGES

A history of all actions taken by `cron` is recorded in `/usr/lib/cron/log`.

FILES

`/usr/lib/cron`	Main `cron` directory
`/usr/lib/cron/log`	Accounting information
`/usr/lib/cron/queuedefs`	Definitions for all queues managed by `cron`
`/usr/spool/cron`	Spool area

SEE ALSO

`at`(1), `crontab`(1), `sh`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`queuedefs`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

csa – Overview of Cray Research system accounting

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The UNICOS operating system supports two accounting packages, Cray Research system accounting (CSA) and standard UNIX System V accounting. CSA is designed to meet the unique accounting requirements of Cray Research sites. It is a set of C programs and shell scripts that, like standard UNIX System V accounting, provides methods for collecting per-process resource usage data, recording connect sessions, monitoring disk usage, and charging fees to specific logins. CSA also provides the following facilities that are not available from the standard UNIX accounting package:

- Per-job accounting
- Device accounting
- Daemon accounting for monitoring the Network Queueing System (NQS) and the UNICOS tape subsystem
- Disk accounting by account ID
- Arbitrary accounting periods
- Flexible system bill unit (SBU) system
- One file containing all data for an accounting period
- Front-end formatting interfaces
- Offline archiving of accounting data

The UNICOS kernel performs process accounting. On termination of a process, one record per process is written to a file, usually /usr/adm/acct/day/pacct. At the completion of various daemon specific events, Network Queuing System (NQS) and tape daemons write daemon accounting records. The csabuild(8) command combines the data and generates a session record file, which is used as input by other CSA programs to generate reports, bills, and data for front-end systems.

FILES

/etc/csaboots	Captures system boot times.
/etc/udb	User validation file that contains user control limits; contains login name to user ID conversions.
/etc/wtmp	Contains login and logoff history information.
/usr/lib/acct	Contains most of the accounting commands listed in *UNICOS Resource Administration*, Cray Research publication SG–2302.

`/usr/lib/acct/day/pacct`	Contains current process accounting information.
`/usr/adm/acct/day/nqacct*`	CSA NQS accounting files.
`/usr/adm/acct/day/tpacct*`	CSA tape accounting files.

SEE ALSO

`acct`(8), `acctcms`(8), `acctdusg`(8), `accton`(8), `acctsh`(8), `acctwtmp`(8), `csaaddc`(8), `csaboots`(8), `csabuild`(8), `csacon`(8), `csaconvert`(8), `csacrep`(8), `csadrep`(8), `csaedit`(8), `csafef`(8), `csaibm`(8), `csajrep`(8), `csaline`(8), `csanqs`(8), `csapacct`(8), `csaperiod`(8), `csaperm`(8), `csarecy`(8), `csaswitch`(8), `csatape`(8), `csaverify`(8), `diskusg`(8), `dodisk`(8), `fwtmp`(8), `lastlogin`(8), `nulladm`(8), `shutacct`(8), `startup`(8), `turnacct`(8), `turndacct`(8)

`acctcom`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`acct`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

`acct`(5), `utmp`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csaaddc – Adds cacct records

SYNOPSIS

/usr/lib/acct/csaaddc [-a] [-o *ofile*] [-t] [-v] [[[-A] [-g]] [-j] [-u]] *ifiles*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csaaddc command adds cacct records and outputs cacct records. You also can generate cacct records with the csacon(8) and acctdisk(8) commands.

The csaaddc command accepts the following operand:

ifiles Specifies the files to be processed. The files are in cacct format.

Output Options

-a Outputs information in ASCII. By default, the output is written to a binary output file.

-o *ofile* Specifies the output file. By default, the output is written to stdout.

-t Totals all records into one record.

-v Sets verbose mode on. When you also specify -a, verbose output is written to stderr.

Consolidation Options

You can specify multiple consolidation options. These options should be the same as those used to create the input files with csacon(8). If you do not specify any consolidation options, the default is -Au.

-A Totals by account ID.

-g Totals by group ID. You must use this option with at least one of the other consolidation options, because not all records have a group ID.

-j Totals by job ID.

-u Totals by user ID.

ifiles This operand is a list of input files in cacct format. The file names are separated by spaces.

NOTES

The consolidation options used with csacon(8) to generate the input files should correspond to the consolidation options used with csaaddc. If they do not, the resulting data can be misleading and difficult to interpret.

Be aware that the `csacon -a` option corresponds to the `csaaddc -A` option.

You must consolidate disk data generated by `acctdisk -A` by using `csaaddc` with the `-A` option. `acctdisk -A` sets the *uid* and *jid* fields of all records to `0`; therefore, these records should not be consolidated by job ID or user ID.

EXAMPLES

The following example merges two input files created with `csacon`(8), using the `-a` and `-g` consolidation options. The output is written to file `outfile`.

```
csaaddc -A -g -o outfile cacct1 cacct2
```

SEE ALSO

`acctdisk`(8), `csacon`(8), `csacrep`(8), `diskusg`(8), `dodisk`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csaboots – Records system boot times for the accounting subsystem

SYNOPSIS

```
/etc/csaboots [-o csafile] [-u] [-v]
/etc/csaboots [-o csafile] [-U ufile] [-v]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csaboots command records system boot times for the accounting subsystem by writing a record of each boot time into the /etc/csainfo file. The default is to write the current time to the output file. rc (see brc(8)) invokes the csaboots command during system startup.

The csaboots command accepts the following options and arguments:

-o *csafile*	Changes the output file from /etc/csainfo to *csafile*.
-u	Writes all boot times found in /etc/utmp (utmp(5)) to the output file.
-U *ufile*	Writes all boot times found in *ufile* to the output file. *ufile* must be in utmp format.
-v	Sets verbose mode. Informational messages are written to stdout.

FILES

/etc/utmp	Records information about current system users
/etc/csainfo	Record of system boot times

SEE ALSO

utmp(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csabuild – Generates a session record file

SYNOPSIS

/usr/lib/acct/csabuild [-a] [-A] [-C *ctmppath*] [-D *level*] [-i] [-n] [-N *nqspath*] [-o *nday*] [-P *pacctpath*] [-s *sessionfile*] [-S *segmentsize*] [-t] [-T *tapepath*] [-u *uptimepath*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csabuild command organizes the accounting files from the following sources, into session records (session.h):

- Network Queuing System (NQS)
- Online tapes (tpdaemon(8))
- Kernel accounting
- Connect accounting

The daemon accounting programs use session records to generate reports, bills, and data for front-end systems.

An integer suffix is attached to the path name arguments; a 0 indicates recycled information, and 1 and higher indicates current data.

The csabuild command accepts several types of options: file name, performance, job ending, and debugging.

File Name Options

The csabuild command accepts the following file name options:

-C *ctmppath*	Specifies connect time record path name for preprocessed data (output from csaline(8)). The default is /usr/adm/acct/work/Pctime.
-N *nqspath*	Specifies NQS path name for the preprocessed data (output from csanqs(8)). The default is /usr/adm/acct/work/Pnqacct.
-P *pacctpath*	Specifies pacct file path name. The default is /usr/adm/acct/work/Wpacct.
-s *sessionfile*	Specifies session file name. The default is /tmp/Super-record. This is the csabuild output file.
-T *tapepath*	Specifies tape daemon file path name. The default is /usr/adm/acct/work/Ptpacct.
-u *uptimepath*	Specifies uptime path name. The default is /usr/adm/acct/work/Puptime.

Performance Options

The `csabuild` command accepts the following performance options:

`-S` *segmentsize* Changes default segment size. The default is 1000 jobs per segment. Debug level 3 displays the percentage of each segment used. If more jobs are run per day, increase this number. If less jobs are run per day, decrease this number.

`-t` Prints timing information for the two major phases of `csabuild`.

Job Ending Options

The `csabuild` command accepts the following job ending options:

`-a` Assumes crash option. The default operation is if a job does not have an associated end-of-job record, but the system was rebooted, the job is assumed to be terminated. With this option, these jobs are not marked as terminated.

`-o` *nday* Terminates the session if a session is older than *nday* days. NQS requests submitted more than *ndays* ago also are terminated. You can use this option to terminate old jobs that are known to be finished.

Debugging Options

The `csabuild` command accepts the following debugging options:

`-A` Abort option. If `csabuild` exits with an error, a core dump is generated.

`-D` *level* Controls messages printed during program execution. Level 1 is verbose, level 10 is not appropriate for any execution, except small test cases.

`-i` Ignores bad records. If `csabuild` runs into a record that it detects as bad, it can recover from the error by discarding the record and continuing to process input. When it discards a record, it prints a diagnostic message.

`-n` Suppresses the NQS sort and condense phase. This option prevents NQS jobs that span multiple system boots from being condensed into one job. (This function is intended only for error recovery.)

NOTES

The `pacct1` and `uptime1` files must exist, other files can be ignored; although without them, data on the associated daemons is not gathered. `csaverify`(8) can verify most of the input files. Generally, `csaedit`(8) and `csapacct`(8) can verify and repair bad input files.

BUGS

`csabuild` is limited by its input. Unless the data files are accurate, the sessions cannot be organized correctly.

FILES

`/usr/adm/acct/day` Directory that contains current unprocessed accounting data

SEE ALSO

`csaedit`(8), `csaline`(8), `csanqs`(8), `csapacct`(8), `csarun`(8), `csatape`(8), `csaverify`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csacon – Condenses a session record file into a cacct file

SYNOPSIS

/usr/lib/acct/csacon [[[-a] [-j] [-u]] [-g]] [-s *sessionfile*] [-v] [-A] [-D *level*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csacon command condenses the information from a session record file into cacct format. You can use the csacrep(8) command to generate ASCII reports from the consolidated file. csacon accepts several types of options: consolidation, input, and output options.

Consolidation Options

The csacon command accepts the following consolidation options. You can specify multiple consolidation options. If you do not specify any consolidation options, the default is to use -au.

-a Consolidates session records by using the account ID as a key.

-A Consolidates all jobs, including those that have not completed. By default, only jobs that have completed are consolidated.

-D *level* Sets the debugging level. Level 1 is slightly verbose; level 10 is very verbose. Debug output is written to standard error. By default, debugging is turned off.

-g Consolidates session records by using the group ID as a key. Because not all records have a group ID, you must use this option with at least one of the following consolidation options: -a, -j, or -u.

-j Consolidates session records by using the job ID as a key.

-u Consolidates session records by using the user ID.

Input Option

The csacon command accepts the following input option:

-s *sessionfile*
Specifies the name of the session record file, which is the input file. csabuild(8) created the file. The default is /tmp/Super-record.

Output Option

The csacon command accepts the following output option:

-v Sets verbose mode on. Verbose output is written to standard error.

EXAMPLES

The following example consolidates all records in the session record file `srec`. The session records are condensed by the three-tuple account ID, job ID, and user ID. Output is written to the `cacct` file.

```
csacon -A -a -j -u -s srec > cacct
```

SEE ALSO

`csaaddc`(8), `csabuild`(8), `csacrep`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csaconvert – Converts UNICOS 8.0, 8.3, 9.0, 9.1, 9.2, and 9.3 accounting file(s) to UNICOS 10.0 format

SYNOPSIS

/usr/lib/acct/csaconvert [-L *level*] [-m] [-v] [-N *nqacctfile*] [-o *outfile*]

/usr/lib/acct/csaconvert [-L *level*] [-m] [-v] [-a *tacctfile*] [-o *outfile*]

/usr/lib/acct/csaconvert [-L *level*] [-m] [-v] [-c *cacctfile*] [-o *outfile*]

/usr/lib/acct/csaconvert [-L *level*] [-m] [-v] [-n *Pnacctfile*] [-o *outfile*]

/usr/lib/acct/csaconvert [-L *level*] [-m] [-v] [-p *pacctfile*] [-o *outfile*]

/usr/lib/acct/csaconvert [-L *level*] [-m] [-v] [-s *cmsfile*] [-o *outfile*]

/usr/lib/acct/csaconvert [-L *level*] [-m] [-v] [-t *tpacctfile*] [-o *outfile*]

/usr/lib/acct/csaconvert [-L *level*] [-m] [-v] [-o *outfile*]

/usr/lib/acct/csaconvert [-L *level*] [-m] [-v] [-x *tacctfile*] [-o *outfile*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csaconvert command converts UNICOS 8.0, 8.3, 9.0, 9.1, 9.2, and 9.3 accounting files to UNICOS 10.0 format. You can specify either a file name or a path name for the input and the output. If you specify a path name, you must use the -m option. For files that already have been converted, the csaconvert command will display their name(s); it will not perform the conversion. csaconvert accepts several types of options: processing, input, and output options.

Processing Options

The csaconvert command accepts the following processing options:

-L *level* Specifies the UNICOS release level under which the input files were generated. The default release level is UNICOS 8.0.

-m Specifies that the input and output file names are path names. A numeric suffix, starting at either 0 or 1, is automatically appended to the file name. If you omit the -o option, all output is written to the stdout file (standard output).

-v Verifies the file revision level. (No conversion is done.)

Input Options

The csaconvert command accepts the following input options. Only one input option may be specified. By default, the input is the pacct accounting file Wpacct1 (-p).

-N *nqacctfile* Specifies a raw NQS file created by the NQS daemon.

-a *tacctfile* Specifies a tacct file created by the acctprc2(8) command.

-c *cacctfile* Specifies a cacct file created by the csaaddc(8) or csacon(8) command.

-n *Pnacctfile* Specifies a preprocessed NQS file created by the csanqs(8) command.

-p *pacctfile* Specifies a pacct accounting file. This is the default input with Wpacct1 as the file name.

-s *cmsfile* Specifies a cms file created by the acctcms(8) command.

-t *tpacctfile* Specifies a tape accounting file created by the tape daemon.

-x *tacctfile* Specifies a tacct file for conversion to cacct format. This capability is necessary if you are converting from UNIX System V accounting to Cray Research system accounting.

Output Option

The csaconvert command accepts the following output option:

-o *outfile* Specifies the output file. The default is stdout.

NOTES

All UNICOS 10.0 accounting tools are able to process accounting data generated on systems running UNICOS 8.0, 8.3, 9.0, 9.1, 9.2, and 9.3. As needed, the data is converted automatically to UNICOS 10.0 format. Since the UNICOS 9.0 release, you are no longer required to run the csaconvert command to convert the prior accounting data to the current release format.

However, if you access the prior accounting data on a regular basis, for performance reasons you should convert the data once using the csaconvert command. This allows you to avoid the overhead of repeatedly converting the data automatically. In this instance, explicit conversion is preferred.

EXAMPLES

Example 1: The following example converts a UNICOS 8.0 cacct file to UNICOS 10.0 format. The output is written to cacct100.

```
/usr/lib/acct/csaconvert -c cacct -o cacct100
```

Example 2: The following example converts a UNICOS 9.3 cacct file to UNICOS 10.0 format. The output is written to cacct100.

```
/usr/lib/acct/csaconvert -L 9.3 -c cacct -o cacct100
```

Example 3: The following example converts UNICOS 8.0 `pacct` files named `Wpacct0`, `Wpacct1`, `Wpacct2`, and so on, found in `/usr/adm/acct/sum/data/0413/1800`. The output is written to the `Npacct0`, `Npacct1`, and `Npacct2` files, and so on.

```
/usr/lib/acct/csaconvert -p /usr/adm/acct/sum/data/0413/1800 -o Npacct -m
```

Example 4: The following example converts a UNICOS 9.3 System V `tacct` file to UNICOS 10.0 `cacct` format:

```
/usr/lib/acct/csaconvert -L 9.3 -x tacct -o cacct
```

SEE ALSO

`acctprc`(8), `csaaddc`(8), `csabuild`(8), `csacon`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csacrep – Reports on consolidated accounting data

SYNOPSIS

```
/usr/lib/acct/csacrep [-a | -u] [-b] [-c] [-d] [-f] [-g] [-h] [-j] [-m] [-n] [-w] [-x]
[-y] [-C] [-J] [-M]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csacrep command generates reports from data in cacct format, such as output from the csacon(8) command. The default output reports the account name, the user ID, and the login name, and it suppresses the headers.

The csacrep command accepts the following two types of options: sorting and printing.

Sorting Options

The csacrep command accepts the following two sorting options. You cannot use the -a and -u options together.

-a Sorts output first by the account ID and then by the secondary key of the user ID. By default, csacrep does not sort. You cannot use this option with the -u option.

-u Sorts output by user ID and then by the secondary key of the account ID. By default, csacrep does not sort. You cannot use this option with the -a option.

Printing Options

The csacrep command accepts the following printing options.

-b Reports SBU data.

-c Reports CPU time memory integral and connect time in seconds.

-d Reports the cumulative online and offline disk usage and the number of samples. Input files that contains disk usage data are generated by using acctdisk(8) or by merging cacct disk output files with other cacct files, using the csaaddc(8) command.

-f Reports full data.

-g Reports group name.

-h Displays headers.

-j Reports number of processes and jobs.

-m Reports CPU breakdown multitasking.

-n Reports prime and nonprime data.

`-w` Reports I/O wait time and I/O wait memory integral.

`-x` Reports blocks transferred and physical and logical I/O.

`-y` Reports SDS data.

`-C` Reports system call and interrupt CPU times.

`-J` Reports job ID.

`-M` Reports Cray MPP system usage statistics. If there is no attached MPP system, the `-M` option reports 0.

NOTES

Zero is a valid value for the number of jobs. If a job is executed with multiple user ID/account ID pairs, the number-of-jobs value is incremented for only one such combination per job.

EXAMPLES

The following example generates a report from a daily accounting file:

```
csacrep -hcw < /usr/adm/acct/sum/data/0203/1315/cacct
```

SEE ALSO

`acctdisk`(8), `csaaddc`(8), `csacon`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csadrep – Reports daemon usage

SYNOPSIS

/usr/lib/acct/csadrep [-a] [-A] [-D *level*] [-j] [-n] [-o *ofile*] [-s *sfile*] [-t] [-V *level*]

/usr/lib/acct/csadrep [-A] [-D *level*] [-o *ofile* [-N]] [-s *sfile*]

/usr/lib/acct/csadrep [-a] [-D *level*] [-j] [-n] [-o *ofile*] [-t] [-V *level*] *files*

/usr/lib/acct/csadrep [-D *level*] [-o *ofile* [-N]] *files*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csadrep command reports usage of the NQS and tape daemons. Input is either from a session file created by csabuild(8) or from a binary file created by csadrep with the -o option. The files operand specifies the binary files.

The usage report is written to stdout.

The csadrep command accepts three types of options: input, output, and report selection options.

Input Option

The csadrep command accepts the following input option.

-s *sfile* Specifies the name of the session record file. The csabuild(8) command creates this file. The default is /tmp/Super-record.

Output Options

The csadrep command accepts the following output options.

-A Reports all jobs, including those that have not completed. By default, only jobs that have completed are reported.

-D *level* Sets the debugging level. Level 1 is slightly verbose, and level 10 is very verbose. Debug output is written to stderr. By default, debugging is turned off.

-o *ofile* Specifies the name of the binary output file. csadrep can process this file, using the files parameter.

-N Does not generate a usage report. You must use the -o option with this option.

-V *level* Sets the verbose level of the usage report. The levels are 0 through 3. Level 0 is terse, and level 3 is extremely verbose. The default is level 0.

Report Selection Options

The `csadrep` command accepts the following report selection options.

`-a` Reports usage for all daemons. This is equivalent to `-jnt`.

`-j` Reports NQS and interactive job usage. This is the default.

`-n` Reports NQS daemon usage.

`-t` Reports tape daemon usage.

EXAMPLES

Example 1: The following example generates an NQS and tape daemon usage report from the session record file `srec`. The verbose level is set to 3, and binary output file `drep.1` is created.

```
csadrep -a -V 3 -s srec -o drep.1
```

Example 2: The following example generates a terse usage report for all the daemons. Input is from three previously created binary files.

```
csadrep -a drep.1 drep.2 drep.3
```

SEE ALSO

`csabuild`(8), `csanqs`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csaedit – Displays, verifies, and deletes records from various accounting files

SYNOPSIS

/usr/lib/acct/csaedit [-a | -x [-t]] [-b *offset*] [-n *nqsfile* | -N *pnqsfile* | -T *tpfile*] [-o *ofile*] [-r *reclist*] [-v]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csaedit command verifies and outputs accounting records in binary or ASCII form. Only valid records are displayed. You can delete selected records from the output file.

To ensure that all of the bad records were removed, reverify the accounting file after deleting any records. Verification can be done with either the csaedit or csaverify(8) command.

The csaedit command accepts three types of options: input, output, and record selection options.

Input Options

At most, only one input option can be specified. The csaedit command accepts one of the following input options:

-n *nqsfile* Specifies an accounting NQS file that has not been preprocessed. The format of this file is specified in the /usr/include/acct/dacct.h file.

-N *pnqsfile* Specifies a preprocessed NQS accounting file. This file is created by csanqs(8). This is the default with Pnqacct1 as the file name.

-T *tpfile* Specifies a tape accounting file. The format of this file is specified in the /usr/include/acct/dacct.h file.

Output Options

The csaedit command accepts the following output options:

-a Specifies ASCII output. The default is to output binary data.

-o *ofile* Specifies the output file. The default is stdout.

-t Outputs CPU times. If you specify -N, queue wait time is also displayed. You must use this option with either the -a or the -x option.

-v Specifies verbose mode. Verbose output is written to stderr.

-x Specifies no execute mode. Only the records to be deleted are displayed. The selected records are not actually deleted. Output is written to stderr.

Record Selection Options

The `csaedit` command accepts the following record selection options:

`-b` *offset* Specifies the byte offset of the record to be deleted. This offset can be obtained from `csaverify`(8).

`-r` *reclist* Specifies the record numbers of the records to be deleted. *reclist* is a comma-separated list.

EXAMPLES

Example 1: The following example outputs the preprocessed NQS file in ASCII. CPU times are reported.

```
csaedit -N Pnqacct1 -at
```

Example 2: The following example deletes records 2, 10, and 15 from an NQS file. The output is written to file `nqacct.NEW` and verbose output is written to file `err`.

```
csaedit -n nqacct1 -r 2,10,15 -v -o nqacct.NEW 2> err
```

FILES

`/usr/include/sys/accthdr.h`	Defines the accounting header
`/usr/include/acct/dacct.h`	Defines the daemon accounting header

SEE ALSO

`csanqs`(8), `csapacct`(8), `csaverify`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csafef2 – Produces summarized session records

SYNOPSIS

/usr/lib/acct/csafef2 [-A] [-s *sessionfile*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csafef2 command is a template that shows a site how to write a program that summarizes session file records and outputs them as ASCII text. Records are summarized by user name, job ID, and account ID. Information is not reported for records with account IDs of –2 (queued and waiting NQS jobs) and 0. (csafef2 is not intended to execute in its released state. See NOTES.)

The csafef2 command template accepts the following options:

-A	Reports both terminated and active sessions. The default is to report ony terminated sessions.
-s *sessionfile*	Specifies the session file name. The default is /tmp/Super-record, which is the output from the csabuild(8) command.

NOTES

The csafef2 command is not a stand-alone command, but a template. It executes only after local source modification at sites with source licenses. Due to license restrictions on code included in csafef2, this template available only at source sites.

SEE ALSO

csabuild(8), csafef(8), csaibm(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csafef – Formats data for front-end accounting system

SYNOPSIS

/usr/lib/acct/csafef [-a] [-s *sessionfile*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csafef program is a template that shows a site how to build a program that formats data for a front-end accounting system. (It is not intended to execute in its released state. See NOTES.)

The csafef program template accepts the following options:

-a Includes session records only for jobs that have terminated. If you omit -a, records for all jobs are formatted, including those jobs that have not terminated.

-s *sessionfile* Specifies the session file name. The default is /tmp/Super-record, which is the output from the csabuild(8) command.

NOTES

The csafef program is not a stand-alone command, but a template. It executes only after local source modification at sites with source licenses. Due to license restrictions on code included in csafef, these templates are available only at source sites.

BUGS

The -u, -v, and -D, options are allowed also, but they are not used by csafef. These unused options are printed in the csafef usage message.

SEE ALSO

csabuild(8), csafef2(8), csaibm(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csagcon – Consolidates accounting data for session and pacct files

SYNOPSIS

csagcon [[[-a] [-c] [-j [-N]] [-u]] [-g]] [-d *flags*] [-m *nword2*] [-o *outfile*] [-s *sort_type*] [-A] [-M *nword1*] [-R *request_file*] [-S *srec_file*] [-T *table_file*]

csagcon [[[-a] [-c] [-j [-N]] [-u]] [-g]] [-d *flags*] [-m *nword2*] [-o *outfile*] [-s *sort_type*] [-C] [-M *nword1*] [-R *request_file*] [-S *srec_file*] [-T *table_file*]

csagcon -P [[[-a] [-c] [-j] [-u]] [-g]] [-d *flags*] [-m *nword2*] [-o *outfile*] [-s *sort_type*] [-M *nword1*] [-R *request_file*] [-T *table_file*] *pacct_files*

csagcon -E [-d *flags*] [-m *nword2*] [-o *outfile*] [-M *nword1*] [-O *nrec*] [-R *request_file*] [-T *table_file*] *pacct_files*

csagcon -I [-d *flags*] [-m *nword2*] [-o *outfile*] [-M *nword1*] [-O *nrec*] [-R *request_file*] [-T *table_file*] *pacct_files*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csagcon command consolidates the accounting data in either a session file, which is created by the csabuild(8) utility, or in a group of pacct (per-process accounting data) files according to user-specified keys. You may specify the data keys and the fields to be consolidated.

To generate reports from the consolidated data file, use the csagfef(8) command. The csagcon -E and -I options allow you to select data for individual processes. By using either of these options and csagfef, you can produce tailored per-process accounting reports similar to acctcom(1) output.

By default, csagcon consolidates a session file named Session-Record by account ID and user ID. Only data for terminated sessions is collected, and the data items that are consolidated are similar to those produced by csacon(8). The unsorted output is written to a file named gacct.

The five types of csagcon command options are as follows:

- Consolidation options
- Record selection options
- Input options
- Output options
- Performance options

Consolidation Options

Consolidation options cannot be used with the `-E` and `-I` options. You can specify multiple options to consolidate the accounting data you have collected, depending upon the capacity of your front-end computer system and the data that is important to report for your site. If you do not specify a consolidation option, the default is to consolidate data for terminated sessions by account ID and user ID (`-au` options). The `csagcon` command accepts the following consolidation options:

`-a` Consolidates records by using account ID as a key.

`-c` Consolidates records by using job class as a key; job class is either interactive or Network Queuing System (NQS). If this option is used with the `-P` option, then the job class for all records is interactive.

`-g` Consolidates records by using group ID as a key. Because not all records have a group ID, you must use this option with at least one of the following consolidation options: `-a`, `-c`, `-j`, or `-u`.

`-j` Consolidates records by using job ID as a key.

`-u` Consolidates records by using user ID as a key.

`-A` Consolidates all sessions, including those that have not completed. By default, only sessions that have completed are consolidated. This option must be used with the `-S` option and is mutually exclusive with the `-C` option.

`-C` Consolidates only active sessions. By default, only sessions that have completed are consolidated. This option must be used with the `-S` option and is mutually exclusive with the `-A` option.

`-N` Consolidates each portion of an NQS request according to its job ID. This option must be used with the `-j` option and cannot be used with the `-P` option. By default, all portions of a request are processed as though they had the same job ID. This option is useful when a request has multiple job IDs, as in the case of rerun requests or requests that use `pipeclient` or `netclient`.

Record Selection Options

By default, `csagcon` consolidates the accounting data. When this occurs, per-process information is no longer available.

The following options generate unconsolidated output. They cannot be used with the `-P`, `-S`, or `-s` consolidation option.

`-E` Allows access to per-process data found in the `pacct eof` (end of job) record.

`-I` Allows access to per-process data found in the `pacct` base.

Input Options

If you do not specify an input option, the default is to use a session file named `Session–Record` as the input file (the `-S` option). The `csagcon` command accepts the following input options:

`-P` *pacct_file* Lists the `pacct` files to be consolidated. The `-E`, `-I`, `-P`, and `-S` options are mutually exclusive. *pacct_file* is a comma separated list of `pacct` filenames.

`-S` *srec_file* Names the session file to be consolidated. The `-E`, `-I`, `-S`, and `-P` options are mutually exclusive. The default filename is `Session-Record`.

`-T` *table_file* Names the table initialization file. By default, *table_file* is `/usr/lib/acct/table_init`. This option is used only in developing source code. It is recommended that you not use an alternate table initialization file because `csagcon` expects the data variable names defined in this file.

Output Options

If you do not specify output options, the default is to consolidate only the default items and to write the unsorted data to a file named `gacct`. The `csagcon` command accepts the following output options:

`-d` *flags* Specifies the debug flags. The flags are as follows:

Flag	Description
`0000`	No debugging (default)
`0001`	Variable mapping information
`0002`	Prefix mapping information
`0004`	Consolidated data array information
`0010`	Timing information
`0020`	Memory allocation information
`0040`	File information
`0100`	Identifying keys from the input records
`0200`	Output information
`0400`	Tape device group name information

To specify multiple flags, add the numerical values. For example, to produce memory allocation and file information, set the flag to 060 (020 + 040). Debugging output is written to `stderr`.

By default, debugging is disabled.

`-o` *outfile* Names the output file where the consolidated data is written. By default, the output is written to a file named `gacct`.

`-s` *sort_type* Sorts the output file. By default, the output is unsorted. This option cannot be used with the `-E` and `-I` options. Valid *sort_types* are as follows:

sort_type	Action
`acid`	Sorts first by numeric account ID then user ID. Must be used with the `-a` or `-u` option.
`acname`	Sorts first by account name then by username. Must be used with the `-a` or `-u` option.
`jclass`	Sorts first by job class then by job ID. Must be used with the `-c` and `-S` option.

`uid` Sorts first by numeric user ID then by account ID. Must be used with the `-a` or `-u` option.

`username` Sorts first by username then by account name. Must be used with the `-a` or `-u` option.

`-R` *request_file* Names the file that contains a list of data fields to be consolidated. If `-R` is not specified, `csagcon` consolidates the default items, which are listed in *UNICOS Resource Administration*, Cray Research publication SG–2302. These default items are similar to the items that `csacon`(8) consolidates.

Performance Options

You can control how much memory the program allocates each time it reserves a block of memory for various data structures. The `csagcon -d 020` option shows run-time memory allocation information.

When per-process data is generated with the `-E` and `-I` options, you can specify the number of records `csagcon` processes before outputting any data.

The `csagcon` command accepts the following performance options:

`-m` *nword2* Specifies the number of words of memory that the program reserves on all allocations except the first. By default, 131072 words (256 clicks) are allocated.

`-M` *nword1* Specifies the number of words of memory that the program reserves on the first allocation. By default, 393216 words (768 clicks) are allocated.

`-O` *nrec* Specifies the number of per-process data records to process before writing to the output file. By default, 2000 records are processed.

This option can only be used with the `-E` or `-I` options.

EXAMPLES

The following example consolidates the data found in a session file named `Super-record.0815`. Only the default items for terminated sessions are consolidated. The output, which is written to the file `gacct.0815`, is sorted first by user ID, then by account ID.

```
csagcon -S Super-record.0815 -o gacct.0815 -s uid
```

NOTES

Users may require privilege to access the `/dev/kmem` file. If a user does not have the appropriate privilege, `csagcon` will terminate with an error.

FILES

`acct`(5)	Per-process accounting (`pacct`) file.
`/usr/lib/acct/table_init`	Default table initialization file.

SEE ALSO

`acctcom`(1)

`csabuild`(8), `csacon`(8), `csagfef`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csagfef – Formats consolidated accounting data

SYNOPSIS

csagfef [-c] [-d *flags*] [-f *infile*] [-v] [*source_file*] ...

csagfef [-d *flags*] [-f *infile*] [-v] [-D *name*[=*def*]] [*source_file*] ...

csagfef -h [-f *infile*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csagfef generic front-end formatter formats the consolidated accounting data into either an ASCII report or a binary data file. The consolidated data is created by the generic data consolidator, csagcon(8).

The data is formatted based on the specifications found in csagfef source scripts, which are written in a language based on awk. The *source_file* operand specifies the names of these scripts. By default, the source scripts are read from stdin (standard input).

The language is described in detail in *UNICOS Resource Administration*, Cray Research publication SG–2302. It is similar to the language recognized by the tsar(8) command.

The csagfef command accepts the following options:

-c Compiles the source files only. This option is used to debug source files. The input data file is not formatted.

-d *flags* Specifies the debug flags. The flags are as follows:

Flag	Description
0001	Lexical scanning
0002	Expression compilation
0004	Table entry
0010	Code execution
0020	Stack contents
0040	Input file parsing
0100	Symbol table searching
0200	Table allocation

To specify multiple flags, add the numeric values. For example, to enable lexical scanning and code execution debugging, set the flag to 011 (001 + 010).

By default, debugging is disabled.

-f *infile* Specifies the name of the input file to be formatted. The input file was created by csagcon(8). The default input filename is gacct.

`-h` Produces information about the input file including the variable names, constant variables, and number of data records. No source files are compiled. Output is written to `stdout` (standard output).

`-v` Specifies that verbose output be written to the `stderr` file when the source file is processed.

`-D` *name*[=*def*] Defines a symbol name to be used in the source file during execution. *def* may be a number or a character string. Character strings must be delimited by escaped double quotes. (See the EXAMPLES section.)

By default, *def* is defined as the number 1.

source_file ... Specifies source script or scripts to be used to format the input file.

EXAMPLES

In this example, the file `gacct.0815` is being formatted according to the source file `mk_rpt`. The symbol `uname` is defined as the string `user1`.

```
$ csagfef -f gacct.0815 -D uname=\"user1\" mk_rpt
```

FILES

`/usr/src/cmd/acct/src/csa/csagfef/examples`
Directory containing example source scripts

`gacct` The default input file created by `csagcon`(8) and used with the [`-f` *infile*] option

SEE ALSO

`csagcon`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

The AWK Programming Language, by A. V. Aho, B. W. Kernighan, P. J. Weinberger, Addison-Wesley, 1988

NAME

csaibm – Converts session records into IBM format

SYNOPSIS

/usr/lib/acct/csaibm [-a] [-A] [-D *level*] [-o *outfile*] [-r] [-s *sessionfile*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csaibm command is a template that shows a site how to write a program that converts a session file to IBM format. It is not intended to execute in its released state.

The csaibm command template accepts the following options:

-a Does not translate ASCII strings to EBCDIC. The default is to output all strings in EBCDIC.

-A Converts session records for all jobs, including those that have not terminated. The default is to convert records only for terminated sessions.

-D *level* Sets the debug level. Level 1 is slightly verbose; level 10 is very verbose. By default, debugging is turned off.

-o *outfile* Writes the EBCDIC records to file *outfile*. The default is to write the output to stdout.

-r Does not report rerun portions of an Network Queuing System (NQS) request separately. Data from all portions are added together and written to one ibmiduse record. The default is to write separate usage records for each portion of a rerun NQS request.

-s *sessionfile* Specifies the session file name. The default is /tmp/Super-record, which is the output from csabuild(8).

BUGS

When you specify -r and there are NQS jobs that have rerun portions, ibmnqs records do not have the correct stop time. The first ibmnqs record have the stop time of the entire request. All subsequent ibmnqs records have a stop time of 0. When you do not specify -r, the correct stop times are written.

SEE ALSO

csabuild(8), csafef(8), csafef2(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csajrep – Prints a job report from the session record file

SYNOPSIS

/usr/lib/acct/csajrep [-b] [-c] [-e] [-h] [-m] [-q] [-t] [-w] [-x] [-y] [-A] [-B] [-C] [-F] [-J] [-L] [-M] [-S *file*] [-T] [-W] [-Z]

/usr/lib/acct/csajrep [-a *acid*] [-b] [-c] [-e] [-h] [-j *jid*] [-m] [-q] [-s *reqid*] [-t] [-u *uid*] [-w] [-x] [-y] [-A] [-B] [-C] [-J] [-L] [-M] [-S *file*] [-T] [-W] [-Z]

/usr/lib/acct/csajrep [-N [-A]] [-S *file*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csajrep command reports session accounting information from the session file, which is created by the csabuild(8) command.

The csajrep command accepts the following three types of options: input, selection, and printing. If you omit the options, input is read from /tmp/Super-record and all records from each completed session are reported. You can select a subset of sessions by specifying the user ID, account ID, job ID, or Network Queuing System (NQS) request ID for which you seek accounting information.

Input Options

The following option specifies an input file for the job accounting report:

-S *file* Specifies the name of a session file created by csabuild(8). The default file is /tmp/Super-record.

Selection Options

By default, the -a, -j, -s, and -u selection options report entire sessions. The -e option limits the report to records that match the selection criteria. The -e option must be used with at least one of the following options: -a, -j, -s, or -u. You cannot specify the -F option in combination with any of the following options: -a, -j, -s, or -u.

Otherwise, the csajrep command accepts the following selection options:

-a *acid* Specifies a numeric account ID or an account name.

-e Selects only the records that meet the selection criteria as defined by the -a, -j, -s, and -u options. The -e option must be used with at least one of these four options. By default, csajrep outputs all of the records from sessions that contain at least one record that meets the selection criteria.

-j *jid* Specifies a (numeric) job ID.

-s *reqid* Specifies an NQS request number (*reqid*).

`-u` *uid* Specifies a numeric user ID of a user login name.

`-F` By default, `-F` displays all records for completed sessions. This option cannot be used with any of the following options: `-a`, `-e`, `-j`, `-s`, and `-u`. When `-F` is used with the `-A` option, all records for both active and completed sessions are reported. When `-F` is used with the `-Z` option, all records are reported except those with a job ID (*jid*) of 0.

`-N` Selects only NQS requests. Information about each segment of a request is reported by `job_id`/`user_id`/`account_id` combination. For a more complete description of information available on NQS requests, see *UNICOS Resource Administration*, Cray Research publication SG–2302.

`-Z` Ignores records for which the job ID is equal to `0`.

Printing Options

The `csajrep` command accepts the following printing options:

`-b` Reports system billing unit (SBU) usage.

`-c` Reports CPU usage.

`-h` Suppresses report headers.

`-m` Reports multitasking CPU information.

`-q` Reports queue wait time and queue type for NQS jobs.

`-t` Prints summary information.

`-w` Reports the I/O wait time while a process is locked in memory and the memory high-water mark.

`-x` Reports I/O statistics.

`-y` Reports SDS usage statistics.

`-A` Reports both active and completed sessions. By default, only completed sessions are reported.

`-B` Reports process and session starting times.

`-C` Reports system call and interrupt CPU times.

`-J` Reports job ID.

`-L` Puts form feeds at the end of each session.

`-M` Reports Cray MPP system usage statistics. If there is no attached MPP system, the `-M` option reports 0.

`-T` Prints only summary information for each session.

`-W` Reports I/O wait time while a process is not locked in memory.

NOTES

A session may contain multiple user IDs and account IDs, because the user may have executed commands such as su(1) or newacct(1). Also, NQS sessions have multiple job IDs when the request is rerun.

pacct end-of-job records, NQS accounting records, and connect-time records do not contain account IDs. Thus, when the -a and -e options are used together, no end-of-job, NQS, or connect-time information is reported.

Only NQS information is reported when the -s and -e options are used together, because only NQS accounting records contain the NQS request ID.

Accounting records sometimes have a job ID of 0 when Cray system accounting (CSA) cannot determine the correct job ID.

When the -u option is used without the -e option, all records for sessions containing at least one accounting record for the specified user are displayed.

For example, if user1 executes the command rsh cray who from a remote host, then the command /usr/lib/acct/csajrep -u user1 -JBc -S Super-record would produce output similar to:

```
JOB ID   ACCOUNT  LOGIN    COMMAND         START           USER-TIM SYS-TIM
         NAME     NAME     NAME            TIME            [SECS]   [SECS]
======== ======== ======== ======== ==================== ======== ========
     134 Xydev    user1    who      Jul 19 10:10:27 1993    0.008    0.012
     134 System   root     rshd     Jul 19 10:10:27 1993    0.001    0.010
         END OF JOB AT Mon Jul 19 10:10:28 1993
```

The rshd(8) command was executed by root on behalf of user user1; thus it is reported by the csajrep -u option.

When used with -u, the -e option suppresses the printing of all accounting records which are not for the specified user. In the previous example, the command /usr/lib/acct/csajrep -eu user1 -JBc -S Super-record would produce output similar to:

```
JOB ID   ACCOUNT  LOGIN    COMMAND         START           USER-TIM SYS-TIM
         NAME     NAME     NAME            TIME            [SECS]   [SECS]
======== ======== ======== ======== ==================== ======== ========
     134 Xydev    user1    who      Jul 19 10:10:27 1993    0.008    0.012
```

EXAMPLES

Example 1: The following example generates a list of commands by job that user jdoe executed. The list includes job ID, start time, and both terminated and nonterminated jobs in the output:

```
csajrep -u jdoe -ABJ
```

Example 2: The following example prints information about NQS job `4140`:

```
csajrep -s 4140 -BJcqtwx
```

SEE ALSO

`csabuild`(8), `csaline`(8), `csanqs`(8), `csarun`(8), `csatape`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csaline – Preprocesses connect-time sessions

SYNOPSIS

/usr/lib/acct/csaline [-l *file*] [-o *file*] [-p] [-t] [-u *file*] [*path*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csaline command processes a utmp(5) file and outputs a list of connect sessions in ctmp.h format sorted by ending time. This list is written to standard output. You also can obtain line usage, reboot, and boot time information.

The *path* operand specifies the path name of a connect-time accounting file. The default path is /usr/adm/acct/work/Wwtmp. The file names are generated by appending a number to the end of the path name. A file with an appendix of 1 (*path*1) must exist; other files need not exist. For example, the path /w/Wwtmp represents the files /w/Wwtmp0, /w/Wwtmp1, and so on. /w/Wwtmp1 must exist; otherwise, csaline fails.

The csaline command accepts the following options:

-l *file* Writes the line usage summary to *file*. This file contains a summary of line usage showing line name, number of minutes used, percentage of total elapsed time used, number of sessions charged, number of logins, and number of logoffs. This file helps track line usage, identify bad lines, and find software and hardware inconsistencies and errors. Hang-up, termination of login(1), and termination of the login shell each generate logoff records. Thus, the number of logoffs is often three to four times the number of sessions. See the init(8) and utmp(5) man pages for more information.

-o *file* Writes an overall record for the accounting period that gives the starting time, ending time, number of reboots, and number of date changes.

-p Prints the input in ASCII showing the line name, login name, and time. The time is in both number and date/time formats. Processing of the data is not done.

-t Uses the last time found in the input when calculating the connect time for active login sessions. This ensures reasonable and repeatable numbers for noncurrent files. The default is to use the current time.

-u *file* Writes system boot times found in the input file to *file*.

NOTES

If the file `path0` exists, it must be in `ctmp.h` format. All other input files should be in `utmp`(5) format.

EXAMPLES

The following example shows how to extract as much information as possible from file `wtmp`:

```
csaline -t -l lineuse -o reboots /usr/adm/acct/work/1220/1305/Wctime > Pctime1
```

FILES

`/etc/wtmp`	Login records format

SEE ALSO

`init`(8)

`utmp`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

csam – Displays system activity data on a dumb terminal

SYNOPSIS

/usr/bin/csam [-f *replayfile*] [-h *host*] [-i *interval*] [-l *logfile*] [-p *passes*] [-d] [-s] [-u] [-C] [-D] [-F] [-H] [-K] [-L] [-M] [-P] [-T] [-W] [-X] [-Y]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csam command displays various system performance statistics. It uses the curses(3) library to drive the terminal so that many terminal types may be supported. The screen is refreshed every *interval* seconds, for the specified number of *passes*. If *passes* is not specified, csam runs continuously. Specify the system to be monitored with the *host* option. If *host* is not specified, the local system is monitored.

The csam command communicates through rpc(3C) with the sam server, samdaemon(8), running on the *host* system to obtain the information to be displayed. The server can run on a Cray Research system or on a connected operator workstation (OWS).

The csam command checks the size of the screen and the LINES environment variable and can adjust most displays to fill the screen.

The csam command accepts the following options:

-d Sets the debug flag. Debug messages are written to the log file.

-f *replayfile* Specifies a file name to be used for replay of previously recorded sam data. The default is client.recplay. You can create the replay file by using either the csam or xsam(8) command. The data written to the record file are those necessary to produce the displays on the client at the time of recording. An xsam client can record data from several hosts at the same time. Replay is at a 1-second refresh rate, but you can adjust this rate by using the <+> and <-> keys. When it reaches the end of the data in the replay file, csam displays a message at the top of the screen. You can record or replay data by using the record and replay control panel, which you select with the <z> key. The <d> key returns you to normal client displays and the help screen is presented automatically.

-h *host* Specifies the network name of the host to be monitored. If the host is a server running on an OWS, you must specify the network name of the OWS. The default host is the local system.

-i *interval* Sets the refresh rate to *interval* seconds. The default interval is set by the server.

-l *logfile* Specifies the name of the debugging messages log file. The default log file name is client.log.

-p *passes* Specifies the number of times csam will refresh the screen. The default is –1, which gives an infinite display.

`-s` Sets the `user/kernel/idle/wait` option on the kernel display. (This may be the only option for systems with more CPUs than can be displayed on the screen. See the `-K` option.)

`-u` Sets user and system CPU time option for the kernel display. (See the `-K` option.) The `-u` option is the default.

`-C` Selects the configuration display, which shows aspects of the hardware and software configuration of the target system.

`-D` Selects the disk display, which shows each disk device and the transfer rate in the last interval. The transfer rate is presented in Mbyte/s and as a bar graph on a logarithmic scale.

`-F` Selects the logical device cache display. This display is similar to the display provided by the `ldcache`(8) command in refresh mode. To select the next cached file system, press <n>, and then press <r> to reset the display for the current file system.

`-H` Selects the help display, which gives a brief explanation of the various displays and commands available.

`-K` Selects the kernel display. This display has the following parts:

- One part that shows various kernel counters from the `sysinfo`, `syserr`, `syswait`, `pws` tables. Four columns of numbers are displayed; those shown as floating-point numbers are counts per second in the refresh interval, those shown as integers are absolute values. Any number that refers to memory or swap usage is in click units (512 64-bit words).
- One part that displays a number of bar graphs. The first two graphs are the hit rates (read and write) for the system buffer cache. How the remaining graphs are used depends on the screen size, the number of CPUs in the system and the values of the `-s` and `-u` options. If there are enough lines on the screen to display all the CPUs and the `-u` option (the default) is used, the user (`*`) and system (`=`) CPU usage for each of the CPUs is displayed. If the `-s` option is in effect or there are not enough lines on the screen to display all the CPUs, the `kernel/user/wait` and idle percentages for the whole system are displayed.

`-L` Selects the logical device display. This display shows the number of Mbyte/s transferred during the previous interval for each logical device as a number and shows the same information also as a bar graph in a logarithmic scale. File systems cached by the `ldcache`(8) command are displayed on two lines: the first line is marked with the character `C` and shows the data transferred between the cache and the user (display character `*`), the second line is marked with the character `D` and shows the data transferred between the cache and the disk (display character `=`).

-M Selects the memory display. A large section to the left shows a map of the common memory usage, low memory in the upper left corner and the high end of memory at the bottom right. Each character on the map represents the number of clicks (512 64-bit words) of memory given by the scale shown in the legend box. Different characters indicate the state of the process (or one of the processes) occupying that space. A capital letter indicates the start of a process and a small letter indicates a continuation of the same process. For example, the string SSSRrrr indicates three (or more) small sleeping processes followed by one larger process that is runnable.

To the right of the common memory map is a legend box and a box showing percentage memory and swap utilization. The oversub field is the oversubscription factor for central memory (in other words, the amount of memory that would be required to hold all the processes in the system).

At the bottom of the display are a series of counters showing the number of processes in various states, both in memory and swapped.

-P Selects the process display. This display is similar to the output from the ps(1) command. This display is not refreshed. You can scroll to the next screen by using the <n> key and you can reset to the first full screen by using the <p> key. You can also specify a numeric process ID followed by a <RETURN>, and the display will switch to a refreshing snap of the process selected. If csam does not find the process or the process terminates while the snap display is active, then csam displays an appropriate message and the screen returns to an updated process display.

-T Selects the tape display, which shows each tape device and the transfer rate in the last interval. The data is presented as Mbyte/s and as a bar graph on a logarithmic scale. (Deferred implementation.)

-W Selects the swap map display. This display is identical to the memory map display, except that the map section maps the swap device instead of central memory. On all Cray Research systems with partitioned swap devices the start of each swap partition (after the first) is indicated by the character *.

-X Selects the top processes display. This display shows a sorted list of processes, each line containing a process name, its process ID, the percentage of one CPU used by that process in the last interval and a bar graph. The bar graph represents the type of CPU usage. User CPU time is marked with an *, idle time with a ., and system CPU time with an =. A total of all active processes, including those that do not fit on the screen, but excluding idle processes, is shown at the top of the screen. Multitasked processes are marked with the character M.

`-Y` Selects the system call display. This display shows a sorted list of system call activity, including the total time spent processing system calls in the last interval, shown as a percentage of one CPU, and the total number of calls per second. Each line on the display provides information for one system call and contains the number of calls per second and the percentage of time spent processing those calls. This second percentage is a percentage of the total system time figure. Therefore, if the total system time is 25% and the system time for the `write` system call is 20%, then the time spent in the `write` system call is 5% of one CPU.

Interactive Input

After `csam` is running on your terminal, you can use the following keys to change displays, move within a display, increase or decrease refresh rates, and exit:

Key	Description
<c> or <C>	Selects host system configuration display.
<d> or <D>	Selects disk display.
<e> or <E>	Quits program.
<f> or <F>	Selects ldcache display.
<g> or <G>	Leaves single-step mode.
<h> or <H>	Selects help screen.
<k> or <K>	Selects kernel display.
<l> or <L>	Selects logical device display.
<m> or <M>	Selects memory display.
<n> or <N>	Advances to next page (disk, ldcache, logical device, process, and tape displays).
<p> or <P>	Selects process display.
<q> or <Q>	Quits program.
<r> or <R>	Resets bar graphs and ldcache display.
<s> or <S>	Selects `user/kernel/wait/idle` usage on kernel display.
<t> or <T>	Selects tape display. (Deferred implementation.)
<u> or <U>	Selects user and system CPU usage on kernel display.
<w> or <W>	Selects swap map display.
<x> or <X>	Selects top process display.
<y> or <Y>	Selects system call display.
<z> or <Z>	Selects record/replay control panel.
<+>	Increases refresh interval by 1 second.

<-	Decreases refresh interval by 1 second. The refresh interval cannot be lowered below the refresh rates set by the server.
<.>	Enters single-step mode. This freezes the display until the space bar is pressed to advance the screen.
Space bar	Advances display in single-step mode.
Numeric input	Selects a process ID for the snap display. This is accepted only in the process display. The ID must be terminated by a carriage return. The erase and kill characters are accepted during this numeric input.

BUGS

Because system tables can change while they are being read, `csam` occasionally may produce a misleading display. This usually is corrected during the next refresh.

SEE ALSO

`ldcache`(8), `sam`(8), `samdaemon`(8), `xsam`(8)

`ps`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csanqs – Preprocesses the NQS accounting files

SYNOPSIS

/usr/lib/acct/csanqs [-n *file*] [-t] [-D *level*] [*pathname*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csanqs command processes the Network Queuing System (NQS) accounting files and generates one output record for each segment of an NQS job.

The csanqs command accepts the following options and operand:

-n *file* Writes a list of null (rejected) NQS jobs to the specified *file*.

-t Calculates queue wait time as the difference between the current time and the time the request entered the pipe queue for requests that are queued and have never executed. The wait time is recalculated when the process begins execution. If you use -t with recycled NQS data, the queue wait time for queued but never executed requests will be erroneous. The wait times will be corrected when the requests start execution and csanqs is run again. For requests that have executed on a CPU and are currently queued, or when -t is omitted, queue wait time is reported only after the request begins execution on a CPU.

-D *level* Sets the debug level. Level 1 is slightly verbose; level 10 is very verbose. By default, debugging is turned off.

pathname Specifies the path name of the NQS accounting file. The default path name is /usr/adm/acct/work/Wnqacct. To generate the file names, append a number to the path name. A file with an appendix of 1 (*pathname*1) must exist. Other files need not exist. For example, the path name /w/nqacct represents the files /w/nqacct0, /w/nqacct1, and so on. /w/nqacct1 must exist; otherwise, csanqs fails.

NOTES

The csanqs command calculates queue wait time by subtracting the time the request entered the pipe queue from the time the request began executing on your Cray Research system. For a request that has been checkpointed, the amount of time the request spent checkpointed is also considered queue wait time. The queue wait time includes the amount of time that queues were stopped or disabled, the time that the Cray Research system was down, and the time the request spent waiting because it was submitted with the qsub -a option (see qsub(1)).

EXAMPLES

A typical usage of `csanqs` is as follows:

```
csanqs /usr/adm/acct/work/1201/1305/Wnqacct > Pnqacct1
```

FILES

`/usr/adm/acct/day/nqacct*`	Current NQS accounting files

SEE ALSO

`qsub`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csapacct – Verifies and deletes records from a pacct file

SYNOPSIS

/usr/lib/acct/csapacct [-o *offset*] [-r *recnum*] [-v] *inputfile outputfile*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csapacct command reads records from a pacct file and verifies each record. Records that are not valid are ignored. Only "good" records, (those not containing truncated or otherwise corrupted data) are written to the output file.

If you specify a byte offset, the record that resides at this location is deleted. Byte offsets of "bad" records (those containing data that is not valid) are obtained from the csaverify(8) program.

To be sure that all of the bad records were deleted, reverify the pacct file after deleting records. Verification can be done with either the csapacct or the csaverify(8) command.

The csapacct command accepts the following options and operands:

-o *offset*	Specifies the byte offset of the record to be deleted. The offset can be obtained from csaverify(8).
-r *recnum*	Specifies the record number of the base pacct record to be deleted.
-v	Specifies verbose mode. Verbose output is written to stderr.
inputfile	Input file. This file must be in acct(5) format.
outputfile	Output file.

EXAMPLES

The following example shows how to verify and delete bad records from pacct file pacct1. The output is written to file pacct.NEW. Verbose mode is turned on.

```
csapacct -v pacct1 pacct.NEW
```

SEE ALSO

csaedit(8), csaverify(8)

acct(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

csaperiod – Runs periodic accounting

SYNOPSIS

/usr/lib/acct/csaperiod [-e *MMDDhhmm*] [-r] [-s *MMDDhhmm*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csaperiod command directs the processing of the daily consolidated accounting files, which are created by the csarun(8) command. The csaaddc(8) command merges the daily files into one file. The csacrep(8) command then generates a report based on the merged file.

The progress of csaperiod is recorded in the pdactive file. When an error is detected, a message is written to the operator, and mail is sent to root and adm. Further data processing is halted.

The cron(8) command usually initiates csaperiod.

The csaperiod command accepts the following options:

-e *MMDDhhmm*
Selects consolidated accounting data generated at or before the specified date, *MMDDhhmm*.

-r Removes the daily data files after processing is done. The default is to leave the daily data files in the /usr/adm/acct/sum/data directory.

-s *MMDDhhmm*
Selects consolidated accounting data generated at or after the specified date, *MMDDhhmm*.

NOTES

By default, csaperiod processes all the /usr/adm/acct/sum/data/* files. The mail recipients (root and adm) can be changed by modifying the MAIL_LIST parameter in the /etc/config/acct_config file.

FILES

/etc/config/acct_config	Accounting configuration file
/usr/adm/acct/fiscal/data/*MMDD*/*hhmm*/cms	Periodic command usage data in cms record format
/usr/adm/acct/fiscal/data/*MMDD*/*hhmm*/pdacct	Periodic condensed data files
/usr/adm/acct/fiscal/rpt/*MMDD*/*hhmm*/rprt	Periodic report files
/usr/adm/acct/nite/pdactive	Log file

`/usr/adm/acct/nite/E*`*MMDDhhmm*	Error messages
`/usr/adm/acct/sum/data/`*MMDD*`/`*hhmm*`/cacct`	Daily condensed data files

SEE ALSO

`acctcms`(8), `cron`(8), `csaaddc`(8), `csacon`(8), `csacrep`(8), `csarun`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csaperm – Changes group ID and permissions of accounting files

SYNOPSIS

/usr/lib/acct/csaperm [-v]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csaperm command sets the group IDs to adm (administrator) in the /etc/csainfo and /etc/wtmp files, and in all the accounting files in the /usr/adm/acct directory. It also sets the file permissions so that a user in the group adm with permission bit acct set can run accounting. This makes it unnecessary to have super-user permissions to run accounting.

The csaperm command accepts the following option:

-v Specifies verbose mode. File names are reported as changes are made.

FILES

/etc/csainfo	System boot times
/etc/wtmp	Login information
/usr/adm/acct/*	Accounting directories and files

SEE ALSO

udbgen(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csarecy – Recycles unfinished sessions into next accounting run

SYNOPSIS

/usr/lib/acct/csarecy [-r] [-s *file*] [-u *path*] [-A] [-C *path*] [-D *level*] [-N *path*] [-P *path*] [-T *path*]

/usr/lib/acct/csarecy [-r] [-s *file*] [-u *path*] [-C *path*] [-D *level*] [-N *path*] [-P *path*] [-R] [-T *path*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csarecy command retrieves job information from the accounting files of the current accounting period and places it in the accounting files of the next accounting period. The default input is /tmp/Super-record. csabuild(8) marks unfinished accounting jobs (those that do not terminate in a given period of system activity). csarecy takes these records from the session file and puts them into the next period's accounting files directory. This process is repeated until the job finishes.

csarecy also prints reports about unfinished accounting jobs, in the following format:

```
SYSTEM BOOT TIME STARTING AT MMDDYY hh.mm
PRESERVED ACCOUNTING SESSIONS (jobs that are continued)
=====================================================
  JOB ID     USERS       ACCOUNTS              STARTED
---------    -----     -------------  ------------------------
```

The csarecy command accepts the following options:

-r Produces a report on all recycled jobs.

-s *file* Specifies session record *file* (from csabuild(8)) as the input file. The default is /tmp/Super-record.

-A Asks you whether you want to select each job for recycling. When this option is used, you should run csarecy interactively. The -R option cannot be used with this option.

-C *path* Specifies the path name of the output file for connection accounting information. The system adds a 0 to the end of the file name, so the actual file name is *file*0. The default is /usr/adm/acct/work/Pctime.

-D *level* Sets debugging level. Level 1 is slightly verbose; level 10 is very verbose.

-N *path* Specifies the path name of the output file for NQS accounting information. The system adds a 0 to the end of the file name, so the actual file name is *file*0. The default is /usr/adm/acct/work/Pnqacct.

`-P` *path* Specifies the path name of the output file for `pacct` accounting information. The system adds a 0 to the end of the file name, so the actual file name is *file*`0`. The default is `/usr/adm/acct/work/Wpacct`.

`-R` Produces report only; does not recycle jobs. The `-A` option cannot be used with this option.

`-T` *path* Specifies the path name of the output file for tape subsystem accounting information. The system adds a 0 to the end of the file name, so the actual file name is *file*`0`. The default is `/usr/adm/acct/work/Ptpacct`.

`-U` *path* Specifies the path name of the output file for `uptime` accounting information. The system adds a 0 to the end of the file name, so the actual file name is *file*`0`. The default is `/usr/adm/acct/work/Puptime`.

NOTES

By default, recycled jobs are ignored by most accounting programs.

SEE ALSO

`csaaddc`(8), `csabuild`(8), `csacon`(8), `csacrep`(8), `csafef`(8), `csajrep`(8), `csaline`(8), `csanqs`(8), `csaperiod`(8), `csarun`(8), `csatape`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csarun – Processes the daily accounting files and generates reports

SYNOPSIS

/usr/lib/acct/csarun [-A] [-V *level*] [*MMDD* [*hhmm* [*state*]]]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csarun command, usually initiated by cron(8), directs the processing of the daily accounting files. csarun processes connect, daemon, and process accounting files.

If errors occur, csarun does not damage the active accounting files. It records its progress by writing descriptive diagnostic messages to the active file. When an error is detected, a message is written to the operator, and mail is sent to root and adm. Further data processing is halted.

Before invoking csarun on a new accounting period, ensure that the previous invocation of csarun has completed successfully. If this is not done, information about unfinished sessions will not be accurate.

A series of lock files are used to protect against reinvocation. The lock and lock1 files prevent simultaneous invocation.

The csarun command accepts the following options and operands:

-A Accounts for both terminated and active jobs. By default, only terminated jobs are reported. csarun does not recycle active sessions.

-V *level* Controls verification level of accounting data files.

MMDD [*hhmm* [*state*]]
Sets month, day, hour, minute, and state for which csarun will rerun the accounting. If csarun is restarted, the month and day are necessary; other portions are optional.

The csarun command breaks its processing into separate, restartable states using statefile to remember the last state completed. It accomplishes this by writing the state name into statefile. csarun looks in statefile to determine what must be processed next. The states are executed in the following order:

State Name	Description
SETUP	Moves active accounting files into a work directory.
WTMPFIX	Verifies the integrity of the /etc/wtmp file (see utmp(5)). If necessary, date changes are corrected.
VERIFY	Verifies the integrity of the data files.

PREPROC	Preprocesses the /etc/wtmp, Network Queuing System (NQS) accounting, and tape accounting files.
ARCHIVE1	User exit that executes a site-dependent accounting program or script to archive the raw and preprocessed accounting files.
BUILD	Organizes the accounting data into a session record file.
ARCHIVE2	User exit that executes a site-dependent accounting program or script to archive the session record file.
CMS	Generates command summaries.
REPORT	Generates daily accounting reports.
DREP	Generates daemon usage report.
FEF	User exit that executes a site-dependent accounting program or script to format the session record file into a format that is suitable for use on a front end.
USEREXIT	User exit that executes a site-dependent accounting program or script.
CLEANUP	Cleans up temporary files and exits.

Before restarting csarun after a failure, check the active file for diagnostics, then fix any corrupted data files such as pacct or wtmp. The lock files must be removed before csarun can be restarted. If csarun is restarted, you must specify the *MMDD* operand, which specifies the month and day for which csarun will rerun the accounting. The entry point for processing is based on the contents of statefile. To override this entry point, include the desired state on the command line to designate where processing should begin.

NOTES

The mail recipients (root and adm) can be changed by modifying the MAIL_LIST parameter in the /etc/config/acct_config file. You also can change the other parameters defined in the accounting configuration file for your site.

csarun checks the number of free blocks in the file system that contains the accounting files to ensure that it consists of more than 500 blocks. By default, csarun assumes that the file system is /usr; if this is not the case, change symbol ACCT_FS in /etc/config/acct_config accordingly. To change the minimum number of free blocks on ACCT_FS (the default is 500), modify MIN_BLKS in the configuration file.

To remove bad records in accounting data files encountered by csarun, use csaedit(8), csaverify(8), and csapacct(8).

BUGS

If possible, do not restart `csarun` in the SETUP state. Instead, run SETUP manually and restart `csarun` by using the following command line:

`csarun` *MMDD hhmm* WTMPFIX

If `csarun` terminates abnormally and leaves the lock files in place, the next execution of `csarun` will remove these locks, but it also will terminate abnormally.

EXAMPLES

Example 1: The following example shows how to start `csarun`:

```
nohup csarun 2> /usr/adm/acct/nite/fd2log &
```

Example 2: The following example shows how to restart `csarun` at the state specified in `statefile`:

```
nohup csarun 0601 1345 2>> /usr/adm/acct/nite/fd2log &
```

Example 3: The following example shows how to restart `csarun` at a specific state:

```
nohup csarun 0601 1345 BUILD 2>> /usr/adm/acct/nite/fd2log &
```

FILES

`/etc/config/acct_config`	Accounting configuration file
`/etc/wtmp`	Connect time information
`/usr/adm/acct/day/*`	Directory that contains current accounting files
`/usr/adm/acct/nite/active`	Record of accounting progress
`/usr/adm/acct/nite/clastdate`	Record of last date and time that accounting ran
`/usr/adm/acct/nite/lock`	Lock file that prevents simultaneous invocation
`/usr/adm/acct/nite/lock1`	Lock file that prevents simultaneous invocation
`/usr/adm/acct/nite/statefile`	Record of last state that `csarun` was working on or completed
`/usr/adm/acct/sum/data/*`	Directory that contains daily condensed data files
`/usr/adm/acct/sum/rpt/*`	Directory that contains daily accounting reports
`/usr/adm/acct/work/*`	Directory that contains temporary files from daily accounting
`/usr/lib/acct/csa.archive1`	Site-generated user exit program or script to be executed during the `ARCHIVE1` state.
`/usr/lib/acct/csa.archive2`	Site-generated user exit program or script to be execute during the `ARCHIVE2` state.

`/usr/lib/acct/csa.fef`	Site-generated user exit program or script to be execute during the `FEF` state.
`/usr/lib/acct/csa.user`	Site-generated user exit program or script to be execute during the `USEREXIT` state.

SEE ALSO

`acctcms`(8), `cron`(8), `csaaddc`(8), `csabuild`(8), `csacon`(8), `csacrep`(8), `csadrep`(8), `csaedit`(8), `csafef`(8), `csajrep`(8), `csaline`(8), `csanqs`(8), `csapacct`(8), `csaperiod`(8), `csaperm`(8), `csarecy`(8), `csaverify`(8), `fwtmp`(8)

`acct`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

`acct`(5), `utmp`(5), `wtmp`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

csasocket – Processes socket accounting data

SYNOPSIS

/usr/lib/acct/csasocket [-d *family*] [-f *address*:*port*] [-g *address*] [-G *group*] [-h] [-i *interface*] [-l *address*:*port*] [-n] [-N *program name*] [-o *output file*] [-p *protocol*] [-P *pid*] [-r *start*:*end*] [-s] [-S] [-t *type*] [-u *user*] [-v] [-1] [*input file*]

IMPLEMENTATION

All UNICOS systems

DESCRIPTION

The csasocket command processes all socket accounting records contained in the given *input file*, which defaults to /usr/adm/acct/day/soacct.

The csasocket command accepts the following options and operands:

-d *family*
: Identifies the domain (i.e. address family) of the sockets to be processed. Any record found for a socket which does not match *family* will be ignored.

  family can be specified as either the numerical value of the desired domain, or as one of the following:

Family	Description
inet	Process any/all Internet sockets.
unix	Process any/all UNIX domain sockets.

-f *address*:*port*
: Identifies the foreign address and/or port of the sockets to be processed. Any record found for a socket which was not connected to *address* and/or *port* will be ignored (this includes any socket which was never connected, see NOTES below).

  If *port* is specified, the delimiter ":" must be included.

  If the -d option was used to specify a specific domain other than inet, this option is not allowed.

  address can be specified as either the foreign host's name or its internet address in dot notation.

  port can be specified as either the service's name assigned to the port in /etc/services or the numerical value of the port.

`-g` *address*
: Identifies the gateway used by the sockets to be processed. Any record found for a socket which did not use *address* as a gateway will be ignored (this includes those sockets for which a gateway was not known, see NOTES below). Since the route a socket uses may change, this option may not show all sockets that actually used *address* as a gateway.

  If the `-d` option was used to specify a specific domain other than `inet`, this option is not allowed.

  address can be specified as either the gateway's host name or its internet address in dot notation.

`-G` *group*
: Identifies the group ID of the sockets to be processed. Any record found for a socket whose group ID is not *group* will be ignored.

  group can be specified as either the group's name, as found in `/etc/group`, or its numerical value.

`-h`
: Indicates that the contents of each accounting record's header, found in *input file*, are to be written to standard error.

`-i` *interface*
: Identifies the interface used by the sockets to be processed. Any record found for a socket which did not use *interface* to access the network will be ignored (this includes those sockets for which an interface was not known, see NOTES below). Since the interface a socket uses may change, this option may not show all sockets that actually used *interface* to access the network.

  If the `-d` option was used to specify a specific domain other than `inet`, this option is not allowed.

  interface must be specified as the name of the interface followed by its unit number. That is, it must be specified as it is displayed by `netstat`(1B).

`-l` *address*:*port*
: Identifies the local address and/or port of the sockets to be processed. Any record found for a socket which was not opened at *address* and/or *port* will be ignored (this includes those sockets for which the local address/port were not known, see NOTES below).

  If *port* is specified, the delimiter ":" must be included.

  If the `-d` option was used to specify a specific domain other than `inet`, this option is not allowed.

  address can be specified as either the local host's name or its internet address in dot notation.

  port can be specified as either the service's name assigned to the port in /etc/services or the numerical value of the port.

`-n` Indicates that any value displayed should not be translated into that value's equivalent name. This option applies to the following values that may be displayed:

Value	Value displayed if `-n` specified
address	The internet address in dot notation.
port	The numerical value of the port.
group	The numerical value of the group.
user	The numerical value of the user's ID.

`-N` *program name* Identifies the program's name that last issued a close on the sockets to be processed. Any record found for a socket which was not closed by *program name* will be ignored.

Only the first 16 bytes of the program's name will be used to match which sockets are to be processed.

`-o` *output file* Identifies the name of the file where all of the records that were processed are to be written. Any record found in *input file* that matched all of the selection criterion specified on the command line will be written (appended) to *output file*.

If *output file* does not exist, it will be created with the owner and group of the executing user and permissions 0666 masked with the executing user's `umask`(1). `csaperm`(8) can be used to change this file's group ID and/or permissions.

`-p` *protocol* Identifies the protocol (e.g. TCP) of the sockets to be processed. Any record found for a socket which was opened using a protocol other than *protocol* will be ignored.

protocol can be specified as either the numerical value of the desired protocol or as one of the following:

Protocol	Description
`icmp`	Process any/all ICMP protocol sockets.
`tcp`	Process any/all TCP protocol sockets.
`udp`	Process any/all UDP protocol sockets.

`-P` *pid* Identifies the process ID of the last process executing which closed the sockets to be processed. Any record found for a socket whose process ID is not *pid* will be ignored.

`-r` *start*:*end* Identifies the date/time range during which the sockets to be processed were opened and closed. Any socket which was opened before *start* or was closed after *end* will be ignored.

If *end* is specified, the delimiter ":" must be included.

Both *start* and *end* can be specified as follows:

[*cc* [*yymmddH*]]*HMM*

Where each character's meaning is consistent with the `date`(1) command. Any string of digits from 3 to 10, or 12, may be specified. The digit's location in the string is interpreted as indicated above.

`-s` Indicates that a summary of all sockets processed is to be written to standard output following all other information written.

`-S` Indicates that only the summary of all sockets processed is to be written to standard output. That is, information detailing each record processed will not be displayed.

`-t` *type* Identifies the type (e.g. stream) of the sockets to be processed. Any record found for a socket type other than *type* will be ignored.

type can be specified as either the numerical value of the desired type or as one of the following:

Type	**Description**
raw	Process any/all raw sockets.
dgram	Process any/all datagram sockets.
stream	Process any/all stream sockets.

`-u` *user* Identifies the owner of the sockets to be processed. Any record found for a socket whose owner is not *user* will be ignored.

user can be specified as either the user's name, as found in the UDB, or the numerical value of the user's ID.

`-v` Indicates that the following information is to be written to standard error:

1. A detailed description of the criterion used to select sockets for processing.
2. Whether any data (and the amount) was found in the *input file* that was ignored. This would indicate that some data existed in the *input file* between the two valid accounting records indicated.
3. Whether any accounting record (and the amount) was truncated. This would indicate that the start of another accounting record was found imbedded in the indicated record.
4. Whether any accounting record, which was truncated, was repaired by padding the record with binary zeros or skipped.
5. Whether any accounting error record was found in the *input file*. The information contained in this record will be displayed.

If this option is not specified, any data found to exist between any two valid accounting records is silently ignored, any truncated record is silently repaired (i.e. it is reconstructed to it original length by appending bytes of binary zeros or it is skipped), and any error record found is silently ignored.

`-1` Indicates that that the information pertaining to each record written to standard output is to be written as one line per record in *keyword=value*, whitespace delimited, pairs.

The following keywords will be contained in each record:

Keyword	Description
`FAMILY`	The socket's address family/domain.
`TYPE`	The socket's type.
`PROTOCOL`	The socket's protocol.
`CREATED`	The date/time (*ccyymmddHHMMSS.xxx*) the socket was created.
`DESTROYED`	The date/time (*ccyymmddHHMMSS.xxx*) the socket was destroyed.
`OPTIONS`	The socket's options in hexidecimal.
`UID`	The socket's owner's UID.
`GID`	The socket's group ID.
`PROGRAM`	The program executing when the socket was destroyed.
`PID`	The process ID of the executing process that destroyed the socket.
`R_PEEKED`	The number of bytes of data returned to the executing program when the `MSG_PEEK` flag was set.
`R_COUNT`	The number of reads/receives performed on the socket.
`R_BYTES`	The number of bytes of data received on the socket.
`W_COUNT`	The number of writes/sends performed on the socket.
`W_BYTES`	The number of bytes of data sent on the socket.

The following, additional, keywords will be contained in each socket that existed in the internet domain:

Keyword	Description
`L_ADDRESS`	The socket's local address.
`L_PORT`	The socket's local port number.
`F_ADDRESS`	The socket's foreign address.
`F_PORT`	The socket's foreign port number.
`GATEWAY`	The gateway that the socket used.
`INTERFACE`	The interface that the socket used.

input file Identifies the file to be read in and scanned for socket accounting records matching the selection criterion specified on the command line.

The default file is `/usr/adm/acct/day/soacct`.

The `csasocket` command reads each accounting record found in the specified *input file* and, for each "socket" accounting record found, determines whether it matches ALL of the selection criterion specified on the command line. If ANY field does not match the given criterion, the record is ignored. For example:

```
csasocket -d inet -t stream
```

Will result in all stream sockets that existed in the internet domain being processed. While:

```
csasocket -d inet
```

Will result in all sockets that existed in the internet domain being processed.

Only those records selected for processing will be written to the *output file*, provided the `-o` option is specified on the command line.

The contents of each record selected will be displayed on standard output, provided the `-S` option is not specified.

If either the `-s` or `-S` option is specified, a summary of all records selected will be written to standard output. This summary will include the minimum, maximum, average and totals of all numerical values.

NOTES

Only the accounting records that matched all of the selection criterion specified on the command line are written (appended) to the *output file*, whether or not the `-v` option is specified. Error records will not be written to the *output file*.

You can use the `csaperm`(8) command to change all of the accounting files' group ID and permissions as necessary.

All data, contained in each of the accounting records written, reflects the state of that socket at the point in time the last program having the socket open closes it. As a result, some of the fields may not contain any information because that information was not known at the time the accounting record was written. For example, in general a UDP protocol socket is never connected (see `connect`(2)) which will result in its foreign address/port being zero. This would also be true for a TCP protocol socket which is listening for incoming connections (see `listen`(2)). In general, the following fields may not be defined at the point in time the accounting record is written:

1. The foreign address/port (`-f`) will be zero for any socket which was never connected, or for which the connection was broken (for example, the executing program received a `SIGPIPE` signal; see `signal`(2)).
2. The gateway (`-g`) used will be zero for any socket which does not have a foreign address identitied, or for which the route to that foreign address was lost.
3. The interface (`-i`) used will be zero for any socket which does not have a foreign address identified, or for which the interface was taken down while the socket was open.

4. The local address/port (`-l`) will be zero for any socket which was never bound (see `bind`(2)), or for which the connection was broken (for example, the executing program received a `SIGPIPE` signal).

   This occurs because the state of the socket can change over time. As noted earlier, the information contained in each accounting record describes the state of the socket at the time the last `close`(2) is issued. As a result, the information contained in an accounting record could show that the socket did send/received some data, yet not have any/all of these fields defined. For example, a UDP socket which is used to send/receive packets from several different addresses would not have a foreign address/port, gateway, nor interface identified. A socket which was once connected but is terminated abnormally may not even have the local address/port identified, as this information could be cleared out before the program issues the `close`(2).

RETURN VALUES

This program will exit with a zero exit status if no unrecoverable error occurred. If any unrecoverable error was encountered, this program will display an error message and terminate with a non-zero exit status.

EXAMPLES

Example 1:

The following command extracts all internet socket accounting records found in `/usr/adm/acct/day/soacct` and writes them to `soinet`. Also, information describing any invalid data, and any error records, found is displayed on standard error. Only a summary of the records written to `soinet` is displayed on standard output.

```
csasocket -o soinet -v -d inet -S
```

Example 2:

The following command displays the contents of all socket accounting records found in `soinet` along with a summary of all the records found.

```
csasocket -s soinet
```

FILES

```
/etc/group
/etc/hosts
/etc/services
/etc/udb.public
/usr/adm/acct/day/soacct
```

SEE ALSO

`csa`(8), `csaperm`(8)

`date`(1), `umask`(1), `netstat`(1B) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`bind`(2), `close`(2), `connect`(2), `listen`(2), `signal`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

`gid2nam`(3C), `nam2gid`(3C), `nam2uid`(3C), `uid2nam`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

NAME

csaswitch – Checks the status of, enables, and disables process, daemon, and record accounting

SYNOPSIS

Positional formats:

/usr/lib/acct/csaswitch [-D *level*] [[-o *options*] | [-t *threshold*]] on *name pathname*
/usr/lib/acct/csaswitch [-D *level*] off *name*
/usr/lib/acct/csaswitch [-D *level*] check *name*
/usr/lib/acct/csaswitch [-D *level*] [-a] status

Non-positional formats:

/usr/lib/acct/csaswitch [-D *level*] [[-o *options*] | [-t *threshold*]] -c on -n *name* -p *pathname*
/usr/lib/acct/csaswitch [-D *level*] -c off -n *name*
/usr/lib/acct/csaswitch [-D *level*] -c check -n *name*
/usr/lib/acct/csaswitch [-D *level*] [-a] -c status

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csaswitch command checks the current status of, enables, and disables process (kernel), daemon, and record accounting.

The csaswitch command accepts the following options or operands:

-D *level* Sets the debug level for displaying messages during the command's execution. These messages are written to stderr (standard error).

The level can be 0 to 10 with a higher number increasing the number of messages displayed. The default is 0, which results in no messages being displayed.

[-c] on Enables the indicated accounting method.

[-c] off Disables the indicated accounting method.

[-c] check Displays the current status of the indicated accounting method.

[-c] status Displays the current status of all accounting methods.

[-n] *name* Specifies the accounting method being checked, enabled, or disabled.

Valid daemon names are:

Name	Description
dm	Data migration accounting
kernel	System (process) accounting

`nqs`	Network Queuing System (NQS) accounting
`socket`	Socket (network) accounting
`tape`	Tape accounting

Valid record names are:

Name	**Description**
`dio`	Device I/O accounting
`mpp`	Device MPP accounting
`mt`	Multi-tasking accounting
`perf`	Performance accounting
`sds`	Secondary data storage accounting

[`-p`] *pathname* Specifies the file name the accounting method is to use to hold accounting data.

`csaswitch` creates this file if it does not exist already, sets the owner and group of the file to `adm`, and sets the mode to 0660.

`-t` *threshold* (Implementation deferred) Specifies the number of CPU seconds or the memory size below which accounting records will not be written.

`-o` *options* Specifies additional options to be processed by the accounting method being enabled. If more than one option is specified, the options must be whitespace delimited and the entire set enclosed in quotes.

This option is currently only supported for socket accounting.

For socket accounting, the following options are valid:

Option	Field	Description
`inet`	Family	Account for all Internet sockets.
`unix`	Family	Account for all UNIX domain sockets.
`raw`	Type	Account for all raw sockets.
`dgram`	Type	Account for all datagram sockets.
`stream`	Type	Account for all stream sockets.
`icmp`	Protocol	Account for all ICMP protocol sockets.
`tcp`	Protocol	Account for all TCP protocol sockets.
`udp`	Protocol	Account for all UDP protocol sockets.

An accounting record will be written for each socket closed that matches this selection criterion. If no options are specified, an accounting record will be written for every socket as it is closed.

Only one option of a given field type can be specified, and the options can be combined to

reduce the sockets selected for accounting (i.e., specifying `"inet stream"` will produce an accounting record for each Internet stream socket closed, while just specifying `stream` will produce an accounting record for every stream socket closed no matter which address family the socket was created in).

`-a` Specifies the status of all accounting methods is to be displayed, not just a predefined subset.

NOTES

The positional and nonpositional formats of the command can be mixed such that the positional parameters appear last and in the order shown above, if the associated option (i.e. `-c`, `-n` or `-p`) is not specified on the command line.

The check and status functions do not require any special privileges.

You must be a privileged user, and authorized for group `adm`, in order to enable/disable accounting. The privileges required are documented in `acctctl`(2).

The commands `turnacct`(8) and `turndacct`(8) can be used to front end this command in order to enable and/or disable accounting.

You can use the `csaperm`(8) command to change all of the accounting files' group ID and permissions as necessary.

RETURN VALUES

For a `check` request, `csaswitch` returns 0 if the accounting method is currently `ON`. If the accounting method is currently `OFF` or its status could not be determined, a non-zero value is returned.

For all other requests, `csaswitch` returns 0 if the request was successfully performed; otherwise, a non-zero value is returned.

EXAMPLES

Example 1: The following command turns on kernel (process) accounting such that all data will be written to file `/usr/adm/acct/day/pacct`:

```
csaswitch on kernel /usr/adm/acct/day/pacct
```

Example 2: The following command checks the status of kernel (process) accounting:

```
csaswitch -c check -n kernel
```

Information similar to the following is displayed:

```
#       Accounting status for Wed  Wed Nov 16 13:36:47 1994
#       Name    State   Value
        kernel  On
```

Example 3: The following command turns on socket (network) accounting such that all data associated with sockets opened for communication over the Internet will be written to file `/usr/adm/acct/day/sockacct`:

```
csaswitch -o inet on socket /usr/adm/acct/day/sockacct
```

Example 4: The following command checks the status of socket (network) accounting:

```
csaswitch check socket
```

Information similar to the following is displayed:

```
#       Accounting status for Wed  Wed Nov 16 13:36:47 1994
#       Name    State   Value
        socket  On      inet
```

Example 5: The following command checks the status of all accounting:

```
csaswitch -a -c status
```

Information similar to the following is displayed:

```
#       Accounting status for Wed Nov 16 13:38:46 1994
#       Name    State   Value
        kernel  On
        nqs     On
        tape    On
        dm      Off
        socket  On      inet
        cray1   Off
        cray2   Off
        site1   Off
        site2   Off

        dio     On
        mpp     On
        mt      On
        perf    On
        sds     On
        mem     Off
        time    Off
        cray3   Off
        cray4   Off
        site3   Off
        site4   Off
```

SEE ALSO

csaperm(8), turnacct(8), turndacct(8), udbgen(8)

acctctl(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

NAME

csaverify – Checks accounting records for valid data

SYNOPSIS

/usr/lib/acct/csaverify [-n *nqsfile*] [-C *ctmpfile*] [-N *pnqsfile*] [-P *pacctfile*] [-T *tapefile*] [-s *nrec*] [-v] [-D]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The csaverify command checks the specified accounting file(s) for valid data. Messages are written to standard output when data is found that is not valid.

To delete pacct records, use csapacct(8). To delete Network Queuing System (NQS) records, use csaedit(8). You also can remove bad tape data by using csaedit(8).

If you omit file name options, all file types are verified using the defaults in the following list.

File name options are as follows:

-n *nqsfile* Specifies the file name of an NQS accounting file. This is the unprocessed version of *pnqsfile*. The default is /usr/adm/acct/work/Wnqacct.

-C *ctmpfile* Specifies the file name of a connect-time preprocessed accounting file (output from csaline(8)). The default is /usr/adm/acct/work/Pctime.

-N *pnqsfile* Specifies the file name of a preprocessed NQS accounting file (output from csanqs(8)). The default is /usr/adm/acct/work/Pnqacct.

-P *pacctfile* Specifies the file name of a pacct accounting file. The default is /usr/adm/acct/work/Wpacct.

-T *tapefile* Specifies the file name of a tape accounting file. The default is /usr/adm/acct/work/Wtpacct.

Performance options are as follows:

-s *nrec* Specifies that no more than *nrec* invalid records are reported for each file type. The default for *nrec* is 2.

-v Sets verbose mode. Output is written to standard output.

-D Sets debug mode. Output is written to standard output.

BUGS

If a partial accounting record is encountered, the partial record may or may not be flagged as not valid. However, all subsequent records will be noted as being not valid because the header words will be out of alignment.

EXAMPLES

The following example verifies the `Wpacct1 pacct` file with verbose mode turned on.

```
csaverify  -P  Wpacct1  -v
```

FILES

`/usr/include/acct/dacct.h`	Daemon accounting header file
`/usr/include/sys/acct.h`	Accounting records header file
`/usr/include/sys/accthdr.h`	Accounting records header definition file for the `/usr/include/sys/acct.h` file

SEE ALSO

`csaedit`(8), `csaline`(8), `csanqs`(8), `csapacct`(8), `csarun`(8), `csatape`(8)

`acct`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

cvtmldir – Converts between wildcard and multilevel directory (MLD) structures

SYNOPSIS

```
cvtmldir [-f] [-m] wildcard_path mldir_path
cvtmldir [-f] [-w] mldir_path wildcard_path
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The cvtmldir command converts from a wildcard directory to an MLD (-m option), or from MLD to a wildcard directory (-w option). It does this by linking (where possible), or copying files between directory trees. When a copy is required, the security attributes of the copied files are preserved.

When the -m option is specified, cvtmldir creates the multilevel symbolic link named in *mldir_path* and its associated directory. The command then copies the files recursively from *wildcard_path* to *mldir_path*, setting the process label for each file copied. If the full relative path to the file being created under the MLD does not exist, cvtmldir creates the directories in that path at the label of the file.

When the -w option is specified, cvtmldir creates a wildcard directory and copies the directory structure below each labeled subdirectory into the wildcard directory. When a name collision occurs between two files from two different labeled subdirectories, the second file name is made unique, and a warning is printed showing the old name and the new name.

When the -f option is specified, cvtmldir allows you to use the same directory for both *wildcard_path* and *mldir_path*. This allows the conversion of directories in place, and is useful for converting file system root directories.

When converting to a multilevel directory, cvtmldir fails if the directory that is the target of the multilevel symbolic link (*mldir_path*.mld) already exists. When converting to a wildcard directory, cvtmldir fails if the output directory (*wildcard_path*) already exists. The -f option causes cvtmldir to continue processing even if the actual output directory already exists. This is useful for converting from wildcard to MLD and back under the same path name (that is, *wildcard_path* is the same as the symbolic link target of *mldir_path*).

The actual directory that is used as the target of a multilevel symbolic link (*mldir_path*.mld) may exist when the -f flag is specified, but the link itself must not exist. The link is created as part of the conversion when the -m option is specified.

The cvtmldir command accepts the following options:

-f Allows the user to specify the same directory for both *wildcard_path* and *mldir_path*.

-m Converts from a wildcard directory to a multilevel directory.

-w Converts from a multilevel directory to a wildcard directory.

The following example shows the in-place conversion of `/tmp` from a wildcard directory to a MLD:

```
umount /dev/dsk/tmp
mv /tmp /tmp.mld
mount /dev/dsk/tmp /tmp.mld
cvtmldir -f -m /tmp.mld /tmp
spset -l syslow /tmp.mld
```

The following example shows the in-place conversion of `/tmp` from a MLD to a wildcard directory:

```
cd /tmp.mld
ls
cd /
rm /tmp
cvtmldir -f -w /tmp.mld /tmp.mld
umount /tmp.mld
mv /tmp.mld /tmp
mount /tmp
```

NOTES

Because of the way `cvtmldir` resolves path name collisions, it is possible for a different wildcard directory structure to exist after the translation from a wildcard to a MLD and back to a wildcard directory.

EXIT STATUS

The `cvtmldir` command exits with one of the following values:

Value	Description
0	Successful completion.
1	Incorrect command usage.
2	Unable to convert the multilevel directory to a wildcard directory.
3	File name collisions were detected and files were renamed in the process of converting the multilevel directory to a wildcard directory.

SEE ALSO

`mlmkdir`(8), `mlrmdir`(8)

`ln`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

General UNICOS System Administration, Cray Research publication SG–2301

NAME

ddms – Provides online disk maintenance capabilities for Cray PVP systems with an IOS model E

SYNOPSIS

/etc/ddms -a disk

/etc/ddms -a info *devicename*

/etc/ddms -a offload -c *cyl* -h *hed* -s *sec* [-e] [-v] *devicename*

/etc/ddms -a restore -c *cyl* -h *hed* [-s *sec range*] [-e] [-v] *devicename*

/etc/ddms -a flaw -c *cyl* -h *hed* -s *sec* [-d *defect address*] [-g *device information*] [-z] [-v] [-u] *devicename* | *devicename-spindle*

/etc/ddms -a unflaw -c *cyl* -h *hed* -s *sec* [-v] [-z] *devicename* | *devicename-spindle*

/etc/ddms -a write -c *cyl range* | -C [-F *flags*] [-h *hed range*] [-S] [-s *sec range*] [-E *errcount*] [-m *mode*] [-p *pattern*] [-b *seed*] [-l *loop*] [-v] *devicename* | [*devicename-spindle*]

/etc/ddms -a read [-c *cyl range* | -C] [-F *flags*] [-h *hed range*] [-S] [-s *sec range*] [-E *errcount*] [-b *seed*] [-p *pattern*] [-l *loop*] [-o] [-v] *devicename* | [*devicename-spindle*]

/etc/ddms -a surf -c *cyl range* | -C [-F *flags*] [-h *hed range*] [-S] [-s *sec range*] [-E *errcount*] [-m *mode*] [-p *pattern*] [-b *seed*] [-l *loop*] [-v] *devicename* | [*devicename-spindle*]

/etc/ddms -a rewrite -c *cyl* -h *hed* -s *sec* [-v] *devicename*

/etc/ddms -a readft [-c *cyl range*] [-f *filename*] [-h *hed range*] [-s *sec range*] [-t *flaw table*] [-u] *devicename* | [*devicename-spindle*]

/etc/ddms -a checkft *devicename*

/etc/ddms -a vid [-c *cyl range* | -C] [-h *hed range*] [-s *sec range*] [-v] *devicename* | [*devicename-spindle*]

/etc/ddms -a format -c *cyl range* | -C [-h *hed range*] [-s *sec range*] [-v] *devicename* | *devicename-spindle*

/etc/ddms -a spmap [-i|-o|-r] [-v] *devicename*

/etc/ddms -a reconstruct [-v] *devicename*

/etc/ddms -a spinup *devicename-spindle*

/etc/ddms -a spindown *devicename-spindle*

/etc/ddms -a scrub [-r] *devicename*

/etc/ddms -a label *devicename*

`/etc/ddms -a ldfrmt` *devicename-spindle*

`/etc/ddms -a backupft` *devicename*

`/etc/ddms -a makeuft -f` *serialnumber devicename*

`/etc/ddms -a restoreft -f` *filename devicename*

`/etc/ddms -a aft` [`-w`] [`-r`] *devicename*

`/etc/ddms -a wrecc -c` *cyl* `-h` *hed* `-s` *sec* [`-b` *bit*] [`-p` *pattern*] [`-l` *length*] [`-m` *mode*] *devicename*

IMPLEMENTATION

Cray PVP systems with an I/O subsystem model E

DESCRIPTION

The `ddms` (disk diagnostic and maintenance system) command provides disk maintenance capabilities by providing for surface testing, flaw maintenance, ID verification, and flaw table integrity capabilities. Errors detected during `ddms` execution are logged into a file named `ddmslog` in your current working directory. You can display errors by using either `olhpa` or `errpt` (for example, `/etc/diag/olhpa -d ddmslog`).

This command requires that you either have `diag` privilege set in the `permbits` field of the user database (UDB), or you are running as super user. In addition, you need `guard` privilege if you plan to run in unguarded mode and write outside the bounds of offloaded sectors (super-user privilege also allows you to run in unguarded mode). See `udbgen`(8).

`-a` Specifies the action to be performed. Valid actions are as follows:

`disk`	Displays valid disk devices and their type as defined in the `/dev/ddd` directory.
`info`	Displays information about the selected disk drives.
`offload`	Assigns a data sector to a spare map sector and moves the data to that sector.
`restore`	Restores a single sector or range of off-loaded sectors.
`flaw`	Updates all flaw tables, reformats the target track/sector, and reassigns unhideable and unslippable sectors to the spare map. Data is preserved for owner access.
`unflaw`	Removes a flaw added with the `flaw` command.
`write`	Writes data patterns and/or random data to the specified `-c`, `-h`, and `-s` range. The `write` action is destructive of data and must be used with caution.
`read`	Allows for the reading of all sectors in the `-c`, `-h`, and `-s` range specified.
`surf`	Writes data patterns to disk, followed by a read-data-and-compare routine. The `surf` action is destructive of data and must be used with caution.

`rewrite`	Rewrites a suspect sector in an attempt to correct a weak write or nonmedia-type ECC error.
`readft`	Reads the flaw table specified by the `-t` option from the disk.
`checkft`	Tests the integrity of all flaw tables for range checks, ascending order, and expected versus actual entries.
`vid`	Verifies sector IDs against what is expected, according to the contents of the User Flaw table on disk.
`format`	Reformats the cylinder, head, and sector range specified, according to the contents of the User Flaw table.
`spmap`	When used with the `-i` option, initializes the spare map by writing the spare map header. The `spmap` action then assigns alternative blocks to all unhideable blocks found in the User Flaw table. When used with the `-o` option, `spmap` reorders the current spare map into sequentially ascending order. The data will be moved to the newly assigned sectors. When used with the `-r` option, `spmap` forces a read of the spare map to kernel memory.
`reconstruct`	Changes from 4-spindle mode to 5-spindle mode. This action is used only with disk arrays. This action should be issued only when a failing spindle in an array has been repaired or replaced. If a spindle failure occurs, use the `pddconf`(8) command to down the failing spindle.
`spinup`	Remotely spins up a spindle on an array that has been swapped out or otherwise spun down. This action must be performed on a spun down disabled spindle before other actions can be done to the spindle.
`spindown`	Remotely spins down a spindle. Typically, this action is used to spin down a disabled spindle from an array prior to removing it.
`scrub`	Writes a pattern across an entire array to set data parity across the array. This action is used only with disk arrays. After the write is complete, the spare map is initialized according to the unhideables from the User Flaw table. The `-r` option is available to read the entire array to check for proper data parity. The `scrub` option does not write to the hardware CE and flaw table cylinders. The `scrub` action is destructive of data and must be used with caution.
`label`	Initializes an array to 5-spindle mode without scrubbing or reconstructing the data. This action is used with disk arrays only. Typically, you should not have to use this action. Instead, you should use the `scrub` action to initialize the array, and use the `reconstruct` action to change an active drive from 4-spindle to 5-spindle mode. The `label` action is destructive of data and must be used with caution.
`ldfrmt`	Loads DCA-2 or DCA-3 device format specifications. This action may need to be done after swapping a spindle from an array.

`backupft` Allows the backing up of the User Flaw table on disk to a file. The file name is dependent on the disk serial number.

`makeuft` Creates a new User Flaw table on the target disk by reading and generating the flaw table information from the sector IDs. This action will overwrite the current User Flaw table.

`restoreft` Allows the restoring to disk of a previously backed-up User Flaw table. The file is identified by the `-f` option.

`aft` Allows you to update or create an `/etc/aft` file (ASCII flaw table file) from the current unhideables table. This file is typically used with the `bb`(8) command to skip bad block assignments. If no option is specified with this action, an ASCII flaw table list is generated and displayed to `stdout`.

`wrecc` Reads the data and ECC on the specified sector, toggles the specified bit(s), and writes the data and ECC. The sector is then read to verify that an ECC error is generated. If the `-p` option is used, the data pattern is written before the ECC is generated on the selected pattern type.

`-b` Supplies a seed value during read and write operations when the `-p rand` option is used. When used with the `wrecc` action, specifies which bit to toggle.

`-c` Specifies a single cylinder or a range of cylinders. The values must be in octal (for example, `-c200-400`, `-c 10`, `-call`).

`-C` Can be used instead of the `-c` option to specify only CE cylinders.

`-d` Used with the `flaw` action to specify the defect address for types of devices that require one.

`-e` Used with the `offload` action to avoid moving the data to the alternative sector.

`-E` Defines the number of errors allowed before `ddms` will abort a `read`, `write`, or `surf` action. The default setting is 500.

`-f` When used with the `-a restoreft` action, supplies the file name for the User Flaw table to be restored. When used with the `-a makeuft` action, supplies the drive serial number. When used with the `-a readft` action, indicates a user flaw file to be displayed, rather than the table on disk.

`-F` *flags* Specifies control flag settings for a specific execution instance. The control flags are defined in the `/usr/include/sys/eslice.h` include file. The default setting is 0137.

`-g` Used with the `flaw` action to specify the head and channel in error. DD-49 type drives require that the general status on an error report be supplied. DD-60 type drives require that a 3-character octal field be specified defining the head(s) containing the defect. This *Combined Head Mask* is supplied with the `olhpa` long report.

`-h` Specifies the desired head or range of heads in octal. (for example, `-h 0-7`, `-h5`, `-hall`).

`-i` Indicates the `spmap` action that initializes the UNICOS spare map.

`-l` Specifies a decimal pass count. When used with the `wrecc` action, specifies the number of bits to be toggled.

`-m` Required to write outside the bounds of off-loaded sectors. The valid options are `guard`, `unguard`, and `destroy`. To validate the `unguard` mode, either you must be running as root or have `guard` privilege set up in the `permbits` field of the user database (UDB). See `udbgen`(8) for more information on setting up permbits. The `destroy` mode works like `unguard` mode, except that it allows writing to the spare map and flaw table cylinders. This option should be used with extreme caution as it will allow customer data to be destroyed.

`-o` Indicates to the `read` action to read in an oscillating fashion. Indicates to the `spmap` action to reorder the spare map sectors into sequentially ascending sectors.

`-p` Specifies the data pattern for a `write`, `surf`, `read`, or `wrecc` action. If `-p` is not specified with the `write` and `surf` actions, `dflt` (all) patterns will be used. If the `-p` option is used with the `read` or `wrecc` action, the data read will be compared against that pattern. Valid options are `zeros`, `ones`, `hilo`, `peak`, `hole`, `bump`, `rand`, `addr`, and `dflt`.

`-r` When used with the `spmap` action, forces a read of the spare map to kernel memory. When used with the `scrub` action, specifies to read the device after the write scrub has completed. When used with `aft` action, reads and displays the contents of the `/etc/aft` file to `stdout`.

`-S` When used with the `read`, `write`, and `surf` actions, specifies to ignore the flaw tables and try to run to unhideable sectors.

`-s` Specifies the sector range desired. Specify the range in octal. Default: all sectors.

`-t` Specifies the flaw table to read in the `readft` action. Valid arguments are `user`, `factory`, `system`, and `unhide`. Default: user.

`-u` If specified with the `readft` action and the `-t user` option, `-u` displays only flaws added by CRI. If specified with the `flaw` action, `-u` indicates an unhideable flaw for defect addressing type devices (for example, a DD-60 drive).

`-v` When specified with most actions, the command output is verbose.

`-w` When used with the `aft` action, writes the `/etc/aft` file with the unhideable entries of the User Flaw table for the specified device.

`-z` Bypasses the request for confirmation on specific actions. By default, default, when a `surf`, `read`, `write`, `vid`, and `format` action are specified, you are required to enter `y` to continue. If the `-z` option is used, this prompt does not appear.

devicename

Specifies the device inode name typically set up in `/dev/ddd`. For disk array type devices (DCA-3), some actions require the specific spindle number to be supplied in addition to the device name. `ddms` checks the *devicename* option for a `-`*spindle*, where *spindle* is the specific spindle number for the request. For example, `0334.1-3` specifies to send the request to spindle 3. See `mknod`(8) for more information on setting up device nodes.

filename Specifies the file created by the `backupft` action.

NOTES

If this command is installed with the default privilege assignment list (PAL), you must have an active `secadm`, `sysadm`, or `diagadm` category to use this command.

EXAMPLES

Example 1: Displays disk information.

```
ddms -ainfo /dev/ddd/0130
```

Example 2a: Flaws out cyl 10 hed 6 sec 22 with channels a1 and a2 of a DD-49.

```
ddms -aflaw -c10 -h6 -s22 -g000140 /dev/ddd/0136
```

Example 2b: Flaws out cyl 10 hed 1 sec 22 as hideable with defect address 321 and head 5 in error for a DD-60.

```
ddms -aflaw -c10 -h1 -s22 -d321 -g040 0136.1
```

Example 2c: Flaws a disk array sector. Notice that the spindle indicator is a required entry.

```
ddms -aflaw -c10 -h1 -s22 -d321 -g040 0136.1-2
```

Example 3: Removes the previously added DD-49 flaw.

```
ddms -aunflaw -c10 -h6 -s22 /dev/ddd/0136
```

Example 4: Reads sequentially all cylinder heads and sectors.

```
ddms -aread 0134
```

Example 5: Reads in an oscillating fashion all cylinders and sectors on head 5.

```
ddms -aread -o -call -h5 -sall 0136
```

Example 6: Reads and compares data to the addressing pattern for all of cylinder 100.

```
ddms -aread -c100 -paddr 0136.4
```

Example 7: Writes all data patterns to cylinders 10 through 20, all heads, and all sectors.

```
ddms -awrite -c10-20 -hall -m unguard 0130
```

Example 8: Initially, writes an addressing pattern to cylinders 100 through 200, all heads, and all sectors. Then the command reads the cylinders, heads, and sectors, and compares data.

```
ddms -asurf -c100-200 -paddr -munguard /dev/ddd/0136
```

Example 9: Shows an alternative to typing in the ddms command and action. You can symbolically link ddms to all the action names preceded by 1 character of your choice, as follows:

```
ddms -aread /dev/ddd/0136
```

could be entered as:

```
aread /dev/ddd/0136
```

if you set up a symbolic link in the following manner:

```
ln -s ddms aread
```

The a character preceding the read command is required so that the read command or any other ddms command will not be mistaken for a UNICOS command or shell built-in command. The a character can be any character value.

SEE ALSO

bb(8) for information on bad block files
mknod(8) for information on setting up device nodes
pddconf(8) for information on downing a failing spindle
udbgen(8) for information on setting up permbits in the user database (UDB)

Online Maintenance Tools Guide for Cray PVP Systems, Cray Research publication SD–1012. (This document contains information private to Cray Research, Inc. It can be distributed to non-CRI personnel only with approval of the appropriate Cray manager.)

NAME

ddoffload – Provides offloading of a single spindle of a DD-4*x* disk drive for Cray PVP systems

SYNOPSIS

```
/etc/diag/ddoffload -s source_device -g good_spindle -F fn
/etc/diag/ddoffload -s source_device -b bad_spindle -g good_spindle [-p]
/etc/diag/ddoffload -s source_device -b bad_spindle -g good_spindle [-f fn] [-p]
/etc/diag/ddoffload -s source_device -b bad_spindle [-d destination_device] -g good_spindle
[-p]
/etc/diag/ddoffload -h
```

IMPLEMENTATION

Cray PVP systems (except CRAY J90 series and CRAY EL series)

DESCRIPTION

The ddoffload (disk-drive offload) command is designed to copy (offload) all data from a spindle that is going bad to a spare spindle through a series of cable swaps and flaw table manipulations. Use ddoffload only when the failing spindle is in a condition to allow data read operations. This command is valid only for DD-40, DD-41, DD-42, and DD-4R device types.

-s *source_device*
: Required. Defines the source device that contains the bad (failing) spindle.

-b *bad_spindle*
: Required except with -F *fn*. Defines the bad (failing) spindle on the source device.

-g *good_spindle*
: Required. Defines the good spindle location on the destination device. This location is used for the data transfer between the bad and spare spindles. Also defines the spindle path to use for the -F *fn* option to create a spare spindle flaws (*fn*) file.

-d *destination_device*
: Required if a two-cabinet offload is being performed. Defines the destination device that contains the good spindle. Only this location is used for the transfer.

-F *fn* Creates a spare flaws file with the specified name. This option only creates a spare flaws file; it does not start the offload process. Performing a ddoffload -s *source_device* -g *good_spindle* -F *fn* in advance saves two cable swaps when you are doing a full offload (one cabinet).

-f *fn* Specifies the existing spare flaws file. Use the specified spare flaws file already created with the -F option. When -f *fn* is not specified, the default file spareflaws is created and used.

-p Disables ddoffload from plocking into memory during the data transfer. By default, ddoffload plocks into memory.

`-h` Displays a help screen for `ddoffload`.

EXAMPLES

Example 1: Creates a file called `spare40`, which contains all unhideable flaws from the spare spindle. You can use the spare spindle later for a full offload. No data offload is performed. To save time when doing an actual offload for a bad spindle, it is recommended that you create a file ahead of time.

```
ddoffload -s 40-A1-27 -g 2 -F spare40
```

Example 2: Performs a data offload from device `40-A1-27`, spindle `3`, to the spare spindle, using the location of spindle `1`. The spare flaws file, `spare40`, which was created in the previous example, is used. Specifying `-p` causes `ddoffload` not to plock into memory during the data transfer and verification portion of the process (one-cabinet method).

```
ddoffload -s 40-A1-27 -b 3 -g 1 -p -f spare40
```

Example 3: Performs a data offload from device `40-A1-27`, spindle `1`, to device `40-A1-26`, spindle `1`. This command saves two cable swaps, because two DD-40 cabinets (channels) are used (two-cabinet method).

```
ddoffload -s 40-A1-27 -b 1 -d 40-A1-26 -g 1
```

Example 4: Performs a data offload from device `/dev/ddd/0334.3`, spindle `2`, to device `/dev/ddd/0335.0`, spindle `0`. This example is for an IOS model E system with two DD-41 drives (two-cabinet method).

```
ddoffload -s 0334.3 -b 2 -d 0335 -g 3
```

SEE ALSO

Disk Drive Offload User Guide (Version 2.0), publication CDM–1028–000. (This manual is Cray Research Proprietary; dissemination of this documentation 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

`ddstat` – Displays configuration information about disk type character and block special devices

SYNOPSIS

`/etc/ddstat` [`-d`] [`-l`] [`-m`] [`-r`] *special0 special1* ...

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

CRAY J90 series

CRAY EL series

DESCRIPTION

The `ddstat` command parses specified disk device inodes and formats information about each. Logical devices are divided into their individual components and presented in a disk-specific format that is similar to the output of the `file`(1) command.

The `ddstat` command accepts the following options and operands:

`-d` Displays extensive internal debugging information.

`-l` Formats and displays any intermediate `/dev/ldd` nodes.

`-m` Prints a modified display format. The modified display takes the form of the shell script commands that would, when executed, replicate the displayed device special nodes being displayed. The `-m` option forces `-l` display mode.

`-r` When used with the `-m` (modified display format) option, adds the shell script command line (`/bin/rm`) necessary to remove the device special node prior to recreating it with `/etc/mknod`.

special ... The path to the defined character or block device. Possible paths include the following:

```
/dev/ce/*
/dev/ddd/*
/dev/dsk/*
/dev/ift/*
/dev/mdd/*
/dev/pdd/*
/dev/sdd/*
/dev/ssdd/*
/dev/spare/*
```

The disk device man pages contain the device-specific configuration parameters as presented to the mknod(8) command. See dsk(4), mdd(4), pdd(4), rdd(4), sdd(4), and ssdd(4) for device specific parameter definitions.

EXAMPLES

Example 1: The following examples show the output of a ddstat command:

```
% ddstat /dev/dsk/tmp60_6
/dev/dsk/tmp60_6 b 34/84 /dev/ldd/tmp60_6
        /dev/pdd/scr1230.0 c 32/90 10 01230 107180 12512 00 0 0 0
        /dev/pdd/scr1232.1 c 32/91 10 01232 107180 12512 00 0 1 0
        /dev/pdd/scr0230.4 c 32/94 10 0230 107180 12512 00 0 4 0
        /dev/pdd/scr0232.5 c 32/95 10 0232 107180 12512 00 0 5 0
        /dev/pdd/scr0234.6 c 32/96 10 0234 107180 12512 00 0 6 0
        /dev/pdd/scr0236.7 c 32/97 10 0236 107180 12512 00 0 7 0
```

Example 2: In the following example, scr_1232.1 is a character device. It has a major device number of 32 and a minor device number of 91. It is of disk type 10 (DD-60) on *iopath* 01232 (cluster 1, IOP 2, channel 32). The start address is sector 107180, the length is 12512 sectors, the flags are 00, no *altpath* is defined, and the disk is on unit 1 of the disk channel in the IOP.

```
% ddstat /dev/pdd/scr_1232.1
        /dev/pdd/scr1232.1 c 32/91 10 01232 107180 12512 00 0 1 0
```

SEE ALSO

file(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

dsk(4), mdd(4), pdd(4), rdd(4), sdd(4), ssdd(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

`ded` – Runs a program in dedicated mode

SYNOPSIS

`/etc/ded` [`-v`] [`-t` *timelimit*] *command* [*arguments*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `ded` command runs a program in dedicated mode and ensures that only one user is running a dedicated program (with `ded`) at a time. If two or more users attempt to use `ded` at the same time, all but one will wait and a message to that effect is placed on standard error.

The *command* with *arguments* is executed with as many CPUs dedicated to its use as it requires. The *command* is limited to a wall-clock time of 10 seconds. (Sites that are licensed for UNICOS source can change this value by changing the line `#define TIMELIMIT 10` in the file `ded.c`.)

The `ded` command accepts the following options:

`-v`	Sets verbose mode. This option causes `ded` to display extra messages about its actions.
`-t` *timelimit*	Lowers the time limit allowed the *command* by *timelimit* seconds. This function is useful in ensuring that very short commands do not waste time.

NOTES

If this command is installed with a privilege assignment list (PAL), a user who is assigned the following privilege text upon execution of this command is allowed to perform the action shown:

Privilege Text	**Action**
`exec`	Allowed to use this command.

If this command is installed with a PAL, a user with one of the following active categories is allowed to perform the actions shown:

Active Category	**Action**
`system`, `secadm`	Allowed to use this command. Allowed to specify any command file.
`sysadm`	Allowed to use this command. Allowed to specify any command file, subject to security label restrictions on the file's path. Shell redirected I/O is subject to security label restrictions.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command and is allowed to specify any command file.

The default generation procedures do not set the setuid bit for the ded command, which is required for it to work. Thus, sites cannot install it unknowingly. A site that wants to enable this command must execute the following command lines. First, to change the owner of the /etc/ded file to root:

```
chown root /etc/ded
```

Second, to change the permissions mode of the /etc/ded file to make it setuid:

```
chmod 4755 /etc/ded
```

The ded command places the *command* in a separate job before executing it. If job accounting is desired, *command* should be a script that enables job accounting and prints job accounting information before exiting.

The ded command depends on the real-time interface defined by /dev/cpu. Programs run in dedicated mode using ded may not be truly dedicated if other real-time processes exist in the system.

A log of invocations of ded is maintained in /etc/ded_log.

SEE ALSO

privtext(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

cpu(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

devacct – Controls device and performance accounting

SYNOPSIS

/usr/lib/acct/devacct [-v] -b *type*

/usr/lib/acct/devacct [-v] -l *type filesystem*

/usr/lib/acct/devacct [-v] -t *type*

/usr/lib/acct/devacct [-v] -L *filesystem*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The devacct command lets system administrators control device and performance accounting.

The devacct command accepts the following options and arguments:

-v	Enables verbose mode.
-b *type*	Begins accounting for the specified *type*.
-l *type filesystem*	Labels *filesystem* with the specified *type*. Only block special devices must be labeled.
-t *type*	Terminates accounting for the specified *type*.
-L *filesystem*	Lists the type currently associated with the specified *filesystem*.

The *type* argument is the device name associated with one of the BLOCK_DEVICE*x* or CHAR_DEVICE*x* variables or the string associated with PERF_NAME0 in the /etc/config/acct_config file. If the *type* contains shell separators, you must enclose the argument in double quotation marks. The *filesystem* argument is the name of a block special device.

BUGS

Only native Cray Research file systems are supported for device accounting.

EXAMPLES

Example 1: The following example labels a file system with type dd40 with ldcache, and then enables accounting for this type:

```
/usr/lib/acct/devacct -l "dd40with ldcache" /dev/dsk/mydd40
/usr/lib/acct/devacct -b "dd40with ldcache"
```

Example 2: The following example disables performance accounting:

```
/usr/lib/acct/devacct -t perf_01
```

Example 3: The following example enables accounting for the Hardware Performance Monitor (HPM):

```
/usr/lib/acct/devacct -b hpm
```

FILES

`/etc/config/acct_config` Accounting configuration file

SEE ALSO

`acctcom`(1), `ja`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`devacct`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

devnm – Prints device name

SYNOPSIS

/etc/devnm *path*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The devnm command identifies the special file associated with the mounted file system in which the operand *path* resides. As a special case, when *path* is "/" (the root file system), both the block device name and the swap device name are printed for the / argument name if swapping is performed on the same disk section as the root file system. Argument names must be full path names.

The devnm command is most commonly used by /etc/rc (see brc(8)) to construct a mount table entry for the root device.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Allowed to specify any path.
sysadm	Allowed to specify any path, subject to security label restrictions. Shell redirected I/O is subject to security label restrictions.

If the PRIV_SU configuration option is enabled, the super user is allowed to specify any file system.

EXAMPLES

If /usr is the mount point for /dev/dsk/usr_sd, the following command line:

```
/etc/devnm /usr
```

produces the following:

```
/dev/dsk/usr_sd /usr
```

FILES

/dev/dsk/*

SEE ALSO

brc(8)

NAME

dgdemon – Invokes the diagnostic daemon

SYNOPSIS

/etc/dgdemon

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The diagnostic daemon, dgdemon, is used by the System Maintenance and Remote Testing Environment (SMARTE) to execute all online diagnostic tests and concurrent maintenance utilities from SMARTE. It sends all diagnostic output to the maintenance workstation (MWS) or operator workstation (OWS). dgdemon also starts the SMARTE threshold server, thserver, to get real-time notification of error records from the operating system through a socket connection to the error-logging daemon, errdemon(8).

The server collects the records received from errdemon and thresholds errors. Exceeded thresholds are communicated to SMARTE on the MWS or OWS.

Typically, dgdemon is started by the rc script at boot time. dgdemon can be terminated by dgstop(8). Only a super user or a privileged user can start dgdemon, and only one diagnostic daemon can be active at any time on an MWS or OWS port. If errdemon is stopped, the server will reconnect to errdemon when errdemon is restarted.

The dgdemon daemon invokes the centralized identification and authorization library routines to validate the user ID and password.

NOTES

If this command is installed with the default privilege assignment list (PAL), you must have an active secadm, sysadm, or diagadm category to use this command.

MESSAGES

dgdemon: Event processing failed
: dgdemon encountered a bad file descriptor while performing a select(2) system call. dgdemon tried to remove the bad file descriptor. If the file descriptor continues to cause an error, dgdemon will perform an exit(2). To restart dgdemon, execute dgdemon as root. If the problem persists, contact your system support staff.

dgdemon: Cannot get host name
: The gethostname(2) system call failed. dgdemon performed an exit. Contact your system support staff.

`dgdemon: Diagnostic daemon already started on` *host_name*
A copy of `dgdemon` was already running on *host_name*. `dgdemon` performed an `exit`.

`dgdemon: Cannot export TCP/IP connection`
`dgdemon` was unable to export its TCP/IP connection. `dgdemon` performed an `exit`. Ensure that SMARTE is executing on the MWS or OWS. If SMARTE is currently executing on the MWS or OWS, execute `dgstop`, and ensure that `dgdemon` is no longer executing. Then execute `dgdemon` as `root`. If the problem persists, contact your system support staff.

`dgdemon: Cannot connect to the Log Manager`
`dgdemon` was unable to connect to SMARTE on the MWS or OWS. `dgdemon` did not perform an `exit`. When SMARTE is started on the MWS or OWS, `dgdemon` automatically connects to it. This problem does not cause `dgdemon` to exit. Contact your system support staff to start SMARTE on the MWS or OWS.

SEE ALSO

`dgstop`(8) for information on terminating the diagnostic daemon
`errdemon`(8) for information on invoking the error-logging daemon

`ia_failure`(3C), `ia_mlsuser`(3C), `ia_success`(3C), `ia_user`(3C) for information on centralized identification and authorization in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`err`(4) for information on the error-logging interface in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

System Maintenance and Remote Testing Environment (SMARTE) Guide, Cray Research publication SD–1017. (This document contains information private to Cray Research, Inc. It can be distributed to non-CRI personnel only with approval of the appropriate Cray manager.)

NAME

dgstop – Terminates the diagnostic daemon

SYNOPSIS

/etc/dgstop

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The dgstop command sends a software termination signal (SIGTERM) to the diagnostic daemon, dgdemon(8). Only a super user can terminate the diagnostic daemon.

Do not terminate dgdemon independently from errdemon(8). To terminate dgdemon, terminate errdemon by using errstop(8).

SEE ALSO

dgdemon(8) for information on invoking the diagnostic daemon
errdemon(8) for information on invoking the error-logging daemon
errstop(8) for information on terminating the error-logging daemon

NAME

diagccmt – Online I/O diagnostic client

SYNOPSIS

All GigaRing based Cray Research systems:

/etc/diag/diagccmt [-h] [-t]

Cray system workstation (SWS):

/opt/CYRIccmt/bin/diagccmt [-h] [-t]

IMPLEMENTATION

All GigaRing based Cray Research systems
Cray system workstation (SWS)

DESCRIPTION

The diagccmt program is the diagnostic client for the online I/O exercisers.

When you connect to diagccmt through the xdi(8) graphical interface, a list of I/O exercisers is displayed. Choosing one of these exercisers then allows you to change desired parameters and execute the exerciser. The exerciser executes on the machine on which diagccmt is executing.

The diagccmt command accepts the following options in any order:

-h Specifies that the diagccmt command-line usage information is to be printed to standard output. Once this information is displayed, diagccmt exits.

-t Specifies that a trace file is to be created containing a variety of information reflecting the flow of execution.

NOTES

You must be in the craydiag group to execute this command.

The path to this command and the appropriate environment variable settings are provided by the craydiag and crayadm modules.

ENVIRONMENT VARIABLES

PATH When searching for available exercisers, the PATH environment variable is scanned.

FILES

`/tmp/diagccmt.stdout`	Contains the standard output text from the `diagccmt` daemon
`/tmp/diagccmt.stderr`	Contains the standard error text from the `diagccmt` daemon
`/tmp/diagccmt`	Directory created by `diagccmt` that becomes its working directory
`/tmp/diagccmt/tracefile.*`	Contains tracing information

SEE ALSO

Concurrent Maintenance Tools User's Guide, publication SD–2627
SWS-ION Administration and Operations Guide, publication SG–2204

`ddms`(8)
`vht`(8)
`vst`(8)
`vtt`(8)
`xdms`(8)
`xdi`(8)

NAME

dig – Sends domain name query packets to name servers

SYNOPSIS

/usr/ucb/dig [*@server_name*] [*domain_name*] [*dns_type*] [*dns_class*] [*%ignored-comment*] [*-dig_option*] [*+query_option*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The domain name system (DNS), currently used by a majority of the Internet's component networks, is a large and complex autonomously administrated distributed database that provides a network wide name service. Its size and the autonomously administered nature of the system make it an ideal breeding ground for misconfiguration and other problems.

The dig (domain information groper) command is a flexible command-line tool, which can be used to gather information from the DNS servers. dig has two modes: simple command-line mode, which makes a single query; and batch mode, which produces a query for a list of several queries. All query options are accessible from the command line.

The usual simple use of dig takes the form:

dig *@server_name domain_name dns_type dns_class*

The elements on the command line have the following meanings:

server_name Specifies the server name. May be either a domain name or a dot-notation Internet address. If you omit this optional field, dig attempts to use the default name server for your machine.

Note: If a domain name is named(8), this is resolved by using the domain name system resolver. If your system does not use the DNS system for host name lookup, you may have to specify a dot-notation address. Alternatively, if a server is at your disposal, all that is required is that the /etc/resolv.conf file be present and that it indicate where the default name servers reside, so that *server_name* itself can be resolved. See the resolv.conf(5) man page for information on the /etc/resolv.conf file.

Warning: Changing /etc/resolv.conf affects the standard resolver library and potentially, several programs that use it. As an option, you can set the LOCALRES environment variable to specify a file that is to be used instead of /etc/resolv.conf (LOCALRES is specific to the dig resolver and is not referenced by the standard resolver). If the LOCALRES variable is not set, or the file is not readable, /etc/resolv.conf is used.

domain_name Specifies the domain for which you are requesting information.

dns_type Specifies the type of information (DNS query type) that you are requesting. If you omit *dns_type*, the default is `a`. The following list shows some values for *dns_type*; see RFC 1035 for a complete list.

Value	Description
`a`	T_A (network address)
`any`	T_ANY (all/any information about specified domain)
`mx`	T_MX (mail exchanger for the domain)
`ns`	T_NS (name servers)
`soa`	T_SOA (zone of authority record)
`hinfo`	T_HINFO (host information)
`txt`	T_TXT (arbitrary number of strings)

dns_class Specifies the network class requested in the query. If you omit *dns_clas*, the default is `in`. The following list shows some values for *dns_class*; see RFC 1035 for a complete list.

Value	Description
`in`	C_IN (Internet class domain)
`any`	C_ANY (All/any class information)

Note: The `any` value can specify a class and/or a type of query. `dig` parses the first occurrence of `any` to mean *dns_type* = T_ANY. To specify *dns_class* = C_ANY, you must either specify `any` twice, or set the *dns_class* value by using `-c` option of the `-`*dig_option* value.

%ignored-comment

Includes an argument that is not parsed. This may be useful if you are running `dig` in batch mode. Instead of resolving every *@server_name* in a list of queries, you can avoid the overhead of doing so, and still have the domain name on the command line as a reference. Example:

```
dig @128.9.0.32 %venera.isi.edu mx isi.edu
```

-dig_option Specifies an option that affects the operation of `dig`. The following options are currently available:

`-f` *XXX* Specifies the batch mode file. File *XXX* contains a list of query specifications (`dig` command lines) that will be executed in sequence. Lines that begin with '`;`', '#', or '`\n`' are ignored. Other options can still appear on the command line and are in effect for each batch query.

`-T` *XXX* Specifies time (in seconds) between start of successive queries in batch mode. Can be used to synchronize two or more batch `dig` commands. Default is 0.

`-p` *XXX* Specifies port number. Queries a name server that is listening to a nonstandard port number.

`-P`[*ping_string*] Makes a call to the shell to execute a `ping`(8) command for response time comparison after the query returns. The following `ping`(8) command prints the last three lines of statistics:

```
ping -s myserver 56 3
```

If the optional *ping_string* is present, it replaces `ping -s` in the shell command. You must specify a server name on the command line.

`-t` *XXX* Specifies type of query. Can specify either an integer value to be included in the *dns_type* field or use an abbreviated mnemonic value as listed in the *dns_type* description.

`-c` *XXX* Specifies class of query. Can specify either an integer value to be included in the *dns_class* field or use the abbreviated mnemonic value as listed in the *dns_class* description.

`-envsav` Specifies default environment file. This flag specifies that the environment after all of the arguments are parsed should be saved to a file to become the default environment. Useful if you do not like the standard set of defaults and do not desire to include such a large number of options each time `dig` is used. The environment consists of resolver state variable flags, timeout, retries and the flags that detail `dig` output (see +*query_option*). If you set the `LOCALDEF` environment variable to the name of a file, this is the location to which the default environment is saved. If not, the file `DiG.env` is created in the current working directory.

Note: `LOCALDEF` is specific to the `dig` resolver, and it does not affect operation of the standard resolver library.

Each time `dig` is executed, a check is made for `LOCALDEF`. If it is defined and the file is readable, the environment is restored before any arguments are parsed.

`-[no]stick` Specifies restoration of the default environment. This flag is useful only in batch queries. It indicates whether to restore the default environment (see `-envsav`) before parsing and sending each query. The default environment includes the standard defaults, those from `LOCALDEF`, or those set by `-envset`. The default is `-nostick`.

`-envset` Sets default environment. You can use this flag, in conjunction with `-stick`, for batch query runs. When `-envset` is specified, the environment after the arguments are parsed becomes the default environment for the duration of the batch file.

+query_option Specifies change to query packet or `dig` output specifics. `+` is used to specify an option to be changed in the query packet or to change `dig` output specifics. Many of these are the same options accepted by `nslookup`(1). If an option requires an argument, the form is as follows:

+keyword[*=value*]

You can abbreviate most keywords. Parsing of the `+` options is very simplistic. A value must not be separated from its keyword by white space. The following keywords currently are available. In column 1, the keyword in parentheses is an abbreviation that you may use (for example, (`deb`)). In column 2, the value in brackets is the default value for that keyword (for example, [`4`]).

Valid Keywords	**Meaning**
[`no`]`debug` (`deb`)	Turns on/off debugging mode [`deb`]
[`no`]`d2`	Turns on/off extra debugging mode [`nod2`]
[`no`]`recurse` (`rec`)	Uses/does not use recursive lookup [`rec`]
`retry=`*retries* (`ret`)	Sets number of retries to *retries* [`4`]
`time=`*sec* (`ti`)	Sets time-out length to *sec* seconds [`4`]
[`no`]`ko`	Keeps/does not keep open option (implies `vc`) [`noko`]
[`no`]`vc`	Uses/does not use virtual circuit [`novc`]
[`no`]`defname` (`def`)	Uses/does not use default domain name [`def`]
[`no`]`search` (`sea`)	Uses/does not use domain search list [`sea`]
`domain=`*name* (`do`)	Sets default domain name to *name*
[`no`]`ignore` (`i`)	Ignores/does not ignore truncation errors [`noi`]
[`no`]`primary` (`pr`)	Uses/does not use primary server [`nopr`]
[`no`]`aaonly` (`aa`)	Uses/does not use authoritative query only flag [`noaa`]
[`no`]`sort` (`sor`)	Sorts/does not sort resource records [`nosor`]
[`no`]`cmd`	Echoes/does not echo parsed arguments [`cmd`]
[`no`]`stats` (`st`)	Prints/does not print query statistics (for example, RTT) [`st`]
[`no`]`qr`	Prints/does not print outgoing query [`noqr`]
[`no`]`reply` (`rep`)	Prints/does not print reply [`rep`]

[no]header (he)	Prints/does not print certain parts of header [he]
[no]Header (H)	Prints/does not print all/any of header [H]
[no]ttlid (tt)	Prints/does not print TTLs and pkt IDs [tt]
[no]ques (qu)	Prints/does not print question section [qu]
[no]answer (an)	Prints/does not print answer section [an]
[no]author (au)	Prints/does not print authoritative section [au]
[no]addit (ad)	Prints/does not print additional section [ad]
pfdef	Sets print flags to default
pfmia	Sets print flags to minimal default
pfset=*value*	Sets print flags to *value*
pfand=*value*	Bitwise and print flags with *value*
pfor=*value*	Bitwise or print flags with *value*

The following pseudo-code example summarizes the retransmission strategy (retry/time) used by the resolver library (see resolver(3C)) when sending datagram queries.

Note: When you use the dig command, the number of servers and/or addresses is always 1.

```
for i = 0 to retry-1
      for j = 1 to num_servers
            send_query
            wait((time * (2**i)) / num_servers)
      rof
rof
```

The keywords pfset, pfand, and pfor make manipulation of the various print options less tedious. The currently defined values are as follows:

PRF_STATS	0x0001	RTT, query host, server host information
PRF_CMD	0x0008	dig command-line echo
PRF_QUES	0x0010	Questions section
PRF_ANS	0x0020	Answers section
PRF_AUTH	0x0040	Authoritative section
PRF_ADD	0x0080	Additional records section
PRF_HEAD2	0x0200	Header flags, section RR counts
PRF_TTLID	0x0400	ttl and packet ID number
PRF_HEADX	0x0700	Any/all packet header information
PRF_QUERY	0x1000	Outgoing query packet information
PRF_REPLY	0x2000	Reply packet information
PRF_SORT	0x8000	Sort various response sections
PRF_DEF	0x27f9	Default dig settings

PRF_MIN	0xa133	Minimalistic dig settings for (future) automated server testing

If you want to see information other than statistics when you are setting the print options, you should examine the outgoing packet (0x1000), incoming packet (0x2000), or both packets, plus the specific sections of the packet in which you are interested.

NOTES

The dig command uses a slightly modified version of the bind resolver(3C) library to gather count and time statistics. Otherwise, it is a straight-forward effort of parsing arguments and setting appropriate parameters. dig uses resolver routines res_init(), res_mkquery(), and res_send(), and it accesses the _res structure. You can compile dig with the standard resolver library, but this will change the output format, make the print options meaningless, and not gather RTT and packet count statistics.

BUGS

When you are running dig in batch mode, if an error occurs in one of the resolver routines, dig exits. The preferred behavior is to suspend only that particular query and to continue with the query list. The fix involves modifying the resolver routines to return a status that indicates and error, rather than simply exiting.

The ping option simply makes a call to the shell. This should be replaced with internal ping code.

FILES

/etc/resolv.conf	Initial domain name and name server addresses
LOCALRES	Environment variable that specifies the file to use in place of /etc/resolv.conf
LOCALDEF	Environment variable that specifies the default environment file

SEE ALSO

named(8), ping(8)

host(1B), nslookup(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

resolver(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

resolv.conf(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

RFC 1035, Domain Names–Implementation and Specification, Mockapetris, November 1987

NAME

diskusg – Generates disk accounting data by user, account, or group ID

SYNOPSIS

/usr/lib/acct/diskusg [-A] [-D] [-G] [-U|-a] [-z] [-i *ignorelist*] [-d *debuglevel*] [[-E] | [-F *s1*:*s2*:...*sn*] | [-T *t1*:*t2*:...*tn* [-t *timetype*]]] [-H] [-h] [-u *outputfile*] [-v] [-S] [-w *string*] [-x] [-y] *filesystems*

/usr/lib/acct/diskusg [-A] [-D] [-G] [-U|-a] [-i *ignorelist*] -s [-d *debuglevel*] [[-E] | [-F *s1*:*s2*:...*sn*] | [-T *t1*:*t2*:...*tn* [-t *timetype*]]] [-H] [-h] [-u *outputfile*] [*files*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The diskusg command generates disk accounting data that can used by the UNIX System V (standard) accounting system or by Cray system accounting (CSA). Disk accounting data is obtained either by scanning the inode region of the logical disks or by merging diskusg output records.

diskusg output, which is written to standard output (stdout), is usually the input to acctdisk(8). The acctdisk command converts diskusg -D output to tacct format, and converts -A and -U (or -a) output to cacct format; acctdisk does not convert -G output nor the output of options -E, -F, and -T. Standard accounting uses tacct format, while CSA uses cacct format. The tacct and cacct records can be merged for reporting with other accounting records of the same format. diskusg normally is run in the dodisk(8) procedure.

diskusg accepts four types of options:

- Consolidation options
- Data input options
- Data output options
- Site-reserved options

In the descriptions of these options, account ID (*acid*), group ID (*gid*), and user ID (*uid*) are numeric values, and account name (*account*), group name (*group*), and login name (*login*) are alphanumeric strings.

Consolidation Options

The following diskusg options allow you to consolidate information in different ways:

-A Reports the following disk usage information per account, which can be converted by acctdisk(8) to cacct format:

acid account #blocks #mblocks

For each account, the following information is provided:

acid Account ID.

account Account name.

#blocks Total number of disk blocks allocated to this account.

#mblocks Total number of migrated disk blocks allocated to this account. (This value will always be 0 if the Data Migration Facility (DMF) is not enabled in this file system.)

`-D` (Default) Prints the following on the standard output, one line per user, which can be converted by `acctdisk`(8) to `tacct` format:

uid login #blocks

For each user, the following information is provided:

uid User ID.

login Login name.

#blocks Total number of disk blocks allocated to this user.

`-G` Reports the following `diskusg` information per group, which cannot be processed by `acctdisk`(8):

gid group #blocks #mblocks

For each group, the following information is provided:

gid Group ID.

group Group name.

#blocks Total number of disk blocks allocated to this group.

#mblocks Total number of migrated disk blocks allocated to this group. (This value will always be 0 if DMF is not enabled in this file system.)

`-U` or `-a` Reports the following disk usage information per user ID, which can be converted by `acctdisk`(8) to `cacct` format:

uid login acid account #blocks #mblocks

The `-U` designation for this option supercedes `-a` and is the preferred usage. The information provided about the user/account pair is as follows:

uid User ID.

login Login name.

acid Account ID.

account Account name.

#blocks Total number of disk blocks allocated to this user/account pair.

#mblocks Total number of migrated disk blocks allocated to this user/account pair. (This value will always be 0 if DMF is not enabled in this file system.)

`-z` Allows the files with an account ID of 0 to be accumulated separately from the primary account for the user. By default, files with an account ID of 0 are accumulated to the primary account of the user. If a user does not have a primary account ID, then the account ID of 0 is used and always reported.

Data Input Options

The following options allow you to alter the input to `diskusg`:

`-i` *ignorelist*

Ignores file systems by these names. The names are those written to the super block by the `labelit`(8) command. The file system names in *ignorelist* must be separated by a comma or a blank space. If the list contains blank spaces, the list must be enclosed by double quotes.

`-s` Specifies that the input data is already in `diskusg` output-record format. The `-S` and `-s` options are mutually exclusive. Input is either from *files* or the standard input. The `diskusg` command combines all lines for a single user into a single line. If the input files were created by using any of the `-A`, `-D`, `-G`, or `-U|-a` options, that option must also be specified. The input files made with the `-h` or `-H` option have the character # placed at the start of the header lines. The # character as the first character of a line will cause the line to be treated as a comment and the line will be ignored.

Data Output Options

The following options allow you to alter the output produced by `diskusg`: The `-E`, `-F`, and `-T` options are all mutually exclusive, and these report formats cannot be processed by `acctdisk`(8).

`-d` *debuglevel*

Turns on debugging. Debugging levels 0 through 5 are available. Level 0 is the default and generates no debug messages, level 5 is the highest level and generates all available debugging messages. The debugging messages are sent to standard error (`stderr`).

`-E` Reports disk usage in an extended format for the report options (`-A`, `-D`, `-G`, or `-U|-a`). The `-E`, `-F`, and `-T` options are all mutually exclusive. For each record ID (account, group, or user ID), the following additional information is provided:

Report Item	Description
`Total Files`	Total number of inodes.
`Directory Files`	Total number of directory inodes.
`Online Files`	Total number of file inodes with online date.
`Offline Files`	Total number of file inodes with offline data.
`Total Blocks`	Total number of data blocks used.
`Online Blocks`	Total number of data blocks residing online.
`Offline Blocks`	Total number of data blocks residing offline.

`ACL Blocks`	Total number of blocks used by access control lists (ACLs).
`PAL Blocks`	Total number of blocks used by privilege assignment lists (PALs).

`-F` *s1*`:`*s2*`:...`*sn*

Reports disk usage per file size range (*s1* and so on) for the report options (`-A`, `-D`, `-G`, or `-U|-a`). The `-E`, `-F`, and `-T` options are all mutually exclusive. The file size ranges are specified by the site using this option. See Example 3. You may specify up to ten file size ranges; the sizes must be given in data blocks and must be in ascending order. (See NOTES for more information about data block sizes.) For each record ID, the following additional information is provided:

Report Item	**Description**
`Total Blocks`	Total number of data blocks used.
s1 `Blocks`	Number of data blocks used in the range greater than or equal to 0 and less than or equal to *s1*.
s2 `Blocks`	Number of data blocks used in the range greater than *s1* and less than or equal to *s2*.
sn `Blocks`	Number of data blocks used in the range greater than *sn-1* and less than or equal to *sn*.
`Max Size Blocks`	Number of data blocks greater than *sn* used.

`-T` *t1*`:`*t2*`:...`*tn*

Reports disk usage per file age range (*t1* and so on) for the report options (`-A`, `-D`, `-G`, or `-U|-a`). The `-E`, `-F`, and `-T` options are all mutually exclusive. The file age ranges are specified by the site using this option. See Example 4. You can combine the `-t` option with `-T` to select a particular time field. You may specify up to ten ages; the ages must be given in hours and must be in ascending order. For each record ID, the following additional information is provided:

Report Item	**Description**
`Total Blocks`	Total number of data blocks used.
t1 `Blocks`	Number of data blocks used in the range greater than or equal to 0 and less than or equal to *t1*.
t2 `Blocks`	Number of data blocks used in the range greater than *t1* and less than or equal to *t2*.
tn `Blocks`	Number of data blocks used in the range greater than *tn-1* and less than or equal to *tn*.
`Max Time Blocks`	Number of data blocks greater than *tn* used.

`-t` *timetype*
Must be used with the `-T` option. Specifies for the `-T` option the time field from the inode to use in the data collection. Possible values for *timetype* are:

`access` Uses the last access time of the data. This value is the default.

`inode` Uses the update time of the inode.

`update` Uses the update time of the data.

`-H` Displays an extended header. The header includes the data report header from the `-h` option, a report description, machine information, a report date line, a report format line. It also shows totals resulting from inode searches, including totals for unknown (charged to no one) accounts, users, or groups, depending upon what options you specified, and an overall total line. All of these lines start with a # character, so that they will be ignored as comments if the report is used as input to `diskusg` with the `-s` option.

`-h` Displays a report header that describes the report columns.

`-u` *outputfile*
Writes terse records to *outputfile* of files that are charged to no one (unknown); they display what is known about the inodes searched. Records include the disk–partition name, the inode number (*inode_num*), the user ID number (*uid*), the inode partition (*ipart_num*), the inode region (*ireg_num*) and the inode offset (*ioff_num*).

`-v` Sets verbose mode. A list of files that are charged to no one is written to `stderr`. This list includes files that have invalid user ID/account ID combinations.

Site-reserved Options

The following options have been reserved for site-specific use:

`-S` Site-reserved option that allows you to create a site-specific report format. The `-S` and `-s` options are mutually exclusive.

`-w` *string* Site-reserved option that accepts a string as input. The meaning of this option and accepted string values are determined by the site.

`-x` Site-reserved option for a True or False value. The option is True if it is present. The meaning of this option is determined by the site.

`-y` Site-reserved option for a True or False value. The option is True if it is present. The meaning of this option is determined by the site.

NOTES

Sites can tailor the contents of the reports by modifying the `site.c` module and recompiling and re-installing `diskusg`. (See the Site-reserved Options subsection.) For more information on how to use the site-reserved reports in the `site.c` source file, see *UNICOS Resource Administration*, Cray Research publication SG–2302.

In reporting the number of data blocks, the program has no way of differentiating among varying block sizes. Various devices may have different block sizes. Reports are comparable only if the underlying device uses the same block size.

EXAMPLES

Example 1: The following example displays the disk usage of `/` and `/usr`:

```
diskusg -U /dev/dsk/usr /dev/dsk/root > /usr/adm/acct/day/dtmp
```

Example 2: The following example generates an extended format report of usage by user ID with column titles:

```
diskusg -UEh /dev/dsk/ptmp
```

Example 3: The following example generates a file size format report of usage by group ID with column titles and extended header:

```
diskusg -Gh -F 100:1000:10000 /dev/dsk/ptmp
```

```
# Group ID      Group Name      Total   0 <    100   <   1000   <  10000   <    Max
#                               Blocks       Blocks     Blocks     Blocks     Blocks
         0            root     275568        32180      64984      46472     131932
         2             bin      26336          892        868      24576          0
         3             sys        144          144          0          0          0
         9        operator        280           80        200          0          0
       168             dce         40           40          0          0          0
      1007         craysrc       3228         2532        696          0          0
      1013              os      21392         4580      14064       2748          0
      1015         network      33344        12228       9408      11708          0
      1090             mpp          8            8          0          0          0
     11121         tsttool      46020        43292       2728          0          0
     11824         testing         32           32          0          0          0
     11951           vsxg0          8            8          0          0          0
     12584         craydev         44           44          0          0          0
```

Example 4: The following example generates a file age format report of usage by account ID with ages of 7 days, 30 days, and 180 days with report headers:

```
diskusg -AH -T 168:720:4320 /dev/dsk/ptmp
```

```
#  Disk Usage report by diskusg - Version 1: (SN-1703 (cool.cray.com)) on
#     Mon May  2 11:03:48 1994
#
#  Scanning filesystems:
#
#  /dev/dsk/ptmp
#
#
#  Account ID Usage report by diskusg - Version 1: (SN-1703 (cool.cray.com)) on
#     Mon May  2 11:03:48 1994
#
#  File Age Report (-T) - Selected by Access time, with the sample parameters in Hours.
#
# Account ID  Account Name      Total   0 <   168 H  <   720 H  <  4320 H  <    Max
#                              Blocks       Blocks     Blocks     Blocks     Blocks
         0         A-00000         16           16          0          0          0
      8306           Admin        400          296         44         40         20
      8359         Country         84           60          0          4         20
      8366          Cust_f          4            4          0          0          0
      8367           Demos         48           36          0         12          0
      8373          SysAdm         16           16          0          0          0
      8374            Intl        304          260          0         44          0
      8383          Netdev     116184        97132      18908        112         32
      8388          Syssup      11608        11304        304          0          0
      8394         Userint      42640        24020      13976       1728       2916
      8395           Users      48712        48460         12        240          0
      8397           Xydev     187516       164408      22344        180        584
#
# -- Totals ---               407532       346012      55588       2360       3572
#
```

FILES

`/etc/udb`	User validation file containing user control limits; used for user ID to login name conversions.
`/etc/acid`	Account ID information file containing account names; used for account ID to name conversions.
`/etc/group`	Account ID information file containing group names; used for group ID for name conversions.

SEE ALSO

`acct`(8), `acctdisk`(8), `acctsh`(8), `dodisk`(8), `labelit`(8), `mount`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

dmap – Maps physical and logical devices

SYNOPSIS

```
/etc/dmap [-o] [-p name]
/etc/dmap [-o] [-l name]
/etc/dmap [-o] [-l minor_number]
```

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The dmap command produces a display providing information about the configuration of the disk subsystem. The type of display is determined by which of the following options and arguments you use with dmap:

`-o` Displays cylinder numbers in octal format. (Default is decimal.)

`-p` *name* Displays slices of all logical devices that reside on the named physical device. The display uses the following format:

```
minor  slice  cyls/block  blocks  total  name
-----  -----  ----------  ------  -----  ----
```

The format headings represent the following:

minor	Minor device number of logical device
slice	Slice index of logical device
cyls/block	Cylinder range on the physical device (block range for SSD and BMR)
blocks	Number of physical disk blocks
total	Cumulative block count
name	Name of logical device if established

`-l` *name*
or
`-l` *minor_number*

Displays all slices for the logical device specified by either its *name* (that is, /dev/dsk/a) or its *minor_number*. The display uses the following format:

```
slice  cyls/block  blocks  total  Physical device
-----  ----------  ------  -----  ---------------
```

The format headings represent the following:

slice	Slice index of logical device

`cyls/block`	Cylinder range on physical device (block range for SSD and BMR)
`blocks`	Number of physical disk blocks
`total`	Cumulative block count
`Physical device`	Name of physical device

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`	Allowed to use this command.
`sysadm`	Allowed to use this command. Shell redirected output is subject to security label restrictions.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

SEE ALSO

General UNICOS System Administration, Cray Research publication SG–2301

NAME

dnsquery – Queries domain name servers by using resolver library calls

SYNOPSIS

dnsquery [-n *nameserver*] [-t *type*] [-c *class*] [-r *retry*] [-p *retry period*] [-d] [-s] [-v] *host*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The dnsquery program provides an interface to nameservers by using Berkeley Internet Name Daemon (BIND) resolver library calls. The program supports queries to the name server with an operation code of QUERY. This program is intended to be a replacement or supplement to programs like nstest, nsquery, and nslookup. All arguments except for *host* and *nameserver* are case-sensitive.

The dnsquery program accepts the following options:

-n *nameserver* Names the name server to be used in the query. Name servers can appear as either Internet addresses of the form *w.x.y.z* or can appear as domain names. The default is specified in the /etc/resolv.conf file.

-t *type* Specifies the type of resource record of interest. *Types* can be one of the following values. The default is ANY.

A	Address
NS	Nameserver
CNAME	Canonical name
PTR	Domain name pointer
SOA	Start of authority
WKS	Well-known service
HINFO	Host information
MINFO	Mailbox information
MX	Mail exchange
RP	Responsible person
MG	Mail group member
AFSDB	DCE or AFS server
ANY	Wildcard

The *type* argument is case-insensitive.

`-c` *class* Specifies the class of resource records of interest. The *class* argument can be one of the following values. The default is `IN`.

`IN`	Internet
`HS`	Hesiod
`CHAOS`	Chaos
`ANY`	Wildcard

The *class* argument is case-insensitive.

`-r` *retry* Specifies the number of times to retry if the name server is not responding. The default is 4.

`-p` *retry period* Specifies the period to wait before timing out. The default is `RES_TIMEOUT`.

`-d` Turns on debugging. This sets the `RES_DEBUG` bit of the resolver's options field. The default is debugging turned off.

`-s` Uses a stream rather than a packet. This uses a TCP stream connection with the name server rather than a UDP datagram. This sets the `RES_USEVC` bit of the resolver's options field. The default is a UDP datagram.

`-v` Synonym for the `s` flag.

host Specifies the name of the host (or domain) of interest.

MESSAGES

If the resolver fails to answer the query and debugging has not been turned on, `dnsquery` will print a message like the following:

```
Query failed (rc = 1) : Unknown host
```

The value of the return code is supplied by `h_errno`.

BUGS

Queries of a class other than `IN` can have interesting results because ordinarily a name server has only a list of root nameservers for class `IN` resource records.

Query uses a call to `inet_addr()` to determine if the argument for the `-n` option is a valid Internet address. Unfortunately, `inet_addr()` seems to cause a segmentation fault with some (bad) addresses (for example, `1.2.3.4.5`).

FILES

`/etc/resolv.conf`	To get the default name server and search lists
`arpa/nameser.h`	List of usable RR types and classes
`resolv.h`	List of resolver flags

SEE ALSO

`named`(8)

`nslookup`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`resolver`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

NAME

dodisk – Performs disk accounting

SYNOPSIS

/usr/lib/acct/dodisk [-a] [-v] [*special_files*]

/usr/lib/acct/dodisk [-A] [-v] [*special_files*]

/usr/lib/acct/dodisk [-o [*mount_points*]]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

By default, the dodisk command performs disk accounting on the special files listed in the /etc/checklist file. It is typically run by using the cron(8) command. checklist should contain a list of special files, one file name per line.

When running Cray Research system accounting (CSA), either the -a or -A option must be specified with dodisk. Failure to do so can cause CSA to abort.

The dodisk command accepts the following options and operands:

-a Reports disk usage in the following format:

uid login account acid #blocks

The information provided is as follows:

uid Numerical user ID of the user

login Login name of the user

account Account name

acid Account ID

#blocks Total number of disk blocks allocated to this user

-A Reports disk usage in the following format:

acid account #blocks

The information provided is as follows:

acid Account ID

account Account name

#blocks Total number of disk blocks allocated to this user

`-o` Causes `dodisk` to invoke `acctdusg` rather than `diskusg` to account for disk usage by login directory. This option causes `dodisk` to perform more slowly.

mount_points
Specifies one or more file system names, rather than special devices, where disk accounting is performed. If *mount_points* are not specified, "/" is the default file system.

`-v` Sets verbose mode. `dodisk` prints verbose records on standard error of files that are charged to no one.

special_files
Specifies one or more special files where disk accounting is performed.

NOTES

If you want to convert the output produced by `dodisk -A` to `cacct` format, you must use `acctdisk`(8) with the `-A` option.

EXAMPLES

The following example is a possible entry for the `/usr/spool/cron/crontabs/root` file so that `cron`(8) automatically runs `dodisk`:

```
30  10  *  *  1-5  /usr/lib/acct/dodisk -a -v 2> /usr/adm/acct/nite/dsklog
```

SEE ALSO

`acct`(8), `acctdisk`(8), `acctsh`(8), `cron`(8), `csa`(8), `diskusg`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

dump – Invokes an incremental file system dump

SYNOPSIS

/etc/dump [-A *altfile*] [-a] [-B *buffer count*] [-c] [-d *density*] [-D *device*] [-e] [-f *file*] [-g *devgrp*] [-l *label_type*] [-m *capacity*] [-M *m_fmt*] [-n] [-P *p_fmt*] [-R *r_fmt*] [-s *size*] [-t *dump_level*] [-T *t_fmt*] [-u] [-v *vsn_list*] [-W] [-w] [-Z *max_list*] [-z] *filesystem*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The dump command copies to magnetic tape or disk all files changed after a certain date in a specified *filesystem*, which is a required operand. (The date is found in the /etc/dumpdates file.)

The dump command accepts the following options:

-A *altfile* Specifies the name of a file to contain a second copy of the output from the beginning of dump. The EXAMPLES section shows how this command is used. You can use this file as input to /etc/restore -v to obtain, without a tape mount, a list of the files contained in the full dump output.

-a Dumps files copied by the Data Migration Facility (DMF) as if they were already migrated offline, that is, inode information only, without data. Use this option to avoid an unnecessary duplication of data when the data migration facility is managing the *filesystem*. Files that are eligible for this special processing include both dual-state files and the premigration copies of files fully backed up by DMF.

-B *buffer count* Determines the number of 32,678-byte blocks that are output in a single listio(2) operation. Default is 1 for pipes, and 20 for other output files.

-c Causes dump to use cartridge tapes rather than round tapes. This is equivalent to -g CART.

-d *density* Obsolete and optional. Valid arguments are 1600 or 6250. The default interpretation is the density of the nine-track output device. Thus, if the -d 1600 option is present along with -g TAPE, the capacity of one output volume is set to 40 Mbytes. Other values are ignored.

-D *device* Obsolete and optional. The default interpretation is the name of a specifically requested output device.

-e Requests a message to stderr at the approximate location of every volume change. The following is a sample output line produced by executing dump with this option:

```
dump (/root to root_dmp):  volume 1 ends near i-node 227
```

The approximate location is an estimate determined by the `-m` parameter, or by default calculation. Using this option facilitates skipping tapes during a restore operation that needs only one file.

Running `restore` with the `-t` option provides an inode to file name mapping that can help determine on which tape the file's data is located, when correlated with the `-e` option. If the `-f` option is used with the `-e` option, you must specify `-m` *capacity* for the command to supply location information.

`-f` *file* Places the dump on *file*, instead of tape. If you specify *file* as `-`, `dump` writes to standard output.

`-g` *devgrp* Specifies an explicit assignment to the character string that is used as the device group to reserve for the output device. The default value is `TAPE`.

`-l` *label_type* The *label_type* specified on the `-l` option is used in the default for the `-M` option. The tape daemon supports the following *label-type* values:

`nl` Nonlabeled tapes

`sl` IBM standard labels

`al` ANSI labels (default)

You cannot use the `-l` option with the `-f` option (dump to a file).

`-m` *capacity* Explicitly states in megabytes the capacity of a single output volume.

`-n` Obsolete and optional. Causes `dump` to broadcast one of the following messages by using `/etc/wall -g oper`:

When the program is aborting: `dump program is falling to floor in pieces`

When the program completes successfully: `dump has completed`

`-s` *size* Obsolete and optional. The default value is 2400. The default interpretation is the number of feet in an output device when `-g TAPE` is set.

`-t` *dump_level* Causes an incremental dump. Valid dump levels are 0 through 9; the default value is 9. If you specify 0, the entire file system is dumped. A level number above 0, incremental backup, tells `dump` to copy all files new or modified since the last dump of a lower level.

`-u` If the dump completes successfully, writes the date and time of the beginning of the dump on the `/etc/dumpdates` file. Lines in the `/etc/dumpdates` file are limited to 4096 characters.

`-v` *vsn_list* Causes `dump` to use *vsn_list* as a list of volume serial numbers (VSNs) to be used for the dump. Each VSN is a set of 1 to 6 alphanumeric characters separated by colons (:). You cannot use the `-v` option with the `-f` option (dump to a file). If `-v` is not used, `dump` asks the operator to type in a VSN list.

Example:

```
dump -v root1:root2:root3 /dev/dsk/root_fs
```

`-W` Tells the operator the file systems that must be dumped. (This information is gathered from the `/etc/dumpdates` and `/etc/fstab` files.) This option causes `dump` to print out, for each file system in `/etc/dumpdates`, the most recent dump date and level, and to highlight those file systems that should be dumped. If this option is set, all other options are ignored, and `dump` exits without further processing.

`-w` Similar to the `-W` option, but it prints only the file systems that must be dumped.

`-z` This option is the same as specifying `-Z 4096`.

`-Z` *max_list* Skips all regular nonmigrated files larger than the specified size.

The following four options are the format specifiers:

`-M` *m_fmt* Builds a `tpmnt` request. The default value is as follows:

```
tpmnt -r in  -g +g -v +i -f +I -n -l +l +?D'-D' +D +?d'-d' +d -P +O
```

`-P` *p_fmt* An option that is used as the prompt if a VSN list is requested from the operator. The default value is `please enter +e vsn's`. See the EXAMPLES section for uses of this option.

`-R` *r_fmt* Reserves an output device. The default value is `rsv +g`.

`-T` *t_fmt* Appends information to the end of the `/etc/dumpdates` line. The default value is +40i.

The format specifiers make use of a string substitution capability, based loosely on the format string for the `date` command. The following substitutions are recognized:

`+D` From the `-D` option, optional, may be `NULL` pointer.

`+d` From the `-d` option, optional, may be `NULL` pointer.

`+e` The estimated number of volumes.

`+f` The base name of the file system dumped.

`+g` The string from the `-g` option. Typical values are `TAPE`, `CART`, and `SILO`.

`+i` From the `-v` option or the operator input string.

`+I` Short form of `+i`, terminated at first colon (`:`).

`+l` From the `-l` option, typical values are `nl`, `sl`, and `al`.

`+O` The temporary file name for output (generated by the `dump` command).

`+t` The dump type. A single character from the `-t` option.

A number between the + character and the field identifier specifies a maximum length. The following conditional substitutions also are recognized:

`+?D_`*AA*`_` If the `-D` option has a value, the character string delimited by the _ characters is copied.

`+?d_AA_` If the `-d` option has a value, the character string delimited by the _ characters is copied.

`+?+_+_` The conditional `+?+` always tests true. This implements an escape mechanism. The example places a `+` in the output.

Because the tapes are under the control of the tape daemon, the `dump` program does not request operator assistance to handle tape errors. Disk errors cause a message to be printed and the buffer to be emptied. Disk errors do not cause the program to stop.

The tape interface logic of the `dump` command includes a `setjob`(2) system call. This permits independent `/bin/rsv` invocations and helps avoid reaching job tape limits when running multiple copies of the dump program from the same login.

When a `dump` is performed on an active file system, all files that satisfy the following two criteria will be available for restoration from the back-up medium:

- The file must be complete at the time of the `sync`() call at the beginning of the dump program.
- Neither the file, nor any components in the file path, may change until dump has completed.

NOTES

To use other, specific tape mount options not covered by `dump`, you can perform your own `rsv`(1) and `tpmnt`(1) operations, as follows:

```
rsv
tpmnt -l al -v a1:a2 -M -p /tmp/tapedev
dump -t 9 -u -f /tmp/tapedev /dev/dsk/slash_a
rls -a
```

Alternately, you can use the format specifiers. The following `dump` command alone yields the same results as the preceding example:

```
dump -t 9 -u -v a1:a2 -R "rsv" -M "tpmnt -l al -v +i -M -p +O" /dev/dsk/slash_a
```

On a file system with active data migration, the migrated files are not backed up, only moved. The `dump` command does not recall migrated data, but it does make a copy of the on-disk inode that contains the migration recall key. The `-a`, `-Z`, and `-z` options permit customer-specified processing on a system that has data migration. The capability also exists to uniquely specify desired policies in the `duex.c` user-exit source file.

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`	Dump any file system.
`sysadm`	Dump any file system, subject to security label restrictions on the file system path and device, and the device labeling policy. Shell redirected output is subject to security label restrictions.

If the PRIV_SU configuration option is enabled, the super user is allowed to dump any file system.

CAUTIONS

The dump and restore(8) commands share an internal inode number, which is used to pass information between levels of an incremental restore. If an inode region is added to your file system, the internal number of a given inode may change, making an incremental restore impossible. If the super-block time-stamps indicate that this may have happened, the dump command prints a warning. If the maximum inode number has changed, an incremental restore fails; you can use restore -x or restore -i.

The restore program contains an error-recovery algorithm that searches forward in units of 4096 bytes for the beginning of a cu_spcl structure (declared in /usr/include/dumprestor.h). If the dump output is buffered or rebuffered using a buffer size that is not a multiple of 4096 bytes, and if one such buffer-full of data is lost, the error recovery algorithm of restore will be disabled.

EXAMPLES

Example 1: To do a full dump to cartridge tape of each file system weekly, with an incremental dump daily, perform the following:

Once a week (everything in mass storage), enter the following command:

```
dump -t 0 -u -c /dev/dsk/slash_a
```

Repeat for each file system.

Daily (everything in file system that has been modified):

```
dump -t 9 -u -c /dev/dsk/slash_a
```

Repeat for each file system.

Example 2: To write dumps to round tapes with IBM standard labels, using a specified volume serial number list (this example is an incremental dump):

```
dump -d 6250 -t 9 -l sl -v a1:a2 -u /dev/dsk/slash_a
```

Example 3: The following example shows the use of the -P option:

```
#  /etc/dump " -P "enter +e VSN(s) from VSN pool for filesystem +f" -u -t 9 /dev/dsk/dumptest
dump (/dumptest to tape): Date of this level 9 dump: Mon Mar  9 11:15:01 1992
dump (/dumptest to tape): Date of last level 3 dump: Mon Jan 27 15:35:11 1992
dump (/dumptest to tape): Dumping /dumptest
dump (/dumptest to tape): to tape
dump (/dumptest to tape): mapping (Pass I) [regular files]
dump (/dumptest to tape): mapping (Pass II) [directories]
dump (/dumptest to tape): estimated 3158 sectors on 0.08 volume(s).
dump (/dumptest to tape): enter 1 VSN(s) from VSN pool for filesystem /dumptest
```

At this point, the operator would enter a VSN list and the process would continue.

Example 4: The following example dumps the in-use inodes from `/dev/dsk/xx` into disk file `ofile` and lists the files on `stdout`:

```
/etc/mknod apipe p
/etc/restore -T -t -f apipe &
/etc/dump -t 0 -A apipe -f ofile /dev/dsk/xx
```

The `dump` and `restore`(8) programs work locally. The `rdump`(8) and `rrestore`(8) programs work when the tape drive can be accessed with `/etc/rmt`. When the Cray Research tape daemon owns the tape drive, the following procedure writes a dump from a network-connected machine:

```
here> rsv CART
here> tpmnt -n -P tname ...
here> remsh there /etc/dump -t0 -f- /dev/dsk/xx  "|"  dd bs=409600 >tname
here> rls -a
```

FILES

`/etc/dumpdates`	Dump date record
`/etc/fstab`	File system information (used by the `-w` and `-W` options)

SEE ALSO

`rdump`(8), `restore`(8), `tpdaemon`(8)

`rls`(1), `rsv`(1), `tpmnt`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`dump`(5), `fstab`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

ebmchk – Scans an a.out file looking for EBM instructions

SYNOPSIS

/etc/ebmchk [-v] [-o *a.out_output_file*] *a.out_input_file*

IMPLEMENTATION

CRAY C90 series

DESCRIPTION

The ebmchk command scans the code space in an a.out file looking for enable bidirectional memory (EBM) instructions. If an EBM instruction is found in a valid code sequence, ebmchk prints the parcel address (relative to the beginning of the code segment); otherwise, it silently terminates.

The ebmchk command accepts the following options:

-v Prints all occurrences of the EBM pattern and a description of how ebmchk interpreted it. For actual EBM instructions, it also finds and prints the name of the preceding entry point from the traceback records, in addition to the parcel address specified by the default.

-o *a.out_output_file*
Creates an a.out file that is identical to the original a.out file with EBM instructions replaced by DBM instructions.

EXAMPLES

Example 1: To execute ebmchk on all files in /bin, enter the following:

```
$ su
Password:
# find /bin -type f -exec ./ebmchk {} ;
```

Example 2: To check an a.out file for occurrences of the EBM instruction (in verbose mode), enter the following:

```
ebmchk -v a.out
```

Example 3: To replace occurrences of the EBM instruction with DBM instructions, enter the following:

```
ebmchk -o b.out a.out
```

NAME

econfig – Verifies boot-time CSL directives for IOS model E systems and generates mknod commands for specified configuration

SYNOPSIS

/etc/econfig [-d] [-n] [-v] [*pfile*]

IMPLEMENTATION

Cray PVP systems with an I/O subsystem model E

CRAY J90 series

CRAY EL series

DESCRIPTION

The econfig command performs bounds checking (gaps and overlaps) and some validity and syntax checks on boot-time configuration specification language (CSL) directives read from its standard input or from the file *pfile*. This command is used to generate mknod requests that reflect the device configuration defined in the *pfile* (deadstart parameter file).

The econfig command accepts the following options and operand:

-d Generates mknod commands on stdout necessary to create the file system described in *pfile*.

-n Do not perform validity checking on device definitions in the supplied parameter file.

-v Generates comment headers on stdout describing the type of mknod commands that follow.

pfile File (that is, a deadstart parameter file) that contains CSL directives that describe a file system.

NOTES

If econfig returns a large number of error messages, they are most likely due to a "cascade effect" where the first few lines are valid errors, and the rest are "errors" due to those first few lines not being correct. For example, if the configuration of an I/O cluster is incorrect, this is listed as an error, but so is every subsequent reference to that cluster; simply correcting the cluster's configuration will cause the other reference errors to disappear. If you get a long error listing, concentrate on correcting the first few errors listed; then try it again.

CAUTIONS

The econfig utility does not check for overlapping disk slices.

NAME

errdemon – Invokes the error-logging daemon

SYNOPSIS

/etc/errdemon [*file*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The error-logging daemon errdemon collects error records from the operating system by reading special file /dev/error and places them in *file*. No analysis of the error records is done by errdemon; that responsibility is left to errpt(8) and/or the olhpa(8) diagnostic utility.

errdemon will launch the monitoring and notification program watchstream(8) if the file /etc/watcherror.re exists.

The daemon is terminated by errstop(8). Only an appropriately authorized user can start the daemon, and only one daemon can be active at any time.

file Specifies an output file for errdemon. If *file* does not exist, it will be created; otherwise, error records are appended to it, so that no previous error data is lost. Default: /usr/adm/errfile.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to start the error-logging daemon.

If the PRIV_SU configuration option is enabled, the super user is allowed to start the error-logging daemon.

FILES

/dev/error	Source of error records
/usr/adm/errfile	Default repository for error records

SEE ALSO

`errpt`(8) for information on processing the error report generated by `errdemon`
`errstop`(8) for information on terminating the error-logging daemon
`olhpa`(8) for information on the system error-log formatter
`thresholding`(7) for an introduction to automated monitoring and notification
`watchstream`(8) for information on the monitoring and notification program

`err`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014, for information on the error-logging interface

UNICOS Resource Administration, Cray Research publication SG–2302
Tape Subsystem Administration, Cray Research publication SG–2307

System Maintenance and Remote Testing Environment (SMARTE) Guide, Cray Research publication SD–1017. (This document contains information private to Cray Research, Inc. It can be distributed to non-CRI personnel only with approval of the appropriate Cray manager.)

NAME

errpt – Processes errors report generated by errdemon(8)

SYNOPSIS

/etc/errpt [-s *date*] [-s l*number*] [-e *date*] [-e l*number*] [-a] [-d *pdevlist*] [-f] [-h] [-o] [-q] [-r] [-t] [*files*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The errpt command processes data collected by the error-logging mechanism (errdemon(8)) and generates a report of that data. The default report is a detailed listing of errors posted in the *files* specified on the command line. Options apply to all files. If you do not specify *files*, errpt will try to use /usr/adm/errfile as *files*.

A summary report listing the options that might limit its completeness and the time stamped on the earliest and latest errors encountered is appended to the error report.

Memory location on the CRAY J90 series is reported as three values (for example, 01/01/05) corresponding to CRAY J90s with a 2X2, 4X4, and 8X8 backplane (respectively). The correct memory location value should be selected based on the hardware configuration of the machine.

A report can be limited to certain records by the invocation of any of the following options:

-s *date* Ignores all records posted earlier than *date*. *date* has the format *mmddhhmmyy*[*ss*] and is consistent in meaning with the date(1) command. The first *mm* is the month number; the second *mm* is the minute number.

-s l*number* Specifies that an error report start l*number* days prior to the date and time specified by either of the -e options. If the -e *date* or -e l*number* option is not specified, the start time and date are set to *number* days less than the current time and date. Note that this argument consists of a lowercase letter "l" (not the number "1"), with no space between the "l" and the following *number* value.

-e *date* Ignores all records posted later than *date*. *date* has the format *mmddhhmmyy*[*ss*] and is consistent in meaning with the date(1) command. The first *mm* is the month number; the second *mm* is the minute number.

-e l*number* Specifies that an error report end l*number* days prior to the current date and time. Note that this argument consists of a lowercase letter "l" (not the number "1"), with no space between the "l" and the following *number* value.

-a Produces a detailed report that includes all error types.

`-d` *pdevlist* Limits a detailed report to data about devices given in *pdevlist*. *pdevlist* can be one of two forms: a list of device identifiers separated from one another by commas, or a list of device identifiers enclosed in double quotation marks and also separated from one another by commas. `errpt` is familiar with the following device types: `comm`, `fddi`, `hippi`, `tape`, `lsp`, `dsk`, `ethernet`, and `atm`. (`dsk` includes only driver error records and not detailed IOS error records.) (See the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014.) For a list of supported disk devices, see `diskspec`(7). The device type `ios` implies all devices connected to the IOS (all of the preceding identifiers). An additional identifier is `mem`, which includes a detailed report of memory errors.

For IOS model E systems, the `-d` *pdevlist* option does not display the relative block number of a file system. The `bb`(8) command must be used to find the relative block number.

`-f` In a detailed report, limits the reporting of device errors to unrecovered errors.

`-h` In a detailed report, limits the reporting of errors to actual hardware device errors.

`-o` Produces a one line summary for each error. The summaries are sorted by device type and also sorted chronologically within each device type.

`-q` Produces a Quick Summary Report of all errors in the specified files. The Quick Summary includes an error total for each type of device; that is, memory, disk, and tape. For some devices, the total unrecovered errors are also displayed.

`-r` Formats the error records in raw mode. This will be useful if the error log has errors from devices or machine types that `errpt` is not prepared to format. You can also use the `-r` option to ensure that `errpt` is formatting new records properly.

`-t` Requests the formatted output of the buffered logs from the control units of the IBM ESCON 3490, 3490E, and 3590 devices in addition to the other output requested. There is no way to print only the buffered log records.

files Specifies the files that `errpt` will process. If *files* is not specified, `errpt` tries to use `/usr/adm/errfile`.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

Two types of error packets are associated with disk errors: C packets and A packets. C packets are sent by the IOS to log all disk errors. A packets are sent by the IOS in response to an I/O completion. The `errpt` program reports A-packet errors when no options are used; `errpt` reports both A- and C-packet errors when the `-a` option is used.

Recovered read errors will be reported in the A-packet response. However, because all write operations are write behind, recovered write errors are not reported in the A-packet response.

FILES

`/usr/adm/errfile`	Default error file
`/usr/src/uts/sys/erec.h`	Error record include file

SEE ALSO

`bb`(8) for information on creating relative bad block files from ASCII flaw table files
`errdemon`(8) for information on invoking the error-logging daemon

`date`(1) for information on printing and setting the date in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`errfile`(5) for information on error log file format in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

`diskspec`(7) for information on the physical characteristics of disk drives on Cray PVP systems (available only online)

General UNICOS System Administration, Cray Research publication SG–2301
UNICOS Resource Administration, Cray Research publication SG–2302
Tape Subsystem Administration, Cray Research publication SG–2307

NAME

errstop – Terminates the error-logging daemon

SYNOPSIS

/etc/errstop

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The errstop command sends a software termination signal (SIGTERM) to the error-logging daemon errdemon(8). Only a super user can terminate the error-logging daemon. errstop also terminates the diagnostic daemon dgdemon(8).

SEE ALSO

dgdemon(8) for information on invoking the diagnostic daemon
errdemon(8) for information on invoking the error-logging daemon

NAME

esdmon – Interactively monitors the logical-layer External Semaphore Device

SYNOPSIS

/etc/esdmon [-c] [-d] [-h] [-l] [-m] [-n *unicos-name*] [-p] [-r *repeat-interval*] [-s *SFS-Arbiter*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The esdmon command is used to interactively monitor the logical-layer External Semaphore Device. esdmon uses the special devices /dev/sfs and /dev/smp to interface to the respective device drivers.

Executing the esdmon command requires super-user permissions.

-c (Continuous) Causes esdmon to use the *curses* library for full-screen displays. Usually, esdmon operates in line mode. When invoked in *continuous* display mode, the initial display shows the *heart-beat* and *port* tables, and a two-line synopsis of the External Semaphore Device assignment. At the end of each *repeat-interval*, keyboard input is sampled for one of the following commands:

- ^l Clear and refresh the display.
- \+ Increase the *repeat-interval* by 1 second.
- \- Decrease the *repeat-interval* by 1 second.
- \> Move the the next higher display.
- < Move the the next lower display.
- 0-4 Move directly to the indicated display.
  - 0 The initial display shows the *heart-beat* and *port* tables, and a two-line synopsis of the External Semaphore Device assignment.
  - 1 Display the first of three lock assignments displays.
  - 2 Display the second of three lock assignments displays.
  - 3 Display the third of three lock assignments displays.
  - 4 A full screen display of all available state information for all 64 semaphores in the External Semaphore Device.
- h,? Display the *help* menu.
- q Quit, exit the program.

-d (Debug) Causes extra information to be displayed. Usually, this information is not needed and can interfere with other displays.

`-h` Displays the *heart-beat* table.

`-l` Displays the *lock* table.

`-m` Displays the *media* table.

`-n` *unicos-name*
Allows for an alternate UNICOS binary to be used for the *nlist* symbol table search.

`-p` Displays the *port* table.

`-r` *repeat-interval*
When used with the `-c` option, specifies the *repeat* interval used for refreshing the full-screen updates. The default value is 2 seconds.

`-s` *SFS-Arbiter*
Specifies the name of the *SFS Arbitration Service* to be monitored.

The *SFS-Arbiter* name must match one of the valid entries in the `/etc/config/sfs` configuration file.

FILES

`/etc/config/sfs` The Shared File System configuration file.

SEE ALSO

`sfs`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

exportfs – Exports and unexports directories to NFS clients

SYNOPSIS

/etc/exportfs [-a] [-i] [-o *options*] [-u] [-v] [*pathname*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The exportfs command makes a local directory or file available for mounting over the network by network file system (NFS) clients. The exportfs command cannot export remote mount file systems. It usually is invoked at boot time by an entry in the /etc/config/daemons file, and it uses information that is contained in the /etc/exports file to export *pathname* (which you must specify as a full path name). Only appropriately authorized users can run exportfs at any time to alter the list or characteristics of exported directories and files. Directories and files that are currently exported are listed in the /etc/xtab file.

With no options or arguments, exportfs prints the list of directories and files that are currently exported.

The exportfs command accepts the following options:

-a Exports all *pathnames* that are listed in /etc/exports, or if you specify -u, unexports all of the currently exported *pathnames*.

-i Ignores the options in /etc/exports. Usually, exportfs consults /etc/exports for the options that are associated with the exported *pathname*.

-o *options* Specifies a comma-separated list of optional characteristics for the *pathname* that is being exported. You can select the *options* from among the following:

access=*client*[:*client*]...
Gives mount access to each client that is listed. The default value allows any machine to mount the specified directory.

anon=*uid* Specifies *uid* as the effective user ID when a request comes from an unknown user. Root users (*uid* 0) are always considered unknown by the NFS server, unless they are included in the root option (following). The default value for this option is –2. To disable anonymous access, and to allow clients to mount only the exported directory, set the value of anon to –1.

cksum Checksums packets that are returned to clients.

kerberos
Specifies that Kerberos Remote Procedure Call (RPC) (AUTH_KERB) messages should be used for NFS transactions. The kerberos option may not be used with the "krb" option.

`krb` Specifies Kerberos authentication that is required for access to this export.

`nosync` Specifies that write operations to this file system are delayed. This option can significantly improve write performance but its use can result in loss of data if the server crashes before the data is written to disk.

`ro` Exports *pathname* with read-only characteristics. If you omit this option, *pathname* is exported with read-write characteristics.

`root=`*hostname*[:*hostname*]...
Gives root access only to the root users from a specified *hostname*. The default is that no hosts are granted root access.

`rw=`*hostname*[:*hostname*]...
Exports *pathname* with read-mostly characteristics. *Read-mostly* means exported with read-only characteristics to most machines, but with read-write characteristics to those specified. If you omit this option, *pathname* is exported with read-write characteristics to all.

`-u` Unexports the indicated *pathnames*.

`-v` Prints each directory or file as it is exported or unexported (verbose).

pathname Specifies path name.

NOTES

You cannot export a directory that is either a parent directory or a subdirectory of one that is currently exported and within the same file system.

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`	Allowed to use this command.
`sysadm`	Allowed to use this command. Shell redirected I/O is subject to security label restrictions.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

The UNICOS kernel must be configured with `NFS_SECURE_EXPORT_OK`.

The *client* argument can specify the name of a host or the name of a netgroup. `netgroups` is not supported for the `hostname` field in `/etc/exportfs`. For information on how to use a netgroup file, see `netgroup`(5).

FILES

`/etc/exports`	Static export information
`/etc/xtab`	Current state of exported *pathnames*

SEE ALSO

`nfsid`(1), `privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`exports`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

`fddidump` – Utility to dump FDDI (FCA-1) shared memory

SYNOPSIS

`/etc/fddidump` [`-a` *address*] [`-l` *length*]

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

DESCRIPTION

The `fddidump` utility dumps the contents of shared memory on an FDDI (FCA-1) channel adapter. To perform this function, `fddidump` uses the ioctl interface to the FDDI driver (`fd.c`) running on the Cray Research system.

Command-line options are as follows:

`-a` *address* Sets the shared memory address from which the dump will begin. Default is 0x1000.

`-l` *length* Sets the length (in bytes) of the dump. Default is 512.

SEE ALSO

CRI documents for FDDI:

`xfddidump`(8)

`fddi`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

ANSI documents for FDDI:

FDDI MAC (Media Access Protocol) Specification (FDDI-MAC), document number X3.139–1987, November 5, 1986

FDDI PHY (Physical Layer Protocol) Specification (FDDI-PHY), document number X3.148–1988, June 30, 1988

FDDI PMD (Physical Medium Dependent) Specification (FDDI-PMD), document number X3.166–1990, September 28, 1989

FDDI SMT (Station Management) Specification (FDDI-SMT), document number X3T9.5/84–49, Rev 6.2, May 18, 1990

Other related documents for FDDI:

RFC 1188 Proposed standard for the transmission of IP datagrams over FDDI networks. October 1990; D. Katz

Logical Link Control Specification (802.2 LLC), document number 802.2–1985, July 16, 1984

NAME

`fddiload` – Utility to load FDDI (FCA-1) microcode

SYNOPSIS

`/etc/fddiload` [`-b` *binary*] [`-z` *device*]

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

DESCRIPTION

The `fddiload` utility loads microcode running on the FDDI (FCA-1) channel adapter into its shared memory. To perform this function, `fddiload` uses the ioctl interface to the FDDI driver (`fd.c`) that runs on the Cray Research system.

The `fddiload` utility accepts the following options:

`-b` *binary* Sets the full path name to the file containing the binary image of the FCA-1 microcode. Default is `/etc//micro/FCA1.ucode`.

`-z` *device* Sets the UNICOS device name for the driver. Default is `/dev/fddi0/fd00`.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

SEE ALSO

CRI documents for FDDI:

`privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`fddi`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

ANSI documents for FDDI:

FDDI MAC (Media Access Protocol) Specification (FDDI-MAC), document number X3.139–1987, November 5, 1986

FDDI PHY (Physical Layer Protocol) Specification (FDDI-PHY), document number X3.148–1988, June 30, 1988

FDDI PMD (Physical Medium Dependent) Specification (FDDI-PMD), document number X3.166–1990, September 28, 1989

FDDI SMT (Station Management) Specification (FDDI-SMT), document number X3T9.5/84–49, Rev 6.2, May 18, 1990

Other related documents for FDDI:

RFC 1188 Proposed standard for the transmission of IP datagrams over FDDI networks. October 1990; D. Katz

Logical Link Control Specification (802.2 LLC), document number 802.2–1985, July 16, 1984

NAME

`fddimap` – Utility to gather FDDI station management information

SYNOPSIS

`/etc/fddimap` [`-d`] [`-e`] [`-p`] [`-u` *port*] [`-i` *infc*] [`-r`] [`-c`] [`-S`] [`-N`] [`-m` *meth*] [`-s` *name* | `MAC address`]

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

DESCRIPTION

The `fddimap` utility transmits FDDI station management (SMT) frames onto an FDDI ring from a Cray Research system that is directly attached to that same FDDI ring, to gather information about a particular station or all stations on the FDDI network. This utility also can build a logical ring map of the FDDI network.

To perform its function, `fddimap` uses the utility socket interface to the station management daemon (SMTD) running on the same Cray Research system (see `smtd`(8)).

The `fddimap` utility accepts the following options:

`-d` Enables debug mode; verbose output.

`-e` Sends a single SMT ECHO packet to the specified station.

`-p` Sends a continuous set of SMT ECHO packets to the specified station. This is very similar to a `/etc/ping` command, except it does not involve the IP protocol.

`-u` *port* Sets the UDP port number to which socket requests are sent. Default is 3000. See `smtd`(8).

`-i` *infc* Sets the interface ordinal of the FDDI interface to use when sending requests on the FDDI ring. Default is 0.

`-r` Prints a logical ring dump of all of the stations that exist on the FDDI ring.

`-c` Retrieves the MAC counters from the specified station. See option `-s`.

`-S` Retrieves the configuration SIF and operation SIF information from the specified station. See option `-s`.

`-N` Retrieves the NIF information from the specified station. See option `-s`.

`-m` *meth* Prints a logical ring map of the FDDI ring by using one of two methods specified by `meth`. Method 1 sends NIF requests counter-clockwise around the ring, starting with this station's upstream neighbor. Method 2 sends SIF requests clockwise around the ring, starting with this station's downstream station.

`-s` *name* | `MAC address`
Specifies the name (or FDDI MAC address) to which to send requests. (See options `-c`, `-S`, `-N`, `-e`, and `-p`.) When specifying a MAC address, use hexadecimal addresses in Canonical form, with colon (:) separators. (See the MAC Address File subsection.)

The MAC Address File

Rather than having to remember the 48-bit MAC addresses of all FDDI stations on the ring, you can create a file that contains these addresses (similar to the `/etc/hosts` file) to give names to each of the FDDI stations on the ring. This file is called `/etc/ethers`, and is standard on most UNIX based systems. The format of the file is as follows:

```
#
#    The addresses in this file must be in Canonical (Ethernet) form.
#
0:40:a6:0:0:e0          sn1703b-1
00:40:a6:00:00:10       sn1703b-2
ff:ff:ff:ff:ff:ff       broadcast
```

Comment lines may be placed anywhere in the file as long as the comments starts in the first column of a line. Comments are designated by a # character in column 1. Leading zeros within each byte are not required, but do make it easier to read for the users.

FILES

`/etc/ethers`	File that contains MAC addresses of all FDDI stations on an FDDI ring.

SEE ALSO

`smtd`(8), `xfddimap`(8)

`fddi`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

The ANSI documents for FDDI:

FDDI MAC (Media Access Protocol) Specification (FDDI-MAC), document number X3.139–1987, November 5, 1986

FDDI PHY (Physical Layer Protocol) Specification (FDDI-PHY), document number X3.148–1988, June 30, 1988

FDDI PMD (Physical Medium Dependent) Specification (FDDI-PMD), document number X3.166–1990, September 28, 1989

FDDI SMT (Station Management) Specification (FDDI-SMT), document number X3T9.5/84–49, Rev 6.2, May 18, 1990

Other related documents for FDDI:

RFC 1188 Proposed standard for the transmission of IP datagrams over FDDI networks. October 1990; D. Katz

Logical Link Control Specification (802.2 LLC), document number 802.2–1985, July 16, 1984

NAME

`fddistat` – Displays information about IOS model E FDDI devices

SYNOPSIS

`/etc/fddistat`

IMPLEMENTATION

Cray PVP systems with IOS model E

DESCRIPTION

The `fddistat` utility gets information from the UNICOS FDDI driver tables and from the Connection Management (CMT) processor on the FCA-1 FDDI channel adapter in an IOS model E. Information is obtained by using an `ioctl` system call and is displayed on the user's screen. For proper viewing, you need a display of at least 40 lines and 80 columns.

Command abbreviations are shown on the bottom line of each `fddistat` screen. All `fddistat` commands are a single character in length and are case-sensitive. Commands are as follows:

`h` Displays the help menu.

`r` Refreshes the display.

`e` Exits the `fddistat` program.

`s` Displays device and logical path statistics.

`p` Displays boot-time and run-time device and logical path parameters.

`q` Displays logical path read and write buffer queues.

`d` Sets device ordinal or displays device configuration and general state information.

`l` Sets logical path ordinal or displays logical path information.

`f` Displays information about the actual FDDI interface.

`x` Toggles extended view on and off.

`D` Enables device ordinal selection by using the plus (`+`) or minus (`-`) key.

`L` Enables logical path selection by using the plus (`+`) or minus (`-`) key.

`+` Increments device or logical path ordinal.

`-` Decrements device or logical path ordinal.

`>` Increases screen refresh rate.

`<` Decreases screen refresh rate.

Selecting Screen Refresh Rate and Ordinals

Near the bottom of the screen, just above the command help line, `fddistat` displays the current screen refresh rate, the current device ordinal, and the current logical path ordinal. You can change these settings at any time.

To select a new screen refresh rate, simply press the less than (<) or greater than (>) key to decrease or increase the rate.

Notice that either Selected Device or Selected Logical Path is highlighted; the one that is highlighted is the one that will be changed when you press the + or - key. Enter `D` on the command line to select the device ordinal or enter `L` to select logical path. After selecting one of these choices, use the + or - key to increment or decrement the ordinal.

Extended View

For debugging purposes, some fields are displayed in hexadecimal form when in extended mode.

`Statistics` Screen

Statistics are shown for both the device as a whole (left) and for the selected logical path (right). The `Statistics` screen displays the following items:

Item	Description
`Read Msgs`	Number of messages (frames) received from the FDDI network. Instantaneous read message rate is shown in parenthesis.
`Read Bytes`	Number of bytes received from the FDDI network. Instantaneous read bytes are shown in parenthesis.
`Write Msgs`	Number of messages (frames) sent to the FDDI network. Instantaneous write message rate is shown in parenthesis.
`Write Bytes`	Number of bytes sent to the FDDI network. Instantaneous write bytes are shown in parenthesis.
`Read Timeouts`	Number of times the IOS has returned a read request with a read time-out error code.
`Write Timeouts`	Number of times the IOS has returned a write request with a write time-out error code.

`No. Read Errors`
`No. Write Errors`
`No. Misc Errors`
	Total number of read, write, and miscellaneous errors that have occurred.

`User Read Errors`
`User Write Errors`
`User Misc Errors`
	The last four read, write, and miscellaneous error codes returned to the user in `errno`.

`Spec Read Errors` `Spec Write Errors`	

Spec Misc Errors	The last four read, write, and miscellaneous specific error codes that occurred. These errors go together with the user error codes. The specific error code is a more descriptive error code than errno.

Parameters Screen

Two versions of parameters are shown, RUN and BOOT. The RUN version shows the value of the parameter as it exists in the running system. The BOOT version shows the value of the parameter when the system was booted. If they are different, an ioctl was used by some process to change them. The Parameters screen displays the following items:

Item	Description
Device TREQ	Requested value for the target token-rotation timer (TTRT) that this station issues in its claim frames.
Device Copy Criteria	Value written to the copy criteria register in the channel adapter. (See the fddi(4) man page for a further explanation.)
Device Pad Count	Number of bytes the channel adapter hardware inserts at the beginning of all frames received and stripped from the beginning of all frames transmitted.
Device Max Write	Maximum number of write requests that can be posted to the IOS before queuing occurs in the mainframe driver.
Device Max Reads	Maximum number of reads requests that can be posted to the IOS before queuing occurs in the mainframe driver.
Path RFT	List of all frame types that the logical path receives.
Path Options	List of currently active options on the logical path.
Path Read Timeout	Value for read time-outs used on the logical path.

Queues Screen

The Queues screen shows a summary of the read and write buffer queues for the logical path. A buffer is either a read or write request. The Queues screen displays the following items:

Item	Description
No. Active	Number of buffers (I/O requests) currently active in the IOS.
No. Queued	Number of buffers (I/O requests) currently queued in the mainframe and waiting to be issued to the IOS.
First	Address of the first buffer (I/O request) on queue.
Last	Address of the last buffer (I/O request) on queue.
Next	Address of the next buffer (I/O request) on queue. This is the next buffer that will be issued to the IOS on completion of a buffer of the same type.

Device Screen

If you enter the Device command followed by a number, the ordinal of the selected device is set to that number (for example, d 1 sets the device ordinal to 1). The Device screen shows the following items:

Item	Description
IOPATH	I/O cluster, IOP, or IOP channel in which the FDDI interface is installed.
Flags	Currently active device flags. The FDDI driver uses these flags internally to coordinate its activities.
MAC Address *(IEEE)*	48-bit Media Access Control (MAC) address of an interface in Canonical form (the form in which the address appears in Cray main memory).
MAC Address *(FDDI)*	Media Access Control (MAC) address of an interface in FDDI (MSB) form (the form in which the address appears when on the FDDI medium).
UNA Address *(IEEE)*	48-bit Media Access Control (MAC) address of an interface's Upstream Neighbor in IEEE form.
UNA Address *(FDDI)*	48-bit Media Access Control (MAC) address of an interface's Upstream Neighbor in FDDI form.
Old UNA Address *(IEEE)*	48-bit Media Access Control (MAC) address of an interface's previous Upstream Neighbor in IEEE form.
Old UNA Address *(FDDI)*	48-bit Media Access Control (MAC) address of an interface's previous Upstream Neighbor in FDDI form.
DNA Address *(IEEE)*	48-bit Media Access Control (MAC) address of an interface's Downstream Neighbor in IEEE form.
DNA Address *(FDDI)*	48-bit Media Access Control (MAC) address of an interface's Downstream Neighbor in FDDI form.
Old DNA Address *(IEEE)*	48-bit Media Access Control (MAC) address of an interface's previous Downstream Neighbor in IEEE form.
Old DNA Address *(FDDI)*	48-bit Media Access Control (MAC) address of an interface's previous Downstream Neighbor in FDDI form.
MAC Services Available	Indicates whether MAC services are available; if not available, the MAC is offline for some reason. When offline, frames cannot be sent or received.
LLC Services Available	Indicates whether LLC services are available; if LLC services are not available, protocols such as TCP/IP cannot use the network interface.

`DAD Results` (MAC_DA_Flag)
: Shows results of the FDDI Duplicate Address Detection test, which is constantly being executed as a part of Station Management (SMT).

`No. Logical Paths Configured`
: Shows maximum number of logical paths configured on the device. This number usually comes from the parameter file at boot time.

`No. Logical Paths Open`
: Shows current number of logical paths open on the device.

`Paths That Are Open`
: Shows current list of logical paths open on the device.

`Logical Path` Screen

If you enter the `Logical Path` command followed by a number, the ordinal of the selected path is set to that number (for example, `l 5` sets the path ordinal to `5`). The `Logical Path` screen displays the following items:

Item	Description
`Flags`	Logical path flags that currently are active. The FDDI driver uses these flags internally to coordinate its activities.
`PID`	Process ID of the process that currently has the logical path open.
`Command`	UNICOS command name that currently has the logical path open. This should correspond to the PID.

`Fddi` Screen

The `Fddi` screen displays information about the state of the actual FDDI interface on the channel adapter. A brief summary of what is displayed in this screen follows. For a more detailed description of these parameters, see the ANSI FDDI SMT (Station Management) Specification.

Item	Description
`T_MAX`	Maximum target token-rotation timer (TTRT) that this station supports.
`TVX`	Maximum time in which this station should see a valid frame or token.
`T_REQ`	Value transmitted in this station's claim frames. This is the token-rotation timer (TRT) at which this station desires the ring to operate.
`T_NEG`	Value that the claim process negotiates. This is a result of all stations claiming with their respective `T_REQ` or a higher claim.
`CFM State`	Current state of the configuration management (CFM) state machine for the station. CFM manages the internal token path within the station.
`RMT State`	Current state of the ring management (RMT) state machine for the station. RMT manages the state of the ring at all times and determines when the ring is operational. RMT also is responsible for recovering the ring once it has seen a catastrophic fault.

`UNA` MAC address of this station's upstream neighbor shown in both IEEE and FDDI forms.

`MY Address` MAC address of this station shown in both IEEE and FDDI forms.

`DNA` MAC address of this station's downstream neighbor shown in both IEEE and FDDI forms.

`TX_Ct` (Frames Transmitted Count)
Number of frames that this station transmitted successfully.

`Token_Ct` (Token Count)
Number of valid tokens that this station received.

`RX_Ct` (Frames Received Count)
Count of all complete frames received, including MAC frames, void frames, and frames that this station stripped.

`Late_Ct` (Late Count)
When the ring is operational, number of times TRT expired before the ring became operational the last time. When the ring is nonoperational, number of times TRT has expired while waiting for the ring to become operational. This count remains 15 until a MAC reset occurs.

`Copied_Ct` (Frames Copied Count)
Number of frames this station copied successfully.

`Lost_Ct` (Lost Frame Count)
Number of instances in which a format error is detected in a frame or token jeopardizing credibility of PDU reception.

`NotCopied_Ct` (Frames Not Copied Count)
Number of frames intended for this station but not copied successfully.

`Error_Ct` (Error Isolated Count)
Number of error frames this station detected that no previous station detected. Error frames may be any of the following types: FCS error detected and received E indicator clear; frame of a length that is not valid and received E indicator clear; or received E indicator not equal to `SET` or `RESET`.

`Ring_Ct` Number of times the ring has gone from not operational to operational.

`TVX_Ct` Number of times TVX has expired. TVX is the FDDI timer that ensures a valid frame or token is received periodically.

`CEM State` Current state of the configuration element management (CEM) machine for the given port. CEM performs the interconnection of PHYs and MACs to configure the ports and MACs within the station.

`ECM State` Current state of the entity coordination management (ECM) state machine for the given port. ECM controls the optical bypass switch of the physical medium dependent (PMD) layer and signals the physical connection management (PCM) when the media is available. ECM also starts the PCM's for the A and B ports when the optical bypass switching is complete.

`Current RX LS`
Sampling of the FDDI line state currently being received by the given PHY.

`PCM State` Current state of the physical connection management (PCM) machine for the given port. PCM initializes the connection of neighboring ports and manages signaling between ports.

`T_VAL` Hexadecimal value representing bits transmitted to the neighboring PHY during the `SIGNAL` state of the PCM state machine. This code describes different things about this station to the neighboring station (for example, the port type (A, B, M, or S) of this port, the type of link confidence test (LCT) that this port requires (short, long, medium, or extended, and so on).

`R_VAL` Hexadecimal value representing bits received from the neighboring PHY during the `SIGNAL` state of the PCM machine.

`Mode` Type of attachment (`TREE` or `PEER`) to the ring. `PEER` mode indicates the station is connected to another port of similar type. `TREE` mode indicates the station is connected to a concentrator's M port; and thus, is part of a branch of a `TREE` off the dual ring.

`Neighbor Type`
Port type (A, B, M, or S) of the neighboring PHY.

`LEM Count` Number of link errors detected on this port since the station was last reset.

`LEM Reject Count`
Number of times this port was removed from the ring due to exceeding the LER threshold.

`LER Avg.` Long-term average link error rate estimate for this port. This count ranges from 10 e–4 through 10 e –15 bits.

`LER Alarm` Link error rate at which this link connection exceeds a preset alarm threshold. The alarm value ranges from 10 e–4 through 10 e –15 bits.

`LER Cutoff` Link error rate at which this link connection is flagged as faulty. The cut-off value ranges from 10 e–4 through 10 e –15 bits.

`EBError Count` (Elasticity Buffer Error Count)
Number of times the elasticity buffer overflowed.

SEE ALSO

`fddimap`(8), `xfddimap`(8), `smtd`(8)

`fddi`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

The ANSI documents for FDDI:

FDDI MAC (Media Access Protocol) Specification (FDDI-MAC), document number X3.139–1987, November 5, 1986

FDDI PHY (Physical Layer Protocol) Specification (FDDI-PHY), document number X3.148–1988, June 30, 1988

FDDI PMD (Physical Medium Dependent) Specification (FDDI-PMD), document number X3.166–1990, September 28, 1989

FDDI SMT (Station Management) Specification (FDDI-SMT), document number X3T9.5/84–49, Rev 6.2, May 18, 1990

Other documents related to FDDI:

RFC 1188 Proposed standard for the transmission of IP datagrams over FDDI networks. October 1990. D. Katz

Logical Link Control Specification (802.2 LLC), document number 802.2–1985, July 16, 1984

NAME

ff – Lists file names and statistics for a file system

SYNOPSIS

/etc/ff [-d] [-a *n*] [-c *n*] [-i *inode_list*] [-I] [-l] [-m *n*] [-n *file*] [-p *prefix*] [-s] [-u] *special*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The ff command (fast find) reads the i-list and directories of the special file *special*, assuming it is a file system, saving inode data for files that match the selection criteria. The *special* file can be a block special file (see mknod(8)). Output consists of the path name for each saved inode, plus any other file information requested, using the options below. Output fields are positional. Except for hard links, which are printed at the end, the output is produced in inode order; fields are separated by tabs. The number of files selected and, optionally, the number of hard links detected are sent as error output on separate lines. The default line produced by ff is as follows:

path-name i-number

With all options enabled, output fields area are as follows:

path-name i-number size uid

The *n* argument in the following option descriptions is used as a decimal integer. Where *n* indicates number of days, a day is defined as a 24-hour period and the *n* argument may be signed, with the following meanings when *n*, for example, has a value of 7:

+7 Equal to or more than seven days

7 Between seven and eight days

–7 Less than seven days.

The ff command accepts the following options:

-d Enables additional debugging messages.

-a *n* Selects if the inode has been accessed in *n* days.

-c *n* Selects if the inode has been changed in *n* days.

-i *inode_list* Generates names for only those inodes specified in *inode_list*.

-I Does not print the inode number after each path name.

-l Generates a supplementary list of all path names for multiply linked files.

`-m` *n*	Selects if the inode has been modified in *n* days.
`-n` *file*	Selects if the inode has been modified more recently than the *file* argument.
`-p` *prefix*	Adds the specified *prefix* to each generated path name. The default is "`.`".
`-s`	Prints the file size, in bytes, after each path name.
`-u`	Prints the owner's login name after each path name.
special	Name of the file system that contains the i-list and directories to be read by the `ff` command.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`	Allowed to specify any file system.
`sysadm`	Allowed to specify any file system. Shell redirected output is subject to security label restrictions.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to specify any file system.

BUGS

Only a single path name out of any possible path names is generated for a multiply linked inode, unless you specify the `-l` option. When you specify `-l`, no selection criteria apply to the names generated. All possible names for every linked file on the file system is included in the output.

EXAMPLES

Example 1: In the following example, `ff` generates a list of the names of all files on the `/dev/dsk/root` file system:

```
ff -I -l /dev/dsk/root
```

Example 2: In the following example, `ff` produces an index of files and inode numbers that are on the `/dev/dsk/usr` file system and have been modified in the last 24 hours:

```
ff -m -1 -l /dev/dsk/usr
```

Example 3: In the following example, `ff` obtains the path names for inodes 451 and 76 on the `/dev/dsk/usr` file system:

```
ff -i 451,76 /dev/dsk/usr
```

SEE ALSO

`mknod`(8)

`find`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

NAME

fingerd – Daemon program for finger(1B)

SYNOPSIS

/etc/fingerd

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The fingerd command is the daemon program for the remote finger(1B) utility. It responds to requests for active user information by using the Internet finger protocol.

The inetd(8) command invokes fingerd. inetd listens on multiple ports and when a request is made on the finger port, it forks fingerd. fingerd uses the TCP port number 79 by default, as specified in the services(5) database.

FILES

/etc/inetd.conf	inetd configuration file that contains the fingerd entry
/etc/services	File that maps service names to port numbers

SEE ALSO

inetd(8)

finger(1B) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

services(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

`fping` – Send an echo packet to a GigaRing node

SYNOPSIS

`/etc/fping` [`-c` *count*] [`-i` *wait*] [`-q`] [`-s` *size*] *special*

IMPLEMENTATION

CRAY T90 systems with GigaRing-based I/O

CRAY J90 systems with GigaRing-based I/O

DESCRIPTION

The `fping` command sends a series of echo message packets to the GigaRing node represented by the specified *special* file. These special files are, by convention, located in the `/dev/fmsg` directory; see `fmsg`(4) for further information. Upon receiving the echo packet, the remote GigaRing node will echo the packet back to the sender. The `fping` command prints out timing information for each echo packet sent.

The `fping` command accepts the following arguments:

`-c` *count* Specifies how many echo packets to send. The `fping` command stops after upon reaching this count. The default count is 8.

`-i` *wait* Wait *interval* number of seconds between each packet. The default interval is 1 second.

`-q` Quiet output. Just the summary line is displayed upon termination.

`-s` *size* Specifies the size of the payload, in 64 bit words, of the message packets sent. The default payload size is 1 word but the echo packet protocol and time stamps use an additional 4 words. The maximum value for size is 120.

special File representing the GigaRing node.

EXAMPLES

The following command sends 8 echo packets to GigaRing I/O node (ION) at ring 2, node 2:

```
# /etc/fping /dev/fmsg/e0202
4+1 words from 020: seq = 0: time = 164.082 + 173.033 us
4+1 words from 020: seq = 1: time = 174.838 + 173.620 us
4+1 words from 020: seq = 2: time = 173.576 + 176.560 us
4+1 words from 020: seq = 3: time = 174.527 + 152.162 us
4+1 words from 020: seq = 4: time = 171.096 + 173.013 us
4+1 words from 020: seq = 5: time = 174.411 + 153.058 us
4+1 words from 020: seq = 6: time = 172.447 + 174.411 us
4+1 words from 020: seq = 7: time = 173.042 + 152.998 us
FPING /dev/fmsg/e0202 (20): 5 data words
```

The `fping` command prints out 2 timing components. The first measures the time it takes to send the echo message packet from the `fmsg` driver in the Unicos kernel and receive the response back at the `fmsg` driver. This time includes Unicos message overhead, GigaRing message time to send the message packet, I/O node overhead to receive and retransmit the message, GigaRing message time for the message packet return, system interuupt overhead, and Unicos message overhead to receive the message. The second time component is the user system call time (a write) plus the time to reconnect the user (the `fping` command). Both times are in microseconds.

FILES

`/dev/fmsg/*`

SEE ALSO

`fmsg`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

fsck – Checks file system consistency and interactively repairs it

SYNOPSIS

/etc/fsck [-c] [-d] [-f] [-i] [-n] [-p] [-q] [-s] [-S] [-u] [-W] [-y] *special*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The fsck command audits and interactively repairs inconsistent conditions in an NC1FS file system.

If the file system is not consistent, fsck asks for confirmation before attempting any correction. In all cases, a y or yes response causes the correction to be made; a response of n or no leaves the file system unchanged. If you do not have write permission, fsck may ask for approval to change the file system, but the action is refused by the I/O subroutine after it tests a software lock.

Before a file system is mounted for general read/write usage, all of the questions asked by fsck should be answered affirmatively. An answer of n or no is appropriate for surveying a file system that will be corrected later. For example, you might answer questions with n or no if you wish to mount the file system as read-only to rescue some files before fsck causes them to be deleted or if you wish to repair the file system manually.

At the conclusion of the program, fsck summarizes the size of, and the free space in, the file system.

Some forms of disk corruption require that you run the fsck command several times.

The fsck command accepts the following options and operand:

-c Used in conjunction with -y, assumes a clear response to all yes, no, or clear questions that fsck asks.

-d Debug. Causes fsck to obtain the device characteristics from a prompted dialogue on stdin, rather than from a stat(2) system call. This permits fsck to be tested against something less than a mountable file system.

-f Performs a fast check. Checks only the inodes and allocations. Duplicate allocations are announced, but no inodes may be cleared without examination of the directories. The free lists of the dynamic block may be rebuilt.

-i Causes fsck to ignore most duplicate allocations. This option allows you to isolate corrupt inodes that cause multiple duplicate allocations. If you specify this option, you should run fsck again after clearing the corrupt inodes.

-n Assumes a no response to all questions that fsck asks.

-p Specifies "preen" mode, which is designed to match the mfsck(8) program interface.

`-q` Suppresses ("quiets") most of the `fsck` output, including warnings and errors that apply to specific files and directories, all errors from phase 5, and all problems with I/O. If you specify this option, and if no responses from the operator are required, the program writes less than twenty lines of output.

`-s` Ignores the actual free lists and (unconditionally) constructs new ones by rewriting the dynamic blocks of the devices.

`-S` Conditionally reconstructs the free list. Specifically, if no problems were detected in the file system, the free list is rebuilt. The test is made against a flag set whenever `fsck` asks a question.

`-u` Does not quit after reading the dynamic block, even if the file system appears to have been unmounted cleanly.

`-W` Suppresses all inode warning messages. You can generate and correct the warning-level problems after the file system is mounted.

`-y` Assumes a `yes` response to all questions that `fsck` asks.

special Name of the special file or the file system descriptor file that may be used to open the file system. A required operand.

The `fsck` command performs the following functions.

1. Verifies that *special* is a file system:
   - Locates super blocks and dynamic blocks
   - Verifies that inode blocks, bad blocks, super blocks, and dynamic blocks may be allocated without error
   - Verifies that total inode count is consistent with blocks allocated for inodes
   - Verifies that file system sizes are consistent with the value returned by the `stat`(2) system call
2. Examines each active inode:
   - Verifies that the mode field is valid
   - Verifies that file sectors may be allocated without conflict and within valid areas of the file system
   - Verifies that the last byte of the file is contained within the last allocation
3. Examines directory inodes:
   - Verifies pointers and signatures (the `cd_signature` field) in each directory sector
   - Verifies for each entry in a directory that a nonzero inode field refers to an accessible inode
   - Validates the directory tree structure
   - Checks directory allocation for missing sectors
4. Scans for directory errors. Unlinked directories are offered for inclusion in the `lost+found` directory. Other directory problems may require that the directory be cleared.
5. Performs a final pass through the active inodes:

- Offers unlinked nondirectory files for inclusion in the `lost+found` directory. Other file problems may require that the file be cleared.
- Verifies link count field of inodes.

6. Verifies dynamic information.
7. Rebuilds all dynamic information.
8. Terminates; prints a summary of the file system state.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`, `sysops`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

CAUTIONS

Some inconsistencies are noted by `fsck` but are not corrected. The `last byte not in last allocation` condition may be caused by pre-allocating a scratch file, and can be cleared by copying the file. The `. . .directory tree structure. . .` error may be caused by a root-user using `/etc/link` and `/etc/unlink`. It can be cleared in the same fashion. Neither of these conditions need prevent the file system from being mounted.

Most corrective actions result in some loss of data; you can determine the amount of data lost and the severity of the loss from the diagnostic output.

Any file system other than the root may be examined by `fsck` while it is mounted, but it will not be changed unless it is unmounted. The root file system is the only one `fsck` will change while mounted. On Cray PVP systems, the reboot step will be requested if necessary.

If unlinked inodes are relinked, existing space in an existing `lost+found` directory in the root directory is required.

FILES

`/dev/dsk/*` Block special device names

SEE ALSO

`crash`(8), `mdd_pre`(8), `mfsck`(8), `mkfs`(8), `mknod`(8)

`stat`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

`fs`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

fsdaemon – File system monitor daemon

SYNOPSIS

/etc/fsdaemon [-c *crit_cmd*] [-l] [-n [+]*incr*] [-p *path*] [-q] [-s *seconds*] [-w *warn_cmd*]

/etc/fsdaemon [-c *crit_cmd*] [-n [+]*incr*] [-p *path*] [-q] [-s *seconds*] [-t] [-w *warn_cmd*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The fsdaemon daemon monitors file systems and starts warning or critical commands if the amount of space in use exceeds a threshold value. When the daemon is initiated, it has no information about which file systems to monitor or what the warning or critical thresholds might be. The fsmon(8) command must be used to configure the daemon as needed. See the fsmon(8) man page in conjunction with this man page because these two programs are components of a single feature.

The fsdaemon daemon accepts the following options and arguments:

-c *crit_cmd* Causes the daemon to execute a command, *crit_cmd*, when a critical threshold is reached on a file system. Each occurrence of a critical condition results in the invocation of a separate instance of *crit_cmd* with the name of the file system (*filesystem*) appended to the *crit_cmd* string preceded by one space. After a command is started, the daemon no longer monitors that file system until a reset is received. To resume monitoring, the command must at some point execute the fsmon command with the -r or -R option and the file system name.

-l Locks daemon in memory. If this option is specified, the daemon locks itself in memory after it has finished initialization. This option is illegal if the -t option is used.

-n [+]*incr* Executes a nice(2) system call with the value -*incr* unless *incr* is preceded by +. For example, an *incr* value of 4 executes a nice(-4) system call and an *incr* value of +4 executes a nice(4) system call.

-p *path* Uses the specified *path*, rather than the default path, for the pipes and files used in communication between fsdaemon and fsmon. This allows you to test a file system monitor version without affecting a production version that may be running.

-q Allows the operator to terminate fsdaemon by using fsmon with the -q option. If this option is not specified, the daemon cannot be terminated with fsmon.

-s *seconds* Specifies cycle time between file system checks. The default is 5 seconds.

`-t` Test mode. Allows the daemon program to be run in the background by a user during development and testing, without the program executing system calls that can be used only by super users. If this option is specified, `fsdaemon` does not detach itself from the initiating terminal or become a true daemon.

`-w` *warn_cmd* Causes the daemon to execute *warn_cmd* when a warning threshold is reached on a file system. Each occurrence of a warning condition results in the invocation of a separate instance of *warn_cmd* with the name of the file system (*filesystem*) appended to the *warn_cmd* string preceded by one space. After a command is started, the daemon no longer monitors that file system until a `reset` is received. To resume monitoring, the command must at some point execute the `fsmon` command with the `-r` or `-R` option and the file system name.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to start the message daemon.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to start the message daemon.

MESSAGES

The usual usage messages are written to the `stderr` file during the initialization phase. After daemon state is reached, all messages are written either to the log file or to the command output pipe.

FILES

`/usr/spool/fsmonitor/Fd.log`	Active log file
`/usr/spool/fsmonitor/Fsdmn.cmd_to`	Daemon command input pipe
`/usr/spool/fsmonitor/Fsdaemon.pid`	Daemon's process ID
`/usr/tmp/Fm_in`*xxxxx*	Daemon command output pipe

SEE ALSO

`fsmon`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

fsed – Debugs NC1FS file systems

SYNOPSIS

/etc/fsed [*file1 file2* ...]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The fsed command debugs NC1FS file systems. It provides display and patch capabilities inside the files specified on the command line.

The input parse mechanism separates commands by using new lines and semicolons. Within a command, blanks and tab characters are used to separate tokens. A # character begins a comment. Most commands consist of a 2-character command code, followed by an optional modifier character and a single token command operand.

The program maintains six special addresses. The dot-address (.) is where the next display will begin. The double-quotation address (") is the result of the last address expression typed. All changes and name references use the double-quotation address. The d-address, the i-address, the SB-address, and the DB-address refer to the last locations that were displayed in the directory, inode, super-block, and dynamic-block format, respectively. These addresses are displayed in response to the id command with an a modifier.

The program maintains a single sector-sized buffer. The data in this buffer may be changed and examined repeatedly without being written to the disk. If the data in the buffer was changed, issuing a command that will destroy this data provokes an expected write message.

Commands

The fsed command contains the following commands:

he — Help. A command description is sent to standard output.

gd*mm nn* — General display. The next *nn* items at dot-address are displayed in a format specified by the modifier *mm*. The gd command accepts multiple and contradictory modifier characters. The default modifier is ow. Other modifier characters are as follows:

- o,O — Display in octal.
- x,X — Display in hexadecimal.
- d,D — Display in decimal.
- c,C — Display as characters, made printable.
- f,F — Display as floating point, 8 bytes per item.

b,B Each item consists of 1 byte.

p,P,q,Q Each item consists of 2 bytes.

h,H Each item consists of 4 bytes.

w,W Each item consists of 8 bytes.

fd*m nn* Formatted display. The next *nn* items at dot-address are displayed in a special format as indicated by the modifier *m* . The modifier can have any of the following values:

i,I Display as an inode.

d,D Display as a directory.

SB Display as a super block. Modifiers SBa, SBb, and SBc cause different parts of this information to be displayed.

DB Display as a dynamic block. The four dynamic block displays are DBa, DBb, DBc, and DBd.

(NULL) Repeats the last command if it was a display.

da*m expr* Defines address. If there is no modifier, *m*, the expression is a byte address within the current partition. The modifier can have the following values:

b The expression is a block number.

i The expression is an inode number.

n The expression is a name of a field in the structure at "- address.

Expressions may begin with a special address designator: ., ", i, or d.

gc *expr* General change. If *expr* is a string, the strlen(*expr*) bytes at double-quotation address are changed. If *expr* is not a string, a single word at double-quotation address (") is changed.

nc *name expr* Named change. The field named *name* at double-quotation address is changed to the new value, *expr*. Special-case coding provides for changing the signature in a directory entry with nc signature; no *expr* is required.

wr Write. The sector buffer containing possibly changed data is returned to the disk. You must use a write command to make these changes permanent.

nm*m req* Names. nm shows the list of recognized names. The optional modifier, *m*, (which can be any optional modifier) results in a full listing. The optional request, *req*, may have the following values:

i Inode names

d Directory names

SB Super-block names

DB Dynamic block names

`ni` *pathname*	Name inode. Begin in the root directory. Set dot address, double-quotation address, and inode address to the inode for the specified file by using the *pathname*.
`cp` *dev*	Change partition. Each file is identified by the device number in its super block. On CRAY Y-MP systems, you can use the `cp` command to examine several file systems simultaneously.
`id`*m*	Internal displays. The required modifier, *m*, can have the following values: `a` Designates the internal addresses `i` Designates the inode information for the current partition `p` Designates the partition summary
`fo` *path*	File out. The file from the current inode address is copied to the file that *path* specifies.
`op`*m path*	Open. An external file is made available by using a `fopen`(*path*, *m*) call. The `cl`, `ou`, and `in` commands also reference this external file.
`cl`	Close. The current external file is closed.
`ou`	Out. The sector buffer contents are written to the external file.
`in`	In. The sector buffer is filled with data from the external file.
`!` *arguments*	Escape to shell. The indicated *arguments* are sent to the system subroutine to be performed by a separate shell.
`qu`	Quit. The program performs the requested action.

MESSAGES

When an error occurs, an informative message is printed and the input buffer is flushed.

EXAMPLES

Example 1: The following example reads the `mbox` file in directory `/xxx` on Cray PVP systems:

```
ni /xxx/mbox;dan a0.blk;gdc 200
```

Repeated new lines will display successive characters.

Example 2: The following example restores the super block in sector 1 from a copy in sector 8, using the file `mike`:

```
opw mike; dab 8; ou ; cl
opr mike; dab 1; in ; wr; cl
```

Example 3: The following example restores the magic word in a dynamic block on Cray PVP systems:

```
dab DB; nc magic 0x6e6331646231636e
wr
```

NAME

`fslogd` – File system error logging daemon

SYNOPSIS

`/etc/fslogd`

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The file system error logging daemon, `fslogd`, collects file system error records from the operating system by reading the character special pseudo device `/dev/fslog`. It prompts the operator for desired recovery actions depending on the type of error encountered. These recovery actions allow increased system resiliency by enabling the system to continue running after file system errors have been detected. This mode of operation is selectable on a file system basis when you use the `mkfs`(8) or `setfs`(8) commands to disable the file system panic flag.

`fslogd` handles three types of errors: file system errors, directory errors, and inode errors detected by the kernel. File system errors cause `fslogd` to prompt the operator to request that the system be panic'ed, the corrupted file system data structure contents be formatted and displayed, or the file system in error be unmounted for maintenance by `fsck`(8) or `fsed`(8). The operator should choose the appropriate option based on the critical level of the file system involved and the extent of the damage. There are currently no directory errors reported by the kernel in `/dev/fslog`. The `fslogd` daemon reports inode errors to the operator to request that the system be panic'ed, the corrupted inode data structure contents be formatted and displayed, or the associated file in error be removed.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	**Action**
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

FILES

`/dev/fslog`	Source of file system error log records

SEE ALSO

`mkfs`(8), `setfs`(8)

`fslog`(4), `fslrec`(5), `inittab`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

`fsmap` – Displays file system free-blocks

SYNOPSIS

`/etc/fsmap` *fsname*

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `fsmap` routine provides a display of the available free-blocks on a specified file system, *fsname*.

The format `fsmap` produces is as follows:

```
start  block   start  block   start  block   start  block
-----  ----- | -----  ----- | -----  ----- | -----  -----
```

Entries are displayed horizontally in four columns to conserve space. Each entry consists of a starting block number (`start`) and the number of contiguous blocks (`block`).

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`	Allowed to specify any file system.
`sysadm`	Allowed to specify any file system. Shell redirected output is subject to security label restrictions.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to specify any file system.

EXAMPLES

Example 1: The following example shows a common use of `fsmap`:

```
fsmap /dev/dsk/usr_mail

Free block layout for: /dev/dsk/usr_mail

  Start    Count     Start    Count     Start    Count     Start    Count
 -------  ------- | -------  ------- | -------  ------- | -------  -------

Partition: 0  Blks: 0-4559  Dev: 40-A1-27

      5        1 |      38        1 |     102        1 |     105        1
    110        1 |     144        1 |     166        1 |     178        1
    237        3 |     243        1 |     245        3 |     250        1
    253     1239 |    1513      903 |    2417        1 |    2419      966
   3406       64 |    3472      361 |    3854      623 |    4478       82

Partition: 1  Blks: 4560-18239  Dev: 40-A1-27

   4873        3 |    4884        1 |    4886        9 |    4896        2
   4910     4717 |    9648        1 |    9651        6 |    9658        5
   9670        3 |    9678        2 |    9682        1 |    9684        9
   9812       12 |    9946        2 |   10124        2 |   10129        2
  10133        4 |   10142       46 |   10189       43 |   10253       10
  10426       42 |   10510      147 |   10678      183 |   10862     5863
  16726      133 |   16860     1380
```

Example 2: The next example shows another use of `fsmap`:

```
fsmap /dev/dsk/qtest3

Free block layout for: /dev/dsk/qtest3

 Start    Count     Start    Count     Start    Count     Start    Count
-------  ------- | -------  ------- | -------  ------- | -------  -------

Partition: 0  Blks: 0-10943  Dev: 40-A2-30

    186     6047 |    6234     2802 |    9037        5 |    9043      132
   9176     1768
```

When there is more than one physical partition in the specified logical device, then the message "`Partition: 0  Blks:#-# Dev:` *device_name*" is repeated for each partition.

SEE ALSO

`bmap`(8), `dmap`(8)

NAME

fsmon – Interfaces with the file system monitor fsdaemon(8)

SYNOPSIS

```
/etc/fsmon -a [-c nnn] [-d] [-D] [-i cw] [-p path] [-t] [-w nnn] filesystems
/etc/fsmon -a [-c nnn] [-d] [-D] [-n cw] [-p path] [-t] [-w nnn] filesystems
/etc/fsmon -a [-c nnn] [-D] [-e] [-i cw] [-p path] [-t] [-w nnn] filesystems
/etc/fsmon -a [-c nnn] [-D] [-e] [-n cw] [-p path] [-t] [-w nnn] filesystems
/etc/fsmon -m [-c nnn] [-d] [-D] [-i cw] [-f state] [-p path] [-t] [-w nnn] filesystems
/etc/fsmon -m [-c nnn] [-d] [-D] [-n cw] [-f state] [-p path] [-t] [-w nnn] filesystems
/etc/fsmon -m [-c nnn] [-D] [-e] [-i cw] [-f state] [-p path] [-t] [-w nnn] filesystems
/etc/fsmon -m [-c nnn] [-D] [-e] [-n cw] [-f state] [-p path] [-t] [-w nnn] filesystems

/etc/fsmon [-D] -q [-p path] [-t]
/etc/fsmon [-D] [-p path] [-t] -R filesystems
/etc/fsmon [-D] [-p path] [-t] -r state filesystems

/etc/fsmon [-D] -l [-p path] [-t]
/etc/fsmon [-D] [-p path] [-s state] [-t] [filesystems]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The fsmon command is the command-level interface to the file system monitor daemon fsdaemon(8). It provides three functions:

- Adds or modifies information used by the daemon to monitor file systems. The first set of eight command lines in the SYNOPSIS section provides these functions.
- Affects the internal state of the daemon. The second set of three command lines in the SYNOPSIS section provides these functions.
- Displays log or file system status. Status displays can be selected based on state (using the -s option) or name. The third set of two command lines in the SYNOPSIS section provides these functions.

The fsmon command accepts the following options:

`-a` Adds specified *filesystems* to the list of monitored file systems. Each file system receives the critical threshold value (specified with the `-c` option), warning threshold value (specified with the `-w` option), and enabled state (specified with the `-e` option) or disabled state (specified with the `-d` option) supplied on the command line. (See those options for defaults and value ranges.) The special file system name `all` implies those file systems found in the kernel mount table, and only file systems not already in the table of monitored file systems are added. It is an error to add a file system that exists in the daemon's table. The `-a` option cannot be used with the `-m` option.

`-c` *nnn* Specifies the critical threshold value. This value specifies the percentage of allocated file system space, which, when reached, triggers the execution of the critical command. (The critical command is specified when `fsdaemon`(8) is initiated.) The value range is 0.0 < *nnn* < 100.0; the default value is 95.0%.

`-d` Disables monitoring of the specified *filesystems*. This option must be used with either the `-a` or `-m` options and cannot be used with the `-e` option. File system monitoring is enabled by default.

`-D` Specifies debug mode. This option may be used with any other option.

`-e` Enables monitoring of the specified *filesystems*. This option is the default; it must be used with either the `-a` or `-m` options and cannot be used with the `-d` option.

`-f` *state* Forces critical (`c`) or warning (`w`) state on the specified *filesystems*. This option is valid only with the `-m` option and causes the same action by the daemon as if the file system had actually reached the critical or warning threshold. If both the `c` and `w` values are supplied, both conditions are forced; however, the daemon acts only on the critical threshold because it checks for that condition first.

`-i` *cw* Specifies that an operator message is sent for the specified state. *cw* takes the following values:

- `c` Specifies that a message is sent for the critical state.
- `w` Specifies that a message is sent for the warning state.
- `cw` Specifies that a message is sent for both the critical and warning states. This is the default action if `-i` and `-n` options are not specified.

`-l` Displays the log file. The current content of the log file is written to the `stdout` file.

`-m` Modifies specified entries in the list of monitored file systems. One or more *filesystems* must be provided. Each file system receives the critical threshold value (specified with the `-c` option), warning threshold value (specified with the `-w` option), and enabled state (specified with the `-e` option) or disabled state (specified with the `-d` option) supplied on the command line. (See those options for value ranges.) Only those options specifically included are changed in the specified file system table entries. The special file system name `all` can be used to modify every entry in the table. If you want to disable monitoring for all file systems, use the following command line:

```
fsmon -m -d all
```

It is an error to name a file system that has not been added to the daemon's table (by using the `-a` option). This option cannot be used with the `-a` option.

`-n` *cw* Specifies that no operator message is sent for the specified state. *cw* takes the following values:

`c` Specifies that no message is sent for the critical state.

`w` Specifies that no message is sent for the warning state.

`cw` Specifies that no message is sent for either the critical or warning state.

`-p` *path* Specifies an alternative path name for the files and pipes used for communication between `fsmon` and `fsdaemon`(8). This option is intended for testing and development when an alternative copy of `fsdaemon`(8) that does not interfere with the installed daemon may be useful. See the `fsdaemon`(8) man page for more information on this capability.

`-q` Terminates `fsdaemon`(8). This option may be enabled by an `fsdaemon`(8) option as elected by the system administrator. A message is displayed if this option has not been accepted by the daemon.

`-R` Resets threshold detection on the specified *filesystems*. When `fsdaemon`(8) starts a critical or warning command, it does not initiate another command on the same file system until a reset has been received. This allows the command to execute without the possibility of another command of the same type interfering with it. (The critical command may start while the warning command is running.) When this option is specified, all internal information concerning the specified *filesystems* is discarded, except the enabled or disabled state and the threshold values, and on the next cycle, `fsdaemon`(8) reevaluates these file systems. This command should be used with care because it removes information as to whether commands are running on the affected file systems.

The special file system name `all` causes `fsdaemon`(8) to reset every file system it knows.

`-r` *state* Removes the selected "command running" state from the specified *filesystems*. Possible states are as follows:

`c` Removes "critical command warning" state.

`w` Removes "warning command running" state.

Either or both of these states may be specified, but at least one must appear.

This form of resetting only removes the command running state or states specified and does not discard other state information. This is the recommended way to end a warning or critical command. The special file system name `all` causes `fsdaemon`(8) to remove the specified state from every file system it knows.

`-s` *state* Shows the status of file systems in the specified *state*. The following states are valid:

`c` Critical threshold detected.

`d` Disabled file system entries.

`e` Enabled file system entries.

`w` Warning threshold detected.

Any combination of states may be specified, but at least one must appear. The default is `cdew`.

When `fsmon` is executed with this option, a status display is returned. The status of the specified file systems is indicated in the display's `Status` column. The following list describes the values that appear in each position in the column. Position one always contains `D`, `E`, or `*`. A position that contains a "`-`" means that the status does not apply to the corresponding file system.

`E-------` Monitoring is enabled.

`D-------` Monitoring is disabled.

`*-------` Monitoring error.

`-C------` Critical threshold detected.

`--c-----` Manual critical forced.

`---X----` Critical command executing.

`----W---` Warning threshold detected.

`-----w--` Manual warning forced.

`------X-` Warning command executing.

`-------?` Internal error.

`-t` Enables terse mode. When this option is specified, `fsmon` does not show display headers or any informative messages.

`-w` *nnn* Specifies warning threshold value. This value specifies the percentage of allocated file system space, which, when reached, triggers the execution of the warning command. (The warning command is specified when `fsdaemon`(8) is initiated.) The value range is 0.0 < *nnn* < 100.0; the default value is 85.0%.

filesystems The special file system name `all` represents all known file system names and can be used in any command line as a value for the *filesystems* operand.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	**Action**
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

FILES

`/usr/spool/fsmonitor/Fd.log`	Active log file
`/usr/spool/fsmonitor/Fsdaemon.pid`	Daemon's process ID
`/usr/spool/fsmonitor/Fsdmn.cmd_to`	Daemon command input pipe
`/usr/tmp/Fm_in`*xxxxx*	Daemon command output pipe

SEE ALSO

`fsdaemon`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`fsmpptest` – MPP multi-PE application file system and disk device test including performance measurement and data comparison

SYNOPSIS

`mpprun -n 4 fsmpptest` [`-b` *bsize*] [`-e` *esize*] [`-i` *isize*] [`-f` *fsize*] [`-m` *mtfer*] [`-s`] [`-r`] [`-p`] [`-S`] [`-A` *nbufs*] [`-a` *nbufs*] [`-B`] [`-R`] [`-E`] [`-X` *cbits*] [`-x` *cblks*] [`-I` *part*] [`-d`] [`-w`] [`-N`] [`-c`] [`-P` *prefx*] [`-n` *npass*] [`-t` *dtype*] [`-u` *diou*] [`-z`] [`-Z`] [`-Y` *fblks*] [`-M` *MHz*] [`-D` *rlnmx*] [`-C` *comnt*] [`-T` *rftag*] [`-o`] [`-V`] *file1 file2 file3 file4* (for a 4 PE application)

IMPLEMENTATION

Cray MPP systems

DESCRIPTION

`fsmpptest` is a multiple PE, multiple file test for MPPs. `fsmpptest` tests a file system's or disk device's basic functionality, data integrity, and performance. The performance of several items are measured and reported.

By default, file system sequential synchronous write, write - preallocated, and read performance are measured for transfer sizes of 4 KBytes to 32 KBytes. The transfer size is incremented by 4 KBytes at a time. The default file sizes are 3,200 KBytes. Each file gets a different data pattern. Each transfer contains the virtual PE number of the PE doing the I/O and also contains the transfer number.

`fsmpptest` is based on `fstest`(8), which is based on `pddtest`(8). The main difference in options between `fsmpptest` and `fstest` are that `fsmpptest` doesn't support random I/O. Random I/O could be added if needed. Also, `-P` specifies the file name prefix, whereas in `fstest` it specifies the data pattern number.

The following areas are tested and measured by `fsmpptest`:

```
stat
unlink
open
ialloc
write
read
close
```

The following information is reported:

- environment information
- write rates
- write - preallocated rates

- read rates
- other times
- system times
- detailed times

`fsmpptest` puts the information it produces into seven files. They are:

`fst.w.out.rftg`	Writes rates
`fst.wp.out.rftg`	Writes - preallocated rates
`fst.r.out.rftg`	Reads rates
`fst.o.out.rftg`	Other times
`fst.s.out.rftg`	System times
`fst.d.out.rftg`	Detailed times
`fst.t.out.rftg`	Terminal output (`stdout` and `stderr`)

OPTIONS

Transfer Size Options

`-b` *bsize* Beginning transfer size in bytes. KBytes (suffixed with a K), 4 KByte blocks (suffixed with a B), 512 byte blocks (suffixed with a b), MBytes (suffixed with a M), M2Bytes (suffixed with a m), GBytes (suffixed with a G), or G2Bytes (suffixed with a g).

`-e` *esize* Ending transfer size in bytes. KBytes (suffixed with a K), 4 KByte blocks (suffixed with a B), 512 byte blocks (suffixed with a b), MBytes (suffixed with a M), M2Bytes (suffixed with a m), GBytes (suffixed with a G), or G2Bytes (suffixed with a g).

`-i` *isize* Transfer size increment in bytes. KBytes (suffixed with a K), 4 KByte blocks (suffixed with a B), 512 byte blocks (suffixed with a b), MBytes (suffixed with a M), M2Bytes (suffixed with a m), GBytes (suffixed with a G), or G2Bytes (suffixed with a g).

`-f` *fsize* File size which determines the total number of transfers. This can be in bytes; KBytes (suffixed with a K), 4 KByte blocks (suffixed with a B), 512 byte blocks (suffixed with a b), MBytes (suffixed with a M), M2Bytes (suffixed with a m), GBytes (suffixed with a G), or G2Bytes (suffixed with a g).

`-m` *mtfer* If set, the maximum number of transfers to do at each transfer size.

I/O Type Options

`-s` Sequential I/O.

`-r` Random I/O. Not supported.

`-p` Pipe I/O. Not supported.

`-S` Synchronous I/O.

`-A` *nbufs* Asynchronous I/O using recalls specifying the number of buffers to use. Not supported.

`-a` *nbufs* Asynchronous I/O using signals specifying the number of buffers to use. Not supported.

`-B` Library buffered I/O. Not supported.

I/O Options

`-R` Raw I/O `O_RAW | O_WELLFORMED`.

`-E` SFS file exclusive open (`O_SFSXOP | O_SFS_DEFER_TM`).

`-X` *cbits* Preallocation bit mask specifying which partition or partitions to put the file on. This facilitates user striping of a file. This should probably be changed so all files aren't striped on the same partitions.

`-x` *cblks* Number of 4 KByte blocks to allocate per partition for the file. This can be used to preallocate a file or to determine the stripe factor for user striping.

`-I` *part* File preallocation using `ialloc` starting at the specified partition with the partition number being incremented by one for each additional file.

`-d` Disk device I/O (to a `/dev` device). Do I/O to a character or block special file *iofile*. Only reads are done unless `-w` is specified to do writes. When selecting this option, stats, unlinks, and preallocated writes are not done. This option can be selected to do single direction I/O to a disk device or file system. To do just reads use `-d` and to do just writes use `-wN`.

`-w` For disk device I/O, do writes as well as reads.

`-N` For disk device I/O, don't do reads.

Other Options

`-c` Compare data. Compares data written to data read. The first and last words of the transfer contain the virtual PE number of the PE doing the I/O. The second and second to last words of the transfer contain the transfer file position for that transfer. This option will affect read overall transfer rates - use the read "Overall Transfer Rate" following the `dc` read or the "Mean Individual Transfer Rates" in the detailed times report when evaluating performance with this option selected. The file can be evaluated using `od -t o8` *iofile*.

`-P` *prefx* File name prefix. All files are prefixed with this character string.

`-n` *npass* Number of passes through the test.

`-t` *dtype* Disk type to use to determine sizes. This is specified as DD*nnn* or ND*nn*. The disk overall size will be used except if `-f` or `-m` is specified, then one of these will be used. Optimal size transfers (usually cylinder) will be done if `-u` is not specified. Look at `diskspec.h` for the disk types available.

`-u` *diou* Disk I/O unit specifier to use for sizes. Use the first letter of sector, track, cylinder, Parity group, Stripe width, or optimal I/O unit to specify this.

`-z` For DFS - close the file after writing it to sync up the I/O. Only use the "Total Transfer Rates" when evaluating performance with this option selected.

`-Z` For DFS - flush the DFS cache before reading the file. Note that all PEs will do this.

`-Y` *fblks* Flush the system buffer cache before reading the file. If set to 1, 10,000 blocks are flushed per PE. If set to a number greater than 1, this many blocks of cache are flushed per PE. Flushing just the size of the cache instead of the default 10,000 blocks saves time. To determine the size of the system buffer cache and thus how many blocks to flush use `sysconf |grep cache`. If the test is only utilizing one File Server, then divide the cache size by the number of PEs doing I/O to flush the cache as quickly as possible.

`-M` *MHz* CPU clock in MHz - overrides the OS value.

`-D` *rlnmx* If set to 0, turns off the detailed I/O timings report.

`-C` *comnt* Comment describing what's being tested.

`-T` *rftag* Result files name tag.

`-o` Don't print output to the terminal. All output will go to just the `fst.t.out*` file.

`-V` Revision level.

file1 file2 file3 file4
File names to test (for a 4 PE application).

NOTES

In general, for doing performance testing, `fsmpptest` should be run on a system without other activity on it.

Data comparison affects read overall transfer rates - use the read "Total Transfer Rate" following the `dc` read or the "Mean Individual Transfer Rates" in the detailed times report when evaluating performance if data comparison is done. The results including data comparison are noted with a `dc`. Reads with data comparison include the overhead of data comparison.

To print `fsmpptest` result files use:

```
enscript -Br -f Courier8 fsmppt.*.out
```

`fsmpptest` result files can be easily graphed using the Wingz (HSTools) spreadsheet on Sun systems. They can be preprocessed for graphing using the `fsedgraph` command.

`fsmpptest` supports the following size suffixes:

K KByte = 1,024 bytes

B Block = 4,096 bytes

b block = 512 bytes

M MByte = $1{,}000^2$ bytes

m M2Byte = $1{,}024^2$ bytes

G GByte = $1{,}000^3$ bytes

g G2Byte = $1{,}024^3$ bytes

EXAMPLES

Example 1: Do a quick performance test of a file system using a 4 PE test:

```
mpprun -n 4 fsmpptest /fs/file1 /fs/file2 /fs/file3 /fs/file4
```

Example 2: Do a quick performance test of a file system using a 4 PE test with data comparison and prefixing the file names with /*tmp*/:

```
mpprun -n 4 fsmpptest -c -P /tmp/ f1 f2 f3 f4
```

Example 3: Do a four file performance test making sure not to overflow the system buffer cache by specifying a file size maximum of 250 blocks:

```
mpprun -n 4 fsmpptest -f 250B -P /tmp/ f1 f2 f3 f4
```

Example 4: Test a file system using a 4 PE test doing raw I/O with transfers ranging from one sector to 32 KBytes against file sizes of 3200 KBytes doing sequential synchronous I/O. Also put a comment at the top of each report file explaining what was tested and tag the report file names with a time stamp. In addition, prefix the files with the file system.

```
mpprun -n 4 fsmpptest -R -b 4096 -e 32768 -i 4096 -f 3276800 -s -S x
-C "Cray File System Raw I/O Test" -T ".‘date +%H%M‘" x
-P /tmp/ f1 f2 f3 f4
```

Example 5: Test a file system using an 8 PE test doing raw I/O with transfers ranging from 1/4 M2Byte to 1 M2Byte against file sizes of 100 M2Bytes doing sequential synchronous I/O. Test only 100 transfers at each transfer size. Also put a comment at the top of each report file explaining what was tested and tag the report file names. In addition, prefix the files with the file system.

```
mpprun -n 8 fsmpptest -R -b 256K -e 1m -i 256K -f 100m -m 100 x
-C "Large File Raw I/O Test" -T ".lrg" x
-P /fs/ f1 f2 f3 f4 f5 f6 f7 f8
```

Example 6: Test a file system using a 4 PE test doing raw I/O with transfers ranging from 32 KBytes to 256 KBytes against file sizes of 25600 KBytes doing sequential synchronous I/O. Also put a comment at the top of each report file explaining what was tested and tag the report file names. In addition, prefix the files with the file system.

```
mpprun -n 4 fsmpptest -R -b 32K -e 256K -i 32K -f 25600K x
-C "Medium File Raw I/O Test" -T ".med" x
-P /fs/ f1 f2 f3 f4
```

Example 7: Test the performance of a file system using a 4 PE test doing raw one M2Byte transfers:

```
mpprun -n 4 fsmpptest -R -b 1m -e 1m -f 100m x
-C "One MByte Raw Transfers Test" -T ".1mb" x
-P /fs/ f1 f2 f3 f4
```

Example 8: Do a quick performance test of four disk partitions including writes and data comparison using a 4 PE test - note that this test will over-write data on these disk partitions:

```
mpprun -n 4 fsmpptest -dwc -P "/dev/xdd/" dd314.0 dd314.1 dd314.2 dd314.3
```

Example 9: Do a quick performance test of four disk partitions at the block device level including writes using a 4 PE test - note that this test will over-write data on these disk partitions:

```
mpprun -n 4 fsmpptest -dw -P /dev/dsk/ dd314.0 dd314.1 dd314.2 dd314.3
```

Example 10: Run `fsmpptest` over night comparing the data using a 16 PE test:

```
mpprun -n 16 fsmpptest -c -n 1000000 -P /fs/ f1 f2 f3 f4 f5 f6 f7 f8 \
f9 f10 f11 f12 f13 f14 f15 f16
```

Example 11: Test a file system using a 2 PE test and printing no output to the terminal:

```
mpprun -n 2 fsmpptest -c -n 1000000 -T .2pes -o /fs/file1 /fs/file2 &
```

Example 12: Test four whole DA-301s using track transfers - note that this test will over-write data on these disks:

```
mpprun -n 4 fsmpptest -dwc -t DA301 -u t x
-C "Four DA-301 Disk Devices Test" -T ".da301s" x
-P /dev/pdd/ da301.0.all da301.1.all da301.2.all da301.3.all
```

Example 13: Test four new DD-314s in a manner similar to `pddtest` using a 4 PE test:

```
mpprun -n 4 fsmpptest -dwc -t DD314 -n 1000000 x
-P /dev/xdd/ dd314.0 dd314.1 dd314.2 dd314.3
```

Example 14: Run `fsmpptest` over night comparing the data to test 100,000 block slices of DD-314s including writes using transfer sizes ranging from one block to one track using a 4 PE test:

```
mpprun -n 4 fsmpptest -dwc -b 1B -e 17B -i 1B -f 100000B -n 1000000 n
-C "DD-314s Test" -T ".‘date +%H%M‘" n
-P /dev/xdd/ dd314.0 dd314.1 dd314.2 dd314.3
```

Example 15: Test a DFS file system using 4 files and a 4 PE test:

```
mpprun -n 4 fsmpptest -T .dfs -P "/:/cray/cool/ptmp/" f1 f2 f3 f4
```

Example 16: Test a DFS file system closing the file after writing it to sync up the I/O using a 4 PE test. Only use the "Total Transfer Rates" when evaluating the reports in this case.

```
mpprun -n 4 fsmpptest -z -T .dfsz -P "/:/cray/cool/ptmp/" f1 f2 f3 f4
```

Example 17: Test a DFS file system flushing the system buffer cache and the DFS cache before reading the file using a 4 PE test:

```
mpprun -n 4 fsmpptest -Y 500 -Z -T .dfsYZ -P "/:/cray/cool/ptmp/" f1 f2 f3 f4
```

Example 18: Test a NFS file system using an 8 PE test:

```
mpprun -n 8 fsmpptest -T .nfs -P "/cray/cool/ptmp/" f1 f2 f3 f4 f5 f6 f7 f8
```

Example 19: Test a NFS file system flushing the 2,000 block system buffer cache before reading the files using a 4 PE test:

```
mpprun -n 4 fsmpptest -Y 500 -T .nfsY -P /cray/frost/tmp/ f1 f2 f3 f4
```

SEE ALSO

`fstest`(8)

NAME

`fsoffload` – Lists files and directories on a logical device

SYNOPSIS

`/etc/fsoffload` [`-a`] [`-c`] [`-d`] [`-D`] [`-f` *pathname*] [`-h`] [`-p` *pdev*] [`-r`] [`-v`] *ldev*
`/etc/fsoffload` [`-a`] [`-c`] [`-d`] [`-D`] [`-f` *pathname*] [`-h`] [`-m`] [`-M`] [`-r`] [`-s` *slice*[`,`*slice*`,...`]] [`-v`] *ldev*

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `fsoffload` command lists files and directories that are resident or partially resident on particular slices of a logical device. A file is considered resident if its inode or any of its data reside on one of the specified logical device components.

The `fsoffload` command accepts the following options and operand:

`-a` Lists all files, directories, and special files found residing on the specified logical device components. The `-r` and `-d` options do not have to be specified with this option, because it performs their functions.

`-c` Lists all children and descendants of directories that are resident on the specified logical device components.

`-d` Lists all directories on the specified logical device components.

`-D` Debug mode; prints additional information about file residency.

`-f` *pathname*
Specifies a file or directory to be searched for files residing on the specified logical device components. If this option is not specified, the mount point found in the kernel mount table is assumed.

`-h` Prints help message.

`-m` Rebuilds files. This option causes files resident on the specified slices to be copied (to their original names). If the specified slices are on a read-only disk, the files are moved off the disk. This option alone does not list the files that are copied; if you wish a list of the files, use this option in conjunction with an option that provides a list, such as `-a`.

`-M` Rebuilds directories. This option causes directories resident on the specified slice(s) to be rebuilt. If the specified slices are on a read-only disk, the directories are moved off the disk.

`-p` *pdev* Specifies a physical device; *pdev* should have a component composing part of the logical device. *pdev* specifies a component name as found in the superblock. You cannot use this option with the `-s` option.

`-r` Lists all regular files on the specified logical device components.

`-s` *slice*[`,` *slice*`,` ...]
Specifies a number representing a particular *slice* of a logical device, as represented in the superblock. The slice value is a base-10 slice number. You can specify multiple values, separated by commas. This option cannot be used with the `-p` option.

`-v` Displays the components of the specified logical device, showing which components are marked for residency search.

ldev A logical device; *ldev* must be a mounted file system. This operand is required.

NOTES

This command can be used to observe sensitive data on a device. Only appropriately authorized users can use this command.

If this command is installed with a privilege assignment list (PAL), a user who is assigned the following privilege text upon execution of this command is allowed to perform the action shown:

Privilege Text	Action
`showall`	Allowed to use this command.

If this command is installed with a PAL, a user with one of the following active categories is allowed to perform the action shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

SEE ALSO

`pddconf`(8), `pddstat`(8)

General UNICOS System Administration, Cray Research publication SG–2301

NAME

fstest – Tests file systems and disk devices

SYNOPSIS

fstest [-b *bsize*] [-e *esize*] [-i *isize*] [-f *fsize*] [-m *mtfer*] [-s] [-r] [-p] [-S] [-A *nbufs*] [-a *nbufs*] [-B] [-R] [-E] [-K] [-X *cbits*] [-x *cblks*] [-I *part*] [-d] [-w] [-N] [-c] [-P *pattn*] [-n *npass*] [-t *dtype*] [-u *diou*] [-v] [-z] [-Z] [-Y *fblks*] [-M *MHz*] [-D *rlnmx*] [-C *comnt*] [-T *rftag*] [-o] [-V] *iofile*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

fstest tests a file system's or disk device's basic functionality, data integrity, and performance. The performance of several items are measured and reported. fstest is a single process, single file test.

By default, file system sequential synchronous write, write - preallocated, and read performance are measured for transfer sizes of 4 KBytes to 32 KBytes. The transfer size is incremented by 4 KBytes at a time. The default file size is 3,200 KBytes. The default data pattern is data pattern 0.

fstest is based on pddtest(8). fstest is similar to pddtest with timers that can also be used at the file system level. It has different defaults and command line options, however. The main item missing from fstest which pddtest has is run on error. This could be added if needed. Other differences are fstest can do random I/O without doing sequential I/O first and can also test a range of transfer sizes. fstest numbers each transfer instead of numbering each sector like pddtest.

The following areas are tested and measured by fstest:

```
stat
unlink
open
ialloc
write
read
close
```

The following information is reported:

- environment information
- write rates
- write - preallocated rates
- read rates
- other times

- system times
- detailed times (optional)

`fstest` puts the information it produces into seven files. They are:

`fst.w.out.rftg`	Writes rates
`fst.wp.out.rftg`	Writes - preallocated rates
`fst.r.out.rftg`	Reads rates
`fst.o.out.rftg`	Other times
`fst.s.out.rftg`	System times
`fst.d.out.rftg`	Detailed times (optional)
`fst.t.out.rftg`	Terminal output (`stdout` and `stderr`)

OPTIONS

Transfer Size Options

`-b` *bsize* Beginning transfer size in bytes. KBytes (suffixed with a K), 4 KByte blocks (suffixed with a B), 512 byte blocks (suffixed with a b), MBytes (suffixed with a M), M2Bytes (suffixed with a m), GBytes (suffixed with a G), or G2Bytes (suffixed with a g).

`-e` *esize* Ending transfer size in bytes. KBytes (suffixed with a K), 4 KByte blocks (suffixed with a B), 512 byte blocks (suffixed with a b), MBytes (suffixed with a M), M2Bytes (suffixed with a m), GBytes (suffixed with a G), or G2Bytes (suffixed with a g).

`-i` *isize* Transfer size increment in bytes. KBytes (suffixed with a K), 4 KByte blocks (suffixed with a B), 512 byte blocks (suffixed with a b), MBytes (suffixed with a M), M2Bytes (suffixed with a m), GBytes (suffixed with a G), or G2Bytes (suffixed with a g).

`-f` *fsize* File size which determines the total number of transfers. This can be in bytes; KBytes (suffixed with a K), 4 KByte blocks (suffixed with a B), 512 byte blocks (suffixed with a b), MBytes (suffixed with a M), M2Bytes (suffixed with a m), GBytes (suffixed with a G), or G2Bytes (suffixed with a g).

`-m` *mtfer* If set, the maximum number of transfers to do at each transfer size.

I/O Type Options

`-s` Sequential I/O.

`-r` Random I/O. For random I/O, the file is only created once.

`-p` Pipe I/O. Not supported.

`-S` Synchronous I/O.

`-A` *nbufs* Asynchronous I/O using recalls specifying the number of buffers to use. Not supported.

`-a` *nbufs* Asynchronous I/O using signals specifying the number of buffers to use. Not supported.

`-B` Library buffered I/O. Not supported.

I/O Options

`-R` Raw I/O (`O_RAW | O_WELLFORMED`).

`-E` SFS file exclusive open (`O_SFSXOP | O_SFS_DEFER_TM`).

`-X` *cbits* Preallocation bit mask specifying which partition or partitions to put the file on. This facilitates user striping of a file.

`-x` *cblks* Number of 4 KByte blocks to allocate per partition for the file. This can be used to preallocate a file or to determine the stripe factor for user striping.

`-I` *part* Preallocate the file on the specified partition using `ialloc`.

`-d` Disk device I/O (to a `/dev` device). Do I/O to a character or block special file *iofile*. Only reads are done unless `-w` is specified to do writes. When selecting this option, `stat`s, `unlink`s, and preallocated writes are not done. This option can be selected to do single direction I/O to a disk device or file system. To do read operations only use `-d` and to do write operations only use `-wN`.

`-w` For disk device I/O, do writes as well as reads.

`-N` For disk device I/O, don't do reads.

Other Options

`-c` Compare data. Compares data written to data read. The first and last words of the transfer contain the transfer file position for that transfer. This option will affect read overall transfer rates - use the read "Overall Transfer Rate" following the `dc` read or the mean from the "Individual Transfer Rates" when evaluating performance with this option selected. The file can be evaluated using `od -t o8` *iofile*.

`-P` *pattn* Data pattern number to use. This is a number from 0 - 9 since there are 10 different data patterns to choose from.

`-n` *npass* Number of passes through the test. For each pass through the test, the data pattern is incremented by one.

`-t` *dtype* Disk type to use to determine sizes. This is specified as DD*nnn* or ND*nn*. The disk overall size will be used except if `-f` or `-m` is specified, then one of these will be used. Optimal size transfers (usually cylinder) will be done if `-u` is not specified. Look at `diskspec.h` for the disk types available.

`-u` *diou* Disk I/O unit specifier to use for sizes. Use the first letter of sector, track, cylinder, Parity group, Stripe width, or optimal I/O unit to specify this.

`-v` Vary I/O - vary between sequential and random I/O. Sequential I/O is done one pass and then random I/O is done the next pass.

`-z` For DFS - close the file after writing it to sync up the I/O. Only use the "Overall Transfer Rates" when evaluating performance with this option selected.

`-Z`	For DFS - flush the DFS cache before reading the file.
`-Y` *fblks*	Flush the system buffer cache before reading the file. If set to 1, 10,000 blocks are flushed. If set to a number greater than 1, this many blocks of cache are flushed. Flushing just the size of the cache instead of the default 10,000 blocks saves time. To determine the size of the system buffer cache and thus how many blocks to flush use `sysconf |grep NBUF`.
`-M` *MHz*	CPU clock in MHz - overrides the OS value.
`-D` *rlnmx*	Detailed I/O timings report. If set to 1, all individual transfer times are reported. If set to a number greater than 1, it reports this many lines worth of individual transfer times for each operation.
`-C` *comnt*	Comment describing what's being tested.
`-T` *rftag*	Result files name tag.
`-o`	Don't print output to the terminal. All output will go to just the `fst.t.out*` file.
`-V`	Revision level.
iofile	File name to test.

NOTES

In general, for doing performance testing, `fstest` should be run on a system without other activity on it. When testing the random I/O performance of a disk, the entire disk should be used for the test.

Data comparison affects read overall transfer rates - use the read "Overall Transfer Rate" following the `dc` read or the mean from the "Individual Transfer Rates" when evaluating performance if data comparison is done. The results including data comparison are noted with a `dc`. Reads with data comparison include the overhead of data comparison.

To print `fstest` result files use:

```
enscript -Br -f Courier8 fst.*.out
```

`fstest` result files can be easily graphed using the Wingz (HSTools) spreadsheet on Sun systems. They can be preprocessed for graphing using the `fsedgraph` command.

`fstest` supports the following size suffixes:

K	KByte = 1,024 bytes
B	Block = 4,096 bytes
b	block = 512 bytes
M	MByte = $1{,}000^2$ bytes
m	M2Byte = $1{,}024^2$ bytes
G	GByte = $1{,}000^3$ bytes
g	G2Byte = $1{,}024^3$ bytes

EXAMPLES

Example 1: Do a quick performance test of a file system:

```
fstest /fs/iofile
```

Example 2: Do a quick performance test of a file system with data comparison:

```
fstest -c /tmp/iofile
```

Example 3: Test a file system with transfers ranging from one sector to 32 KBytes against a file size of 3200 KBytes doing sequential synchronous I/O. Also report the first 10 lines of individual I/O timings for each operation and put a comment at the top of each report file explaining what was tested. In addition, tag the report file names with a time stamp.

```
timex fstest -b 4096 -e 32768 -i 4096 -f 3276800 -s -S -D 10 x
-C "Cray File System Test" -T ".‘date +%H%M‘" x
/tmp/iofile
```

Example 4: Test a file system with transfers ranging from 1/4 M2Byte to 1 M2Byte against a file size of 100 M2Bytes doing sequential synchronous I/O. Test only 100 transfers at each transfer size. Also report the first 10 lines of individual I/O timings for each operation and put a comment at the top of each report file explaining what was tested. In addition, tag the report file names.

```
fstest -b 256K -e 1m -i 256K -f 100m -m 100 -s -S -D 10 x
-C "Large File System Test" -T ".lrg" x
/fs/iofile
```

Example 5: Test a file system with transfers ranging from 32 KBytes to 256 KBytes against a file size of 25600 KBytes doing sequential synchronous I/O. Also report the first 10 lines of individual I/O timings for each operation and put a comment at the top of each report file explaining what was tested. In addition, tag the report file names.

```
fstest -b 32K -e 256K -i 32K -f 25600K -s -S -D 10 x
-C "Medium File System Test" -T ".med" x
/fs/iofile
```

Example 6: Test the performance of a file system using one M2Byte transfers:

```
fstest -b 1m -e 1m -f 100m -s -S -D 10 x
-C "One MByte Transfers File System Test" -T ".1mb" x
/fs/iofile
```

Example 7: Do random I/O to a disk in a file system:

```
fstest -r /fs/iofile
```

Example 8: Test the raw I/O performance of a file system:

```
fstest -R -b 4B -e 32B -i 4B -f 3200B x
-C "Raw I/O Test" -T ".raw" /tmp/iofile
```

Example 9: Test the raw I/O performance of a file striped across 4 file system partitions using a stripe factor of a track (48 blocks for this example). Check which file system partitions are available by doing `df -p /fs`. For this example the file system consists of 5 partitions, 0 - 4. Use partitions 1 - 4 of the file system for this test. The bit mask to select these partitions is determined as follows:

$2^1 + 2^2 + 2^3 + 2^4 = 30$

The bit mask could also have been determined by using:

$(2^5 - 1) - 2^0$

Do transfers of 4 tracks (192 blocks) to a 400 track file.

```
fstest -R -X 30 -x 48 -b 192B -e 192B -f 19200B /fs/iofile
```

Example 10: Do random raw I/O to a DD-302 in a file system. Only do 1,000 random transfers at each transfer size. To determine the file size use df to determine the amount of free space.

```
fstest -rR -f 395920B -m 1000 x
-C "DD-302 File System Random Raw I/O Test" -T ".ranraw" x
/fs/iofile
```

Example 11: Do a quick performance test of a disk partition including writes and data comparison - note that this test will over-write data on this disk partition:

```
fstest -dwc /dev/pdd/dd302
```

Example 12: Do a quick performance test of a disk partition at the block device level including writes - note that this test will over-write data on this disk partition:

```
fstest -dw /dev/dsk/dd302
```

Example 13: Run `fstest` over night comparing the data and varying the I/O between sequential and random to test a file system (the data pattern will also vary):

```
fstest -c -v -n 1000000 /fs/iofile
```

Example 14: Test a file system using two processes with each using a different data pattern and printing no output to the terminal:

```
fstest -c -n 1000000 -T .p1 -o /fs/iofile &
fstest -c -P 1 -n 1000000 -T .p2 -o /fs/iofile2 &
```

Example 15: Test a whole DA-301 using track transfers - note that this test will overwrite data on this disk:

```
fstest -dwc -t DA301 -u t x
-C "DA-301 Disk Device Test" -T ".da301" x
/dev/pdd/da301.all
```

Example 16: Test a new DD-302 as you would using `pddtest`:

```
fstest -dwcv -t DD302 -n 1000000 /dev/pdd/dd302
```

Example 17: Test the performance of a four disk file system using four files. Only write the files once so writes don't dominate the test. Use a large enough file size so the test runs long enough to get meaningful results. Evaluate the run using the mean of the "timex real time" results or the "`fst.s.out` pass total elapsed time" results.

Put the following in a script and run it:

```
# four file test using a transfer size of one block

TAG=".‘date +%H%M%S‘"

timex fstest -dw -b 4096 -e 4096 -f 4096000 -s -S x
-C "Four-File File System Test - File 1" -T ".1$TAG" -o x
/fs/iofile1 &

timex fstest -dw -b 4096 -e 4096 -f 4096000 -s -S x
-C "Four-File File System Test - File 2" -T ".2$TAG" -o x
/fs/iofile2 &

timex fstest -dw -b 4096 -e 4096 -f 4096000 -s -S x
-C "Four-File File System Test - File 3" -T ".3$TAG" -o x
/fs/iofile3 &

timex fstest -dw -b 4096 -e 4096 -f 4096000 -s -S x
-C "Four-File File System Test - File 4" -T ".4$TAG" -o x
/fs/iofile4 &

wait

# end of four file test
```

Example 18: Run `fstest` overnight, comparing the data and varying the I/O between sequential and random to test a DD-314 including writes (the data pattern will also vary and cylinder transfers will be done):

```
fstest -dwcv -t DD314 -n 1000000 /dev/xdd/dd314
```

Example 19: Run `fstest` overnight, comparing the data and varying the I/O to test a 100,000 block slice of a DD-314 including writes using transfer sizes ranging from one block to one track:

```
fstest -dwcv -b 1B -e 17B -i 1B -f 100000B -n 1000000 x
-C "DD-314 Test" -T ".‘date +%H%M‘" /dev/xdd/dd314
```

Example 20: Test a DD-314 including writes doing track random I/O and data comparison:

```
fstest -dwc -r -t DD314 -u t /dev/xdd/dd314
```

Example 21: Test a DFS file system:

`fstest -T .dfs /:/cray/cool/`*ptmp/iofile*

Example 22: Test a DFS file system closing the file after writing it to sync up the I/O. Only use the "Overall Transfer Rates" when evaluating the reports in this case.

`fstest -z -T .dfsz /:/cray/cool/`*ptmp/iofile*

Example 23: Test a DFS file system flushing the system buffer cache and the DFS cache before reading the file:

`fstest -Y 2000 -Z -T .dfsYZ /:/cray/cool/`*ptmp/iofile*

Example 24: Test a NFS file system:

`fstest -T .nfs /cray/cool/`*ptmp/iofile*

Example 25: Test a NFS file system flushing the system buffer cache before reading the file:

`fstest -Y 2000 -T .nfsY /cray/frost/`*tmp/iofile*

Example 26: Test the raw I/O performance of a preallocated file on partition 1 of a file system

`fstest -R -I 1 /`*fs/iofile*

Example 27: Test a ND-40 including writes doing data comparison and varying the I/O using optimal I/O transfers:

```
fstest -dwcv -t ND40_64KR5 /dev/hdd/nd40.1.h1.b1
```

Example 28: Test a ND-40 in a file system doing raw I/O, data comparison, varying the I/O between sequential and random, using optimal size I/O transfers, and preallocating the file on partition 0 (to determine the file size use df to determine the amount of free space):

`fstest -Rcv -t ND40_64KR5 -f 23034352B -I 0 /nd40/`*iofile*

Example 29: Do a quick test of some new ND-40 nodes to make sure they work - note that this test will overwrite data on these disk slices:

```
fstest -dw -t ND40R_R5 -f 10000B /dev/hdd/nd40.p11.f16.1
fstest -dw -t ND40R_R1 -f 1000B /dev/hdd/nd40.p21.f32.1
fstest -dw -t ND40R_R1 -f 1000B /dev/dsk/nd40test
```

SEE ALSO

fsmpptest(8)

NAME

ftpd – Invokes the Internet file transfer protocol server

SYNOPSIS

/etc/ftpd [-d] [-h] [-k] [-l] [-v] [-r[0|1]] [-R] [-s *shift*] [-t *timeout*] [-T *seconds*] [-u*mask*] [-Sc *tos*] [-Sd *tos*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The ftpd command invokes the Internet file transfer protocol server process. The server uses transmission control protocol (TCP) and listens at the port specified in the ftp service specification; see services(5).

The ftpd command invokes the centralized identification and authorization library routines to validate the user ID and password.

The ftpd command accepts the following options:

-d Logs debugging information to syslog. (Identical to the -v option.)

-h Disables printing of host-specific information before the user is validated.

-k Allows unknown hosts (for example, those not found in the host's name database) to access Cray Research systems through ftp when security is enabled. By default, hosts that do not have an entry in the host's name database cannot transfer into a secure UNICOS system.

-l Logs general information to syslogd(8).

-v Logs debugging information to syslog. (Identical to the -d option.)

-r[0|1] Enables or disables raw I/O to files. If you specify -r0, raw I/O is disabled; if you specify -r or -r1, it is enabled. If you omit the -r option, raw I/O is enabled by default.

-R Allows a user to send a file to any command, rather than to just a disk file, and it allows the output of any command to be retrieved. You should not enable this option unless you fully understand the ramifications of it; usually, users are allowed only to send and retrieve files. Because this option allows them to execute commands, you might be extending more privileges than you intend.

-s *shift* Sets the default value for the TCP window scale option. If *shift* is on, it is enabled, with a value of 4. If you omit the -s option, this is the default mode. If *shift* is off, it disables the TCP window scale option. If *shift* is a value between 0 and 14, it enables the TCP window scale option with that value.

-t *timeout*
Sets the inactivity time-out period to *timeout* seconds. The default is 15 minutes.

`-T` *seconds* Sets the maximum number of seconds to which the inactivity time-out period can be set. The default is 7200 seconds (2 hours).

`-u`*mask* Sets the default umask to *mask*. The default is 027.

`-Sc` *tos* Sets the IP type-of-service option for the FTP control connection to the value *tos*, which can be a numeric TOS value or a symbolic TOS name found in the `/etc/iptos` file.

`-Sd` *tos* Sets the IP type-of-service option for the FTP data connection to the value *tos*, which can be a numeric TOS value or a symbolic TOS name found in the `/etc/iptos` file.

`-S` *tos* Sets the IP type-of-service option for both the FTP control connection and the FTP data connection to the value *tos*, which can be a numeric TOS value or a symbolic TOS name found in the `/etc/iptos` file.

Currently, the `ftp` server supports the following `ftp` requests (case is not distinguished):

Request	**Description**
`ABOR`	Aborts the previous command (vacuously).
`ADAT`	Base 64 encoded authentication data.
`ALLO`	Allocates storage (vacuously).
`APPE`	Appends to a file.
`AUTH`	Identifies a supported authentication mechanism.
`CDUP`	Changes to parent of current working directory.
`CWD`	Changes the working directory.
`DELE`	Deletes a file.
`ENC`	Privacy protected command. The argument field is a base 64 encoded Telnet string. The server decodes the string and verifies its integrity. The resulting string is interpreted as an FTP command.
`HELP`	Gives help information.
`LIST`	Lists files in a directory (`ls -lg`).
`MIC`	Integrity protected command. The argument field is a base 64 encoded Telnet string. The server decodes the string and verifies its integrity. The resulting string is interpreted as an FTP command.
`MKD`	Makes a directory.
`MODE`	Specifies the data transfer *mode*.
`NLST`	Gives a name list of files in a directory (`ls`).
`NOOP`	Does nothing.
`PASS`	Specifies a password.

PASV	Prepares for server-to-server transfer.
PBSZ	Maximum number of bytes in buffer for protected file transfer.
PORT	Specifies a data connection port.
PROT	Protection level.
PWD	Prints current working directory.
QUIT	Terminates the session.
REST	Restarts the command.
RETR	Retrieves a file.
RMD	Removes a directory.
RNFR	Specifies the *rename-from* file name.
RNTO	Specifies the *rename-to* file name.
SITE CHMOD	Changes the mode on a file.
SITE COPYBUF	Sets copy buffer size.
SITE FULLBUF	Toggles use of full buffers on writes to disk. By default, the buffer is filled before writing to disk begins.
SITE HELP	Gives help for the SITE commands.
SITE IDLE	Gets and sets the inactivity time-out period.
SITE RAWBUF	Toggles whether raw I/O will be used.
SITE SHOWBUF	Shows buffer sizes on transfers.
SITE SOCKBUF	Sets socket buffer size on data socket.
SITE UMASK	Gets and sets the umask value.
SITE WINSHIFT	Gets the status of the use of the TCP window scale option. It can also be used to enable or disable the option. By default, it is enabled with a value of 4.
STAT	Displays status of connection to a file.
STOR	Stores a file.
STOU	Stores a file and gives it a unique name.
STRU	Specifies the data transfer *structure*.
SYST	Gives operating systems information.
TYPE	Specifies the data transfer *type*.
USER	Specifies the user name.
XCUP	Changes to the parent of the current working directory.

XCWD	Changes the working directory.
XMKD	Makes a directory.
XPWD	Prints the current working directory.
XRMD1	Removes a directory.

The remaining ftp requests specified in Internet RFC 959 are recognized but not implemented. The following ftp requests are not specified in RFC 959:

Unimplemented, obsolete:	MAIL, MLFL, MRCP, MRSQ, MSAM, MSND, and MSOM
Implemented, obsolete:	XCUP, XCWD, XMKD, XPWD, and XRMD
Implemented, forthcoming:	MDTM and SIZE

The ftpd command interprets file names the same way the glob command of ftp(1B) does. This lets you use the *, ?, [,], {, }, and ~ metacharacters.

The ftpd command authenticates users according to the following four rules:

1. The user name must be in the /etc/udb file (see udb(5)) and must have a password (that is, it must not have a null password). In this case, the client must provide a password before any file operations can be performed.
2. The user name must not appear in the /etc/ftpusers file (see ftpusers(5)).
3. The user must have a standard shell (see shells(5)).
4. If the user name is anonymous or ftp, an anonymous ftp account must be present in the /etc/udb file (user ftp). In this case, the user is allowed to log in by specifying any password (by convention, this is given as the user's name).

In the last case, ftpd takes special measures to restrict the client's access privileges. The server performs a chroot(2) system call to the home directory of user ftp. To avoid breaching system security, you must construct the ftp directory with care. The following rules are recommended:

~ftp	The ftp home directory should be owned by root and unwritable by anyone.
~ftp/bin	This directory should be owned by the super user and unwritable by anyone. The ls(1) program must be present to support the list commands. This program should have mode 111.
~ftp/etc	This directory should be owned by the super user and unwritable by anyone. The udb(5) file must be present for the ls command to work properly. This file should be mode 444.
~ftp/pub	This directory should have mode 777 and be owned by ftp. You should then place files that are accessible to the anonymous account in this directory.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

`ftpd` login requests are recorded in the audit log.

BUGS

The `anonymous` account is a security risk and should be avoided.

The server must run with appropriate privilege to create sockets with privileged port numbers. (A *socket* is a bidirectional structure within the host that sends and receives packets.) It maintains an effective user ID of the logged-in user and uses nonprivileged port numbers for the data connection.

FILES

`/etc/config/spnet.conf`	File that contains a list of individuals and groups that are allowed access to the UNICOS system from a given host or workstation and the services allowed at that workstation. It also contains the minimum and maximum security levels and the minimum and maximum compartments for the client host and workstation.
`/etc/ftpusers`	File that contains the names of users who are denied access to `ftp` from a remote host
`/etc/inetd.conf`	Default configuration file for the `inetd` daemon
`/etc/shells`	File that contains a list of shells that are associated with user accounts
`$HOME/.netrc`	User-created file that consists of the `ftp` user authentication table

SEE ALSO

spnet(8), syslogd(8)

ftp(1B), login(1), privtext(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

chroot(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

ia_failure(3C), ia_mlsuser(3C), ia_success(3C), ia_user(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

ftpusers(5), group(5), netrc(5), services(5), shells(5), udb(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

RFC 959 *File Transfer Protocol (FTP)*

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

`fuser` – Identifies processes using a file or file structure

SYNOPSIS

`/etc/fuser` [`-k`] [`-m`] [`-u`] [`-c`] *files* [[`-`] [`-k`] [`-u`] [`-c`] *files*]
`/etc/fuser` [`-s`] [`-k`] [`-u`] [`-c`] [*major*] *minor* [[`-`] [`-k`] [`-u`] [`-c`] [*major*] *minor*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `fuser` command lists the process IDs of the processes using the *files* specified on the command line. For block special devices, all processes using any file on that device are listed. The process ID is followed by `c`, `p`, `r`, or `t` character, to specify how the process uses the file. The character `c` signifies that the process is using the file as its current directory; `p` signifies that the process is using the file as the parent of its current directory (only when in use by the system); `r` signifies that the process is using the file as its root directory; and `t` signifies that the process is using the file as its controlling tty.

When used with the `-s` option, `fuser` lists the process IDs of the processes using the devices with minor device number *minor* and major device number *major*. If *major* is omitted, the major device number for sockets is assumed.

The `fuser` command accepts the following options:

`-k` Sends the `SIGKILL` signal to each process. Only the super user can terminate another user's process (see `kill`(2)).

`-m` Interprets *files* as a mounted file system. This is equivalent to replacing *files* with the block device in which *files* appears.

`-u` Prints the login name, in parentheses, following the process ID.

`-c` Prints the command name, in square brackets, following the process ID.

`-` Respecifies options between groups of files on a single command line. If you specify a single hyphen, the new set of options replaces the old set, canceling any options currently in force.

`-s` Lists the process IDs of the processes using the devices with minor device number *minor* and major device number *major*. If *major* is omitted, the major device number for sockets is assumed.

The process IDs are printed as a single line on the standard output, separated by spaces and terminated with a single new line. All other output is written on standard error.

NOTES

Output from `fuser` is restricted to processes running at a security label that the calling user dominates.

If this command is installed with the default privilege assignment list (PAL), a user with the `showall` privilege text is not subject to output restrictions.

EXAMPLES

Example 1: If typed by a super user, the following example terminates all processes that are preventing `/dev/dsk/usr` from being unmounted; it lists the process ID and login name of each process as it is killed.

```
fuser -ku /dev/dsk/usr
```

Example 2: The following example lists process IDs and login names of processes that have the password file open.

```
fuser -u /etc/passwd
```

Example 3: The following example performs both of the preceding operations with a single command line.

```
fuser -ku /dev/dsk/usr - -u /etc/passwd
```

Example 4: The following example lists process IDs, command names, and login names of processes that have socket 37 open.

```
fuser -s -cu 37
```

FILES

`/unicos`	System namelist
`/dev/kmem`	System image

SEE ALSO

`mount`(8)

`privtext`(1), `ps`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`kill`(2), `signal`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

General UNICOS System Administration, Cray Research publication SG–2301

NAME

fwtmp, wtmpfix – Manipulates connect accounting records

SYNOPSIS

```
/usr/lib/acct/fwtmp [-c] [-i]
/usr/lib/acct/wtmpfix [files]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

Without arguments, fwtmp reads from the standard input and writes to the standard output, converting binary records of the type found in wtmp to formatted ASCII records. The ASCII version lets you edit records that contain data that is not valid (using vi(1)), or generally maintain the file.

The fwtmp command accepts the following options:

-c Specifies that the output is in binary form. The default is ASCII form.

-i Specifies that the input is in ASCII form. The default is binary form.

files Files that contain connect accounting records.

The wtmpfix command examines the standard input or specified files in wtmp format, corrects the time and date stamps to make the entries consistent, and writes to the standard output. You can use a - in place of *files* to indicate the standard input. If time and date corrections are not performed, acctcon1 (see acctcon(8)) exits with an error code when it encounters certain date-change records.

Each time the date is set, a pair of date-change records are written to /etc/wtmp. The first record is the old date denoted by the string old time placed in the line field and the flag OLD_TIME placed in the type field of the utmp.h structure (defined in the include file utmp.h). The second record specifies the new date and is denoted by the string new time placed in the line field and the NEW_TIME flag placed in the type field. wtmpfix uses these records to synchronize all time stamps in the file.

In addition to correcting time/date stamps, wtmpfix checks the validity of the name field to ensure that it consists only of alphanumeric characters or spaces. If it encounters a name that is considered not valid, it changes the login name to INVALID and writes a diagnostic message to the standard error. In this way, wtmpfix reduces the chance that acctcon1 will fail when processing connect accounting records.

FILES

/etc/wtmp	Login records format file
/usr/include/utmp.h	Data about who currently is using the system

SEE ALSO

acct(8), acctcms(8), acctcon(8), acctmerg(8), acctprc(8), acctsh(8), runacct(8)

acctcom(1), ed(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

acct(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

acct(5), utmp(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

gated – Performs routing protocols

SYNOPSIS

gated [-c] [-C] [-n] [-N] [-t*trace_options*] [-f *config_file*] [*trace_file*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The gated routing daemon can be configured to manage one or more routing protocols, including the routing information protocol (RIP) and open shortest path first (OSPF) protocol. The gated daemon replaces routed(8).

Normally, gated detaches from the terminal and runs in the background. However, if trace options are specified without specifying a trace file, gated assumes that tracing is desired to be sent to stderr and remains in the foreground.

The gated daemon accepts the following options and arguments:

-c Specifies that the configuration file will be parsed for syntax errors and then gated will exit. If no errors occurred, gated leaves a dump file in /usr/tmp/gated_dump. The gated daemon does not need to be run as the super user to use the -c option, but it may not be possible to read the kernel forwarding table and interface configuration if it is not run as super user. The -c option implies -tgeneral. All *trace_option* clauses in the configuration file are ignored.

-C Specifies that the configuration file will only be parsed for syntax errors. The gated daemon exits with a status of 1 if there were any errors and 0 if there were not. gated does not need to be run as the super user to use the -C option, but it may not be possible to read the kernel forwarding table and interface configuration if not run as super user.

-f *config_file*
Specifies an alternate configuration file. By default, gated uses /etc/gated.conf.

-n Specifies that gated will not modify the kernel forwarding table. Used for testing gated configurations with actual routing data.

-N Specifies that gated will not daemonize.

-t*trace_options*
Specifies a comma-separated list of trace options to be enabled on startup. If no options are specified, *general* is assumed. No space is allowed between this option and its arguments. This option must be used to trace events that take place before the configuration file is parsed, for example, determining the interface configuration and reading routes from the kernel. See the gated-config(5) man page for valid trace options and a more detailed explanation of tracing.

trace_file
Specifies the file that receives tracing information.

Signal Processing

The gated daemon catches the following signals and does the following special processing:

SIGHUP Reads configuration again. This signal causes gated to reread the configuration file. gated first performs a cleanup of all allocated policy structures. Then the configuration file is parsed again.

OSPF is not capable of reconfiguring; it is shut down and restarted during a reconfiguration. This may have an adverse impact on the routing system.

You can enable and disable any protocol without restarting gated.

SIGINT Takes snapshot of current state. The current state of all gated tasks, timers, protocols and tables are written to /usr/tmp/gated_dump.

On the UNICOS system, this is done by creating (using fork(2)) a subprocess to dump the table information so as not to impact the routing functions of gated.

SIGTERM Graceful shutdown. On receipt of this signal, gated tries a graceful shutdown. All tasks and protocols are requested to shut down.

All protocol routes are removed from the kernel forwarding table on receipt of a SIGTERM signal. Interface routes, routes with RTF_STATIC set (from the route(8) command), and static routes specifying retain remain. To terminate gated with all routes intact, use SIGKILL.

SIGUSR1 Toggle tracing. On receipt of a SIGUSR1 signal, gated closes the trace file. A subsequent SIGUSR1 signal causes it to be reopened, allowing the file to be moved regularly.

It is not possible to use SIGUSR1 if a trace file has not been specified, or tracing is being performed to the stderr file.

SIGUSR2 Check for interface changes. On receipt of a SIGUSR2 signal, gated rescans the kernel interface list looking for changes.

FILES

Many of the default file names in the following list contain the string %s, which is replaced by the name with which gated is invoked. Normally this is gated, but if invoked as gated-test, gated will by default look for /etc/gated-test.conf. These paths may all be changed at compilation time.

These are the default filenames:

/usr/tmp/gated_dump The file to which gated writes status information. The default is /usr/tmp/%s_dump.

/etc/gated.conf The gated configuration file. The default is /etc/%s.conf.

`/etc/gated.pid` The file to which `gated` writes its process ID (PID). The default is `/etc/%s.pid`.

SEE ALSO

`arp`(8), `gdc`(8), `ifconfig`(8), `ospf_monitor`(8), `ripquery`(8), `route`(8)

`netstat`(1B) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`gated-config`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

COPYRIGHT INFORMATION

NAME

gdc – Operational user interface for gated(8)

SYNOPSIS

gdc [-q] [-t *seconds*] *command*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The gdc command provides a user-oriented interface for the operation of the gated(8) routing daemon. It provides support for starting and stopping the daemon, for the delivery of signals to manipulate the daemon when it is operating, for the maintenance and syntax checking of configuration files, and for the production and removal of state dumps and core dumps. gdc can reliably determine the running state of gated(8) and produces a reliable exit status when errors occur, making it advantageous for use in shell scripts that manipulate gated(8). Commands executed using gdc and, optionally, error messages produced by the execution of those commands, are logged using the same syslogd(8) facility which gated(8) itself uses, providing an audit trail of operations performed on the daemon.

If installed as a setuid root program, gdc allows nonroot users who are members of a trusted group (by default the gdmaint group) to manipulate the routing daemon while denying access to others. The name of the user and each executed command are logged using the syslogd(8) daemon.

The gdc daemon accepts the following options and arguments:

-q Runs quietly. With this option, informational messages normally printed to standard output are suppressed and error messages are logged using syslogd(8) instead of being printed to the standard error output. This is often convenient when running gdc from a shell script.

-t *seconds* Specifies the time in seconds gdc waits for gated(8) to complete certain operations, in particular at termination and startup. The default value is 10 seconds.

command Specifies the operation that gated(8) performs.

The *command* argument can have one of the following values, all of which cause signals to be delivered to gated(8):

COREDUMP Sends an abort signal to gated(8), causing it to terminate with a core dump.

dump Signals gated(8) to dump its current state into the file /usr/tmp/gated_dump.

interface Signals gated(8) to recheck the interface configuration. gated(8) normally does this periodically, but the facility can be used to force the daemon to check interface status immediately when changes are known to have occurred.

KILL Causes gated(8) to terminate ungracefully. Used most frequently when the daemon has hung.

`reconfig` Signals `gated`(8) to reread its configuration file, reconfiguring its current state as appropriate.

`term` Signals `gated`(8) to terminate after shutting down all operating routing protocols gracefully. Executing this command a second time causes `gated`(8) to terminate even if some protocols have not yet fully shut down.

`toggletrace` If `gated`(8) is currently tracing to a file, causes tracing to be suspended and the trace file to be closed. If `gated`(8) tracing is suspended, causes the trace file to be reopened and tracing initiated. This is useful for moving trace files.

By default `gated`(8) obtains its configuration from a file usually named `/etc/gated.config`. The `gdc` program also maintains these other versions of the configuration file:

`/etc/gated.conf+` The *new* configuration file. When `gdc` is requested to install a new configuration file, this file is renamed `/etc/gated.conf`.

`/etc/gated.conf-` The *old* configuration file. When `gdc` is requested to install a new configuration file, the previous `/etc/gated.conf` is renamed to this name.

`/etc/gated.conf--` The *really old* configuration file. `gdc` retains the previous *old* configuration file under this name.

The following values for the *command* argument perform operations related to configuration files:

`checkconf` Checks `/etc/gated.conf` for syntax errors. This command is usually executed after making changes to the configuration file but before sending a `reconfig` signal to the currently running `gated`(8), to ensure that there are no errors in the configuration which would cause the running `gated`(8) to terminate on reconfiguration. When this command is used, `gdc` issues an informational message indicating whether parse errors occurred or not. If errors did occur, `gdc` saves the error output in a file for inspection.

`checknew` Performs the same operation as `checkconf`, except that the *new* configuration file, `/etc/gated.conf+`, is checked.

`newconf` Moves the `/etc/gated.conf+` file into place as `/etc/gated.conf`, retaining the older versions of the file as described previously. If the *new* configuration file does not exist or otherwise appears impaired, `gdc` does nothing when this command is specified.

`backout` Rotates the configuration files in the newer direction, in effect moving the old configuration file to `/etc/gated.conf`. This command does not execute if the `/etc/gated.conf-` file does not exist or is zero length, or if the operation deletes an existing, nonzero length `/etc/gated.conf+` file.

`BACKOUT` Performs a `backout` operation even if `/etc/gated.conf+` exists and is of nonzero length.

`modeconf` Sets all configuration files to mode 664, owner to `root`, and group to `gdmaint`. This allows a trusted nonroot user to modify the configuration files.

`createconf` If the `/etc/gated.conf+` file does not exist, this command creates a zero length file with the file mode set to 664, owner to `root`, and group to `gdmaint`. This allows a trusted non-root user to install a new configuration file.

The following values for the *command* argument provide support for starting and stopping `gated`(8), and for determining its running state:

`running` Determines if `gated`(8) is currently running by checking if `gated`(8) has a lock on the file containing its process ID (PID), if the PID in the file is within a reasonable range of PIDs, and if a running process has that PID. Exits with zero status if `gated`(8) is running, nonzero otherwise.

`start` Starts `gated`(8). The command returns an error if `gated`(8) is already running. Otherwise it executes the `gated`(8) binary and waits for the delay interval (10 seconds by default, as set with the `-t` option otherwise) or less, until the newly started process obtains a lock on the PID file. A nonzero exit status is returned if an error is detected while executing the binary, or if a lock is not obtained on the PID file within the specified wait time.

`stop` Stops `gated`(8), gracefully if possible, ungracefully if not. The command returns an error (with nonzero exit status) if `gated` is not currently running. Otherwise it sends a `TERM` signal to `gated`(8) and waits for the delay interval (10 seconds by default, or as specified with the `-t` option otherwise) or less for the process to exit. Should `gated`(8) fail to exit within the delay interval, a second `TERM` signal is sent. Should `gated`(8) fail to exit by the end of the second delay interval, a `KILL` signal is sent. This should force immediate termination. The command terminates with a zero exit status when it detects that `gated`(8) has terminated, nonzero or otherwise.

`restart` If `gated`(8) is running, terminates `gated`(8) using the same procedure as for the `stop` command. If `gated`(8) was not running prior to command exeution, or when the previous `gated`(8) terminates, starts a new `gated`(8) process using the procedures described for the `start` command. A nonzero exit status is returned if any step in this procedure fails.

The following values for the *command* argument allow the removal of files created by the execution of some of the previous commands:

`rmcore` Removes any existing `gated`(8) core dump file.

`rmdump` Removes any existing `gated`(8) state dump file.

`rmparse` Removes the parse error file generated when a `checkconf` or `checknew` command is executed and syntax errors are encountered in the configuration file being checked.

BUGS

Many commands work only when `gated`(8) is installed in the system directory in which it was configured.

FILES

`/etc/gated`	The `gated`(8) binary
`/etc/gated.conf`	Current `gated`(8) configuration file
`/etc/gated.conf+`	Newer configuration file
`/etc/gated.conf-`	Older configuration file
`/etc/gated.conf--`	Much older configuration file
`/etc/gated.pid`	Location of PID for `gated`(8)
`/usr/tmp/gated_dump`	State dump file for `gated`(8)
`/usr/tmp/gated_parse`	Location of parse errors for the configuration file
`/usr/tmp`	Location of `gated`(8) core file

SEE ALSO

`gated`(8), `ospf_monitor`(8), `ripquery`(8), `route`(8), `syslogd`(8)

`gated-config`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

COPYRIGHT INFORMATION

NAME

getconfig – Searches the accounting configuration file for the specified argument

SYNOPSIS

/usr/lib/acct/getconfig *label*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The getconfig command searches the accounting configuration file for *label* and returns the value associated with it. This command is generally used in shell scripts.

By default, getconfig searches the /etc/config/acct_config file. If the shell variable ACCTCONFIG is set to an alternative file, getconfig will search a different configuration file.

EXAMPLES

Example 1: The following example extracts the value for HOLIDAY_FILE from the default configuration file. The shell variable HOLIDAY is set to the extracted value.

```
HOLIDAY=‘getconfig HOLIDAY_FILE‘
```

Example 2: The following example shows how to search an alternative configuration file for the HOLIDAY_FILE variable:

```
HOLIDAY=‘ACCTCONFIG=/tmp/myconfig getconfig HOLIDAY_FILE‘
```

FILES

/etc/config/acct_config Accounting configuration file

SEE ALSO

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

getpal – Gets the privilege assignment list (PAL) category entries of a file

SYNOPSIS

getpal [-c *catlist*] [-p *privlist*] [-t *privtext*] *files...*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The getpal command displays the privilege assignment list (PAL) category entries of the specified regular file(s).

The output consists of a list of PAL entries, one per line, in the following format:

category_name:*privlist*:*privtext*

category_name is the name of the category (for example, secadm). *privlist* is a character string that specifies a list of zero or more privileges (with multiple privileges separated by commas). A *privlist* value of PRIV_NULL indicates no privileges. A *privlist* value of PRIV_ALL indicates all privileges. *privtext* is a sequence of 0 to 8 characters that represent privilege text. A *privtext* value of TEXT_NULL indicates null privilege text.

If multiple file names are specified for *files*, the following line precedes the PAL information for each file.

#*filename*:

The getpal command accepts the following options and operands:

-c *catlist* Outputs PAL entries for each of the categories specified in *catlist*. If this option is not specified, it outputs PAL entries for any category, including other. *catlist* is a character string that represents one or more category names (for example, secadm). Multiple category names must be separated by commas, with no intervening white space.

-p *privlist* Outputs PAL entries that contain any of the privileges specified in *privlist*. If this option is not specified, or if the PRIV_ALL privilege name is specified, it outputs PAL entries containing any privilege sets, including PRIV_NULL. *privlist* is a character string that represents one or more privilege names (for example, PRIV_MAC_READ). Multiple privilege names must be separated by commas, with no intervening white space.

-t *privtext* Outputs PAL entries that contain the privilege text character sequence specified by *privtext*. If this option is not specified, it outputs PAL entries containing any privilege text value, including TEXT_NULL. *privtext* is a sequence of one to eight alphanumeric characters, or the word TEXT_NULL, that represents privilege text.

files Represents the name(s) of the file(s) whose PALs will be displayed.

If no options are specified, then output is produced for every PAL entry of the specified files.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`	Allowed to specify any file.
`sysadm`	Allowed to specify any file, subject to security label restrictions. Shell redirected output is subject to security label restrictions.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to specify any file.

EXIT STATUS

The `getpal` command exits with one of the following values:

Value	Description
0	The requested PAL entries were successfully displayed.
1	A badly formed option or option that is not valid was supplied.
2	When multiple files are supplied, both failure and success occurred.
4	The PAL(s) for the specified file(s) could not be displayed.

EXAMPLES

The following examples assume that `testfile` has the following PAL assigned to it:

```
system:PRIV_ALL:TEXT_NULL
secadm:PRIV_MAC_READ,PRIV_MAC_WRITE:TEXT_NULL
sysadm:PRIV_MAC_WRITE,PRIV_KILL:admin
other:TEXT_NULL
```

Example 1: The following example outputs all PAL entries for the `secadm` category for `testfile` using the `-c` option:

```
$ getpal -c secadm testfile
secadm:PRIV_MAC_READ,PRIV_MAC_WRITE:TEXT_NULL
```

Example 2: The following example displays all PAL entries with the `admin` privilege text for `testfile` using the `-t` option:

```
$ getpal -t admin testfile
sysadm:PRIV_MAC_WRITE,PRIV_KILL:admin
```

Example 3: The following example displays all PAL entries with the PRIV_MAC_WRITE privilege for testfile using the -p option:

```
$ getpal -p PRIV_MAC_WRITE testfile
system:PRIV_ALL:TEXT_NULL
secadm:PRIV_MAC_READ,PRIV_MAC_WRITE:TEXT_NULL
sysadm:PRIV_MAC_READ,PRIV_KILL:admin
```

Example 4: The following example shows getpal executed with no options specified, which displays every PAL entry for testfile:

```
$ getpal testfile
system:PRIV_ALL:TEXT_NULL
secadm:PRIV_MAC_READ,PRIV_MAC_WRITE:TEXT_NULL
sysadm:PRIV_MAC_READ,PRIV_KILL:admin
other:PRIV_NULL:TEXT_NULL
```

Example 5: The following example displays all PAL entries with both the PRIV_MAC_WRITE privilege and the admin privilege text for testfile using the -p and -t options:

```
$ getpal -p PRIV_MAC_WRITE -t admin testfile
sysadm:PRIV_MAC_WRITE,PRIV_KILL:admin
```

Example 6: The following example shows the display when no options are specified and multiple files are specified:

```
$ getpal testfile emptypalfile
# testfile:
system:PRIV_ALL:TEXT_NULL
secadm:PRIV_MAC_READ,PRIV_MAC_WRITE:TEXT_NULL
sysadm:PRIV_MAC_WRITE,PRIV_KILL:admin
other:PRIV_NULL:TEXT_NULL
# emptypalfile:
other:PRIV_NULL:TEXT_NULL
```

SEE ALSO

setpal(8)

NAME

getprivs – Gets the privilege sets of a file

SYNOPSIS

getprivs [-a] [-f] [-s] *files...*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The getprivs command displays the allowed, forced, and set-effective privilege sets of the specified regular file(s).

Output consists of a list of privilege sets, one per line, in the following format:

priv_set_id:*privlist*

priv_set_id can be a, f, or s indicating the allowed, forced, and set-effective privilege sets, respectively. *privlist* is a character string that specifies a list of privileges. Multiple privilege names are separated by commas with no intervening white space. When no privileges are set, the value of *privlist* is the PRIV_NULL character string. The PRIV_ALL character string represents the list of all privileges.

If multiple file names are specified for *files*, the following line precedes the PAL information for each file:

#*filename*:

The getprivs command accepts the following options and operand:

-a Displays the allowed privilege set of each specified file.

-f Displays the forced privilege set of each specified file.

-s Displays the set-effective privilege set of each specified file.

files Represents the name(s) of the file(s) whose privilege set(s) will be displayed.

If no options are specified, then all privilege sets are displayed for each specified file.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Allowed to specify any file.
sysadm	Allowed to specify any file, subject to security label restrictions. Shell redirected output is subject to security label restrictions.

If the PRIV_SU configuration option is enabled, the super user is allowed to specify any file.

EXIT STATUS

The getprivs command exits with one of the following values:

Value	Description
0	The specified privilege state was successfully reported.
1	A badly formed option or option that is not valid was supplied.
2	When multiple files are supplied, both failure and success occurred.
4	Privilege state for the specified file(s) could not be obtained.

EXAMPLES

The following examples assume that *testfile* has the following privilege states:

```
ALLOWED: PRIV_NULL
FORCED: PRIV_ALL
SET-EFFECTIVE: PRIV_MAC_READ,PRIV_MAC_WRITE
```

Example 1: The following example shows getprivs executed with no options specified, which results in all the privilege sets being displayed for testfile:

```
$ getprivs testfile
a:PRIV_NULL
f:PRIV_ALL
s:PRIV_MAC_READ,PRIV_MAC_WRITE
```

Example 2: The following example displays the forced and set-effective privilege sets of testfile using the -f and -s options:

```
$ getprivs -f -s testfile
f:PRIV_ALL
s:PRIV_MAC_READ,PRIV_MAC_WRITE
```

Example 3: The following example displays the set-effective privilege set of testfile using the -s option:

```
$ getprivs -s testfile
s:PRIV_MAC_READ,PRIV_MAC_WRITE
```

Example 4: The following example displays the allowed and forced privilege sets of testfile using the -f and -a options:

```
$  -f -a testfile
a:PRIV_NULL
f:PRIV_ALL
```

Example 5: The following example shows the display when no options are specified and multiple files are specified:

```
$ getprivs testfile noprivsfile
# testfile:
a:PRIV_NULL
f:PRIV_ALL
s:PRIV_MAC_READ,PRIV_MAC_WRITE
# noprivsfile:
a:PRIV_NULL
f:PRIV_NULL
s:PRIV_NULL
```

SEE ALSO

setprivs(8)

NAME

gettable – Gets NIC format host tables from a host

SYNOPSIS

/etc/gettable [-v] *host* [*outfile*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The gettable command obtains the Network Information Center (NIC) standard host tables from a *nicname* server (a name server that is running on a machine that is a NIC).

The gettable commands supports the following option and operands:

-v Gets only the version number instead of the complete host table and puts the output complete host table in the file *outfile* or, by default, hosts.ver.

host The indicated *host* is queried for the table.

outfile If retrieved, the table is placed in the file *outfile* or, by default, hosts.txt.

The gettable command operates by opening a TCP/IP connection to the port that is indicated in the service specification for the nickname server. A request is then made for all names, and the resulting information is placed in the output file.

The gettable command is best used in conjunction with the htable(8) command, which converts the NIC standard file format to that used by the network library look-up routines.

SEE ALSO

htable(8), named(8)

NAME

getty – Sets up an interactive connection

SYNOPSIS

```
/etc/getty [-m] [-L minlvl [-maxlvl]] [-C mincmp [-maxcmp]] [-t timeout] line [speed]
/etc/getty -c file
```

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The getty command is invoked by init(8). It is the second process in the series (*init-getty-login-shell*) that ultimately connects a user with the UNICOS operating system. Initially, getty prints the login message field for the entry it is using from /etc/gettydefs. getty reads the user's login name and invokes the login(1) command with the user's name as the argument.

On a secure system, the following options specify the security label range at which the line may be used:

-m
: Indicates that the line should be marked as a multilevel device. If the -m option is omitted, the line is labeled single-level.

-L *minlvl* [-*maxlvl*]
: Specifies the security level range at which the line will be labeled. The *minlvl* and *maxlvl* values are specified as decimals. If *maxlvl* is omitted, the maximum level is set equal to the minimum. If the -L option is omitted, the minimum and the maximum level are set equal to the system minimum level.

-C *mincmp* [-*maxcmp*]
: Specifies the compartment range at which the line will be labeled. The *mincmp* and *maxcmp* values are specified as integers that correspond to the binary bitmap of compartments. Octal or hexadecimal values can be given by prefixing the value with 0 or 0x, respectively. If *maxcmp* is omitted, the maximum compartment value is set equal to the minimum. If the -C option is omitted, the minimum and maximum compartment values are set to 0.

The getty command accepts the following options and operands:

-t *timeout*
: Specifies that getty should exit if the open on the line succeeds and no one types anything in *timeout* number of seconds.

line
: The name of a tty line in /dev to which getty is to attach itself. getty uses this string as the name of a file in the /dev directory to open for reading and writing.

speed
: Optional; a label to a speed and tty definition in the /etc/gettydefs file. This definition tells getty what the login message should look like and provides the initial tty settings. The default *speed* is 300.

`-c` *file* A check option; scans the *file* as if it were scanning `/etc/gettydefs` and prints out the results to the standard output. If there are any unrecognized modes or improperly constructed entries, it reports them. If the entries are correct, it prints the values of the various flags.

Finally, `login` is called with the user's name as an argument. Additional arguments may be typed after the login name. These are passed to `login`, which places them in the environment (see `login`(1)).

NOTES

The `getty` command is run on all terminals (ttys) directly attached to the IOS.

FILES

`/etc/gettydefs`

SEE ALSO

`init`(8)

`login`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`gettydefs`(5), `inittab`(5), `tty`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

hddmon – HIPPI disk device monitor and control program

SYNOPSIS

hddmon [-o] [-s *slot*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The hddmon program is used to monitor and control HIPPI disk array devices. The hddmon program can be used to monitor statistics detailing hdd(4) device usage and to control the read/write mode and the up/down state of the device.

The hddmon program has two modes of operation: interactive and single-pass. Running hddmon without the -o option brings up an interactive menu-driven refreshed display. Specifying the -o option does a single pass and writes statistics to stdout.

-o The -o option prints the main view screen. If the -s option is also indicated, detailed statistical information about the device occupying the specified slot is printed.

-s *slot* Specifying the -s option by itself selects the indicated *slot* number as the current slot of the display.

The mnu Package

In interactive mode, the hddmon program makes use of a menu-based display package called mnu(4). The mnu(4) package provides command input and screen refresh capabilities using the following primitives:

Primitive	Description
<TAB> or right arrow	menu right
<BACK SPACE> or left arrow	menu left
<RETURN>	next menu level or execute menu item
<ESC>	back to first menu level
<CONTROL-F> or <PAGE DOWN>	next display page
<CONTROL-B> or <PAGE UP>	previous display page
<CONTROL-D> or down arrow	display down 1 line
<CONTROL-U> or up arrow	display up 1 line
?	help

The first letter of a given menu item selects and executes that menu item. A help facility is available for nearly all menu items; to invoke the help facility, select the desired menu item and type `?`. For information on the the `mnu`(4) package, see the `mnu`(4) man page.

The Main View Screen

When you invoke `hddmon` in interactive mode, the main view screen appears. The main view screen lists all HIPPI disk array facilities arranged in the order in which they were opened. Each HIPPI disk facility that has been previously initialized is assigned a "slot" number. The current slot number is denoted by a `*` and is used to view detailed statistics or direct control to a particular device. A sample main view screen appears as follows:

```
hddmon:                                                          09/14/94 13:24:32
slot   view   display   help   configure   els   quit
                                                                 Page: 0  Line: 0
+slot   -slot   #slot
                                                    Sectors moved      Errors
slot iopth fcty rprt ifield type  state     mode     reads   writes   rec  unr
---- ----- ---- ---- ------ ----  --------- ----  -------- --------  ---- ----
   0  0130 0x10 0x10      6 HD64  open   up   rw      8961    19252     0    0
*  1  0130 0x10 0x11      6 HD64  open   up   rw      4096    34194     0    0
   2  0130 0x10 0x00     21 HD32  clsd   up   rw         1        0     0    0
```

The `iopth`, `fcty`, `rprt`, `ifield`, and `type` designate the I/O path, facility number, raid partition, ifield, and device type, respectively.

The `state` of the device indicates whether the device is open or closed and whether the run state is up, down, or suspend. Normally the state appears open and up.

The `mode` display indicates the read/write mode of the device. Possible modes are `rw` (read/write), `ro` (read only), or `na` (no allocate). The default mode is read/write. The mode is used by the filesystem to indicate the read and write capabilities of the device.

The last four columns summarize device activity and indicate the number of sectors transferred as well as recovered and unrecovered errors.

Main Menu Items

The main `hddmon` menu includes the following options:

Option	Description
`slot`	Selects the current slot.
`view`	Selects from several screens of statistical information about the device in the current slot.
`display`	Controls display primitives such as the refresh rate and page number in case the display information overflows the available window.
`help`	Displays menu primitives.

`configure` Sets the state and mode for the current device. Changing the device state can come in handy when the disk array or I/O subsystem is hung up and needs to be restarted. This procedure is detailed in the subsection "Array/IOS restart procedure."

`els` (CRAY J90 systems) Displays statistics and error information and configures the logical channels up and down. The menu screen that this item invokes is described in the subsection "The `els` display."

`quit` Exits the `hddmon` program.

Nearly all of the menu items have help text associated with them. For that reason, not all menu possibilities are listed here. Do not be afraid to explore. Select the menu item and type `?` to enter help mode. When in help mode, you can move easily from one menu selection to another.

Array/IOS Restart Procedure

If it becomes necessary to restart the HIPPI disk array and/or the IOS, you may be able to do this without a system interruption or, possibly, without losing a request, by using the following procedure.

1. Set the proper current slot and set the device state to suspend by selecting the `configure -> state -> suspend` menu item.
2. Restart the array and/or restart the IOS on the operator workstation (OWS):
   a. Use `eboot`(8) to reboot `eiop.hpi`.
   b. Use `econ`(8) to configure the IOP down.
   c. Use `econ`(8) to configure the IOP up.
3. Put the device in `up` state by selecting the `configure -> state -> up` menu item.

See the `econ`(8) and `eboot`(8) man pages for further information on using these commands on the OWS.

Configuration

The `hddmon` command opens the control node for `hdd` devices. By definition the control node is `/dev/ddd/hdd`. It is made implicitly by the `mkspice`(8) command by default whenever `mkspice`(8) is executed or explicitly by the `mkspice`(8) command as follows:

```
mkspice -t HD64 hdd
```

The `/dev/ddd/hdd` control node is assigned a minor number of 0. That means that you should avoid configuring your `/dev/hdd` nodes with a minor number of zero. Failure to do so could cause `hddmon` to exit with an `EBUSY` (device busy) error.

The `els` Display

The following screen appears when you select the `els` menu item and the `display` menu item on the `els` screen:

```
                                                                  04/27/95 14:07:30
main   display   select   configure
                                                                  Page: 0  Line: 0

                                        Errors                 I/O Transfers (bytes)
index  chan    path state  rejects  conn  retries  fail    read  Swrite  Cwrite
----- ------- ---- ----- -------- ----- -------- ----- ------- ------- -------
* 0   064/067 disk open        42     0       42     0    3.1M    7.8M     0.0
             tape close        0     0        0     0    0.0     0.0      0.0
  1   104/107 disk open        30     0       30     0    2.8M    6.1M     0.0
             tape close        0     0        0     0    0.0     0.0      0.0
```

The `els` menu includes the following options:

Option	Description
`main`	Returns you to the top-level `hddmon` display.
`display`	Displays the information as shown in the example.
`select`	Increments, decrements, or selects a specific channel for configuration (configuring up or down). The channel that is currently selected is indicated by a `*`.
`configure`	Configures the logical channel used for IPI-3 disk I/O up or down.

Each channel in the `els` display corresponds to a memory-HIPPI channel on the CRAY J90 series. Each memory-HIPPI can support both IPI-3 disk and tape traffic.

The `els` display includes the following categories:

Category	Description
index	An index number that identifies a specific memory-HIPPI channel. The memory-HIPPI channel that is currently selected is identified by a `*` in the leftmost column. In the example display above, the memory-HIPPI channel with the index of 0 is currently selected. The `configure` option uses the currently selected channel to perform channel up and down functions.
`chan`	The physical input and output memory-HIPPI channel of the form: input/output (for example, 064/067) The channel numbers are in octal.
`path`	Identifies the driver using the channel. Only disk and tape are supported.
`state`	The current state of the disk or tape driver using the channel. The possible values for this field are:

Value	Description
`open`	The channel path has been opened for use.
`close`	The channel path is closed.

DOWN The channel path has been configured down.

`rejects` The number of connection rejects reported by the memory-HIPPI driver.

`conn` The number of not connected errors reported by the memory-HIPPI driver.

`retries` The number of retries attempted by the CRAY J90 series IPI-3 pseudo driver.

`fail` The number of errors reported for the channel path.

`read` The number of read bytes for this channel path.

`Swrite` The number of simple write (single-packet) bytes for this channel path.

`Cwrite` The number of complex write (double-packet) bytes for this channel path.

Invoking the `select` option from the `els` menu adds the following options to the display:

Option	Description
`+channel`	Increments the index to the next channel.
`-channel`	Decrements the index to the previous channel.
`#channel`	Selects a specific channel using the index number.

Invoking the `configure` option from the `els` menus adds the following options to the display.

Option	Description
`up`	Configures the disk channel path up.
`down`	Configures the disk channel path down.

The `configure` option configures only the disk channel path. The tape channel path must be configured by using `hpi3_config`(8).

When the channel path is configured down, it must be configured back up before the channel path is available for normal use.

FILES

`/usr/src/uts/cmd/disk/hddmon.c`

`/usr/src/uts/cmd/disk/mnu.c`

`/usr/include/sys/mnu.h`

`/usr/include/sys/hddstat.h`

SEE ALSO

`mkspice`(8)

`eboot`(8), `econ`(8) in the *Support System Reference Manual*, Cray Research publication SR–3077

`hdd`(4), `mnu`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

hit – Performs a HYPERchannel interface test

SYNOPSIS

/etc/hit [-lrhw] [-1ikpstvx] [-a *addr*] [-d *dsize*] [-n *npass*] [-z *device*] *local remote*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The hit command performs one of three loopback tests on a Network Systems Corporation (NSC) HYPERchannel network adapter. Messages can be looped back: in the local adapter, in any other adapter on the HYPERchannel network, or in another host computer on the HYPERchannel network. hit can also be used with front-end interface (FEI) and VME devices.

The hit command can operate in one of two modes:

- Active mode: sends a message and waits for it to be echoed back (see the -l, -r, and -h options).
- Passive mode: waits for a message and echos it back (see the -w option).

In active mode, hit performs one or more passes, each of which consists of the following steps:

1. Sends a message proper.
2. Sends an associated data segment (if requested); see the -d option.
3. Receives a message proper.
4. Receives an associated data segment (if requested); see the -d option.
5. Checks data in the message proper (if requested); see the -p option.
6. Checks data in the associated data segment (if requested); see the -d and -p options.
7. Updates the data pattern (if requested); see the -i option.

In passive mode, hit performs one or more passes, each of which consists of the following steps:

1. Receives a message proper.
2. Receives an associated data segment (if requested); see the -d option.
3. Sends the message proper.
4. Sends the associated data segment (if requested); see the -d option.

Some of the command line options can alter these sequences in active and passive modes.

The `hit` command accepts the following options (one of them must be specified):

`-l` Performs loopback to the *local* adapter, using ‘‘transmit local message’’ functions. Only the *local* adapter address is required.

A130 note: The A130 is not capable of performing local loopback. The IOS model E driver for the A130 does not do the loopback; therefore, this test will fail.

`-r` Performs loopback to a *remote* adapter, using transmit message functions with byte 8 of the message proper (numbering from byte 0) set to 0xFF. Both *local* and *remote* adapter addresses are required.

`-h` Specifies the active side of a loopback test to a remote `host`, using transmit message functions. It is the responsibility of the remote host to turn the messages around. Both *local* and *remote* adapter addresses are required.

Note: The remote host must be running `hit -w` (or its equivalent) for this option to work correctly.

`-w` Specifies the passive, or `waiting`, side of a loopback test to a remote host. Only the *local* adapter address is required.

Note: The network driver assigns a logical channel number that `hit` must incorporate in the local adapter address. The `hit` command issues a message indicating the actual local adapter address it is using. You must specify the correct address when starting the active partner (that is, `hit -h`. See the About NSC Addresses subsection for more information.

You can use the remaining options to modify default settings.

`-1` Specifies one-way data. This option is used with two cooperating `hit` processes (one with the `-h` option, the other with the `-w` option). The sending process sends a message proper plus data, but expects only a message proper in reply. The receiving process reads in a message proper plus data, but sends only a message proper in reply. This simulates a common occurrence during file transfers, where data is moving in only one direction. Do not confuse this option with the `-k` option (described later).

`-i` Increments each byte of the data pattern at the end of each successful pass. See the Data Formats subsection for a description of the data pattern. This slows down the rate of message exchange. The default is to used a fixed data pattern. You cannot select this option if you have selected the `-w` option.

`-k` Runs `hit` in a very fast and unrestrained manner. Use this option with two cooperating `hit` processes (one with the `-h` option, the other with the `-w` option). (Do not confuse this with the `-l` option.) The sending process sends a message proper (and perhaps data) but never tries to read a reply. The receiving process reads in a message proper (and perhaps data) but never tries to send a reply. This option obtains maximum transfer rates.

Note: Not all drivers handle this kind of maximum-rate I/O. This option was developed to test the Cray Research UNICOS driver. Other drivers may drop messages (or worse).

`-p` Checks each byte of the data pattern at the end of each pass. `hit` compares the received data with the transmitted data. If any discrepancies exist, the number of bytes in error is reported. If you select verbose mode (see the `-v` option), each byte in error is printed, along with the expected value. This slows down the rate of message exchange. The default is not to check received data. You cannot select this option if you have already selected the `-w` option.

`-s` Stops the test after an error is detected. The default is to terminate the current pass of the test when an error is detected and proceed to the next pass.

`-t` Prints timing information after the last pass has completed. The elapsed time is printed (to six decimal places, but the precision depends on the granularity of the system clock). A data transfer rate, in bytes per second and bits per second, is also printed *if* associated data segments were sent and received (see the `-d` option), *and* no errors were detected. The transfer rate calculation is: (size of associated data segment in bytes) × (number of passes) × 2 ÷ (elapsed time in seconds).

`-v` Writes verbose output to `stdout`, slowing down the rate of message exchange. The default is to print only error messages.

`-x` Writes debug output to `stdout`, slowing down the rate of message exchange. This option is useful only if you have a `hit` source listing. The default is to suppress debugging information.

`-a` *addr* Allows you to specify the Internet network number. *addr* is a decimal class A Internet network number from 1 through 127; the default is 84. Usually, the default value is the only one needed; therefore, this option may be safely ignored. See the About Internet Addresses subsection for more information.

`-d` *dsize* Allows you to specify, as a decimal integer *dsize*, the associated data size in bytes. It ranges from 0 (no data) to 16,384 bytes. (This maximum data size is a compile-time parameter and can be changed.) The default is 4096 bytes. For local and remote adapter loopback, the data size is limited by the adapter memory size, typically 4096 bytes. For remote host loopback, the data size is limited by the loopback software running on the remote host.

`-n` *npass* Allows you to specify the number of passes as a decimal integer *npass*. The default is 100 passes.

`-z` *device* Allows the character special device name to be specified. The default is `/dev/hy00` for logical channel 0, `/dev/hy01` for logical channel 1, and so on. If you select this option, the named device must agree with the *local* operand. The default device name may also be modified at compile time.

You must specify one or both of the adapter addresses. See the About NSC Addresses subsection for more information.

local Local adapter address, specified as 4 hexadecimal digits. For example, A400 adapter unit 0x13, port 1, is specified as `1301`.

remote Remote adapter address, specified as 4 hexadecimal digits. For example, A130 adapter unit 0x40, logical channel 4, is specified as `4004`.

About NSC Addresses

NSC HYPERchannel addresses consist of 16 bits. The high-order 8 bits are always the adapter unit number, which is configured with thumbwheel switches on the back of the adapter. The low-order 8 bits vary, depending on adapter type. For example, the A400 adapter uses the low-order 2 bits for a port number (up to four host computers can be connected to a single A400). The remaining 6 bits are used for software routing (the logical channel number).

You need to know the local adapter unit number and port number. The network driver fills in the software routing field in the address. For example, if you are using adapter unit 0x13 port 1, you would specify address 1301. The driver may assign you logical channel number 1; in this case, your actual address would be 1305. On Cray PVP systems, you cannot use logical channel number 0 (for example, `/dev/comm/. . ./lp00` or `/dev/hy00`.) It is reserved by the IOS for USCP; if you use this path, `hit` will report an error 5.

You also need to know the remote adapter unit number. For remote adapter loopback (see the `-r` option), you do not need to know what kind of adapter it is. You can ignore the low-order 8 bits of the address. For example, remote adapter unit 0 x 40 would be specified as 4000. For remote host loopback (see the `-h` option), you need to know both the remote adapter type and the software on the remote host. They dictate the value in the low-order 8 bits of the remote adapter address. If `hit` is running on the remote host in passive mode (see the `-w` option), it tells you what the full address is.

About Internet Addresses

Each HYPERchannel network is assigned a class A Internet network number. (This is a restriction that will be lifted in the future.) `hit` must specify this number to the `hy`(4) network driver. `hit` is configured with a single network number; the default is `84`. Some sites may have more than one HYPERchannel network. In this case, each network has its own class A number (for example, the Cray Research development network is 84, and the production network is 86). Therefore, `hit` accesses the development network by default, and requires the option `-a 86` to access the production network.

Data Formats

The message proper always contains a 48-byte data area, which begins in byte 16 (numbering from byte 0). It is always filled with the default (fixed) data pattern. If you specify an associated data segment (with the `-d` option), the data segment is also filled with the data pattern. The data pattern consists of one-byte integers 0, 1, 2, 3 ..... Message proper byte 16 contains 0, byte 17 contains 1, and so on. Similarly, data segment byte 0 contains 0, byte 1 contains 1, and so on. If more than 256 bytes are in the data segment, the pattern repeats every 256 bytes.

If you specify the `-i` option, each byte in the pattern is incremented after each pass. For the second pass, message proper byte 16 contains 1, byte 17 contains 2, and so on.

MESSAGES

`-l or -r option: data size limit=`*number*`.`
`-w option: cannot specify -i or -p.`
`Associated data rate=`*number* `bytes/second.`
(*number* bits/second). (Only if `-t` option selected)

`Bind error, errno=`*number*`, lchan=`*number*`.`	(Network driver only)
`Can't establish network connection.`	(Network driver only)
`Can't open` *device*`.`	(Not if using network driver)
`Elapsed time=`*number* `seconds.`	(Only if `-t` option selected)
`Invalid data size specified.`	(Only if `-d` option selected)
`Invalid local address` *number*`.`	
`Invalid network specified.`	(Only if `-a` option selected)
`Invalid number of passes.`	
`Invalid remote address` *number*`.`	
`Local adapter address is` *hex_number*`.`	
`Local address not specified.`	
`No logical channels available.`	(Network driver only)
`One of -lrhw must be specified.`	
`Received message from` *hex_number*`.`	(Only if `-w` option selected)
`Remote address not specified.`	
`Setsockopt error, errno=`*number*`.`	(Network driver only)
`Socket error, errno=`*number*`.`	(Network driver only)
`Still waiting for connection.`	(Only if `-w` selected)
`Stop, error detected.`	(Only if `-s` option selected)
`Unknown option` *c*`.`	
`Usage: hit ...`	

All other errors are logged with the time of day, last function, pass number, and a cumulative error count. The last function is the last system call (`sendto` (see `send`(2)), `recvfrom` (see `recv`(2)), `write`(2), or `read`(2)).

As a rule, the network driver does not report errors to the user, but rather writes messages to the console. Look at the system console and `/usr/adm/messages` if there are problems.

Theory

Presently, `hit` can operate with one of two device drivers:

- The network driver (4.2 BSD default)
- The Cray Research UNICOS driver (System V default)

Network Driver Operation

`hit` opens a raw socket. Each active pass consists of a `sendto` (see `send`(2)) call to write out the message proper (and associated data, if any), followed by a `recvfrom` (see `recv`(2)) call to read in the message proper (and associated data, if any). Because few errors are reported to the user level, a 30-second time is started before each network operation. If the timer expires, you should examine the system console and `/usr/adm/messages` to see whether the driver has reported any problems.

Cray UNICOS Driver Operation

The `hit` command opens an entry in the `/dev` directory. The name is `/dev/hy00` for logical channel 0, `/dev/hy01` for logical channel 1, and so on. The logical channel number is taken from the two low-order hexadecimal digits of the local adapter address. Each active pass consists of a `write`(2) call to write out the message proper (and associated data, if any), followed by a `read`(2) call to read in the message proper (and associated data, if any).

NOTES

The test may be stopped at any time with the `SIGINT` signal, which is usually <`CONTROL-c`>.

If `hit` refuses to run, try master-clearing the local adapter manually (press the red reset button on the back of the adapter).

You should match the options on two cooperating copies of `hit`. If a passive copy is started for 1000 passes, the active copy should also specify 1000 passes. The passive copy should always be started first.

EXAMPLES

A good way to use `hit` is to try local loopback first. This checks out the driver, interface, and local adapter. If this works, try remote adapter loopback. This ensures the two adapters can talk. Try remote loopback with several destination adapters, from each host on the network. Finally, try remote host loopback. This proves that two hosts can talk to each other.

Example 1: The following example performs a local loopback with 4-Kbyte data segments for 100 passes:

```
hit -lt 1301
```

Timing information is reported when the test is complete. The local adapter is A400 unit 0x13 port 1.

Example 2: The following example performs a remote loopback with 4-Kbyte data segments for 100 passes:

```
hit -rt 1301 2100
```

Timing information is reported when the test is complete. The local adapter is A400 unit 0x13 port 1, and the remote adapter is A130 unit 0x21.

Example 3: The following example performs a remote loopback with no data for 10,000 passes:

```
hit -rips -d0 -n10000 1301 C100
```

The data pattern in the message proper is checked and then incremented at the end of each pass. The test halts if any error is detected. The local adapter is unit A400 0x13 port 1, and the remote adaptor is A130 unit 0xC1.

Example 4: The following example starts the passive side of a remote host loopback, with 4-Kbyte data segments, for 100 passes:

```
hit -wt 1301 2301
```

Timing information is reported when the test is complete. The local adapter is A400 unit 0x13 port 1, and the remote adapter is unit A130 0x23 logical channel 1. The `hit` command issues a message showing the actual local adapter address (including the logical channel number), as in the following example:

```
hit: local adapter address is 1305
```

Exxample 5: The following example starts the active side of a remote host loopback, with 4-Kbyte data segments, for 100 passes (the remote adapter address was taken from the `hit` diagnostic message in the preceding example):

```
hit -ht 2301 1305
```

Timing information is reported when the test is complete. The local adapter is A130 unit 0x23, logical channel 1, and the remote adapter is A400 unit 0x13, logical channel 1, port 1.

FILES

`/dev/hy`*nn* Cray Research UNICOS HYPERchannel driver entry (*nn* = logical channel number)

SEE ALSO

`hy`(4), `vme`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

hpi3_clear – Clears a IPI-3/HIPPI packet driver device

SYNOPSIS

/usr/lib/hpi3_clear *devicename*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The hpi3_clear command terminates all outstanding requests to *devicename*. Requests that have finished processing but have not been returned to the user are discarded. No further user requests to the device are processed until the device is closed. When using the hpi3_clear command, you must specify a *devicename*.

EXIT STATUS

The hpi3_clear command completes successfully, 0 is returned; otherwise, a nonzero value is returned. Where possible, this exit status code is normalized to the last three digits. Exit status values are documented in the *Tape Subsystem User's Guide*, Cray Research publication SG–2051.

FILES

/dev/hpi3/*devicename*	IPI-3/HIPPI interface devices
/dev/hpi3/reqt	IPI-3/HIPPI interface devices
/etc/config/hpi3_config	IPI-3/HIPPI configuration file

SEE ALSO

hpi3_config(8), hpi3_option(8), hpi3_start(8), hpi3_stat(8), hpi3_stop(8)

hpi3(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

Tape Subsystem Administration, Cray Research publication SG–2307

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

hpi3_config – Configures a IPI-3/HIPPI channel up or down

SYNOPSIS

Cray PVP systems with IOS model E:

/usr/lib/hpi3_config -c *channel-pair* -C *cluster* -i *iop state*

CRAY EL series and CRAY J90 series:

/usr/lib/hpi3_config -c *channel-pair state*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The hpi3_config command configures a IPI-3/HIPPI channel up or down. The operands are as follows:

-c *channel-pair*	Specifies the channel pair that will be configured.
-C *cluster*	Specifies the cluster in which the channel is configured.
-i *iop*	Specifies the IOP in which the channel is configured.
state	Specifies HPI-3 channel state (either up or down).

EXIT STATUS

The hpi3_config command completes successfully, 0 is returned; otherwise, a nonzero value is returned. Where possible, this exit status code is normalized to the last three digits. Exit status values are documented in the *Tape Subsystem User's Guide*, Cray Research publication SG–2051.

FILES

/dev/hpi3/*devicename*	IPI-3/HIPPI interface devices
/dev/hpi3/reqt	IPI-3/HIPPI interface devices
/etc/config/hpi3_config	IPI-3/HIPPI configuration file

SEE ALSO

hpi3_clear(8), hpi3_option(8), hpi3_start(8), hpi3_stat(8), hpi3_stop(8)

hpi3(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

Tape Subsystem Administration, Cray Research publication SG–2307

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

hpi3_option – Modifies a IPI-3/HIPPI packet driver option(s)

SYNOPSIS

/usr/lib/hpi3_option [-a *maximum-number-async-responses*] [-c *maximum-number-cmdlst*] [-i *maximum-number-iop-processes*] [-r *maximum-number-non-cmdlst*] [-t on|off]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The hpi3_option command modifies one or more IPI-3/HIPPI packet driver options. The options are as follows:

-a *maximum-number-async-responses*
: Specifies the maximum number of asynchronous responses that may be outstanding per device. Users cannot enable asynchronous responses that exceed this value.

-c *maximum-number-cmdlst*
: Specifies the maximum number of command list requests that may be outstanding per device. If the number of command list requests issued exceeds this value, an error is returned.

-i *maximum-number-iop-processes*
: Specifies the maximum number of processes that can open an I/O processor (IOP) device concurrently. If this limit is exceeded, an error is returned.

-r *maximum-number-non-cmdlst*
: Modifies the maximum number of noncommand list requests that may be outstanding per device.

-t on|off
: Specifies the tracing state (either on or off).

NOTES

If a process has an IOP device open, you cannot modify the maximum number of processes that may open an IOP.

EXIT STATUS

The hpi3_option command completes successfully, 0 is returned; otherwise, a nonzero value is returned. Where possible, this exit status code is normalized to the last three digits. Exit status values are documented in the *Tape Subsystem User's Guide*, Cray Research publication SG–2051.

FILES

`/dev/hpi3/`*devicename*	IPI-3/HIPPI interface devices
`/dev/hpi3/reqt`	IPI-3/HIPPI interface devices
`/etc/config/hpi3_config`	IPI-3/HIPPI configuration file

SEE ALSO

`hpi3_clear`(8), `hpi3_config`(8), `hpi3_start`(8), `hpi3_stat`(8), `hpi3_stop`(8)

`hpi3`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

Tape Subsystem Administration, Cray Research publication SG–2307

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

hpi3_start – Starts the IPI-3/HIPPI packet driver subsystem

SYNOPSIS

/usr/lib/hpi3_start [-f *config-file*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The hpi3_start command informs the IPI-3/HIPPI packet driver and, for Cray PVP systems with IOS model E (IOS-E), the IPI-3/HIPPI IOP driver of the physical organization of each IOP. For each IOP configured, a corresponding IOP file is created. This file is used to issue packets that affect the IOP as a whole. For each device attached to the IOP, hpi3_start creates a corresponding device file. Device files are used to issue packets that affect the device.

The user provides the configuration in a file, which describes the IOP devices, channels, slaves, devices, and options.

The hpi3_start command accepts the following option:

-f *config-file* Specifies the configuration file. The default configuration file is /etc/config/hpi3_config.

For Cray PVP systems with IOS-E, the default configuration file is of the following format:

```
-IOPS
IOP-name   cluster   IOP

-CHANNELS
IOP-name  input-channel  output-channel  state input-channel-timeout
  output-channel-timeout connection-timeout

-SLAVES
slave-name  IOP-name input-ch1:output-ch1[,input-ch2:output-ch2] i-field

-DEVICES
device-name   slave-name  low-facility-addr  high-facility-addr

-OPTIONS
option    option-value
```

For the CRAY EL series and CRAY J90 series, the default configuration file is of the following format:

```
-CHANNELS
input-channel  output-channel  state  input-channel-timeout
output-channel-timeout  connection-timeout

-SLAVES
slave-name  input-ch1:output-ch1[,input-ch2:output-ch2]  i-field

-DEVICES
device-name  slave-name  low-facility-addr  high-facility-addr

-OPTIONS
option  option-value
```

NOTES

You must stop all configured IOP drivers before executing the `hpi3_start` command.

EXIT STATUS

The `hpi3_start` command completes successfully, 0 is returned; otherwise, a nonzero value is returned. Where possible, this exit status code is normalized to the last three digits. Exit status values are documented in the *Tape Subsystem User's Guide*, Cray Research publication SG–2051.

FILES

`/dev/hpi3/`*device-name*	IPI-3/HIPPI interface devices
`/dev/hpi3/reqt`	IPI-3/HIPPI interface devices
`/etc/config/hpi3_config`	IPI-3/HIPPI configuration file

SEE ALSO

`hpi3_clear`(8), `hpi3_config`(8), `hpi3_option`(8), `hpi3_stat`(8), `hpi3_stop`(8)

`hpi3`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

Tape Subsystem Administration, Cray Research publication SG–2307

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

`hpi3_stat` – Displays device statistics

SYNOPSIS

Cray PVP systems with IOS model E:

`/usr/lib/hpi3_stat` [`-c`] [`-C` *cluster*] [`-d` *devname*] [`-i` *iop*] [`-t`]

CRAY EL series and CRAY J90 series:

`/usr/lib/hpi3_stat` [`-c`] [`-d` *devname*] [`-t`]

All Cray Research systems:

`/usr/lib/hpi3_stat` [`-o`]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `hpi3_stat` command displays device status, configuration, packet driver table, or the packet driver options statistics. The status, configuration, or table values for all devices can be displayed, or a partial display can be requested (for example, you can request a cluster, I/O processor (IOP), or a particular device to be displayed). The status of all devices is the default display.

The `hpi3_stat` command accepts the following options:

`-c` Specifies that the configuration will be displayed. By default, the status is displayed. The `-c`, `-o`, and `-t` options are mutually exclusive.

`-C` *cluster* Specifies the cluster in which all configured devices should be displayed.

`-d` *devname* Specifies the device for which information is displayed.

`-i` *iop* Specifies the IOP in which all configured devices should be displayed.

`-t` Specifies that the packet driver table be displayed. By default, the status is displayed. The `-c`, `-o`, and `-t` options are mutually exclusive.

`-o` Specifies that the options should be displayed. You cannot specify the `-o` option with any other options.

When you request device configuration (`-c`) information, the following information is displayed:

device The device name

ioc The cluster in which the device is configured

iop The IOP in which the device is configured

chpair A channel pair in which the device is configured in the following format:

input-channel:output-channel

ichst The status of the input channel:

- `down` Interrupts logically disabled
- `wconn` Waiting for connection
- `wpkt` Waiting for an IPI-3 packet
- `read` Reading data
- `dump` Reading and ignoring data

ochst The status of the output channel:

- `down` Interrupts logically disabled
- `up` Interrupts enabled
- `conn` Connection established
- `disc` Connection terminated
- `cmd` Read of command complete
- `wcmd` Write of command complete
- `wdta` Write of data complete

sl The ordinal of the slave in which the device is attached

lfc The low facilities address

hfc The high facilities address

If you do not specify `-c`, `-t`, or `-o`, the following information is displayed:

device The device name

ioc The cluster in which the device is configured

iop The IOP in which the device is configured

chpair A channel pair in which the device is configured in the following format:

input-channel:output-channel

ichst The status of the input channel:

- `down` Interrupts logically disabled
- `wconn` Waiting for connection
- `wpkt` Waiting for an IPI-3 packet
- `read` Reading data
- `dump` Reading and ignoring data

ochst The status of the output channel:

`down`	Interrupts logically disabled
`up`	Interrupts enabled
`conn`	Connection established
`disc`	Connection terminated
`cmd`	Read of command complete
`wcmd`	Write of command complete
`wdta`	Write of data complete

If you specify `-t`, the following table information is displayed:

device The device name

ord The table ordinal

flag The flags used by the IPI-3/HIPPI packet driver:

0001	The user has open the device file
0002	An IOP request timed out
0004	Packet interface was enabled
0010	The device is open (IOP open)
0020	A signal was registered
0100	A clear device is in progress
0200	The device has been cleared
0400	Device processing is waiting for an interrupt

sig The signal to send to the user when a packet is returned from the IOP

pid The ID of the process that has the device open

lock The number of locks on the process

rsyn The resynchronization code

async The number of queued asynchronous packets (that is, asynchronous packets returned by the IOP but not received by the user)

enabl The number of asynchronous packets that are currently enabled

usr The number of outstanding user requests

ios The number of outstanding IOP requests

cmd The number of command list requests outstanding to the IOP

EXIT STATUS

The `hpi3_stat` command completes successfully, 0 is returned; otherwise, a nonzero value is returned. Where possible, this exit status code is normalized to the last three digits. Exit status values are documented in the *Tape Subsystem User's Guide*, Cray Research publication SG–2051.

FILES

`/dev/hpi3/`*devicename*	IPI-3/HIPPI interface devices
`/dev/hpi3/reqt`	IPI-3/HIPPI interface devices
`/etc/config/hpi3_config`	IPI-3/HIPPI configuration file

SEE ALSO

`hpi3_clear`(8), `hpi3_config`(8), `hpi3_option`(8), `hpi3_start`(8), `hpi3_stop`(8)

`hpi3`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

Tape Subsystem Administration, Cray Research publication SG–2307

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

hpi3_stop – Stops the IPI-3/HIPPI subsystem

SYNOPSIS

/usr/lib/hpi3_stop [-u] [-w]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The hpi3_stop command shuts down the IPI-3/HIPPI packet driver, disables further opens of the IOPs and IPI-3 devices, waits for current activity to stop, and if requested, stops the IOP drivers.

The hpi3_stop command accepts the following options:

-u Leaves the IOP drivers up.

-w Waits for all activity to stop and for all devices to be closed. By default, if the IPI-3/HIPPI packet driver is not idle, an error is returned.

EXIT STATUS

The hpi3_stop command completes successfully, 0 is returned; otherwise, a nonzero value is returned. Where possible, this exit status code is normalized to the last three digits. Exit status values are documented in the *Tape Subsystem User's Guide*, Cray Research publication SG–2051.

FILES

/dev/hpi3/*devicename* IPI-3/HIPPI interface devices

SEE ALSO

hpi3_clear(8), hpi3_config(8), hpi3_option(8), hpi3_start(8), hpi3_stat(8)

hpi3(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

Tape Subsystem Administration, Cray Research publication SG–2307

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

hpmall – Reports hardware performance statistics for entire machine workload

SYNOPSIS

/etc/hpmall [-d] [-r] [-t *second*] [-V]

/etc/hpmall [-g *group*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The hpmall command reports the Hardware Performance Monitor (HPM) statistics that have been accumulated for all user processes on the machine. Additionally, for super users, it can set the default HPM group under which all processes will be run. hpmall writes its output to standard output.

The hpmall command accepts the following options:

-d Displays HPM statistics for each CPU on a multi-CPU machine. The default is to combine the statistics for all CPUs before they are reported.

-g *group* Number of the HPM monitor group under which all processes will be run at the next connection to the CPU (not used on the CRAY C90 or CRAY T90 series). Note that this option can be executed only by a super user. If this option is specified, all other options are ignored.

group can be any of the following values:

0 Execution summary

1 Hold issue conditions

2 Memory activity

3 Vector events and instruction summary

-r Generates "raw" output suitable for postprocessing by other tools, such as awk(1). See the *Guide to Parallel Vector Applications*, Cray Research publication SG–2182, for a description of this format. If this option is not specified, the output is written as a report.

-t *second* The amount of time to delay between making two samples of the HPM counters. If this option is not specified, the HPM statistics reported are for the entire machine since boot-up. If this option is specified, hpmall samples the counters, waits for *second* seconds, and then samples again. It then reports the difference between these two samples. If the value given for *second* exceeds 1800 (30 minutes), the delay value is set to 1800.

-V Displays the current version of hpmall, as well as a short copyright notice.

NOTES

The meanings of the HPM statistics and their implications are discussed in detail in the *Guide to Parallel Vector Applications*, Cray Research publication SG–2182. On Cray PVP systems (except CRAY C90 and CRAY T90 series), users can override the default HPM counter group for their programs by executing the `hpm`(1) command or by using the perftrace library. Therefore, if you set the default group by using `hpmall`, a user can still accumulate statistics under a different group. These statistics will appear in `hpmall` reports, regardless of the default HPM group that you have set.

On Cray PVP systems (except CRAY C90 and CRAY T90 series), note that if the default HPM counter group is changed to be other than 1, no wait semaphore time will be recorded for accounting, because counter group 1 is used to measure the wait semaphore time.

EXAMPLES

Example 1: The following example shows how to generate a report to the file `hpm.all` that shows the combined HPM statistics for all CPUs on the current machine, since boot-up.

```
/etc/hpmall > hpm.all
```

Example 2: The following example shows how to generate a report to the file `hpm.all` that shows the combined HPM statistics for all CPUs on the current machine, for a 30-second period.

```
/etc/hpmall -t 30 > hpm.all
```

Example 3: The following example shows how to generate a report to the file `hpm.all` that shows the HPM statistics for each CPU on the current machine, since boot-up.

```
/etc/hpmall -d > hpm.all
```

Example 4: The following example shows how to generate raw data for the whole machine workload and post-process it with `perfview`.

```
/etc/hpmall -r > hpm.raw
perfview hpm.raw
```

Example 5: The following example shows how to change the HPM counter group under which all subsequent processes will run. This can be executed only by a super user. (No report is generated.)

```
/etc/hpmall -g 0
```

Note: The previous example is not used on the CRAY C90 and CRAY T90 series.

FILES

`/dev/hpm_all`

SEE ALSO

hpm(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

Guide to Parallel Vector Applications, Cray Research publication SG–2182

NAME

hpmflop – Reports hardware performance statistics gathered for user processes

SYNOPSIS

/etc/hpmflop [-a] [-d *date*] [-t] [-u *maxusers*] [-V] [*datafile*]

/etc/hpmflop [-c] [-d *date*] [-m] [-p] [-t] [-u *maxusers*] [-V] [*datafile*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The hpmflop command reports the average megaflops achieved by user processes, based on statistics gathered by special instrumentation from the Hardware Performance Monitor (HPM). These statistics are optionally gathered on a site-wide basis for all normally terminated user programs on the machine. hpmflop writes its report output to standard output.

Review of the output of this command allows the site to determine which users and which of their programs might be using large amounts of CPU time with low Megaflop rates. Thus, these users can be contacted and encouraged to optimize their programs.

The hpmflop command accepts the following options and operands:

-a Reports on all processes for all user IDs. By default, hpm(1) reports processes belonging only to the user invoking hpmflop.

-c Reports a summary sorted by user ID and by CPU seconds used.

-d *date* Reports on processes for a particular day for the user invoking hpmflop. Used with the -a option, this reports on all processors for that day. Default: reports all processes chronologically.

date must be of the form "*Mmm dd*":

- The value must be surrounded by double quotes because of the embedded blank.
- *Mmm* is the standard alphabetic abbreviation for a particular month as defined by ctime(3C)
- The single space between month and day is required.
- *dd* must be two digits. Include a leading zero for days 1 through 9.

```
hpmflop -d"Jul 08" rawfile
```

-m Reports a summary by user ID sorted by MFLOPS.

-p Reports a summary by user ID and process sorted by process; it can be used with -c and -m.

`-t` Reports on today's processing; it can be used with `-c` and `-m` for daily summaries for the invoking user or with `-a` for a daily list of all processes. It reports processes belonging only to the user invoking `hpmflop` chronologically.

`-u` *maxusers*
Allocates an internal table large enough to hold *maxusers* different user IDs. The default is 1,000 different user IDs. If the raw data file contains records for more than this number of different user IDs, `hpmflop` will issue a fatal error message and terminate. *maxusers* must be a positive, nonzero, decimal value.

`-V` Displays the current version of `hpmflop`, as well as a short copyright notice.

datafile Name of the file containing HPM information for `hpmflop` to process. The information was written to a site-selected file by the `hpmdump` global data-gathering feature described in the following section. By default, the name of the data file is `/usr/spool/hpm.data`.

Data Gathering

The data gathering itself is controlled by the contents of the following files:

```
/lib/segdirs/def_ld
/lib/segdirs/def_seg
```

To enable loading of the global gathering code, these files must include the following `SEGLDR` directive:

```
hardref=_hpmdumpg
```

To activate the global gathering, an additional step is required. You must create a file to receive the data. The file name used is a fixed value: `/usr/spool/hpm.data` and must have its privileges set so that every terminating process has access to and can write to it. This file will increase in size during production processing, so the site personnel with responsibilities for maintaining the file should also be responsible for removing older versions.

This file must exist (even if empty) before the data-gathering code can write statistics into it. If your site chooses to use a different file for this data gathering, the file `/usr/spool/hpm.data` must be linked to your site's chosen file, using the `ln`(1) command.

The algorithm used by the embedded data gathering code may cause some process terminations to be missed and not written into the data file. However, the missed processes will not be statistically significant.

Note: The following two paragraphs do not apply to the CRAY C90 or the CRAY T90 series.

Before the data gathering scheme can supply useful information, the default HPM counter group for all processes should be set to group 0. You can set the default HPM counter group using the `hpmall`(8) command. Note that such a setting is active only until the next system restart. It is acceptable to set the default HPM counter group to group 3, but the megaflops values reported by `hpmflop` will be for only vector floating-point operations.

The site default HPM counter group can also be set permanently during UNICOS installation.

To prevent large numbers of trivial processes from appearing in the collected data, a default minimum total CPU time of 5 seconds is set in the data-gathering software. This value can be overridden by setting an environment variable as follows. The *cputime* value must be a decimal number of seconds. Only processes whose total accumulated CPU time exceed this threshold will be considered for inclusion in the common file.

`/etc/profile`:

```
HPM_MT=cputime ; export HPM_MT
```

`/etc/cshrc`:

```
setenv HPM_MT cputime
```

Raw-format Output

The `hpmflop` command processes data generated by the global data-gathering feature, which is written to the common statistics file. Each record of output contains the data from one HPM counter group for a terminating process. If a process ran under more than one counter group, it may have more than one record written.

The records are written in ASCII with a fixed number of fields in each record. The fields are separated by white space. The fields described below are decimal numbers unless otherwise indicated.

Sample record for the CRAY C90 and CRAY T90 series:

```
hpmg 701379651 13612 834 ./a.out 92905263 11927052 74532647 1109
            180091749 13797955 0 0 3061091 121009 54064351 201810
            31801275 658 8474328 1506763 603712 626749 7333326 13238
            19590 16380 755 12761 11190 225 665 120000043 33 30000000
            120002196 60022765
```

Field	Description
1	HPM global record marker (the string `hpmg`).
2	Time stamp taken when the process terminated. This number can be passed to `ctime`(3C) to see a printable date/time value.
3	Process identification number (PID) of the process whose execution is responsible for the HPM statistics.
4	User identification number (UID) of the user who is running this process. Additional information, such as the user's name, can be extracted from the running system using the `getpwuid` (see `getpwent`(3C)) call.
5	Name of the executable file that was run to generate these HPM statistics. This is a string.
6-37	HPM counter values, with field 6 being counter 0, and field 37 being counter 31.

Sample record for Cray PVP systems (except CRAY C90 and CRAY T90 series):

```
hpmg 701379576 19728 834 ./a.out 1 315 1 7295375 3099449 85613587
            58800482 175050 553262 176035219
```

Field	Description
1	HPM global record marker (the string `hpmg`).
2	Time stamp taken when the process terminated. This number can be passed to `ctime`(3C) to see a printable date/time value.
3	Process identification number (PID) of the process whose execution is responsible for the HPM statistics.
4	User identification number (UID) of the user who is running this process. Additional information, such as the user name, can be extracted from the running system, using the `getpwuid` (see `getpwent`(3C)) call.
5	Name of the executable file that was run to generate these HPM statistics. This is a string.
6	HPM counter group. This will be a number from 0 through 3.
7-14	HPM counter values, with field 7 being counter 0 of the group, and field 14 being counter 7 of the group.
15	Total CPU time accumulated for this process, when run under the HPM counter group given in field 6. Processes may accumulate time under other HPM groups, and such times would be reported in separate termination records.

Note that the raw-format output contains more statistics than the megaflop rate of the processes. This raw data can be processed by `awk`(1) scripts, such as those generated by the `perfscripts`(1) command.

NOTES

`hpmflop` must be executed on the same type of machine, with the same clock speed, on which the original data was gathered. Do not mix machine types. If the user ID values are mapped differently on different machines, likewise, the results from this command may not be useful.

The meanings of the HPM statistics and their implications are discussed in detail in the *Guide to Parallel Vector Applications*, Cray Research publication SG–2182.

Troubleshooting

The data-gathering instrumentation is activated by loading of user programs with the `hardref` SEGLDR directive. However, several errors can occur when this code is preparing the data for writing to the common file. To conserve user program memory space, this error handling is quite primitive. If your site has set up all of the required configuration and data files but no data appears on the common file, some error analysis will be required.

The data-gathering instrumentation code stores error flags in global memory locations in every executable program in which it has been loaded. The easiest way to determine the current processing error is to run a trivial program under a debugger and check the value of the global variable `_hpm_global_error`. In some cases, a system `errno` also occurs, and this value will be stored in the global variable `_hpm_global_sys_errno`.

The possible decimal values for `_hpm_global_error` are as follows:

1 The program could not open the `/dev/hpm_mult` device. Either the machine does not have this device defined, or else some greater I/O error occurred. The `_hpm_global_sys_errno` variable will be set to the `errno` associated with the failed open.

2 The program could not read the `/dev/hpm_mult` device correctly. This may be due to some mismatch in the expected data size, or some other I/O error. The `_hpm_global_sys_errno` variable will be set to the `errno` associated with the failed read.

4 This user execution did not accumulate enough CPU time to meet the criteria for minimum CPU time. The minimum is the decimal environment variable `HPM_MT` or 5 seconds.

5 Unable to open the global data file (for writing) whose name is `/usr/spool/hpm.data`. The `_hpm_global_sys_errno` variable contains the system `errno` value for the failed attempts to open.

The global data file is site-maintained, and must have world-write permissions.

Failures to perform the writes on the global data file are not noted by the termination code. This type of failure can be checked independently by site personnel.

If the variable for `_hpm_global_error` is not present in the program being tested, the feature was not loaded correctly. Check the `/lib/segdirs/def_ld` and `/lib/segdirs/def_seg` files carefully.

Security Considerations

If your site chooses to enable the `hpm` global data gathering feature, the following security issues should be considered:

Because the global data file is world-writeable, the opportunity exists for a user to abuse it, including: Altering the data or rendering it unusable, and/or filling up the file system in which the data is written, making the file system unusable.

For sites relying on the mandatory access control (MAC) protection mechanism, this feature should not be enabled, because it provides a direct channel to bypass the MAC policy enforced by system.

EXAMPLES

The following command requests statistics for all processes on the machine for January 17. Assume that a data file was accumulated for all processes on the machine for dates that included January 17th. The data is contained in the file `raw.data`.

```
hpmflop -d"Jan 17" -a raw.data
```

Output would consist of one or more report lines in the following form:

```
Jan 17 15:45 rds       124.26 Mflops      65.4 Mvops 0.12s ./a.out
```

The fields in the report lines have the following meanings:

Field	Meaning
1-3	Date and time of the normal process termination.
4	User ID as generated by the function `getpwuid` (see `getpwent`(3C)). If this raw performance data is being processed on a different machine from that on which it was gathered, this field may be shown as (NULL). Such a problem could be caused by the mismatch of user ID numbers on the different machines.
5, 6	Number of average megaflops achieved during the execution of the process.
7, 8	Number of average megaflops achieved for vector arithmetic operations.
9	Number of total CPU seconds executed by the process.
10	Command used to invoke the process.

Review of such a report allows the site to determine which users and which of their programs might be using large amounts of CPU time with low megaflop rates. Thus, these users' programs can be targeted for further optimizations.

SEE ALSO

`hpmall`(8) for instructions for setting the default counter group

`awk`(1), `perfscripts`(1) for information about commands that can process raw data produced by the global data-gathering feature *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`ctime`(3C) for information about date and time formatting
`getpwent`(3C) for information about user IDs and user names in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

Guide to Parallel Vector Applications, Cray Research publication SG–2182, for descriptions of all the performance tools

NAME

hsxconfig – Configures HIPPI and HSX channel interfaces

SYNOPSIS

/etc/hsxconfig *device_name status*

IMPLEMENTATION

Cray PVP systems (except CRAY J90 series and CRAY EL series)

DESCRIPTION

The hsxconfig command lets an operator configure HSX high-speed external communication channels up and down. This is useful in preventing simultaneous use of HSX channels and online tape devices, which can result in performance degradation. It also configures the ANSI High Performance Parallel Interface (HIPPI). See hippi(4) for more information.

Setting a channel up or down is appropriate only on the logical path zero device (/dev/hsx0/i00 or /dev/hsx0/o00). The inode permission bits determine who may perform this task.

Additionally, hsxconfig can configure a logical path in auto header mode. In auto header mode, the HSX driver adds a header to each output block and removes it from each input block, allowing user processes to share a hardware channel transparently.

Auto header mode is appropriate only on nonzero logical paths. Only a super user can change the header mode.

EXAMPLES

Example 1: The following example configures the HSX channel referred to as /dev/hsx0/i00 up:

```
hsxconfig /dev/hsx0/i00 up
```

Example 2: The following example configures the HSX channel referred to as /dev/hsx0/i00 down:

```
hsxconfig /dev/hsx0/i00 down
```

Example 3: The following example sets auto header mode on the HSX logical path referred to as /dev/hsx0/i01:

```
cd /dev/hsx0
hsxconfig i01 auto
```

Example 4: The following example restores header processing to the user on the HSX logical path referred to as /dev/hsx0/i01:

```
cd /dev/hsx0
hsxconfig hsx0/i01 hdr
```

SEE ALSO

`hippi`(4), `hsx`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

`htable` – Converts NIC standard format host tables

SYNOPSIS

`/etc/htable` [`-c` *connected_nets*] [`-l` *local_nets*] [*files*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `htable` command converts host *files* in the format specified in Internet RFC 810 to the format used by the network library routines. The `hosts`, `networks`, and `gateways` files are created as a result of running `htable`. The `gethostent`(3C) routines use the `hosts` file in mapping host names to addresses. The `getnetent`(3C) routines use the `networks` file in mapping network names to numbers.

If any of the files `localhosts`, `localnetworks`, or `localgateways` are present in the current directory, the file's contents is prepended to the output file without interpretation. Of these, only the `gateways` file is interpreted. This allows sites to maintain local aliases and entries that are not usually present in the master database. Only one gateway to each network is placed in the `gateways` file; a gateway listed in the `localgateways` file overrides any in the input file.

The `htable` command accepts the following options:

`-c` *connected_nets*	Specifies a list of networks to which the host is directly connected.
`-l` *local_nets*	Specifies a list of networks to be treated as local networks.
files	Specifies files to be converted.

If you use the `gateways` file, a list of networks to which the host is directly connected is specified with the `-c` option. The networks, separated by commas, may be given by name or in Internet-standard dot notation (for example, `-c arpanet,128.32,local.ether.net`). `htable` includes only gateways that are directly connected to one of the networks specified or that can be reached from another gateway on a connected net.

If the `-l` option is given with a list of networks (in the same format as for `-c`), these networks are treated as local, and information about hosts on local networks is taken only from the `localhosts` file. Entries for local hosts from the main data base are omitted. This allows the `localhosts` file to completely override any entries in the input file.

If you omit *file*, `htable` reads from standard input.

The `htable` command is best used in conjunction with the `gettable`(8) command, which retrieves the Network Information Center (NIC) database from a host.

SEE ALSO

`gettable`(8), `named`(8)

`gethost`(3C), `getnet`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

NAME

hyroute – Sets the Internet address to hardware address mapping

SYNOPSIS

/etc/hyroute *interface* [-c] [-d] [-l] [-p] [-s] [-D] [*file*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The hyroute command manipulates the host or gateway address to hardware address mapping.

The hyroute command accepts the following operands and options:

interface Specifies the name of the interface to which this command refers.

-c Compares the system's current information to that contained in *file*.

-d Dumps the system's table (used for debugging routing code).

-l Displays on standard error a trace of the parsing of *file*.

-p Prints a digested version of *file*.

-s (Set) Reads *file* and sets the system's database according to the information in the file.

-D Dumps the system's table in a readable format.

file Specifies the input file. This operand is necessary for all options other than -d. If you do not specify a file name, or if the minus sign (-) is encountered as an argument, hyroute reads from standard input.

The input file is free format. Comment lines begin with a * or # symbol in column 1. Statements end with a semicolon. Three valid statement verbs are direct, gateway, and c_format.

EXAMPLES

The HYPERchannel-DX series of adapters can contain an IP routing engine to perform IP routing between the attached Cray Research system and network media such as Ethernet, FDDI, and T1/T3 links. If you are configuring a Cray Research system with one of these types of adapters, you must place an entry in the configuration file to point to the IP router instead of placing an individual entry for each destination host, as is done for hosts directly attached to HYPERchannel media. For these types of adapters, the configuration file must contain only an entry for the Cray Research system itself and an entry for the IP router. The format of this entry is the same as it is for normal HYPERchannel hosts; however, you should obtain the value to place into the *dest* field of the direct statement from the Network Systems Corporation personnel on site. The address can vary, depending on the configuration of the adapter and the IP routing engine that is being used to route IP packets.

The following statement describes a HYPERchannel host that can be reached directly from an adapter:

```
direct host dest control access [mtu];
```

The *host* value can be a host name (see `hosts`(5)). The *host* can be an Internet address specified in dot notation (see `inet`(3C)). The values *dest*, *control*, and *access* are hexadecimal numbers, and the optional *mtu* value is a decimal number. The data is sent to HYPERchannel address *dest*, using a control value of *control* and an access code of *access* (see adapter manuals for details). The *mtu* field is the maximum size HYPERchannel packet that the host can receive.

The specified remote adapter and the local adapter must both be connected to one or more common trunks or connected to trunks that are connected with link adapters. The following statement describes a host that may be reached indirectly through any one of the hosts indicated by *gate*n:

```
gateway host gate1 gate2 gate3 . . . ;
```

The hosts listed are not gateways in the formal sense (they do not run the Internet gateway protocols), but they are hosts on the HYPERchannel that can bridge between subsections of the HYPERchannel network.

The following statement causes `hyroute` to interpret all subsequent numbers in the input file as integer constants expressed in C syntax. For example, a leading 0x signals a hexadecimal number (such as 0xFF); a leading 0 indicates an octal number (such as 0377); otherwise, the number is decimal (such as 255).

```
c_format;
```

The following statement describes a host that can be reached on an FEI-3 or low-speed channel:

```
direct host dest unused unused [mtu];
```

The following statement describes a host that can be reached over a HIPPI or HSX channel:

```
direct host hwaddr readdev writedev [mtu];
```

The low-order 8 bits of *readdev* and *writedev* contain the minor device numbers of the logical paths to open for reading and writing. The `hwaddr` is the logical channel to use when sending to the specified host.

A sample file follows:

```
* comment
direct azure     6100 0 0 4160;
direct bronze    6101 0 0 4160;
direct cyber     2100 1100 0 4160;
direct dadcad    6102 0 0 4160;
direct tekcad    2400 1100 0 4160;
direct tekcrd    2201 1100 0 4160;
direct tekid     2500 1100 0 4160;
direct teklabs   2200 1100 0 4160;
gateway iddic    tekcrd teklabs cyber tekcad tekid;
gateway iddme    tekcrd teklabs cyber tekcad tekid;
gateway metals   tekcrd teklabs cyber;
```

SEE ALSO

ifconfig(8)

inet(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

hippi(4), hosts(5), hsx(4), hy(4), vme(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

ifconfig – Configures network interface parameters

SYNOPSIS

/etc/ifconfig *interface* [*address_family*] [*address* [*destination*]] [*parameters*]

/etc/ifconfig *interface* [*address_family*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The ifconfig command assigns an address to a network interface and configures network interface parameters. You must use it at boot time to define the network address of each interface present on a machine; use ifconfig also to redefine an interface's address or other parameters.

If you use the format shown in the first synopsis, ifconfig sets the configuration for the specified interface; if you use the format shown in the second synopsis, ifconfig displays the configuration for the specified interface.

The ifconfig command accepts the following arguments:

interface Specifies a string of the form *name-unit* (for example, fddi0).

address_family Specifies a name that represents a protocol; this argument is necessary when an interface may receive transmissions from differing protocols. Possible value is inet. With the first format, if you omit *address_family*, it sets inet configuration. With the second format, if you omit *address_family*, it displays inet configuration.

address For the DARPA Internet family, this option specifies the *address* as either a host name present in the host name database, hosts(5), or a DARPA Internet address expressed in the Internet standard dot notation.

destination Specifies the address of the other end of a point-to-point network.

parameters Sets directives that can further specify network configuration.

You can set the following parameters by using ifconfig:

admin This option marks an interface as restricted to authorized users only. Creating a socket with PRIV_ADMIN effective on PAL-based systems, or UID root on PRIV_SU systems, enables a socket for communication on this interface. In a future release, an additional setsockopt system call will be required to enable communication on the socket. This option also prevents forwarding of IP packets to and from the interface.

-admin Disables the admin option.

`alias` Establishes an additional network address for this interface. This is sometimes useful when you are changing network numbers and you want to accept packets addressed to the old interface.

`authority` *byte0*`[:`*byte* `...]`
Labels the interface to specify the authorities that are allowed on packets that are accepted or sent over the interface. The authority list can consist of up to 8 bytes, with each byte separated by a colon. The last byte must have the low-order bit set to 0, and all preceding bytes must have the low-order bit set to 1. The values are specified in hexadecimal notation.

`arp` (`ifconfig` accepts this value but it is not relevant to any Cray Research interface.) Enables the use of the address resolution protocol in mapping between network-level addresses and link-level addresses (default).

`-arp` (`ifconfig` accepts this value but it is not relevant to any Cray Research interface.) Disables the use of the Address Resolution Protocol.

`bg` Retries attempt to set the interface in the background if the first attempt fails.

`broadcast` *addr*
(Internet only) Specifies the address to use to represent broadcasts to the network. The default broadcast address is the address with a host part of all 1's.

`compart` *min*`[-`*max*`]`
Labels the interface to allow only packets that have at least *min* compartments, and no more than *max* compartments, to be accepted or sent over the interface. If *max* is omitted, it is assumed to be equal to *min*. You can specify the *min* and *max* values in hexadecimal notation by preceding the number with 0x; you can specify them in octal notation by preceding the number with 0; or you can specify them in decimal. See the NOTES section for operational information.

`debug` Enables driver-dependent debugging code; usually, this turns on extra console error logging.

`-debug` Disables driver-dependent debugging code.

`delete` Removes the network address specified. This would be used for an incorrectly specified alias or for one that is no longer needed.

`down` Marks an interface down. When an interface is marked down, the system does not try to transmit messages through that interface. If possible, the interface is also reset to disable reception. This action does not automatically disable routes that use the interface.

`hwloop` For packets destined for the local interface that support hardware loopback, use hardware loopback instead of software loopback.

`iftype` *type* (Cray Research systems only) Sets the interface type to one of the following:

`hy` NSC HYPERchannel (see `hy`(4) and `np`(4))

`hippi`	High Performance Parallel Interface (see `hippi`(4))
`hsx`	High-speed external communications (see `hsx`(4))
`n130`	NSC N130 protocol (see `hy`(4) and `np`(4))
`vme`	FEI-3 (see `vme`(4))
pvc	Permanent Virtual Circuit (see `atmarp`(8))
q2931	ATM standard signalling protocol
spans	FORE Systems proprietary signalling protocol

`level` *min*`[-`*max*`]`
Labels the interface to allow only packets that are within the range *min* to *max* to be accepted or sent over the interface. If *max* is omitted, it is assumed to be equal to *min*. The *min* and *max* values are mnemonic level names (see `secnames`(3C)), or they can be specified in decimal. The range of allowed values includes `syslow` and `syshigh`. See the NOTES section for operational information.

`metric` *n* Sets the routing metric of the interface to *n*; the default is 0. The routing protocol uses the routing metric. Higher metrics make a route less favorable; metrics are counted as additional hops to the destination network or host.

`mtu` *size* Changes the size of the read buffers posted to the low-level driver. The *size* argument represents the size available to the IP layer for *datagrams* (that is, data packets), not including link-level or physical frame headers. The size is rounded up to a multiple of the word size and a warning message printed, if necessary.

For GigaRing I/O systems only, the following maximum limits apply for each media type:

Ethernet 1500 bytes (same as default)

FDDI 4500 bytes (default = 4352)

HIPPI 65536 bytes (default = 65280)

ATM 65536 bytes (default = 9180)

If the size of the posted read buffer is too small to receive an entire datagram, the datagram is discarded and the following error message is logged:

```
WARNING: if_fddi.c: fddi0: Bad read: errno = 5
```

This condition can cause applications that use TCP connections (such as `ftp`) to hang, repeatedly retransmitting the data.

The *size* argument must be greater than the write buffer size on all directly connected hosts. Presumably, all hosts on the network use the same write buffer size for a given host. The `ifconfig` command compares the *size* argument to the host's write buffer size in the `hyroute` table. If the host's write buffer size is greater than the *size* argument, the new `mtu` size is accepted, but the following warning message is printed:

```
ifconfig: WARNING - New mtu may be too small.
```

See the `hyroute` command input file `direct` statement for more information on the write buffer size.

This parameter does not apply to the Ethernet and FDDI network interfaces on the CRAY EL series and CRAY J90 series.

`netmask` *mask*
For Internet:

Specifies how much of the address to reserve for subdividing networks into subnetworks. The *mask* includes the network part of the local address and the subnet part, which is taken from the host field of the address. You can specify the *mask* as one hexadecimal number with a leading 0x, with a dot-notation Internet address, or with a pseudo-network name listed in the network table `networks`(5). The mask contains 1's for the bit positions in the 32-bit address that are used for the network and subnet portions, and 0's for the host portion. The mask should contain at least the standard network portion, and the subnet field should be contiguous with the network portion.

`ptp` For GigaRing systems only:

This option flags the interface as a point-to-point interface, causing the HIPPI to be used in "dedicated" mode and all packets to be transmitted in "hold" mode. This mode assumes only TCP is using the HIPPI channel, there is no HIPPI switch in the network, and the network is connected point-to-point to another host.

For all Cray Research systems:

If a destination address is supplied, you can omit this flag, but a warning message is issued. If this flag is set and no destination address is offered, an error message is displayed and `ifconfig` fails.

`-ptp` Turns off the Point-to-point Interface flag.

`rbuf` *bufcnt*
Changes the maximum number of buffers that can be posted to the driver for reads. You can see the current value by using the `netstat -iv` command.

Care should be taken when using this option, as there is no limit set for the *bufcnt* argument. Whatever value the user enters will be used by the system.

This parameter does not apply to the Ethernet and FDDI network interfaces on the CRAY EL series and CRAY J90 series.

`trailers`
(`ifconfig` accepts this value, but it is not relevant to any Cray Research interface.) Requests the use of a trailer link-level encapsulation when sending (default). If a network interface supports trailers, the system, when possible, encapsulates outgoing messages in a manner that minimizes the number of memory-to-memory copy operations that the receiver performs. Currently, only Internet protocols use this parameter.

`-trailers`
(`ifconfig` accepts this value, but it is not relevant to any Cray Research interface.) Disables the use of a trailer link-level encapsulation.

`up` Marks an interface up. This parameter enables an interface after the `ifconfig down` command is issued. The up action occurs automatically when the first address is set on an interface. If the interface is reset after it has been previously marked down, the hardware is reinitialized.

`wbuf` *bufcnt*
Changes the maximum number of buffers that can be posted to the driver for writes. Use the `netstat -iv` command to see the current value.

Care should be taken when using this option, as there is no limit set for the *bufcnt* argument. Whatever value the user enters will be used by the system.

This parameter does not apply to the Ethernet and FDDI network interfaces on the CRAY EL series and CRAY J90 series.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to observe and configure network interface parameters.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to observe and configure network interface parameters.

The `ifconfig` command displays the current configuration for a network interface when no optional parameters are supplied. If an address family is specified, `ifconfig` reports the details specific only to that address family.

The security label of the interface is displayed such that label information that is within the user's security label reange is shown. Only appropriately authorized users can see the actual label information. Only appropriately authorized users can modify the configuration of a network interface.

The security label can be increased only while an interface is configured up. Restricting a label requires the interface to be configured down. When an interface is restricted, routes for that interface are bounded by the new label. If a route is outside of the new label, the route is deleted automatically.

MESSAGES

Messages indicate that the specified interface does not exist, the requested address is unknown to the interface, potential problems exist with the mtu size, or the nonprivileged user tried to alter an interface's configuration.

SEE ALSO

`brc`(8), `hyroute`(8), `initif`(8)

`netstat`(1B), `privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`hosts`(5), `intro`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

ift – Reads user Factory Flaw table and reports flaws

SYNOPSIS

/etc/ift [-f] [-p] [-s *streams*] *special*

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

DESCRIPTION

The ift command reads permanent flaw information from a CE cylinder and writes it to standard output as an ASCII Flaw table (aft). The *special* operand is a special file that corresponds to the disk's CE cylinder number 1. By default, ift reads the User Flaw table and reports only nonslippable flaws.

The ift command accepts the following options:

-f Reads the Factory Flaw table rather than the User Flaw table.

-p Reports the physical location of all flaws, even those that were slipped.

-s *streams*
In a case such as an array, when a device is made of individual member drives, each with its own flaw table, *streams* is a bit mask that designates the drives from which flaw tables will be read. Bit 2^0 is drive 0, Bit 2^1 is drive 1, and so on.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

Example 1: In the following example, the ift command reads the User Flaw table from device 0101, and it writes the nonslippable flaws to the ASCII Flaw table /etc/aft/0101.

```
ift /dev/ift/0101 > /etc/aft/0101
```

Example 2: In the following example, if device 0130.4 is an array, the flaw tables from drives 1 and 3 will be the only ones read.

```
ift -s 012 /dev/ift/0130.4
```

SEE ALSO

dsk(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

inetd – Performs Internet super-server function

SYNOPSIS

/etc/inetd [-d] [-R *rate*] [*configuration file*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The inetd command, invoked at boot time by /etc/sdaemon, which is invoked by /etc/tcpstart, which is invoked by netstart(8), listens for connections on certain Internet sockets, and when a connection is found on one of its sockets, it decides to which service the socket corresponds and invokes a program to service the request. After the program is finished, inetd continues to listen on the socket (except in selected cases; see the inetd.conf(5) man page for more information). Essentially, inetd allows running one daemon to invoke several others, reducing load on the system.

The inetd command accepts the following options:

-d	Turns on debugging for connection-oriented services.
-R *rate*	Specifies the maximum number of times a service can be invoked in one minute; the default is 90.
configuration file	File from which, on execution, inetd reads its configuration information. This file is by default, /etc/inetd.conf. The configuration file contains the following information about each server: service name, socket type, protocol, wait/nowait, user, server program, and server program arguments. For RPC-based servers, protocol version numbers are also specified. See inetd.conf(5) for more information about format and content.

The inetd command provides several trivial services internally by use of routines within itself. These services are echo, discard, chargen (character generator), daytime (human readable time), and time (machine readable time, in the form of the number of seconds since midnight, January 1, 1900). For details of these services, consult the relevant RFCs, as follows:

echo	RFC 862, *Echo Protocol*, Postel, J. B., May 1983.
discard	RFC 863, *Discard Protocol*, Postel, J. B., May 1983.
chargen	RFC 864, *Character Generator Protocol*, Postel, J. B., May 1983.
daytime	RFC 867, *Daytime Protocol*, Postel, J. B., May 1983.
time	RFC 868, *Time Protocol*, Postel and Harrenstein, May 1983.

The `inetd` command rereads its configuration file when it receives a hangup signal, `SIGHUP`. Services may be added, deleted, or modified when the configuration file is reread. `inetd` also reregisters all of its RPC-based servers with `portmap` at this time.

`inetd` invokes the server process at the security label of the incoming connection or datagram.

ERROR MESSAGES

The `inetd` server logs error messages using `syslog`(3C). Important error messages and their related explanations are as follows:

Error Message	Explanation
`server failing (> 90 connections in 60 seconds), service terminated`	The number of requests for the specified server in the past minute has exceeded the limit. The limit exists to prevent a broken program or a malicious user from swamping the system. This message may occur for any of the following reasons: 1. A number of hosts are requesting the service in a short period of time 2. A "broken" client program is requesting the service too frequently in a short period of time 3. A malicious user is running a program to invoke the service in a "denial of service" attack 4. The invoked service program has an error that causes clients to retry too quickly To change the rate limit, use the [`-R`] option, as described above. Once the limit has been reached, the service will be reenabled automatically following a five minute wait period.
`No such user` *user*`, service ignored`	No entry for *user* exists in the `passwd` file. This message occurs when `inetd` rereads the `configuration` file.
`getpwnam:` *user:* `No such user`	No entry for *user* exists in the `passwd` file. This message occurs when the service is invoked.
`can't set uid number`	The user ID for the entry's user is invalid.
`can't set gid number`	The group ID for the entry's user is invalid.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

FILES

`/etc/inetd.conf` Default configuration file for `inetd`

SEE ALSO

`ftpd`(8), `netstart`(8), `rexecd`(8), `rlogind`(8), `rshd`(8), `telnetd`(8), `tftpd`(8) in the *UNICOS Administrator Commands Reference Manual*, Cray Research publication SR–2022

`privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`inetd.conf`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

infd – Allows the operator to display informative messages

SYNOPSIS

/usr/lib/msg/infd

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The infd command allows the operator to display informative messages. The operator does not reply to informative messages, and the informative message is removed after the operator has seen it. infd displays messages in the order in which it receives them.

NOTES

If this command is installed with a privilege assignment list (PAL), a user who is assigned the following privilege text upon execution of this command is allowed to perform the actions shown:

Privilege Text	Action
showall	Messages are not subject to security label restrictions.
both	Messages are not subject to security label restrictions.

If this command is installed with a PAL, a user with one of the following active categories is allowed to perform the action shown:

Active Category	Action
system, secadm, sysadm, sysops	Allowed to use this command to view all messages.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command to view all messages.

SEE ALSO

msgdaemon(8), msgdstop(8), rep(8)

msgi(1), msgr(1), privtext(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

NAME

infocmp – Compares or displays terminfo(5) descriptions

SYNOPSIS

/usr/bin/infocmp [-c] [-d] [-n] [-C] [-I] [-L] [-r] [-u] [-s *sort_option*] [-v] [-V] [-w *width*] [-1] [-A *directory*] [-B *directory*] [*termnames*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The infocmp command can be used to compare a binary terminfo(5) entry with other terminfo(5) entries, to rewrite a terminfo(5) description to take advantage of the terminfo(5) string capability use, or to display a terminfo(5) description from the binary file (term(5)) in a variety of formats. In all cases, the Boolean fields are displayed first, followed by the numeric fields, followed by the string fields.

Default Options

If no options are specified and zero or one *termnames* are specified, the -I option is assumed. If more than one *termname* is specified, the -d option is assumed.

Comparison Options

The infocmp command compares the terminfo(5) description of the first terminal *termname* with each of the descriptions given by the entries for the other terminals' *termnames*. If a capability is defined for only one of the terminals, the value returned depends on the type of the capability: F for Boolean variables, -1 for integer variables, and NULL for string variables.

-c Produces a list of each capability that is common between the two entries. Capabilities that are not set are ignored. This option can be used as a quick check to see whether the -u option is worth using.

-d Produces a list of each capability that is different. In this manner, if there are two entries for the same terminal or similar terminals, using infocmp shows what is different between the two entries. This is sometimes necessary when more than one person produces an entry for the same terminal and you want to see what is different between the two.

-n Produces a list of each capability that is in neither entry. If no *termnames* are given, the environment variable TERM is used for both of the *termnames*. This can be used as a quick check to see whether anything was left out of the description.

Source Listing Options

The -C, -I, and -L options produce a source listing for each terminal named.

-C Uses the termcap names.

-I Uses the terminfo(5) names.

`-L` Uses the long C variable name listed in the include file <`term.h`>.

`-r` When using `-C`, displays all capabilities in `termcap` form.

If no *termnames* are given, the `TERM` environment variable is used for the terminal name.

The source code produced by the `-C` option may be used directly as a `termcap` entry, but not all of the parameterized strings may be changed to the `termcap` format. `infocmp` attempts to convert most of the parameterized information, but unconverted information is plainly marked in the output and commented out. These strings should be edited by hand.

All padding information for strings is collected and placed at the beginning of the string, where `termcap` expects it. Mandatory padding (padding information with a trailing '/') becomes optional.

All `termcap` variables that are no longer supported by `terminfo`(5), but can be derived from other `terminfo`(5) variables, are displayed. Not all `terminfo`(5) capabilities are translated; only the variables that were part of `termcap` are normally displayed. Specifying the `-r` option removes this restriction, allowing all capabilities to be displayed in `termcap` form.

It is not always possible to convert a `terminfo`(5) string capability into an equivalent `termcap` format, because padding is collected to the beginning of the capability, not all capabilities are displayed, mandatory padding is not supported, and `termcap` strings are not as flexible. Subsequently converting the `termcap` file back into `terminfo`(5) format does not necessarily reproduce the original `terminfo`(5) source.

The following table shows some common `terminfo`(5) parameter sequences, their `termcap` equivalents, and some terminal types that commonly have such sequences:

`terminfo`	`termcap`	Representative terminals
%p1%c	%.	adm
%p1%d	%d	hp, ANSI standard, vt100
%p1%'x'%+%c	%+x	concept
%i	%i	ANSI standard, vt100
%p1%?%'x'%>%t%p1%'y'%+%;	%>xy	concept
%p2 is displayed before %p1	%r	hp

`use` String Capability Option

`-u` Produces a `terminfo`(5) source description of the first terminal *termname* that is relative to the sum of the descriptions given by the entries for the other terminals' *termnames*. It does this by analyzing the differences between the first *termname* and the other *termnames* and producing a description with the string capability `use` for the other terminals. In this manner, it is possible to retrofit generic `terminfo`(5) entries into a terminal's description. If two similar terminals exist but were coded at different times or by different people so that each description is a full description, using `infocmp` shows what can be done to change one description so that it is relative to the other.

A capability is displayed with an at-sign (@) if it no longer exists in the first *termname* but one of the other *termname* entries contains a value for it. A capability's value is displayed if the value in the first *termname* is not found in any of the other *termname* entries, or if the first of the other *termname* entries that has this capability specifies a different value for the capability than that in the first *termname*.

The order of the other *termname* entries is significant. Because the `terminfo`(5) compiler `tic`(8) does a left-to-right scan of the capabilities, specifying two `use` entries that contain differing entries for the same capabilities produces different results, depending on the order in which the entries are given. `infocmp` flags any such inconsistencies between the other *termname* entries as they are found.

Alternatively, specifying a capability after a `use` entry that contains that capability causes the second specification to be ignored. Using `infocmp` to recreate a description can be a useful check to ensure that everything was specified correctly in the original source description.

Another error that does not cause incorrect compiled files, but slows down the compilation time, is specifying extra `use` string capabilities that are superfluous. `infocmp` flags any other *termname* `use` entries that are not needed.

Other Options

`-s` *sort_options* Sorts the fields within each type. The `-s` option accepts the following arguments:

- `d` Leaves fields in the order in which they are stored in the `terminfo`(5) database
- `i` Sorts by `terminfo`(5) name
- `l` Sorts by the long C variable name
- `c` Sorts by the `termcap` name

If an `-s` option is not given, the fields displayed are sorted alphabetically by the `terminfo`(5) name within each type, except in the case of the `-C` or the `-L` options, which cause the sorting to be done by the `termcap` name or the long C variable name, respectively.

`-v` Prints out tracing information on standard error as the program runs.

`-V` Prints out the version of the program in use on standard error and exit.

`-w` *width* Changes the output to *width* characters.

`-1` Causes the fields to be displayed one to a line. Otherwise, the fields are displayed several to a line to a maximum width of 60 characters.

Change Databases Options

The location of the compiled `terminfo`(5) database is taken from the environment variable `TERMINFO`. If the variable is not defined, or if the terminal is not found in that location, the system `terminfo`(5) database, usually in `/usr/lib/terminfo`, is used. The `-A` and `-B` options may be used to override this location.

`-A` *directory* Sets `TERMINFO` for the first *termname*.

`-B` *directory* Sets `TERMINFO` for the other *termnames*.

With this, it is possible to compare descriptions for a terminal with the same name located in two different databases. This is useful for comparing descriptions for the same terminal created by different people. Otherwise, the terminals would have to be named differently in the `terminfo`(5) database for a comparison to be made.

ENVIRONMENT VARIABLES

`TERM` Used by screen editors and other screen-based programs to identify the terminal type in use. This variable need not be set if you are not going to use `vi`(1) or `more`(1). `TERM` maps the terminal's keyboard and screen characteristics to a `TERMINFO` definition. The value of `TERM` must be one of the known terminal definition names. These names are found in the `/usr/lib/terminfo` directory, which is organized into 36 directories (0 through 9 and a through z, as well as A through Z if necessary). Each terminal definition is kept in the directory that matches the first character of the terminal definition name.

`TERMINFO` Used to identify the path to an alternative terminal information (`TERMINFO`) directory. The `/usr/lib/terminfo` default directory file contains terminal definitions. You can create your own terminal definitions, compile them, and store them in one of your own directories. Commonly, this directory is `TERMINFO=$HOME/terminfo`. For the system to recognize an alternative directory, you must then set and export `TERMINFO`.

MESSAGES

`malloc is out of space!`
Not enough memory was available to process all of the terminal descriptions requested. Run infocmp several times, each time including a subset of the desired `termnames`.

`use= order dependency found:`
A value specified in one relative terminal specification was different from that in another relative terminal specification.

`´use=term' did not add anything to the description.`
A relative terminal name did not contribute anything to the final description.

`must have at least two terminal names for a comparison to be done.`
The `-u`, `-d`, and `-c` options require at least two terminal names.

FILES

`/usr/lib/terminfo/?/*` Database of compiled terminal descriptions

SEE ALSO

`tic`(8)

`tput`(1), `tset`(1B) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`curses`(3) (available only online)

`term`(5), `terminfo`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

`init`, `telinit` – Controls process initialization

SYNOPSIS

`/etc/init` [`0123456SsQqM`]

`/etc/telinit` [`0123456sSQqabcM`]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `init` command is a general process spawner that primarily creates processes from a procedure stored in the `/etc/inittab` file (see `inittab`(5)). `init` controls the state (run level) of the system by controlling autonomous processes required by the various states.

`init` considers the system to be in a certain run level at any given time. You can think of a run level as a software configuration of the system in which each configuration allows only a selected group of processes to exist. The processes spawned by `init` for each of these run levels are defined in the `/etc/inittab` file. `init` can be in one of eight run levels, 0 through 6 and `S` or `s`. An appropriately authorized user can change the run level by running `/etc/init` (which is linked to `/etc/telinit`). This user-spawned `init` sends appropriate signals to the original `init` spawned by the operating system when the system was rebooted, telling the original `init` which run level to change.

`init` is invoked inside the UNICOS operating system as the last step in the boot procedure. It is always process ID 1, and all other processes are spawned, directly or indirectly, from it. `init` first checks to see if there was a run-level specified in the initial UNICOS exchange package. On CRAY Y-MP E series, this is set by the OWS program `mfstart`. If this value is nonzero, `init` uses it for the initial run level. `init` next looks for `/etc/inittab` to see whether there is an entry of the type *initdefault* (see `inittab`(5)). If such an entry exists, and no run level was set in the exchange package, `init` uses the run level specified in that entry as the initial run level to enter. If this entry is not in `/etc/inittab` or `/etc/inittab` is not found, `init` requests that you enter a run level from the system console, `/dev/console`.

If an `S` or `s` is entered, `init` goes into the single-user level, which is the only run level that does not require a properly formatted `/etc/inittab` file to exist. If `/etc/inittab` does not exist, by default the only legal run level that `init` can enter is the single-user level. In the single-user level, the console terminal `/dev/console` is opened for reading and writing. To exit from the single-user run level, one of two options can be selected. First, if the shell is terminated (using an end-of-file), `init` reprompts for a new run level. Second, `init` or `telinit` can signal `init` and force it to change the run level of the system.

When booting the system, failure of `init` to prompt for a new run level could be because the device `/dev/console` is missing.

When `init` prompts for the new run level, you can enter only one of the digits 0 through 6 or the letters `S` or `s`. If you enter a digit from 0 through 6, `init` enters the corresponding run level. Any other input is rejected, and you are prompted again. If this is the first time `init` has entered a run level other than single-user mode, `init` first scans `/etc/inittab` for special entries of the type *boot* and *bootwait*. These entries are invoked, provided that the run level entered matches that of the entry before any normal processing of `/etc/inittab` occurs. In this way, any special initialization of the operating system, such as mounting file systems, can occur before users are allowed onto the system. `init` scans the `/etc/inittab` file to find all entries that are to be processed for that run level.

Run level 2 is usually defined by the administrator to contain all the terminal processes and daemons that are spawned in the multiuser environment. In a multiuser environment, the `/etc/inittab` file is usually set up so that `init` creates a process for each terminal on the system.

For terminal processes, ultimately the shell will terminate because of an end-of-file either typed explicitly or generated as the result of hanging up. When `init` receives a child process termination signal, telling it that a process it spawned has died, `init` records the fact and the reason the process died in `/etc/utmp` and `/etc/wtmp`, if such a file exists (see `who`(1)). A history of the processes spawned is kept in `/etc/wtmp`, if such a file exists.

To spawn each process in the `/etc/inittab` file, `init` reads each entry, and for each entry that should be respawned, it forks a child process. After it has spawned all the processes specified by the `/etc/inittab` file, `init` waits for one of its descendant processes to die or until `init` is signaled by `init` or `telinit` to change the system's run level. When one of the preceding conditions occurs, `init` reexamines the `/etc/inittab` file. New entries can be added to the `/etc/inittab` file at any time; however, `init` still waits for one of the preceding three conditions to occur. To provide for an instantaneous response, the `init Q` or `init q` command can wake `init` to reexamine the `/etc/inittab` file.

When `init` is requested to change run levels (using `telinit`), `init` sends the warning signal `SIGTERM` to all processes that are undefined in the target run level. On IOS model E systems, `init` also sends an advisory O packet with the new run level to the OWS. `init` waits `TWARN` (10) seconds before forcibly terminating these processes by using the kill signal `SIGKILL`.

The `telinit` command, which is linked to `init`, directs the actions of `init`. It takes a one-character argument and signals `init` by using the kill system call to perform the appropriate action.

The following arguments serve as directives to `init`:

`0` through `6`	Places the system in one of the run levels 0 through 6.
`a`, `b`, `c`	Processes only those `/etc/inittab` file entries having the `a`, `b`, or `c` run level set.
`Q`, `q`	Reexamines the `/etc/inittab` file.
`S`, `s`	Enters the single-user environment.

`M` Enters maintenance run level.
Note: Reserved for use by Cray Research. See NOTES subsection.

Only an appropriately authorized user can run `telinit`.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to run these commands and to change run levels.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to run these commands and to change run levels.

The maintenance run level (`M`) is similar to run levels 0 through 6, with the exception that the `bootwait` and `boot` functions are not performed if `M` is the intial run level. The maintenance run level is reserved for use by Cray Research; its functions may change without notice.

MESSAGES

If `init` finds that it is continuously respawning an entry from `/etc/inittab` more than 10 times in 2 minutes, it assumes that the command string contains an error and generates an error message on the system console; it refuses to respawn this entry until either 5 minutes has elapsed or it receives a signal from a user `init` (`telinit`). This prevents `init` from eating up system resources when someone makes a typographical error in the `/etc/inittab` file or when a program is removed that is referenced in the `/etc/inittab` file.

FILES

`/etc/inittab`

`/etc/utmp`

`/etc/wtmp`

SEE ALSO

`brc`(8), `cleantmp`(8), `getty`(8)

`login`(1), `sh`(1), `who`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`kill`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

`inittab`(5), `utmp`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

`initif` – Configures network interfaces

SYNOPSIS

`/etc/initif` [`-F` *family*] [`-f` *file*] [`-h` *header*] [`-q`] [*interfaces*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `initif` script configures network interfaces according to information contained in a tabular configuration file. Only the specified interfaces are configured. If no interfaces are specified, all interfaces listed in the configuration file will be configured.

When configuring interfaces, unless you use the `-q` option, `initif` prints a header line, which is not terminated by a new-line character, and prints the name of each interface successfully configured, terminating the list with a period and new-line character. The default header string is as follows:

`Configuring` *type* network interfaces:

type is a string that indicates the type of network interfaces being configured:

`all`	Default case of all listed interfaces.
`selected`	One or more interfaces have been specified on the command line.
`TCP/IP`	For all interfaces in the `inet` address family.

The following options are accepted by `initif`:

`-F` *family*	Specifies that only interfaces of address type *family* will be configured. *initif* will configure up only interfaces whose *family* field matches the specified *family*. (See the File Format subsection.) The match may be case-insensitive; that is, a specification of `-F INET` on the command line matches `inet` entries in the configuration file.
`-f` *file*	Specifies *file* as the configuration file from which to fetch information about interfaces to be configured. If `-f` is not specified, the default configuration file is used.
`-h` *header*	Specifies the string *header*, rather than the default, as the header printed before the list of interfaces configured. White space in a header string must be escaped from the shell.
`-q`	Specifies quiet mode; prints neither a header nor the names of interfaces configured.
interfaces	Specifies interfaces to be configured.

File Format

A configuration file for `initif` consists of a series of lines with the following format:

name hyfile family address destination arguments ...

The elements in an entry have the following meanings:

name Name of the interface (for example, `hy0`).

hyfile Name of the file that contains the hardware address-mapping information to be attached to the interface through the `hyroute`(8) command. A *hyfile* of `-` (one hyphen) indicates that no hardware address-mapping information should be attached to this interface. If *hyfile* begins with a leading slash (/), it is assumed to be the full path name of the file; otherwise, the file name is interpreted relative to the UNICOS configuration directory `/etc/config`.

family Address family to be used for the interface (typically `inet`).

address Network address to be associated with the interface.

destination Network address of the destination address if this interface is a point-to-point link. A *destination* of `-` (one hyphen) indicates that the interface is not a point-to-point link.

arguments Keyword-value pairs of arguments (for example, `netmask 0xffffff00 iftype hy`) to be passed to `ifconfig`(8) when bringing up the interface; see `ifconfig`(8) for a complete list of arguments.

An initial # character on a line indicates that the line is a comment.

MESSAGES

If interface *name* is not listed in configuration file *file*, the `initif` script issues the following message.

`initif: interface ‘`*name*`‘ not found in ‘`*file*`‘`

FILES

`/etc/config/interfaces` Default configuration file

SEE ALSO

`hyroute`(8), `ifconfig`(8)

NAME

`install` – Invokes UNICOS installation and configuration menu system

SYNOPSIS

`/etc/install/install` [`-C`] [`-l` *log_directory*] [`-P` *pager*] [`-r`] [`-T`] [`-t`]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The UNICOS installation and configuration menu system provides a menu-driven interface for installing, configuring, and maintaining the UNICOS operating system.

The installation and configuration system provides the following functions:

- Controls media access
  In read-only mode, installation actions are disabled, although other aspects of the installation menu system can run.
- Facilitates binary and source generation
  The binary and source generation functions change to a new root environment using the `chroot`(8) command. The installation and configuration menu system should not be invoked while in a changed-root environment.
- Verifies installation and configuration procedures

The `install` command accepts the following options:

`-C` Turns off the X Window System version of the UNICOS installation and configuration menu system (the default).

`-l` *log_directory*
Overrides the default directory `/etc/install`, containing the `.inlogpipe` pipe. If `/etc/install` is NFS mounted, this option moves the `.inlogpipe` to a non-NFS mounted area. This area must be under the mount point so that the `chroot`(8) command used to execute builds can access this directory.

`-P` *pager* Sets the `PAGER` variable used by the `man`(7D) command.

`-r` Sets read-only mode. If set, menu selections can be modified, but are not saved in `.sav` files. Menu actions are disabled. This option allows multiple users to enter the installation system at one time. However, only one user may be using the tool in non-read-only mode at a time, and this user must be super user.

`-T` Disables checking of the `TERM` environment variable for specified terminal types and lets users continue if `intcapchk` does not exist or fails.

`-t` Test mode. Disables `chroot` mode, so that when doing builds you are not in a changed-root environment.

FILES

```
/etc/install/*.sav
/etc/install/cfdb/*.cfg
/etc/install/*.mnu
```

SEE ALSO

`chroot`(8)

UNICOS Installation Guide, Cray Research publication SG–2112

UNICOS Configuration Administrator's Guide, Cray Research publication SG–2303

NAME

`iocstat` – Displays information about model E I/O cluster(s) (IOCs) attached to Cray PVP systems and their associated low-speed channel(s)

SYNOPSIS

`iocstat` [`-r` *rate*]

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

DESCRIPTION

The `iocstat` utility gets information from the MIOP table (in `epack.h`) and displays the information on the requester's screen. The MIOP table controls access and I/O operations to low-speed channels to model E I/O clusters. The information is attained by using the `tabread` system call.

The `iocstat` utility accepts the following option:

`-r` *rate* Updates the display each *rate* second(s) with new information from the MIOP table. If `-r` is not specified, the display is put out one time only.

The display has the following format:

```
     connection          # of packets          pkt size      retry
    CL  CH  state       in           out        in   out     in   out
    --  --  -----  ----------  ----------   ---- -----  ---- -----

     0  16   init     1192040     1192123   11.7   7.0      0     0
     1  18   init     2125998     2125806   10.0   7.0      0     0
```

The column headings in the preceding table have the following definitions:

`CL` The number of the I/O cluster. Possible cluster numbers range from 0 through 7 or, on the CRAY C90 series, from 0 through 15.

`CH` Low-speed channel on which the cluster is attached to the mainframe. Possible channel numbers are 16, 18, 20, 22, 24, 26, 28, and 30.

`state` Current state of the channel. The possible states are `init` and `down`. The `down` state means that the channel is not initialized currently.

`# of pkts` Number of I/O packets that have come from this I/O cluster to the mainframe since the last deadstart, both input and output.

`pkt size` Average size of input and output packets received from this cluster. This size is attained by dividing the total number of words transferred on this low-speed channel by the number of packets received. The total word count includes packet header information as well as the text.

`retry` Records the number of retry packets read (in) and written (out) to this cluster since deadstart. Retries are attempted when the input packet contains bad information (for example, a bad magic number or a trailer word that does not equal the header). The retry count reflects the number of errors that have been encountered on this channel.

The `-r` option takes the preceding information and runs it through a refreshing screen display. When using the `-r` option for a refreshing display, the following commands control the screen:

`>` Increase refresh time

`<` Decrease refresh time

`R` Scroll

`r` End scroll

`+` Next page of data

`-` Previous page of data

`q` Exit screen mode

SEE ALSO

`pddstat`(8) to display information about the IOS model E, from the disk table

NAME

ipi3_clear – Clears an IPI-3/IPI packet driver device

SYNOPSIS

/etc/ipi3_clear [-c] [-r] *devicename*

IMPLEMENTATION

All Cray Research systems (except CRAY J90 series and CRAY EL series)

DESCRIPTION

The ipi3_clear command terminates all outstanding requests to an IPI-3/IPI packet driver device. Requests that have finished processing but have not been returned to the user are discarded. No further user requests to the device are processed until the device is closed. You must specify a *devicename* when using the ipi3_clear command.

The ipi3_clear command accepts the following options:

-c Terminates all outstanding activity on the channel.

-r Issues a slave reset to the IPI-3/IPI device. A slave reset causes the device to reset itself to the initial start-up state. All information in the slave is lost.

EXIT STATUS

The ipi3_clear command completes successfully, 0 is returned; otherwise, a nonzero value is returned. Where possible, this exit status code is normalized to the last three digits. Exit status values are documented in the *Tape Subsystem User's Guide*, Cray Research publication SG–2051.

FILES

/dev/ipi3/*devicename*	IPI-3/IPI interface devices
/dev/ipi3/reqt	IPI-3/IPI interface device
/etc/config/ipi3_config	IPI-3/IPI configuration file

SEE ALSO

ipi3_config(8), ipi3_option(8), ipi3_start(8), ipi3_stat(8), ipi3_stop(8)

ipi3(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

Tape Subsystem Administration, Cray Research publication SG–2307

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

ipi3_config – Configures an IPI-3/IPI channel up or down

SYNOPSIS

/etc/ipi3_config [-c *channel*] [-C *cluster*] [-i *iop*] *state*

IMPLEMENTATION

All Cray Research systems (except CRAY J90 series and CRAY EL series)

DESCRIPTION

The ipi3_config command configures an IPI-3/IPI channel up or down. The options are as follows:

-c *channel*	Specifies the channel that will be configured.
-C *cluster*	Specifies the cluster in which the channel is configured.
-i *iop*	Specifies the IOP in which the channel is configured.
state	Specifies IPI-3/IPI channel state (either up or down).

NOTES

If you request a channel up configuration, a selective reset command will be sent to all slaves configured on the channel. The selective reset resets the burst size to the device default.

EXIT STATUS

The ipi3_config command completes successfully, 0 is returned; otherwise, a nonzero value is returned. Where possible, this exit status code is normalized to the last three digits. Exit status values are documented in the *Tape Subsystem User's Guide*, Cray Research publication SG–2051.

FILES

/dev/ipi3/*devicename*	IPI-3/IPI interface devices
/dev/ipi3/reqt	IPI-3/IPI interface device
/etc/config/ipi3_config	IPI-3/IPI configuration file

SEE ALSO

ipi3_clear(8), ipi3_option(8), ipi3_start(8), ipi3_stat(8), ipi3_stop(8)

ipi3(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

Tape Subsystem Administration, Cray Research publication SG–2307

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

ipi3_option – Modifies a IPI-3/IPI packet driver option(s)

SYNOPSIS

/etc/ipi3_option [-a *maximum-number-async-responses*] [-c *maximum-number-cmdlst*] [-i *maximum-number-iop-processes*] [-r *maximum-number-non-cmdlst*] [-t on|off]

IMPLEMENTATION

All Cray Research systems (except CRAY J90 series and CRAY EL series)

DESCRIPTION

The ipi3_option command modifies one or more IPI-3/IPI packet driver options. The options are as follows:

-a *maximum-number-async-responses*
: Specifies the maximum number of asynchronous responses that may be outstanding per device. Users cannot enable asynchronous responses that exceed this value.

-c *maximum-number-cmdlst*
: Specifies the maximum number of command list requests that may be outstanding per device. If the number of command list requests issued exceeds this value, an error is returned.

-i *maximum-number-iop-processes*
: Specifies the maximum number of processes that can open an IOP device concurrently. If this limit is exceeded, an error is returned.

-r *maximum-number-non-cmdlst*
: Modifies the maximum number of noncommand list requests that may be outstanding per device.

-t on|off
: Specifies the tracing state (either on or off).

NOTES

If a process has an IOP device open, you cannot modify the maximum number of processes that may open an IOP.

EXIT STATUS

The ipi3_option command completes successfully, 0 is returned; otherwise, a nonzero value is returned. Where possible, this exit status code is normalized to the last three digits. Exit status values are documented in the *Tape Subsystem User's Guide*, Cray Research publication SG–2051.

FILES

`/dev/ipi3/`*devicename*	IPI-3/IPI interface devices
`/dev/ipi3/reqt`	IPI-3/IPI interface device
`/etc/config/ipi3_config`	IPI-3/IPI configuration file

SEE ALSO

`ipi3_clear`(8), `ipi3_config`(8), `ipi3_start`(8), `ipi3_stat`(8), `ipi3_stop`(8)

`ipi3`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

Tape Subsystem Administration, Cray Research publication SG–2307

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

`ipi3_start` – Starts the IPI-3/IPI packet driver subsystem or a single IOP

SYNOPSIS

`/etc/ipi3_start` [`-C` *cluster*] [`-i` *iop*]
`/etc/ipi3_start` [`-f` *config-file*]

IMPLEMENTATION

All Cray Research systems (except CRAY J90 series and CRAY EL series)

DESCRIPTION

The `ipi3_start` command starts either the IPI-3/IPI subsystem or starts a single I/O processor (IOP) within the IPI-3/IOP subsystem.

If when you are starting the IPI-3/IPI subsystem, the `ipi3_start` command reads and processes the IPI-3/IPI configuration from a file provided by the user and communicates this configuration to the IPI-3/IPI packet driver and to the IPI-3/IPI IOP. For each IOP configured, `ipi3_start` creates a corresponding IOP file, which only issues requests to the IOP. For each device attached to the IOP, `ipi3_start` creates a corresponding device file, which issues packets that affect an IPI-3 device.

If you specify a single IOP, requests are sent to the IOP to redefine the configuration of that IOP.

The `ipi3_start` command accepts the following options:

`-C` *cluster* Specifies the cluster of the IOP to be restarted. You can specify this option only if the IPI-3/IPI subsystem has already been started.

`-f` *config-file* Specifies the configuration file. The default configuration file is `/etc/config/ipi3_config`.

`-i` *iop* Specifies the IOP to be restarted. You can specify this option only if the IPI-3/IPI subsystem has already been started.

NOTES

Before executing the `ipi3_start` command, you must stop all configured IOP drivers.

EXIT STATUS

The `ipi3_start` command completes successfully, 0 is returned; otherwise, a nonzero value is returned. Where possible, this exit status code is normalized to the last three digits. Exit status values are documented in the *Tape Subsystem User's Guide*, Cray Research publication SG–2051.

FILES

`/dev/ipi3/`*devicename*	IPI-3/IPI interface devices
`/dev/ipi3/reqt`	IPI-3/IPI interface device
`/etc/config/ipi3_config`	IPI-3/IPI configuration file

SEE ALSO

`ipi3_clear`(8), `ipi3_config`(8), `ipi3_option`(8), `ipi3_stat`(8), `ipi3_stop`(8)

`ipi3`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

Tape Subsystem Administration, Cray Research publication SG–2307

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

ipi3_stat – Displays device information

SYNOPSIS

```
/etc/ipi3_stat [-c] [-C cluster] [-d devname] [-i iop] [-t]
/etc/ipi3_stat [-o]
```

IMPLEMENTATION

All Cray Research systems (except CRAY J90 series and CRAY EL series)

DESCRIPTION

The ipi3_stat command displays device status, configuration, packet driver table, or the packet driver options statistics. The status, configuration, or table values for all devices can be displayed, or a partial display can be requested (for example, you can request a cluster, IOP, or a particular device to be displayed). The status of all devices is the default display.

The ipi3_stat command accepts the following options:

-c	Specifies that the configuration will be displayed. By default, the status is displayed. The -c, -o, and -t options are mutually exclusive.
-C *cluster*	Specifies the cluster in which all configured devices should be displayed.
-d *devname*	Specifies the device for which information is displayed.
-i *iop*	Specifies the IOP in which all configured devices should be displayed.
-t	Specifies that the packet driver table be displayed. By default, the status is displayed. The -c, -o, and -t options are mutually exclusive.
-o	Specifies that the options should be displayed. You cannot specify the -o option with any other options.

When you request device configuration (-c) information, the following information is displayed:

burst	Displays the burst size.
ch	Displays the device channel number.
chst	Displays the channel status (up or down).
chtyp	Displays the channel adapter type.
device	Displays the device name.
dvst	Displays the device status (up or down).
dvtyp	Displays the device type.
ioc	Displays the cluster in which the device is configured.

iop Displays the IOP in which the device is configured.

ltmo Displays the long device time-out.

sl Displays the slave address.

slst Displays the slave status (`up` or `down`).

stmo Displays the short device time-out.

If you do not specify `-c`, `-t`, or `-o`, the following information is displayed:

ch Displays the channel in which the device is configured.

chst Displays the status (`up` or `down`) of the channel.

chtyp Displays the type of hardware channel adapter.

dvtyp Displays the device type.

device Displays the device name.

ioc Displays the cluster in which the device is configured.

iop Displays the IOP in which the device is configured.

If you specify `-t`, the following table information is displayed:

async Displays the number of queued asynchronous packets. Queued packets are those returned by the IOP but not received by the user.

cmd Displays the number of command list requests that are outstanding to the IOP.

device Displays the device name.

enabl Displays the number of asynchronous packets that are currently enabled.

flag Displays the flags used by the IPI-3 packet driver. You can set *flag* to the following values:

- 0001 The user has opened the device file.
- 0002 An IOP request has timed out.
- 0004 The packet interface has been enabled.
- 0010 The device is open.
- 0020 A signal has been registered.
- 0100 A clear device is in progress.
- 0200 The device has been cleared.
- 0400 A device process is waiting for an interrupt.

lock Displays the number of locks on the process.

ios Displays the number of outstanding IOP requests.

ord Displays the table ordinal.

pid Displays the process ID of the open device.

rsyn Displays the resynchronization code.

sig Displays the signal sent to the user when a packet is returned from the IOP.

usr Displays outstanding user requests.

EXIT STATUS

The `ipi3_stat` command completes successfully, 0 is returned; otherwise, a nonzero value is returned. Where possible, this exit status code is normalized to the last three digits. Exit status values are documented in the *Tape Subsystem User's Guide*, Cray Research publication SG–2051.

FILES

`/dev/ipi3/`*devicename*	IPI-3/IPI interface devices
`/dev/ipi3/reqt`	IPI-3/IPI interface device
`/etc/config/ipi3_config`	IPI-3/IPI configuration file

SEE ALSO

`ipi3_clear`(8), `ipi3_config`(8), `ipi3_option`(8), `ipi3_start`(8), `ipi3_stop`(8)

`ipi3`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

Tape Subsystem Administration, Cray Research publication SG–2307

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

ipi3_stop – Stops the IPI-3/IPI subsystem or a single IOP

SYNOPSIS

/etc/ipi3_stop [-C *cluster*] [-i *iop*] [-k] [-u] [-w]

IMPLEMENTATION

All Cray Research systems (except CRAY J90 series and CRAY EL series)

DESCRIPTION

The ipi3_stop command shuts down either the entire IPI-3/IPI packet driver subsystem or a single I/O processor (IOP) within the IPI-3/IPI subsystem.

If you are bringing down the entire subsystem, ipi3_stop prevents processes from opening the I/O processor (IOP) and IPI-3 devices, waits for current activity to cease, and if requested, issues stop driver requests to all IOPs.

If you are stopping one IOP, ipi3_stop stops the IOP driver for that IOP. Either it kills all processes that have the IOP device open or all processes that have an IPI-3/IPI device configured on this IOP open, or it leaves the processes running but disables the handling of any future requests to the devices. The command also prevents other proecesses from opening the IOP and IPI-3 devices configured on this IOP.

The ipi3_stop command accepts the following options:

-C *cluster* Specifies the cluster number of the IOP to be stopped.

-i *iop* Specifies the IOP to be stopped.

-k Specifies that all processes that have the specified IOP device open or that have an IPI-3 device configured on the IOP open, will be killed. You can only use this option when you are shutting down a single IOP.

If you omit this option, all further requests to these devices are returned with an error.

-u Leaves the IOP drivers up.

-w Waits for all activity to stop and for all devices to be closed. By default, if the IPI-3/IPI packet driver is not idle, an error is returned.

EXIT STATUS

The ipi3_stop command completes successfully, 0 is returned; otherwise, a nonzero value is returned. Where possible, this exit status code is normalized to the last three digits. Exit status values are documented in the *Tape Subsystem User's Guide*, Cray Research publication SG–2051.

FILES

`/dev/ipi3/`*devicename*	IPI-3/IPI interface devices

SEE ALSO

`ipi3_clear`(8), `ipi3_config`(8), `ipi3_option`(8), `ipi3_start`(8), `ipi3_stat`(8)

`ipi3`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

Tape Subsystem Administration, Cray Research publication SG–2307

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

kcompress – Compresses a UNICOS kernel file

SYNOPSIS

/etc/kcompress *kernel1 kernel2*

/etc/kcompress [-d] *kernel2 kernel1*

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The kcompress program compresses a UNICOS kernel file. A decompression routine is added to the compressed kernel so it decompresses itself at boot time. Then a compressed kernel can be used to boot a system just as an uncompressed kernel would be used.

Compression saves space, because the uncompressed version of the kernel does not need to be stored, as it can always be recovered by using the kcompress -d option. Compression also saves time in the boot process, by reducing the amount of data that must be moved from the operator workstation (OWS) to the mainframe.

The kcompress command accepts the following option and operands:

-d Decompresses the kernel specified.

kernel1 Specifies the uncompressed kernel file. This operand is required.

kernel2 Specifies the compressed kernel file. This operand is required.

NOTES

Kernel compression is performed automatically. To compress or decompress any version of the UNICOS kernel earlier than UNICOS 7.0, the kcompress program must be used manually. To determine whether or not a kernel has been compressed, use the size(1) command.

The symbol table is left uncompressed, so programs such as crash will work directly with a compressed kernel. The kcompress program modifies the initial exchange package, making it machine-type-dependent. Therefore, kcompress should always be targeted for the same machine type as the kernel being compressed.

EXAMPLES

Example 1: Use the following command line to compress the file unicos.k.1 to the file unicos.k.2:

```
% /etc/kcompress unicos.k.1 unicos.k.2
```

Example 2: Use the following command line to decompress the file `unicos.k.2` to the file `unicos.k.1`:

```
% /etc/kcompress -d unicos.k.2 unicos.k.1
```

Example 3: Use the `size`(1) command to determine whether or not a kernel has been compressed:

```
% /bin/size unicos.k.*
unicos.k.1: 844021 + 0 + 0 = 844021
unicos.k.2: 276251 + 0 + 567770 = 844021
```

The first kernel is not compressed, and it shows that it has 844,021 words of code. The second kernel has been compressed, and it shows 276,251 words of code and 567,770 words of *bss*. The code size shown is the size of the compressed image, and the *bss* number represents how much it will expand.

SEE ALSO

`size`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

NAME

kerbd – Generates and validates Kerberos tickets for Kerberized NFS

SYNOPSIS

```
kerbd [-d]
```

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The user level daemon, kerbd, talks to the kernel and the Kerberos key distribution center (KDC) to generate and validate Kerberos authentication tickets. kerbd maps Kerberos user names into local user and group IDs. By default, all groups that the requested user belongs to are included in the grouplist credential.

The kerbd daemon accepts the following options:

-d Runs in debug mode. kerbd outputs information about Kerberos tickets when they are processsed.

SEE ALSO

kdestroy(1), kinit(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

kerberos(3K) in the *Kerberos User's Guide*, Cray Research publication SG–2409

krb.conf(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

keyenvoy – Serves as intermediary to keyserv(8)

SYNOPSIS

/etc/keyenvoy

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The keyenvoy process is used by secure Remote Procedure Call (RPC) programs to communicate with the key server, keyserv(8). For security reasons, the key server does not communicate with anything but local root processes; keyenvoy is a setuid root process that acts as an intermediary between a user process and the key server.

NOTES

The keyenvoy process is dynamically created and destroyed by both client and server programs that use secure RPC. The process of creating and destroying the keyenvoy process occurs within the RPC library, and it is transparent to the user.

You can find further information about using secure RPC in the *Remote Procedure Call (RPC) Reference Manual*, Cray Research publication SR–2089.

The keyenvoy program cannot be run interactively.

SEE ALSO

keyserv(8)

Remote Procedure Call (RPC) Reference Manual, Cray Research publication SR–2089

NAME

keyserv – Stores public and private encryption keys

SYNOPSIS

/etc/keyserv [-n]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The keyserv daemon is used for storing the private encryption keys of each user logged into the system. These encryption keys are used for accessing secure network services, such as Secure RPC. When a user logs in to the system, the login(1) program usually uses the password to decrypt the user's private encryption key stored in a network information service (NIS) database, and then the keyserv daemon stores the decrypted key.

Usually, the key for root is read from the /etc/.rootkey file when the daemon is started up. This is useful during power-fail reboots when no one is around to type a password, yet you still want the secure network services to operate normally.

The keyserv daemon is a Remote Procedure Call (RPC) program, which is registered with portmap(8) as program number 100029. The keyserv daemon is normally initiated from the netstart (or equivalent) script at boot time.

The keyserv daemon catches the SIGHUP signal and reregisters itself with portmap(8) when it receives the signal. This enables keyserv to continue running properly when portmap must be restarted.

The keyserv daemon accepts the following option:

-n Prompts for the password to decrypt the root key stored in the NIS, rather than reading the root key from /etc/.rootkey. The decrypted key is then stored in /etc/.rootkey for future use. This option is useful if the /etc/.rootkey file ever becomes out-of-date or corrupted.

FILES

/etc/.rootkey File that stores the root key

SEE ALSO

keyenvoy(8), portmap(8)

keylogin(1), login(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

NAME

killall – Kills all active processes

SYNOPSIS

/etc/killall [-n *namelist*] [-x *pid*[,*pid...*]] [*signal*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The killall command is used by the shutdown(8) procedure to kill all active processes not directly related to shutdown. The killall command terminates all processes with open files so that the mounted file systems will not be busy and can be unmounted.

killall sends *signal* (see kill(1)) to all remaining processes not belonging to the preceding group of exclusions and not specifically excluded by the -x option. If no *signal* is specified, the killall command uses a default of 9.

Only an appropriately authorized user can use this command.

The killall command accepts the following options:

-n *namelist* Specifies an alternative system *namelist* file in place of /unicos.

-x *pid* Excludes the given process IDs from the list of processes that receive the *signal*.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

SEE ALSO

shutdown(8)

kill(1), ps(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

signal(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

NAME

klogind – Remote login server

SYNOPSIS

```
/etc/klogind
/etc/eklogind
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

klogind is the server for the Kerberos version of the rlogin(1B) program. The server provides a remote login facility with authentication provided by Kerberos.

klogind listens for service requests at the port indicated in the klogin or eklogin service specification; see services(5).

Invocation as klogind is intended for normal hosts to which password access is granted if Kerberos authorization fails. Invocation as eklogind provides an encrypted communications channel.

When a service request is received, the server checks the client's source address and requests the corresponding host name (see gethostbyaddr(3C), hosts(5), and named(8)). If the host name cannot be determined, the dot-notation representation of the host address is used.

After the source address is checked, klogind allocates a pseudo-terminal (see pty(4)), and manipulates file descriptors so that the slave half of the pseudo terminal becomes the stdin, stdout, and stderr for a login process.

The parent of the login process manipulates the master side of the pseudo terminal, operating as an intermediary between the login process and the client instance of the rlogin program. When klogind is invoked as eklogind, all data that passes over the network is encrypted. In normal operation, the packet protocol that is described in pty(4) is invoked to provide ^S/^Q type facilities and to propagate interrupt signals to the remote programs. The login process propagates the client terminal's baud rate and terminal type, as found in the TERM environment variable (see environ(7)). The screen or window size of the terminal is requested from the client, and the window size changes from the client are propagated to the pseudo terminal.

Because of export controls, data stream encryption through the use of eklogind is not supported outside of the USA and Canada.

MESSAGES

All diagnostic messages are returned on the connection associated with the `stderr`, after which any network connections are closed. An error is indicated by a leading byte with a value of 1.

Messages are as follows:

`Try again.` This message indicates that a `fork`(2) system call by the server failed.

`/bin/sh: . . .` This message indicates that the user's login shell could not be started.

BUGS

A more extensible protocol must be used.

SEE ALSO

`named`(8)

`gethost`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`kerberos`(3K) in the *Kerberos User's Guide*, Cray Research publication SG–2409

`hosts`(5), `services`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

`environ`(7) (available only online)

NAME

krbipd – Validates Kerberos ticket address for Kerberos servers

SYNOPSIS

/etc/krbipd [-d]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The krbipd daemon is a Remote Procedure Call (RPC)-based server that processes address checking requests from klogind(8), kshd(8), and other Kerberos servers. The daemons klogind(8) or kshd(8) send a request to krbipd when they detect an address mismatch between the address in the Kerberos service ticket and the interface from which the ticket was received. If the addresses do not match, the Incorrect network address message is displayed, and the request is rejected.

If the klogin(1) request to klogind(8) was sent through another interface than the one used to request the service ticket, a mismatch occurs. krbipd checks the address it receives from klogind(8) against a list of Internet Protocol (IP) addresses configured for the machine. If the klogind(8) address matches one of these addresses, krbipd tells klogind(8) the address matched. The klogind(8) daemon processes the authentication request.

The krbipd daemon accepts the following option:

-d Executes in debug mode. krbipd displays information about addresses being processed.

SEE ALSO

klogind(8), kshd(8)

kerberos(3K) in the *Kerberos User's Guide*, Cray Research publication SG–2409

NAME

kshd – Provides remote shell server function

SYNOPSIS

/etc/kshd [-S *tos*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The kshd is the server for the kerberized versions of the rsh and rcp commands. The server provides remote execution facilities with authentication based on Kerberos.

The kshd command accepts the following option:

-S *tos* Directs kshd to set the IP Type-of-Service (TOS) option on the connection to the *tos* value. This value can be a numeric TOS value or a symbolic TOS name found in the /etc/iptos file.

The kshd server listens for service requests at the port indicated in the kshell service specification (typically 544) (see services(5)). When a service request is received, the following protocol is initiated:

1. The server reads characters from the socket up to a null (\0) byte. The resultant string is interpreted as an ASCII number, base 10.
2. If the number received in step 1 is nonzero, it is interpreted as the port number of a secondary stream to be used for the stderr. A second connection is then created to the specified port on the client's machine.
3. The server checks the client's source address and requests the corresponding host name (see gethostbyaddr(3C), hosts(5), and named(8)). If the host name cannot be determined, the dot-notation representation of the host address is used.
4. A Kerberos ticket and authenticator pair are retrieved on the initial socket.
5. A null-terminated user name that consists of a maximum of 16 characters is retrieved on the initial socket. This user name is interpreted as a user identity to use on the server's machine.
6. A null-terminated command to be passed to a shell is retrieved on the initial socket. The length of the command is limited by the upper limit on the size of the system's argument list.
7. kshd validates the user according to the following steps:
   a. The local (server-end) user name is looked up in the password file, and a chdir(2) to the user's home directory is performed.
   b. If either the lookup or chdir(2) fails, the connection is terminated.

c. The `.klogin` file in the home directory mediates access to the account (through `kuserok`(3K)) by the Kerberos principal specified in the ticket or authenticator. If this authorization check fails, the connection is terminated.

8. A null byte is returned on the initial socket and the command line is passed to the normal login shell of the user. The shell inherits the network connections established by `kshd`.

NOTES

On a UNICOS multilevel security (MLS) system, if this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the action shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

On a UNICOS non-MLS system or a UNICOS MLS system with PRIV_SU enabled, the super user is allowed to use this command.

MESSAGES

Except for the last message listed in this section, all diagnostic messages are returned on the initial socket, after which any network connections are closed. An error is indicated by a leading byte with a value of 1 (0 is returned in step 8 when all of the steps prior to the execution of the login shell complete successfully).

`remuser too long`	The name of the user on the remote machine consists of more than 16 characters.
`Command too long`	The command line exceeds the size of the argument list (as configured into the system).
`Login incorrect`	No password file entry for the user name existed.
`No remote directory`	The `chdir`(2) command to the home directory failed.
`Permission denied`	The authorization procedure described previously failed.
`Can't make pipe`	The pipe needed for the `stderr` was not created.
`Try again`	A `fork`(2) by the server failed.
shellname: ...	The user's login shell could not be started. This message is returned on the connection associated with `stderr`, and it is not preceded by a flag byte.

BUGS

A facility to allow all data exchanges to be encrypted should be present. You should use a more extensible protocol.

Because of export controls, data stream encryption is not supported outside of the United States and Canada.

SEE ALSO

`named`(8)

`gethost`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`kerberos`(3K), `kuserok`(3K) in the *Kerberos User's Guide*, Cray Research publication SG–2409

`hosts`(5), `services`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

`labelit` – Reads or writes file system labels and security labels

SYNOPSIS

`/etc/labelit` [`-c` *comparts*] [`-l` *minslvl*] [`-u` *maxslvl*] [`-s`] *device* [*fsname volname*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `labelit` command provides initial labels or changes existing labels for unmounted file systems. You must be an appropriately authorized user to use this command; see the NOTES section.

Whenever you use `labelit` to set a label on a file system, the action is recorded in the security log.

If you do not specify options or operands, `labelit` prints current label values, the file system's minimum and maximum security levels, and valid compartments. If no options or operands are specified, the file system may be mounted. When setting the file system label range, the maximum label must dominate the minimum label.

The `labelit` command accepts the following options and operands:

`-c` *comparts*	Specifies the file system's valid compartment set to be written in the label. This option is not available on pre-UNICOS 6.0 file systems.
`-l` *minslvl*	Specifies the file system's minimum security level to be written in the label. Can be either a decimal integer or level name.
`-u` *maxslvl*	Specifies the file system's maximum security level to be written in the label. Can be either a decimal integer or level name.
`-s`	Permits changing the security level or compartments on a file system after the label has been set initially. You must specify the `-u`, `-l`, and `-c` options when using the `-s` option.
device	Name of block special file for file system; required operand.
fsname volname	Specifies the file system name, *fsname*, and the volume name, *volname*, to be written in the label (see `fs`(5)).

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories may perform the actions shown:

Active Category	**Action**
`system`, `secadm`	Allowed to use this command.

`sysadm` Allowed to use this command. Shell redirected I/O is subject to security label restrictions.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

The following two options have been preserved for compatibility with previous versions of the `labelit` command:

`-f` *fsname* Specifies the file system name to be written in the label.

`-v` *volname* Specifies the volume name to be written in the label (see `fs`(5)).

Minimum and maximum security levels are checked to verify that *minslvl* is less than or equal to *maxslvl*.

Mounted file systems can be labeled with `labelit`; however, the new label information will not take effect until the file system has been remounted (that is, unmounted, then mounted again).

SEE ALSO

`fsck`(8), `mkfs`(8), `mount`(8)

`df`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`fs`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

`lastlogin` – Records last login of each user

SYNOPSIS

`/usr/lib/acct/lastlogin` [`-c` *infile*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `lastlogin` command updates `/usr/adm/acct/sum/loginlog` with the last date on which each user logged in. `lastlogin` is invoked by both the standard UNIX System V accounting and Cray Research system accounting (CSA) packages as part of the daily accounting run. By default, `lastlogin` processes the file `/usr/adm/acct/nite/ctacct.`*MMDD*, which is a standard UNIX System V accounting total accounting file in `tacct.h` format.

The `lastlogin` command accepts the following option and argument:

`-c` *infile* Specifies that `lastlogin` should process *infile*, which is in CSA `cacct.h` format.

EXAMPLES

The following example is a possible entry for the `/usr/spool/cron/crontabs/root` file so that `cron`(8) automatically runs `dodisk`(8):

```
30  10  *  *  1-5  /usr/lib/acct/dodisk -a -v 2> /usr/adm/acct/nite/dsklog
```

FILES

`/usr/adm/acct/sum` Summary directory

SEE ALSO

`acct`(8), `acctsh`(8), `prdaily`(8), `runacct`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

ldcache – Assigns and displays logical device cache

SYNOPSIS

```
/etc/ldcache -l dev [-h high[,low]] [-n units] [-r rate] [-s size] [-t type] [-x max[,min]]
[-p] [-w]
/etc/ldcache [-a] -b
/etc/ldcache [-a] -i
/etc/ldcache [-f file]
/etc/ldcache [-]
```

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The ldcache command assigns and displays SSD or BMR cache characteristics for logical devices; it may be issued at any time. SSD space for logical device cache is allocated from the SDS (see ssbreak(2)) memory pool. Cache for logical devices is specified as a number of units and the size of each unit. The ratio between these values is chosen depending on the predicted number of files usually open on the device at the same time.

All I/O to a file system goes through the logical device cache unless you specify that a particular file should bypass the cache. You can bypass the logical device cache by setting the O_LDRAW and the O_RAW flags with the open(2) system call.

An appropriately authorized user can assign, change, or release cache by using the following options:

-l *dev* Specifies the name or number of the logical device. Optionally, *dev* may be the mount point (file system name) of a logical device. If *dev* is a device or file system name, it must begin with /. If the file system name is used, the file system must be mounted.

-h *high*, *low* Specifies threshold values for dirty units in cache. *high* specifies the maximum number of dirty units that may be in cache at any one time. If the number of dirty units exceeds *high*, new requests to dirty units will sleep until the number falls below the threshold. When the number of dirty units in cache exceeds *low*, the system automatically starts flushing the oldest dirty units down to the level specified by *low*. If *low* is not specified, it defaults to the same value as *high*. To disable the threshold parameters set *high* to 0 (the default).

-n *units* Specifies the number of cache units to be assigned. If 0, all cache for the device is released.

`-s` *size* Specifies the size of each cache unit in 4096-byte blocks. The size specified must be a multiple of the largest I/O unit of any of the underlying devices of any of the underlying physical slices (pdds) for the logical device (specified with `-l`). The I/O unit of a pdd is its sector size in 4096-byte blocks. For example, the I/O unit size (sector size) of a DD-60 disk drive is 4. The I/O unit size (sector size) of an SSD is 1. The system is shipped with a maximum cache unit size of 256.

`-t` *type* Specifies the memory type for cache. The value for *type* can be `BMR` (buffer memory), `MEM` (main memory), or `SSD`. The default is `SSD`.

`-x` *max,min* Defines the `ldcache` aging parameters. *max* specifies the maximum age in seconds any unit in the cache may be before `ldcache` starts automatically flushing units. All units older than *min* seconds are flushed. If *min* is not specified, it defaults to the same value as *max*. To disable trickle sync set *max* to 0 (the default).

`-p` Specifies that cache is to be assigned on a per-disk basis. When this option is used `-n` number of units of cache is assigned to each physical device composing the logical device. Each unit is assigned to a particular physical device and may not be allocated to another physical device in the logical device.

`-w` Specifies that the cache unit is "write-through." When this option is specified, the write block is not kept in the logical device cache, but it is written asynchronously to disk. This option should be used only when `ldcache -a` shows a significantly greater number of reads than writes; using this option decreases performance for other types of applications.

CAUTION: Using this option significantly decreases performance for applications that write data out, then immediately reuse the data.

An appropriately authorized user can also assign, change, or release cache entries by using an input file (or `stdin`) with the following options:

`-f` *file* Specifies the file in which the cache descriptions reside. The file format appears as described below.

`-` Specifies that the cache descriptions should be read from `stdin`. The input format appears as described below.

The cache description file must be in the following format:

logical_device type #_units size_4k_blocks [max[,min] [high[,low]]]

Each field is separated by white space. All lines that begins with a # character (or blank lines) are ignored.

Example:

```
/dev/dsk/ptmp  SSD  500  48  300,240  400,350
```

Any user can display cache statistics by using the following options:

`-a` Displays devices that have any read or write operations, even though no cache is attached. You cannot specify this option with the `-l` or `-r` option.

`-b` Displays cache to user and cache to disk statistics. This gives the best indication of the effectiveness of cache. You cannot specify this option with the `-l` or `-r` options.

`-i` Displays the static configuration parameters for each ldcached device. You cannot specify this option with the `-l` or `-r` options.

`-r` *rate* Specifies the refresh rate (in seconds) for a detailed display. The default rate is 1 second. You cannot specify this option with the `-a`, `-b`, or `-i` options.

If no options are specified, `ldcache` displays general information about all devices with cache in the following format:

```
T  unit   size   reads   writes   hits   misses   rate   name
-  -----  -----  ------  -------  -----  -------  -----  -----
```

The `T` column indicates the memory type; `B` indicates BMR, `M` indicates main memory, and `S` indicates SSD.

If you specify the `-b` option, the output is in the following format:

```
 Cache to user    Cache to disk      Cache/disk ratio
Reads   Writes   Reads   Writes   Read    Write   Total    Name
------  -------  ------  -------  ------  ------  -------  -----
```

If a device is specified (using the `-l` option) without parameters to modify its cache, `ldcache` provides a display of detailed information refreshed at either the default rate of 1 second or at the rate specified with the `-r` option. This display provides information relative to the time `ldcache` is invoked and is useful for monitoring a cache for a particular time slice.

The display format is as follows:

```
device name             time

                         Read data  Write data
                         ---------  ----------
Blocks transferred:
Cache to user:
Cache to disk:
Cache/disk ratio:
Avg request length:
Lst transfer rate:
Max transfer rate:
Cache hits:
Cache misses:
Cache hit rate:
```

This display also accepts the following commands:

n Goes to next device with cache attached

+ Increases refresh interval by 1 second

- Decreases refresh interval by 1 second

c Clears counters to 0

NOTES

If this command is installed with a privilege assignment list (PAL), a user who is assigned the following privilege text upon execution of this command is allowed to perform the action shown:

Privilege Text	Action
set	Allowed to assign, change, and release ldcache entries.

If this command is installed with a PAL, a user with one of the following active categories is allowed to perform the action shown:

Active Category	Action
system, secadm, sysadm, sysops	Allowed to assign, change, and release ldcache entries.

If the PRIV_SU configuration option is enabled, the super user is allowed to assign, change, and release ldcache entries.

SEE ALSO

ldsync(8)

privtext(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

open(2), ssbreak(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

dsk(4), ssd(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

ldsync – Flushes logical device cache to disk

SYNOPSIS

/etc/ldsync [-l *dev*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The ldsync command flushes data from all logical device caches to disk. The -l option can be used to flush cache for a single logical device. Only data that has been written to cache but not disk is affected. ldsync does not invalidate data in the cache.

-l *dev* Specifies the name or number of the logical device.

When you use the -l option to use ldsync on a logical device basis, disable LDSYNCTM. To do this, manually set ldsynctm in the /etc/inittab file or change LDSYNCTM in /usr/src/cmd/init/conf.c to a value greater than 1000000 and rebuild /etc/init.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm, sysops	Allowed to specify any device.

If the PRIV_SU configuration option is enabled, the super user is allowed to specify any device.

CAUTIONS

The ldsync command should always be issued before taking down the system. It should always be issued after a sync(1) command, rather than before it; otherwise, the sync data for an ldcache file system does not make it to the disk until the next ldsync is issued.

SEE ALSO

ldcache(8)

sync(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

sync(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

General UNICOS System Administration, Cray Research publication SG–2301

NAME

link, unlink – Executes link(2) and unlink(2) system calls

SYNOPSIS

/etc/link *oldfile newfile*
/etc/unlink *file*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The link and unlink commands perform their respective system calls on their operands, abandoning all error checking.

NOTES

The unlink command does not remove a directory from the file system, it simply unlinks the reference from the specified directory. Use of unlink(8) on directories by privileged users can cause file system errors (unlinked inodes) which can be fixed using fsck(8). A privileged user should use rmdir(8) to remove a directory from the file system.

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Allowed to link or unlink any file or directory.
sysadm	Allowed to link or unlink any file or directory that has the same security label as the user. Shell redirected I/O is subject to security label restrictions.

If the PRIV_SU configuration option is enabled, the super user is allowed to link or unlink any file or directory.

SEE ALSO

rm(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

link(2), unlink(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

General UNICOS System Administration, Cray Research publication SG–2301

NAME

lmdd – Moves I/O for performance and debugging tests

SYNOPSIS

lmdd [*option=value*] ...

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

lmdd copies a specified input file to a specified output and prints out timing statistics. Primarily used for timing I/O.

The lmdd command accepts the following options:

if=*name* Input file taken from *name*. The following indicate standard input: -, 0, stdin. Default *name* is internal, a special file that mimics /dev/zero by providing a buffer of zeroes without doing a system call.

of=*name* Output file taken from *name*. The following indicate standard input: -, 1, stdin. Default *name* is internal, a special file that mimics /dev/null by removing data without doing a doing a system call.

bs=*n* Input and output block size are *n* bytes. Default is 8192. Block size can be followed by k, m, or g to indicate kilobytes (*1024), megabytes (*1024*1024), or gigabytes (*1024*1024*1024) respectively. bs=*n* is different from dd(1), which has a 512 byte default.

ipat=*n* Expects a known pattern in the file if *n* is non zero. (See *opat* option below.) The pattern is a sequence of four byte integers: the first is 0, the second is 1, and so on. The default is not to check for the pattern. Mismatches are displayed.

opat=*n* Generates a known pattern on the output stream if *n* is non zero. Used for debugging file system correctness. The default is not to generate the pattern.

mismatch=*n*
Stops at the first mismatched value if *n* is non zero. Used with the *ipat* option above.

skip=*n* Skips *n* input blocks before starting copy.

fsync=*n* Calls fsync(2) on the output file before exiting or printing timing statistics if *n* is non zero.

sync=*n* Calls sync(2) before exiting or printing timing statistics if *n* is non zero.

rand=*n* Turns on random behavior. *n* is the size used as the upper bound for seeks. Block size can be followed by k or m to indicate kilobytes (*1024) or megabytes (*1024*1024).

count=*n* Copies only *n* input records.

`print=`*n* Modifies the printout at the end of the run. *n* is an integer between 0 and 5 as follows: 0 means no printout, 1 means latency style printout (useful for randoms), 2 means microsecond latency printout, 3 means Kbytes/second printout, 4 means Mbytes/second printout, and 5 means output suitable as `xgraph`(1). Default is a bandwidth style printout.

`label=`*string*
Prints out the *string* before the results. Useful if you are running multiple `lmdd`s in parallel. Use the label argument `if=`*arg* to sort the results.

WARNING: Because multiple `printfs` are sent to `stderr`, output may be mixed up.

`move=`*n* Moves *n* bytes of data. Useful when you want to cycle through block sizes, but always move approximately the same amount of data.

`bufs=`*n* Cycles through *n* different buffers.

`touch=`*n* Touches each buffer after the I/O.

`hash=`*n* Prints a hash mark for every block read and written (like FTP) if *n* is non zero.

EXAMPLES

The following example measures disk performance:

```
# lmdd of =XXX bs=4m move=100m
100.00 MB in 3.14 secs, 31.89 MB/sec
# lmdd if=XXX bs=4m move=100m
100.00 MB in 1.63 secs, 61.40 MB/sec
```

SEE ALSO

`bds`(8)

`xgraph`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`fsync`(2), `sync`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

NAME

lockd – Processes NFS file lock requests

SYNOPSIS

/etc/lockd [-t *timeout*] [-g *graceperiod*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The lockd daemon processes lock requests that are sent either locally by the kernel, or remotely by another lock daemon. lockd forwards lock requests for remote data to the server site lock daemon through Remote Procedure Call (RPC) and external Data Representation (XDR). lockd then requests the status monitor daemon, statd(8), for monitor service. The reply to the lock request is not sent to the kernel until the status daemon and the server site lock daemon have replied.

The lockd daemon accepts the following options:

%-o *openfile*	Allows you to increase the number of file descriptors you can open on behalf of users.
-t *timeout*	Uses *timeout* seconds as the interval, rather than the default value (15 seconds) to retransmit a lock request to the remote server.
-g *graceperiod*	Uses *graceperiod* seconds as the grace period duration, rather than the default value (45 seconds).

If either the status monitor or server site lock daemon is unavailable, the reply to a lock request for remote data is delayed until all daemons become available.

When a server recovers, it waits for a grace period for all client-site lock daemons to submit reclaim requests. Client-site lock daemons, on the other hand, are notified by the status monitor daemon of the server recovery and promptly resubmit previously granted lock requests.

NOTES

If your system is licensed for ONC+™, lockd will also register for version 4 of the nlockmgr protocol. This protocol is necessary for file locking on network file system (NFS) version 3 file systems.

SEE ALSO

statd(8)

fcntl(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

NAME

lpc – Controls the operation of the line printer

SYNOPSIS

/etc/lpc [*command* [*argument...*]]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The lpc command controls the operation of the line printer system. For each line printer configured in the /etc/printcap file, lpc performs the following tasks:

- Disables or enables a printer
- Disables or enables a printer's spooling queue
- Rearranges the order of jobs in a spooling queue
- Finds the status of printers and their associated spooling queues and printer daemons

Without any operands, lpc prompts for commands from standard input. If you specify arguments, lpc interprets the first operand as a command and the remaining arguments as operands to that command. You can redirect standard input, causing lpc to read commands from the file. The following list contains commands that lpc accepts (commands may be abbreviated):

?[*commands*]
help [*commands*] Prints a short description of each command specified in the argument list, or, if no arguments are given, a list of the recognized commands.

abort *printers*
abort all Terminates an active spooling daemon on the local host immediately and then disables printing for the specified *printers* (preventing new daemons from being started by lpr(1B)). If the specified printer is all, all printers are disabled.

clean *printers*
clean all Removes any temporary files, data files, and control files that cannot be printed (that is, does not form a complete printer job) from the specified *printer* queue(s) on the local machine. If the specified printer is all, all printer queues are affected.

disable *printers*
disable all Turns off the specified *printer* queues. This prevents lpr from entering new printer jobs into the queue. If the specified printer is all, all printer queues are turned off.

`down` *printers message*
`down all` *message*
: Turns off the specified *printer* queue, disables printing, and puts *message* in the printer status file. The message does not have to be quoted; the remaining arguments are treated like `echo`(1). This command allows you to take a printer down and lets others know why (`lpq` indicates that the printer is down and prints the status message). If the specified printer is `all`, all printer queues are turned off.

`enable` *printers*
`enable all`
: Enables spooling on the local queue for the listed printers. This allows `lpr` to put new jobs in the spool queue. If the specified printer is `all`, all printer queues have spooling enabled.

`exit`
`quit`
: Exits from `lpc`.

`restart` *printers*
`restart all`
: Attempts to start a new printer daemon. This action is useful when an abnormal condition causes the daemon to die, leaving jobs in the queue. `lpq` reports that no daemon is present when this condition occurs. If you are super user, try to abort the current daemon first (that is, kill and restart the daemon). If the specified printer daemon is `all`, all printer daemons are started.

`start` *printers*
`start all`
: Enables printing and starts a spooling daemon for the specified *printers*. If the specified printer is `all`, all printers have printing enabled and a spooling daemon started.

`status` *printers*
`status all`
: Displays the status of daemons and queues for the *printer* on the local machine. If the specified printer is `all`, the status of daemons and queues for all printers is displayed.

`stop` *printers*
`stop all`
: Stops a spooling daemon for the specified *printer* after the current job completes and disables printing. If the specified printer is `all`, spooling daemons for all printers are stopped.

`topq` *printer* [*jobnum...*] [*user...*]
: Places the jobs (specified by number) in the order listed at the top of the printer queue. If you specify *user*, the jobs that belong to that user or users are placed at the top of the queue.

`up` *printers*
`up all`
: Enables everything for the specified *printer* and starts a new printer daemon. If the specified printer is `all`, everything is enabled for all printers. This command reverses the effects of the `down` command.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

MESSAGES

`?Ambiguous command`	The command abbreviation matches more than one command.
`?Invalid command`	No such command exists.
`?Privileged command`	The specified command can be executed only by root.

FILES

`/etc/printcap`	Printer description file
`/usr/spool/*`	Spool directories
`/usr/spool/*/lock`	Lock file for queue control

SEE ALSO

`lpd`(8)

`lpq`(1B), `lpr`(1B), `lprm`(1B), `privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`printcap`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

lpd – Provides line printer daemon function

SYNOPSIS

/usr/lib/lpd [-l] [-S *tos*] [*portnumber*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The lpd program is the line printer daemon (spool area handler) and usually is invoked at boot time by using the netstart(8) script. It checks for the existing printers in the printcap(5) file and prints any files that remain after a crash. It then uses the listen(2) and accept(2) system calls to receive requests to print files in the queue, transfer files to the spooling area, display the queue, or remove jobs from the queue. In each case, it creates a child process (by using fork(2)) to handle the request so that the parent can continue to listen for more requests. The Internet port number used to rendezvous with other processes usually is obtained with getservbyname (see getserv(3C)), but you can change it with the *portnumber* argument.

The options are as follows:

-l Directs lpd to log valid requests that are received from the network, which can be used for debugging.

-S *tos* Directs lpd to set the IP Type-of-Service (TOS) option on its network connection to the value *tos*, which can be a numeric TOS value or a symbolic TOS name found in the /etc/iptos file.

portnumber Specifies port number used to rendezvous with other processes.

The lpd program controls access in the following two ways:

- All requests must come from one of the machines listed in the /etc/hosts.equiv or /etc/hosts.lpd file.
- If the rs capability is specified in the printcap entry for the printer being accessed, lpr(1B) requests are honored only for the users who have accounts on the machine by using the printer.

The minfree file in each spool directory contains the number of disk blocks that must be left free so that the line printer queue does not fill the disk. You can edit the minfree file with any text editor.

The UNICOS system takes several measures to enforce mandatory access controls. The lpr(1B) program and lpd daemon use multilevel spool directories to segregate spool files that have different security labels. The lpd daemon operates at the security label of the current printer spool directory so that mandatory access controls for those spool files are enforced. The lpd daemon may use the optional mi and ma printcap capabilities to limit access beyond that enforced by the mandatory access controls; the printcap capabilities do not override the mandatory access controls.

The lock file in each spool directory prevents multiple daemons from becoming active simultaneously, and it stores information about the daemon process for the lpr(1B), lpq(1B), and lprm(1B) commands. (On UNICOS systems using multilevel directories, the lock file is in the .mld directory.) After the daemon successfully set the lock, it scans the directory for files that have the suffix cf. Lines in each cf file specify files to be printed or nonprinting actions to be performed. Each line begins with one of the following key characters that specifies what must be done with the remainder of the line:

C Classification; string to be used for the classification line on the burst page.

c Cifplot file that contains data produced by cifplot.

d DVI file that contains TeX output (device independent (DVI) format from Stanford).

f Formatted file or name of a file to print that is already formatted.

g Graph file that contains data produced by plot.

H Host name or machine name at which lpr(1B) is invoked.

I Indent or the number of characters to indent the output (in ASCII).

J Job name or string to be used for the job name on the burst page.

L Literal or identification information from the password file that prints the banner page.

l Similar to f, but passes control characters and does not make page breaks.

M Mail or sends mail to the specified user when the current print job completes.

N File name of file being printed, or a blank for the standard input (when lpr(1B) is invoked in a pipeline).

n ditroff file that contains device-independent troff output.

P Person or login name of the person who invoked lpr(1B). lprm(1B) verifies ownership with this line.

p Print or name of a file to print by using lpr(1B) as a filter.

r File that contains text data with Fortran carriage control characters.

S Symbolic links.

T Title or string to be used as the title for lpr(1B).

t troff file that contains troff(1) output (cat phototypesetter commands).

U Unlink or name of file to remove after completing printing.

v File that contains a raster image.

W Width or changes in the page width (in characters) used by lpr(1B) and the text filters.

1 troff font R or name of the font file to use instead of the default.

2 troff font I or name of the font file to use instead of the default.

3 troff font B or name of the font file to use instead of the default.

4 `troff` font S or name of the font file to use instead of the default.

If a file cannot be opened, a message is logged with `syslog`(3C). `lpd` tries up to 20 times to reopen a file; after 20 attempts, it skips the file to be printed.

The `lpd` program uses the lock file to prevent multiple daemons from becoming active simultaneously. If the daemon is killed or dies unexpectedly, the lock file does not have to be removed. The lock file is kept in a readable ASCII form and contains the following items:

- The process ID of the daemon.
- The control file name of the current job being printed. This line is updated to reflect the current status of `lpd` for the programs `lpq`(1B) and `lprm`(1B).

NOTES

If this command is installed with a privilege assignment list (PAL), a user who is assigned the following privilege text upon execution of this command is allowed to perform the action shown:

Active Category	Action
`admin`	Allowed to start `lpd`.

If this command is installed with a PAL, a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`secadm`, `sysadm`	Allowed to start `lpd`.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to start `lpd`.

Cray Research systems do not support the `plot`, `cifplot`, `TeX`, and `troff` programs.

`lpd` will fail to print a file if the print device is not secure or the NAL entry for the remote host does not permit the transfer.

FILES

`/dev/lp*`	Line printer devices
`/dev/printer`	Socket for local requests
`/etc/hosts.equiv`	List of machine names that are allowed printer access
`/etc/hosts.lpd`	List of machine names that are allowed printer access, but not under same administrative control
`/etc/printcap`	Printer description file
`/usr/spool/*`	Spool directories
`/usr/spool/*/minfree`	Minimum free space to leave

SEE ALSO

`brc`(8), `lpc`(8)

`lpq`(1B), `lpr`(1B), `lprm`(1B), `pack`(1), `privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`accept`(2), `listen`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

`getserv`(3C), `syslog`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`printcap`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

`lptest` – Generates line printer ripple pattern

SYNOPSIS

`/usr/ucb/lptest` [*length* [*count*]]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `lptest` command writes the traditional "ripple test" pattern on standard output. This pattern prints all 95 printable ASCII characters in sequence. While originally created to test printers, it is useful for testing terminals, driving terminal ports for debugging purposes, or any other task for which a quick supply of random data is needed.

The `lptest` command accepts the following operands:

length Specifies the output line length. The default length is 79.

count Specifies the number of output lines to be generated. The default count is 200. If you specify *count*, you must also specify *length*.

NAME

lrmt – Copies data to rmt(8) command on network

SYNOPSIS

lrmt -n [*uid*@]*machine*:*device* -d *dest* [-p *pname*] [-c *capacity*] [-b *nbs*,*pbs*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The lrmt program is a local interface that handles the network communication side of /etc/dump and /etc/restore when these two programs are functioning as rdump and rrestore, respectively. For example, rdump is implemented as a shell script that pipes data from dump into lrmt and from there onto the network. The program is cataloged separately to permit easier testing and to permit other network data passing applications.

The lrmt program accepts the following parameters:

-n [*uid*]@*machine*:*device*
: Required parameter. *machine* is the network name at which rmt(8) will execute. *device* is the name of a file at *machine*. For example, to write a tape on node_xx, use -n node_xx:/dev/mt0. Though lrmt requires root privileges to use the rcmd() library routine, the optional *uid*@ field permits the remote /etc/rmt to run as the named user.

-d *dest*
: Required parameter. *dest* (destination) can be either -d p, which means that data leaving lrmt will be placed on the pipe, or -d n, which means that data leaving lrmt will go to the network socket.

-p *pname*
: A named pipe to be used for passing data on the local machine. If you do not specify this option, the defaults are stdin (if -d n) or stdout (if -d p).

-c *capacity*
: The capacity, in megabytes, of a remote output tape device. The default value is infinite. If you do not specify this option, the program will pause when each output volume is full. After an operator input signifying that the next output volume is mounted, the program continues.

-b *nbs*,*pbs*
: An optional parameter that specifies the buffer sizes to be used on the network or on the pipe. The default is -b 32768,32768.

SEE ALSO

dump(8), restore(8), rmt(8)

rcmd(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

NAME

makedbm – Creates a network information service (NIS) dbm file

SYNOPSIS

```
/etc/yp/makedbm [-i yp_input_file] [-o yp_output_name] [-d yp_domain_name]
[-m yp_master_name] infile outfile
/etc/yp/makedbm [-u dbmfile]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The makedbm command converts *infile* into a pair of files, .pag and .dir, in dbm(3C) format. Each line of the input file is converted to one dbm record. All characters up to the first tab or space form the key, and the rest of the line is the data. If a line ends with \, the data for that record is continued on to the next line. It is left for the clients of the network information service (NIS) (formerly called yellow pages) to interpret #; makedbm does not treat it as a comment character. *infile* can be -; in which case, standard input is read.

Use makedbm to generate dbm files for NIS, and it generates a special entry with the key *yp_last_modified*, which is the date of *infile* (or the current time, if *infile* is -).

The makedbm command accepts the following options:

-i *yp_input_file*
: Creates a special entry by using the key *yp_input_file*.

-o *yp_output_name*
: Creates a special entry by using the key *yp_output_name*.

-d *yp_domain_name*
: Creates a special entry by using the key *yp_domain_name*.

-m *yp_master_name*
: Creates a special entry by using the key *yp_master_name*. If no master host name is specified, *yp_master_name* is set to the local host name.

-u *dbmfile*
: Undoes the specified dbm file; that is, makedbm prints out a dbm file, one entry per line, with a single space separating keys from values.

infile Specifies files to be converted.

outfile Specifies name of output dbm file.

EXAMPLES

It is easy to write shell scripts to convert standard files in passwd(5) format to the key value form used by makedbm. The following example takes a file in passwd(5) format and converts it to a form that can be read by makedbm to make the NIS file passwd.byname:

```
#!/bin/awk -f
BEGIN { FS = ":"; OFS = "\t"; }
{ print $1, $0 }
```

That is, the key is a user name, and the value is the remaining line in the password file.

SEE ALSO

yppasswd(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

dbm(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

NAME

map_mbone – Obtains routing information from multicast routers and builds a topological map from the information

SYNOPSIS

map_mbone [-d [*debuglevel*]] [-f] [-g] [-n] [-r *retries*] [-t *timeout*] [*router*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The map_mbone command obtains multicast information from the specified router (which should be a host name or a host address), and attempts to build a topological map from the information. If no router is specified, map_mbone floods the routing graph with queries to build the map.

The map_mbone command accepts the following options and arguments:

-d [*debuglevel*] If the -d option is specified with no argument, the debug level defaults to 2. Debug levels have the following effects:

- Level 1 Prints all logged messages to stderr.
- Level 2 Prints all level 1 messages plus notifications of significant events to stderr.
- Level 3 Prints all level 2 messages plus notifications of all packet arrivals and departures to stderr.

-f Floods the routing graph with queries. This is true by default unless the router is specified.

-g Generates output in GraphEd format.

-n Represses look-up of domain name server (DNS) names for routers.

-r *retries* Sets the number of retries when requesting information from a router. Default is 3 retries.

-t *timeout* Sets the time out in seconds when waiting for a response from a router. Default is 4 seconds.

router Specifies the router using either a host name or a host address.

SEE ALSO

mrouted(8), mrinfo(8)

NAME

mddconf – Displays or changes a mirrored disk configuration

SYNOPSIS

mddconf [-w on|off] [[*sign*] *permissions*] [[*name1*] [*name2*] ...]

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

CRAY J90 series

CRAY EL series

DESCRIPTION

The mddconf program changes or reports the current configuration of the components of a mirrored special device (/dev/mdd/name). More specifically, you can use this command to disable a single component that will be removed for repairs, to tune the entire mirror for faster read operations, or to select the component to be used to initialize the mirror.

The mddconf program accepts the following option and operands:

-w on|off
: Enables or disables the write-behind option of write I/O.

  on: Write-behind enabled

  off: Write-behind disabled

[*sign*] *permissions*
: Specifies a new configuration for the mirror. The configuration is installed to the kernel-resident mirror table. This option is location-dependent; it must be placed where the getopt scan ends. It is distinguished from the file names by first character, *sign*.

  The *sign* argument may be plus (+) or minus (-). A + specifies that the bits identified in the configuration are to be ORed into from the current specification. A - specifies that the bits identified in the configuration are to be removed from the current specification.

  The *permissions* argument may be in the alphabetic form rwx (or any combination of r, w, and x) or in the numeric form *nnn*.

  The alphabetic specification of the configuration applies mirror-wide; thus, the r character means that all components of the mirror are enabled for reading, the w character that all components are enabled for writing.

  A numeric specification of the configuration permits the configuration to be specified on a component-by-component basis. The bottom 3 bits belong to the first component device of the mirror, the next 3 bits to the second, and so on. An octal digit (octit) containing a 4 bit enables the device for reading, a 2 bit for writing, and a 1 bit marks the device as active.

[*name1*] [*name2*] ...

These arguments specify the names of one or more block special inodes with major type `dev_mdd`. If you do not specify a name, the program processes all names in the `/dev/mdd` directory. To permit the parameter parse to distinguish the optional first name from the optional configuration specification, the first character of the name must be a slash (`/`) or a period (`.`).

EXAMPLES

Example 1: To report the configuration of all devices in `/dev/mdd`, enter the following:

```
mddconf
```

Example 2: To set a mirrored device ready for member reconstruction, enter the following:

```
mddconf  037       /dev/mdd/device
```

This enables both members as write enabled and read from only member 0. The mode could be set 073 depending on which member is the read member. See `mddcp`(8) command for information on data reconstruct.

Example 3: To set all mirrored devices to an initializing configuration (that is, all mirrors are active only in the first component), enter the following:

```
mddconf  07         # specified in octal
mddconf   7         # specified in decimal
mddconf  0x7        # specified in hexadecimal
```

Example 4: To set a mirror with no write-behind, enter the following:

```
mddconf  -w off       /dev/mdd/device
```

SEE ALSO

`mdd_pre`(8)

NAME

mddcp – Copies leg of a mirrored disk in IOS model E systems

SYNOPSIS

mddcp [*device*]

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

CRAY J90 series

CRAY EL series

DESCRIPTION

The mddcp command is the mirrored disk copy command for IOS-E systems. The command copies the first read-enabled leg of a mirrored disk device to all-write enabled legs of the mirrored group. It synchronizes all members of a mirrored group. Mirror members can become unsynchronized during system crashes or when write errors occur on a disk. The mddconf command sets the read and write modes for individual members of a mirrored group. The device is a /dev/mdd device.

SEE ALSO

mddconf(8)

NAME

mdd_pre, mdd_post – Prepares a mirrored file system for fsck processing, tunes a mirrored file system after fsck processing

SYNOPSIS

```
mdd_pre [-d] [/dev/dsk/name1 ... ]
mdd_post [-d] [/dev/dsk/name1 ... ]
```

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

CRAY J90 series

CRAY EL series

DESCRIPTION

These programs assist in verifying the state of a file system that contains mirrored devices.

The mdd_pre program prepares the file system for processing by fsck(8). The essential step is to configure all mirrored devices to a nonmirrored state. Thus, only one component device of each mirror is enabled for reading; the same component is also the only one enabled for writing. Next, mdd_pre examines the actual physical devices that are components of the mirror. If every dynamic block that can be found in the physical devices indicates that the file system was dismounted cleanly, the mdd_pre program assumes that the fsck(8) step will be skipped and reconfigures all mirrored devices to the tuned configuration.

The mdd_post program performs the steps that are required after fsck(8) runs. First, the current configuration of each mirror is examined. If the mirror is configured for writes to multiple devices, it is assumed that the mirror is already tuned. If not, the mirror is set to read from a single device (one component should already be read enabled) and to write to all devices that are components of the mirror. The program then forks a daemon fragment of itself to complete the last two steps: a copy step to put identical information in all components, and a configure step to set the mirror to the tuned configuration.

All components of mirrored devices that are lacking a 1 bit, or ‘x’ bit, in the mirror configuration are disabled.

The "tuned configuration" for a mirror is determined from the mirror=(*name1:nnn1;name2:nnn2;...*) specification in /etc/fstab, or from spreading all I/O activity equally throughout the mirror: writes to all devices, reads from any device.

The mdd_pre and mdd_post programs support the following option:

-d Enables debug output.

If no /dev/dsk files are specified on the command line, mdd_pre examines the /dev/mdd directory to disable mirrors, and the /etc/fstab file to search for dynamic blocks in file systems.

If no /dev/dsk files are specified on the command line, mdd_post examines the file systems specified in /etc/fstab.

NOTES

These programs exist because of the logical chaos that would result from declaring nonidentical devices to be a mirror. Therefore, a pause at the beginning of the mdd_post program permits the operator, if more fsck(8) processing is required, to abort mdd_post before the mirrored configurations are changed.

Experience in the field suggests that mdd_pre fits in /etc/inittab and mdd_post fits in rc.pst.

EXAMPLES

Suppose that /dev/dsk/fs is a logical device made up of a single three-way mirrored device called /dev/mdd/m. The following sequence shows how mdd_pre and mdd_post can be used to prepare /dev/dsk/fs for mounting:

```
/etc/mdd_pre    /dev/dsk/fs
/etc/fsck       /dev/dsk/fs
/etc/mdd_post   /dev/dsk/fs
/etc/mount      /root/mntpnt    /dev/dsk/fs
```

The next sequence shows the same processing done manually:

```
/etc/mddconf -p 0117 /dev/mdd/m
/etc/fsck /dev/dsk/fs
/etc/mddconf -p 0337 /dev/mdd/m
/etc/mount /root/mntpnt /dev/dsk/fs
/etc/mddcp /dev/mdd/m
/etc/mddconf 0777 /dev/mdd/m
```

NAME

mfsck – Runs file system checks in parallel

SYNOPSIS

/etc/mfsck [-c] [-d *depth*] [-f *fstab-file*] [-n] [-q|-u] [-r] [-v] [*progname*]

IMPLEMENTATION

Cray PVP systems

STANDARDS

X/Open, XPG4

DESCRIPTION

This command was formerly called gencat. It has been renamed mfsck for UNICOS 8.3, UNICOS 9.0, and subsequent releases to conform with the X/Open XPG4 specification. The name gencat is now used by a UNICOS message system command. The functionality of mfsck has not changed, only its name.

The mfsck command runs several copies of fsck(8) in parallel, which can speed up system startup. Typically, there are two or more passes; only the root file system is checked in pass one, and then groups of file systems are checked in parallel in pass 2, pass 3, and so on.

Device conflict checking is done within passes to ensure that no more than one fsck is active at a time on an individual device. With the advent of device conflict checking, the importance of dividing file systems carefully into multiple passes beginning with pass 2 has diminished, because mfsck will determine the maximum parallelism possible.

The /etc/fstab file (see fstab(5)) determines when each file system is checked. This file has a one-line entry for each file system in the following format:

filesystem directory type options frequency passnumber

For example, a line in the /etc/fstab might contain the following:

```
/dev/dsk/bench   /bench   NC1FS  rw  1  2
```

If mfsck detects any problems with one or more file systems, it notes the fact and proceeds to the next pass. After all passes are complete, mfsck reruns fsck(8) for each file system that produced errors on the first run. The fsck(8) command queries the operator as it tries to fix up the damaged file system.

The mfsck command passes the -u option to fsck(8) or *progname* unless the /etc/fstab option fsckopt=q is present for the particular file system.

The `mfsck` command accepts the following options:

`-c` Causes `mfsck` to consolidate all passes except pass 1 into a large pass 2. Disables passes greater than pass 2 in `/etc/fstab`. This effectively causes `mfsck` to consolidate all passes except pass 1 into a large pass 2, but device conflict checking is still performed.

`-d` *depth* Changes the *depth* of device conflict checking. The default is 1; that is, only the first device on which a file system resides is checked for activity conflict when attempting to start another `fsck`(8).

`-f` *fstab-file* Causes `mfsck` to use the file *fstab-file* instead of `/etc/fstab`.

`-n` Disables device conflict checking.

`-q` Causes `mfsck` to omit passing the `-u` option to `fsck`(8) or *progname*, regardless of the `/etc/fstab` options.

`-u` Causes `mfsck` to pass the `-u` option to `fsck`(8) or *progname*. In the case of `fsck`(8), this causes unconditional file system checking.

`-r` Performs read-only file system checks; used primarily for debugging.

`-v` Provides verbose output; used primarily for debugging.

progname Specifies the path name of the command that checks the file system. The default is `/etc/fsck`. This command must support the `-p` (preen) option, the `-r` (readonly) option, and the `-u` (unconditional) option.

The interrupt signal (SIGINT; usually <`CONTROL-c`> causes the current pass to terminate. The `mfsck` command then goes on to the next pass. Interrupt `mfsck` (and the file system check procedure) with caution.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	**Action**
`system`, `secadm`, `sysadm`, `sysops`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

FILES

`/etc/fstab`

SEE ALSO

`fsck`(8)

`fstab`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

mkbinhost – Creates a binary network host file

SYNOPSIS

/etc/mkbinhost [*ascii_host_file*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The mkbinhost command creates a binary file that contains network hosts from an ASCII network host file. By default, mkbinhost creates the binary network host file /etc/hosts.bin from the network hosts listed in the ASCII /etc/hosts file. If you specify the optional *ascii_host_file* argument, mkbinhost creates the binary network host file *ascii_host_file*.bin from the ASCII network host information in *ascii_host_file*.

When looking up network host and address information from the local hosts table, all library routines in gethost(3C) use /etc/hosts.bin if it exists. If it does not exist, the library routines get information from the /etc/hosts file.

Whenever you modify /etc/hosts, you must run mkbinhost to update /etc/hosts.bin. No automatic mechanism is available for detecting an out-of-date hosts.bin file.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

FILES

/etc/hosts	ASCII version of the network host file
/etc/hosts.bin	Binary version of the network host file

SEE ALSO

`named`(8)

`privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`gethost`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

mkdmp – Initializes raw dump device header

SYNOPSIS

For IOS model E:

mkdmp [-b] *special*

For GigaRing based systems:

mkdmp [-f] *special*

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E (IOS-E)

GigaRing based Cray PVP systems

CRAY J90 series

CRAY EL series

DESCRIPTION

The mkdmp utility performs different actions depending on which system it is run: an IOS-E system or a GigaRing based system.

The mkdmp utility accepts the following arguments:

-b Reads a list of bad blocks from standard input.

-f (Force) Overwrites the dump device header to contain MF_INITIAL in the first word of the device. This will also update the other fields of the DmpDevHdr structure contained in the device header.

special Specifies the path name for the dump device. For GigaRing based systems, most path names for block-special devices are in the format /dev/dsk/*name*.

IOS-E systems

The mkdmp utility initializes the dump slice for a disk on an IOS-E system. The mkdmp utility writes a dump header ID and the length of the slice to the first sector of the dump slice. The -b option lets the user map the bad or good blocks of the slice. mkdmp reads from standard input and maps the blocks by starting logical block and by length of segment.

GigaRing based systems

The mkdmp utility queries the first 32 words (256-bytes) of the device and looks for a valid dump device header. The structure for the dump device header is defined in the /usr/include/sys/dump.h file as follows:

```
struct DmpDevHdr  {
    char    FileID[8];      /* Dump device id (ASCII identifier) */
    char    Date[8];        /* Date dump device was initialized  */
    char    Time[8];        /* Time dump device was initialized  */
    uint_64 Timezone;       /* Seconds from GMT                  */
    char    Tzname[2][8];   /* Timezone character names          */
    uint_64 Daylight;       /* Daylight savings time flag        */
    uint_64 BlkSz;          /* Block size of dump device         */
    uint_64 NumBlks;        /* Blocks in dump device             */
    uint_64 DmpFileHdr;     /* Byte address of dump file header  */
    uint_64 Reserved[22];   /* Not used                          */
}
```

The `FileID` (8-byte) word has the current status for the device. The possible values that this field may contain are as follows:

`MF_INITL` Indicates that the device has been initialized and that there are no dumps stored in it.

`MFSYSDMP` Indicates that a dump is currently stored in the device. This is the case when you use third-party I/O to store the boot PE dump to a disk device.

`MFCOPIED` Indicates that the dump has been copied from the dump device into some file system. This is the case when you use third-party I/O and the boot PE dump has been copied from the disk device into a given file system.

If the `FileID` field does not contain any of the preceding values, `mkdmp` assumes that the device has not been initialized and it will write `MF_INITIAL` to the first word of the dump device header. The dump device header will also be initialized if the size of the dump device is different from that given in the `NumBlks` word.

NOTES

(GigaRing based systems) The common GigaRing dump routines assume that the raw dump device being used contains a disk controller that automatically handles things like bad-block mapping and so forth (for example, SCSI).

(IOS-E systems and GigaRing based systems) If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`	Allowed to specify any file.
`sysadm`	Allowed to specify any file, subject to security label restrictions on the file's path. Shell redirected I/O is subject to security label restrictions.

If the PRIV_SU configuration option is enabled, the super user is allowed to specify any file.

EXAMPLES

(IOS-E systems) The following example reads the bad blocks generated from the bb(8) command and maps them into the dump slice header. It writes the dump ID, dump slice, and block map to the disk.

```
bb /dev/dsk/dump | mkdmp -b /dev/dsk/dump
```

SEE ALSO

bb(8)

mfboot(7), mfsysdmp(7) (available only online)

NAME

mkfm – Make fmsg nodes

SYNOPSIS

/etc/mkfm *node_id* [*node_id1 node_id2 node_id3* ...]

IMPLEMENTATION

CRAY T90 systems with GigaRing-based I/O

CRAY J90 systems with GigaRing-based I/O

DESCRIPTION

Given a list of node identifiers, the mkfm command creates the the character special nodes in the /dev/fmsg directory. For information on the files in this directory, see fmsg(4). These nodes provide a general purpose interface from Unicos to the GigaRing message complex and are used by commands such as fping(8) and mmr(8).

node_id GigaRing node identifier. GigaRing node identifiers are generally specified as octal integers comprised of a ring and a node component, as follows:

0*rrrnn* where:

rrr = GigaRing ring number

nn = GigaRing node number

EXAMPLES

The following command makes /dev/fmsg character special nodes for ring 1 node 1, ring 1 node 2, and ring 4, node 3:

```
mkfm 0101 0102 0403
```

The /dev/fmsg directory will be created if it does not already exist, and three character special files will be created: /dev/fmsg/e0101, /dev/fmsg/e0102, and /dev/fmsg/e0403. If the character special files already exist, they will be removed and new ones made.

FILES

/dev/fmsg/*

SEE ALSO

fping(8), mmr(8)

fmsg(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

mkfs – Constructs a file system

SYNOPSIS

/etc/mkfs [-a *strategy*] [-A *nblocks*] [-B *nbytes*] [-C *compart*] [-d] [-F] [-i *inode_factor*] [-L *minslvl*] [-n *nblocks*] [-q] [-Q] [-U *maxslvl*] [-I *num_inodes*] [-b *flaw_list*] [-P *primary_aau*] [-S *secondary_aau*] [-p *primary_parts*] [-s *arbiter:semaphore_count*] [-M] [-m] [-z] *device*

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The mkfs command constructs a file system (NC1FS, or UNICOS shared file system, SFS) by writing on the special file, *device*, specified on the command line. It builds the file system with a boot block, a super block, a root inode, and a bit map of free blocks (a system table that contains 1 bit for each block). By default, mkfs performs a surface analysis and zeros the disk data blocks before initialization. You can disable the surface analysis with the -Q option. You can disable the surface analysis and data block zeroing with the -q option.

mkfs assigns a minimum and maximum security label range to the file system. The default security label range is [0:0, 0:0]. You can use the -L, -U, and -C options to request a nonzero security label range. If a nonzero security label range is specified, the file system is marked as secure (using a magic number) in the file system superblock.

The mkfs command determines if *device* is a UNICOS logical (block special) device, and if so, segments the file system into partitions. Partitions are made by combining blocks from the logical device until blocks from a new physical device have been encountered. The merging of blocks into partitions can by controlled by the -m or -M options.

The mkfs command uses partitions as file system entities. A partition belongs to either the primary area, or the secondary area. By default, the maximum number of primary partitions in a file system is 4, with all remaining partitions defined as secondary partitions. If you define more than 4 primary partitions, the maximum number of primary partitions kept current (updated by dynamic blocks to file system bitmaps) is 4. Primary area partitions may have super blocks, inode regions, pipe data, directory data, indirect blocks, and other non-user file information. Secondary area partitions only contain user data blocks. Allocation units (*aau* * 4096 bytes) can be different for primary and secondary areas; however, all partitions of either area have the same allocation unit.

The mkfs command accepts the following options and operand:

-a *strategy* Specifies an allocation strategy. *strategy* can take one of the following values:

rrf Round-robin all files (default). Directories and inodes go into the first partition of a multi-partition file system whenever possible.

`rrd1` Round-robin first level directories

`rrda` Round-robin all directories

`-A` *nblocks* Specifies big file allocation unit. The minimum number of 4096-byte blocks allocated for a file whose size is greater than or equal to the number of bytes specified for big files. Default is `BIGUNIT` in the `sys/param.h` include file. The value of *nblocks* should be a multiple of *secondary_aau*, if secondary partitions exist; otherwise, it should be a multiple of *primary_aau*. This allocation unit is used for any I/O request to a big file, unless the process makes a request larger than *nblocks*. If the process requests an allocation larger than *nblocks*, then the allocation is equal to the request size.

Any file system allocation equal to or larger than the `BIGUNIT` size specified with this option goes to a secondary partition.

`-B` *nbytes* Specifies the big file threshold. The number of bytes after which a file in this file system is considered big. Default is `BIGFILE` in the `sys/param.h` include file. When using primary and secondary partitions, `BIGFILE` is the threshold at which a file is allocated on the secondary partition. If this value is larger than the initial user allocation request, then initial parts of files will reside on the primary partition with subsequent allocations on the secondary partition, depending on the `BIGFILE` threshold.

`-C` *compart* Specifies the valid security compartment set as an octal mask or a comma-separated list. When the `-C` option is specified on a UNICOS system with device labeling enforcement enabled, the device special file specified by *device* is given the requested compartments as a valid compartment mask. The default compartment mask value is 0. Only appropriately authorized users can set device labels.

`-d` Does not create a `/lost+found` directory.

`-F` Force `mkfs` to bypass the `ismounted`() test. Occasionally, when remaking a file system that was once shared and controlled by an SFS Arbiter which is no longer active, `mkfs` will refuse to remake the file system, as the `ismounted`() subroutine could not determine if the shared file system was in use by some other system without being able to access the SFS Arbiter. The `-F` option allows the administrator to override the `ismounted`() test. If the shared file system is actually in use by another system, using the `-F` option can cause serious trouble for the other system.

`-i` *inode_factor* Specifies the inode factor as a ratio of blocks to inodes. Default is 4 (which is 25%).

`-L` *minslvl* Specifies the minimum security level of the file system (*minslvl*), specified by either an integer or level name. Default is 0. When the `-L` option is specified on a UNICOS system with device labeling enforcement enabled, the device special file specified by *device* is given the requested level as a minimum security level. Only appropriately authorized users can set device labels.

`-n` *nblocks* Specifies *nblocks* as the decimal number of 4096-byte blocks in the file system. If `-n` is not specified, the default number is the number of blocks on the special file.

`-q` Specifies quick mode. Bypass surface check.

`-Q` Specifies quick mode with zeroing. Bypass surface check but zero data blocks.

`-U` *maxslvl* Specifies the maximum security level of the file system (*maxslvl*), specified by either an integer or level name. Default is 0. When the `-U` option is specified on a UNICOS system with device labeling enforcement enabled, the device special file specified by *device* is given the requested level as a maximum security level. Only appropriately authorized users can set device labels.

`-I` *num_inodes* Specifies the number of inodes desired on the file system. `mkfs` will create *num_inodes* inodes, adjusted by inode allocation rules and limits. If `-I` is specified, the `-i` option is ignored.

`-b` *flaw_list* Reads a list of decimal tuples (pairs of numbers) from the file *flaw_list* that specifies the starting block number and the number of blocks in the bad sections of the file system. If the name of the file is –, `mkfs` will read the standard input.

The `mkfs` command avoids using the specified areas, when possible: If information for which the location is critical (such as the super block) falls within the specified area, that area of the disk is used; if the specified area is not a critical area, the area specified is reserved when `mkfs` is executed and is not allocated as new file space. Reserving bad blocks with the `-b` option avoids using the bad block areas on disk that are remapped to spare cylinders. Using the spare cylinders may case an I/O performance degradation when reading or writing the bad block area; the tradeoff can be disk fragmentation caused by bad block avoidance, if there are a large number of flaws on a particular disk.

`-P` *primary_aau* Specifies the area allocation unit for all primary allocation partitions. The area allocation unit is the minimum allocatable unit for these partitions. The *primary_aau* is specified as an integer number of 4096-byte blocks. The value must be a multiple of the physical sector size of the disks composing the file system. The default value for the primary allocation unit is the greater of 4096 bytes, or the physical sector size of the device(s) making up the file system.

The default allocation unit is 1 for disk drives in which the sector size is 4 Kbytes, 4 for disk drives in which the sector size is 16 Kbytes, and 16 for disk drives in which the 64 Kbytes. See the `diskspec`(7) and `hdd`(4) man pages for information on sector sizes for specific disk drives.

The `-P` option can be set to any value as long as it is a multiple of the hardware sector size and the size of each primary partition is a multiple of this value.

`-S` *secondary_aau* Specifies the area allocation unit for all secondary allocation partitions. The area allocation unit is the minimum allocatable unit for these partitions. The *secondary_aau* is specified as an integer number of 4096-byte blocks. The value must be a multiple of the physical sector size of the disks composing the file system. The default value for the secondary allocation unit is the greater of 4096 bytes, or the physical sector size of the device(s) making up the file system.

The default allocation unit is 1 for disk drives in which the sector size is 4 Kbytes, 4 for disk drives in which the sector size is 16 Kbytes, and 16 for disk drives in which the 64 Kbytes. See the `diskspec`(7) and `hdd`(4) man pages for information on sector sizes for specific disk drives.

The `-S` option can be set to any value as long as it is a multiple of the hardware sector size and the size of each secondary partition is a multiple of this value.

`-p` *primary_parts* Specifies the number of partitions which make up the primary allocation area. The specified number of partitions is taken in order from the beginning of the logical device. All partitions in the primary allocation area will use *primary_aau*, if specified. All file system metadata (inodes, directories, block reservation bit maps, and so on) reside in the primary allocation area. The maximum number of primary partitions kept current is 4. Primary partition data is never placed on the secondary partition unless the primary partition becomes full. If the primary partition data space is full, then no further file extensions can be made and no new directories can be created. If the primary partition inode regions are also full, then no further file allocation can be made.

`-s` *arbiter:semaphore_count*
Specifies the file system to be an SFS-type (Shared File System) file system.

The argument to the `-s` option provides the *arbiter* name or number, and the number of semaphores from that arbiter to be assigned to this file system at mount time.

The *arbiter* name or number must match one of the valid configuration entries in the `/etc/config/sfs` configuration file.

`-M` Specifies that each slice of a file system be treated as a partition. The `-M` option is the default on Cray PVP systems with an IOS model E.

`-m` Specifies that the logical device structures are ignored, and the file system has one partition.

`-z` Disables the file system panic flag. This flag, which is enabled by default, causes the kernel to panic when it encounters file system errors. File system errors encountered when the panic flag is disabled are logged to `/dev/fslog` and handled by the `fslogd`(8) daemon.

device Specifies the path name of the block special file to be written.

WARNINGS

Primary allocation units cannot exceed the maximum buffer cache size allowed (nblkmax). nblkmax is calculated as a percentage of buffer cache: nblkmax = nbuf/v.v_blkfctr.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to make any file system and initialize all security attributes.

If the PRIV_SU configuration option is enabled, the super user is allowed to make any file system and initialize all security attributes.

SEE ALSO

fsck(8), mfsck(8)

fs(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

mknod – Builds a directory entry and inode for a special file

SYNOPSIS

```
/etc/mknod name b major minor [device parameters]
/etc/mknod name c major minor [device parameters]
/etc/mknod name p
/etc/mknod name B slice1 [slice2 ...]
/etc/mknod name L member0 [member1 member2 ...]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The mknod command makes a directory entry and corresponding inode for a special file. mknod accepts the following arguments:

name Specifies the *name* of the entry.

b Creates block special files (disks and tape).

B Specifies dynamic disk configuration.

c Creates character special files (other devices).

L Creates a logical disk descriptor file.

p Creates first in, first out (FIFO) special files (named pipes).

major Specifies major device; may be octal, decimal, or symbolic. See the Device Names subsection for a listing of the device names that can indicate major numbers.

minor Specifies minor device (such as unit, drive, or line number); can be either octal or decimal.

device parameters
Specifies up to eight device-specific numbers (specified in decimal or octal), providing the operating system with more information about the device's configuration. The parameter field may also be the path name to a configuration file.

slice Specifies the physical device, start, and length as follows:

device-name,*begin-cyl*,*length*

partition Specifies the full path name of a disk block special file.

The assignment of major device numbers is specific to each system. They are found in the system source file devsw.c.

NOTES

When you use the mknod(8) command to create block or character special files that require a path name, the device node path name cannot exceed 23 characters.

Generally, the `mknod` command requires an appropriately authorized user. The `p`, `B`, `C`, and `L` options, however, can be run by any user.

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	**Action**
`system`, `secadm`	Allowed to create any file.
`sysadm`	Allowed to create any file, subject to security label restrictions on the file's path. Shell redirected I/O is subject to security label restrictions.

If the `PRIV_SU` configuration option is enabled, the super user or a user with the `mknod` permbit is allowed to use this command. A user with the `mknod` permbit is subject to discretionary and mandatory access control restrictions of the file's path, and shell redirected I/O is subject to security label restrictions.

Device Names

The following tables summarize the device name that you use as the major number for each logical device and the special files associated with that device.

You can use the following device names with all Cray PVP systems:

Major number	Device description	Special files
`dev_bmx`	Block-multiplexer tape driver	`/dev/`*tape*|*cart*...
`dev_bmxd`	Block-multiplexer tape daemon driver	`/dev/bmxdem`
`dev_clst`	CPU cluster device driver	`/dev/cpucluster`
`dev_cpu`	Real-time CPU driver	`/dev/cpu/[0-63]`
`dev_err`	Device error logging	`/dev/err`
`dev_hpm`	Hardware performance monitor	`/dev/hpm`, `/dev/hpm_mul`, `/dev/hpm_all`
`dev_hppi`	`hippi` driver	`/dev/hippi`
`dev_ifei`	OSI FEI and DSI driver	`/dev/ifei`...
`dev_log`	System log file	`/dev/klog`
`dev_mig`	Archive daemon	`/dev/dmd`
`dev_mm`	User and kernel null memory driver	`/dev/null`
`dev_proc`	`/proc` file system	`/proc`
`dev_ptc`	Pseudo-TTY master driver	`/dev/pty/`...
`dev_pts`	Pseudo-TTY slave driver	`/dev/ttyp`...
`dev_secd`	Memory error reporting driver	`/dev/secded`
`dev_slg`	Security log driver	`/dev/slog`

Major number	Device description	Special files
dev_sy	Generic TTY driver	/dev/tty
dev_unet	UltraNet driver	/dev/ultra

The following table summarizes the device names that you can use with the I/O subsystem model E.

Major number	Device description	Special files
dev_ios	IOS model E control	/dev/ios
dev_ip3g	IOS model E ipi3 global ioctl driver	/dev/ipi3
dev_ipi3	IOS model E ipi3 device packet driver	/dev/ipi3/...
dev_ldd	IOS model E logical disk driver	/dev/dsk
dev_mdd	IOS model E mirror disk driver	/dev/mdd
dev_np	Network packet driver for low-speed interfaces	/dev/comm/npctl*
dev_pdd	IOS model E physical disk driver	/dev/pdd, /dev/ift, /dev/spare
dev_rdd	IOS model E RAM disk driver	/dev/pdd
dev_sdd	IOS model E stripe disk driver	/dev/sdd
dev_smp	Semaphore device driver	/dev/smp
dev_ssdd	IOS model E SSD driver	/dev/pdd
dev_zp	IOS model E synchronous OWS TTY driver	/dev/tty0[0-4]

EXAMPLES

To configure a three-drive DD-49 file system named tmp using dynamic disk configuration, enter the following:

```
/etc/mknod /dev/dsk/tmp_cluster B 49-A1-32,1,408 49-A1-33,1,408 49-A1-34,1,408
/etc/mknod /dev/dsk/tmp b 5 40 /dev/dsk/tmp_cluster
```

For more information on reconfigurable disks, see *General UNICOS System Administration*, Cray Research publication SG–2301.

On Cray PVP systems with an I/O subsystem model E, to configure a three-drive DD-60 file system named tmp using dynamic disk configuration, you can use the format for physical device nodes found in pdd(4):

mknod *name type major minor dtype iopath start length flags altpath unit*

Enter the following, which sets up sets up a logical descriptor file with three slices:

```
/etc/mknod /dev/ldd/tmp L /dev/pdd/tmp0 /dev/pdd/tmp1 /dev/pdd/tmp2
```

The following command creates a block special file with minor number 22. (Find empty minor slots with `ls -l /dev/dsk/* | sort +4 | more key-in`.)

```
/etc/mknod /dev/dsk/tmp b 34 22 0 0 /dev/ldd/tmp
```

The following commands set up physical devices on minor numbers 20, 21, and 22 on DD-60 disk drives that are attached to IOC 0, IOP 1, channel 32, units 0 through 2.

```
/etc/mknod /dev/pdd/tmp0 c 32 20 10 0132 0 119692 0 0 0
/etc/mknod /dev/pdd/tmp1 c 32 21 10 0132 0 119692 0 0 1
/etc/mknod /dev/pdd/tmp2 c 32 22 10 0132 0 119692 0 0 2
```

See pdd(4) and ldd(4) for information about disk and other device types, and formats of device dependent information.

SEE ALSO

ddstat(8)

dsk(4), hdd(4), ldd(4), mdd(4), pdd(4), sdd(4), ssdd(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

mkspice – Makes inodes for the ce, ift, and spare sector slices

SYNOPSIS

mkspice [-t *dtype*] [-i] *iopath0*[,*altpath0*][.*unit*] [*iopath1*[,*altpath1*][.*unit*] *iopath2*[,*altpath2*][.*unit*] ...]
mkspice -t *dtype* -l *length* [*name*]

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

DESCRIPTION

The mkspice command makes physical disk device inodes describing the spare sector map, factory flaw map, and customer engineering slices for the physical disk devices named for their I/O paths and alternate I/O paths (*iopath* and *altpath*). (See pdd(4) for information on the I/O path and alternate I/O path.)

In its second form, the mkspice command makes a single node representing the SSD or RAM device in /dev/ddd. The created node is used by the installation and configuration menu system.

The mkspice command makes the following directories (if they do not already exist):

/dev/ift	Factory flaw slices
/dev/spare	Spare sector slices
/dev/ce	Customer engineer slices
/dev/ddd	Diagnostic slices

The mkspice command then makes one character special device inode in each directory corresponding to and named for each physical *iopath* specified.

The mkspice command accepts the following options and operands:

-t *dtype* Specifies the device type (for example, dd49). See pdd(4) for a list of supported device types. mkspice must be run once for each different device type. The default device type is currently dd49.

If *dytype* specifies an SSD or RAM device, the -l option is needed to specify the size.

-l *length* The length of the SSD or RAM device in blocks.

-i Creates the /etc/aft directory if it does not exist and initializes the aft file(s) as follows:

```
ift /dev/ift/iopath > /etc/aft/iopath
```

The spare map is then initialized:

```
cat /etc/aft/iopath | spmap -w /dev/spare/iopath
```

name When used for an SSD or RAM device, the node name is optional. The default is SSD or RAM, respectively.

EXAMPLES

Example 1: The following command generates spare sector, ift, and ce slices for a DD-49 physical disk.

```
mkspice -t dd49 0130 0134
```

This command generates the following files:

```
/dev/spare/0130
/dev/spare/0134
/dev/ift/0130
/dev/ift/0134
/dev/ce/0130
/dev/ce/0134
/dev/ddd/0130
/dev/ddd/0134
```

Example 2: The following command generates spare sector, ift, and ce slices for a DD-60 physical disk.

```
mkspice -t dd60 2130.0 2130.1
```

This command generates the following files:

```
/dev/spare/2130.0
/dev/spare/2130.1
/dev/ift/2130.0
/dev/ift/2130.1
/dev/ce/2130.0
/dev/ce/2130.1
/dev/ddd/2130.0
/dev/ddd/2130.1
```

Example 3: The following command generates the `/dev/ddd` node for an SSD:

```
mkspice -t SSD -l 262144 full_ssd
```

This command generates the following file:

```
/dev/ddd/full_sdd
```

SEE ALSO

`ift`(8), `spmap`(8)

`pdd`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

mlmkdir – Creates a multilevel directory (MLD)

SYNOPSIS

mlmkdir *path...*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The mlmkdir command creates a multilevel symbolic link and directory-tree pairs by using its own internal naming convention for the root of the directory trees it creates. *path* is the name of the multilevel symbolic link created by mlmkdir. Only an appropriately authorized user can create a multilevel directory.

mlmkdir places a 0777 mode on the target directory of the multilevel symbolic link. This mode is automatically copied to any new labeled subdirectories created within the multilevel directory (MLD) structure.

If a MLD is created by a mechanism other than mlmkdir, or an authorized user changes the mode of a MLD after its creation, the mode of the directory and the subsequent modes of all new labeled subdirectories is left to the discretion of the user.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Allowed to specify any MLD path.
sysadm	Allowed to specify any MLD path, subject to security label restrictions on the path. Shell redirected I/O is subject to security label restrictions.

If the PRIV_SU configuration option is enabled, the super user is allowed to specify any MLD path.

EXIT STATUS

If successful, mlmkdir exits with a 0. Otherwise, it prints a diagnostic message and exits with a nonzero value.

SEE ALSO

`mlrmdir`(8)

`chmod`(1), `ln`(1), `mkdir`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

General UNICOS System Administration, Cray Research publication SG–2301

NAME

mlrmdir – Removes a multilevel directory (MLD)

SYNOPSIS

mlrmdir *path...*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The mlmkdir command removes a multilevel symbolic link and its related directory. The name of the multilevel directory is derived from the multilevel symbolic link.

When a multilevel directory (MLD) is removed, the directory is removed first, then the multilevel symbolic link is removed. If the directory cannot be removed (for example, it is not empty), it is left intact and the multilevel symbolic link is not touched. In this instance, mlrmdir issues an error and exits.

If the multilevel symbolic link cannot be removed, mlrmdir issues a warning and exits, having removed the directory, but not the symbolic link file.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Allowed to specify any path.
sysadm	Allowed to specify any path, subject to security label restrictions. Shell redirected output is subject to security label restrictions.

If the PRIV_SU configuration option is enabled, the super user is allowed to specify any path.

EXIT STATUS

If successful, mlrmdir exits with a 0; otherwise, it prints a diagnostic message and exits with a nonzero exit status.

SEE ALSO

mlmkdir(8)

rm(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

General UNICOS System Administration, Cray Research publication SG–2301

NAME

mmr – Read or write a GigaRing MMR register

SYNOPSIS

```
/etc/mmr [-a] [-d delta] gr_node [gr_node1 gr_node2 gr_node3 ...]
/etc/mmr [-d delta] [-m mmrno] [-w mmr_data] gr_node
```

IMPLEMENTATION

CRAY T90 systems with GigaRing-based I/O

CRAY J90 systems with GigaRing-based I/O

DESCRIPTION

The mmr command is used to read, display, and optionally write GigaRing MMR (memory mapped registers) registers. The mmr command uses the character special files in the /dev/fmsg directory as an interface to the GigaRing channel(s).

The mmr command has two forms. The first form reads a predetermined number of MMR registers (see the -a option below) from the list of *gr_nodes*. The second form reads or writes one MMR register from one *gr_node*.

A *gr_node* is either a full path name representing a character special file in /dev/fmsg or a GigaRing node id. For information on the files in /dev/fmsg, see fmsg(4).

The mmr command accepts the following arguments:

-a	Read all the MMR registers from the specified GigaRing node. The default is to read a predetermined set of MMR registers.
-d *delta*	Use delta addressing instead of explicit GigaRing addressing.
-m *mmrno*	Read or write only the specified MMR register. The -m option may not be used along with the -a option.
-w *mmr_data*	Specifies writing an MMR register with *mmr_data*. This option requires that the -m option also be specified.
gr_node	Full path name representing a character special file in /dev/fmsg or a GigaRing node id.

EXAMPLES

The following example reads and display the default MMR values from the GigaRing node at ring 2, node 2:

```
# /etc/mmr /dev/fmsg/e0202

  1 GR_SETUP                    34  -Scrubber  +Scrubber  32_bit_client
  2 GR_NODE_ID               20202  Ring 02  Node 02
  3 GR_ID_MASK                   0  (0x0)
 20 GR_PORT_ACCESS               5  Master
 21 GR_PORT_LOCK       25353370015  (0xabadf00d)
 24 GR_RING_MASK                 0  (0x0)
 30 GR_ERROR_COUNTER             0  client    000   +ring    000  -ring 000
 31 GR_NEG_RING_ERRORS           0  Scrub_old 000   CRC_count 00
 32 GR_POS_RING_ERRORS           0  Scrub_old 000   CRC_count 00
 33 GR_CLIENT_ERRORS             0
 70 GR_SEND_BUF_FULL             0  (0x0)
 71 GR_RECV_BUF_FULL             0  (0x0)
```

The following example reads and displays the GigaRing node id from the GigaRing node at ring 2, %node 2:

```
# /etc/mmr -m 2 /dev/fmsg/e0202
   2 GR_NODE_ID               20202  Ring 02  Node 02
```

The following example writes, reads, and displays MMR scratch register 0 on the GigaRing node at ring2, node 2:

```
# /etc/mmr -m 0100 -w 0777  /dev/fmsg/e0202
 100 GR_SCRATCH0                 777  (0x1ff)
```

FILES

/dev/fmsg/*

SEE ALSO

fmsg(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

monacct – Creates monthly summary files

SYNOPSIS

/usr/lib/acct/monacct [*number*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The monacct command creates standard UNIX System V accounting summary files in /usr/adm/acct/fiscal and restarts summary files in /usr/adm/acct/sum. Invoke monacct once each month or accounting period.

The monacct command accepts the following operand:

number Month or accounting period. The default is the current month (01 through 12). The default is useful if monacct is executed using cron(8) on the first day of each month.

EXAMPLES

The following example creates the accounting summary files for May:

```
/usr/lib/acct/monacct 5
```

SEE ALSO

acct(8), acctsh(8), prdaily(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

mount, umount – Mounts and unmounts the file system

SYNOPSIS

mount command:

(1):
/etc/mount [-p]

(2):
/etc/mount -s

(3):
/etc/mount -a [-f *fstab*] [-b] [-t *type*] [-Q *quota_file*]
/etc/mount -a [-f *fstab*] [-b] [-t *type*] [-q] [-v]

(4):
/etc/mount [-r] [-f *fstab*] [-b] -t *type* -o *options* [-v] *special directory*

(5):
/etc/mount [-r] [-f *fstab*] [-q] [-b] [-t *type*] [-o *options*] [-v] *special directory*

(6):
/etc/mount [-r] [-f *fstab*] [-Q *quota_file*] [-b] [-t *type*] [-o *options*] [-v] *special directory*

umount command:

(1):
/etc/umount [-v] *special directory*

(2):
/etc/umount -a [-v]

(3):
/etc/umount -t *type* [-h *host*] [-v]

(4):
/etc/umount -h *host* [-v]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

File systems other than root (/) are considered removable because they can be either available or unavailable to users. The mount command announces to the system that *special*, which is a block special file or a remote resource, is available to users at the mount point *directory*. The directory must already exist; then it becomes the name of the root of the newly mounted file system.

The mount command, when entered with arguments shown in synopses (4), (5), or (6), adds an entry to the table of mounted devices, /etc/mnttab (see mnttab(5)). The umount command removes the entry.

If you invoke the mount command shown in synopsis (1) with the -p option or without options, mount prints the entire mount table. If you invoke mount with an incomplete argument list (that is, if you omit the *type*, *options*, *special*, or *directory* argument in the mount command shown in synopses (5) or (6), mount searches /etc/fstab (see fstab(5)) for the missing arguments.

If you invoke the mount command shown in synopsis (2) with the -s option, mount prints the shared mount table, which includes information about shared file systems only.

If you invoke the mount command with the -a option, as shown in synopsis (3), mount mounts all file systems in /etc/fstab. The -a option on the umount command (see synopsis (3)) removes all entries except root from the kernel mount table.

If you include the -v option, a list of the arguments passed to the mount(2) system call is displayed, including those determined by /etc/fstab.

If /etc/fstab is set up with the proper quota configuration, the mount command activates file system quotas on the configured quota control file in the default enforcement mode only if the -q option is present. See the fstab(5) man page for a description of the quota option.

The mount command accepts the following options and operands:

-p Prints the list of mounted file systems in a format suitable for use in the /etc/fstab file.

-s Prints the shared mount table. The shared mount table includes only shared file systems and displays information about which systems have a particular file system mounted.

-a Tries to mount all file systems listed in the /etc/fstab file. If specified with the -t option, the mount command tries to mount all file systems of the specified type. The -a option enables the mount command to build a dependency tree of mount points. If a file system is already mounted, mount displays a message and tries to mount the next file system. If a mount fails for another reason, the mount command does not mount the file systems under the one that failed.

-f *fstab* Specifies an alternative file that will be used rather than the /etc/fstab file.

-b Bypasses the file system validation.

-t *type* Specifies the type of file system that will be mounted. If you omit this option and mount cannot find the information in /etc/fstab, the root file system type is used. The following values are valid for *type*:

NC1FS Specifies the PVP file system type.

NFS Specifies the network file system (NFS) type.

NFS3 Specifies the network file system (NFS) version 3 type. If your system is not licensed for ONC+™, you will be unable to mount NFS version 3 type file systems.

PROC Specifies the /proc file system type.

INODE Specifies the /inode file system type.

-Q *quota_file* Manually activates quota enforcement. Use this option only if the quota configuration is not defined in /etc/fstab. *quota_file* must be the fully qualified name of the quota file. If the quota file does not have the default name or if it does not reside in the root directory of the file system, you must use this option instead of the -q option. You cannot use this option with the -q option or if the -r option is specified and the quota file resides on this file system.

-q Activates quota enforcement. The quota configuration for the file system usually is defined in /etc/fstab; however, if the quota configuration is not defined in /etc/fstab, the quota control file defaults to .Quota60 in the root directory of the file system. You cannot use this option with the -Q option.

-v Displays a list of the arguments passed to the mount(2) system call, including those determined by the /etc/fstab file.

-r Indicates that *special* must be mounted read-only. If *special* is write-protected, you must use this option.

-o *options* Permits a comma-separated argument list to be specified. The following arguments are available for use with any file system types:

nochk Specifies no file system check.

quota=*file*|noquota
Specifies that usage limits are enforced or not enforced, respectively. The default is noquota.

rw|ro The rw argument specifies read-write file systems; the ro argument specifies read-only file systems. The default is rw.

The following arguments are available for use with NFS file systems:

bg Retries the mount request in the background if the server does not respond.

cksum Performs a checksum operation on outgoing NFS requests.

cray Specifies use of a modified NFS protocol to reduce system overhead. This argument is valid only if the server for this file system is running the UNICOS operating system and one or more cnfsd (see nfsd(8)) daemons.

intr Allows keyboard interrupts on hard mount requests.

kerberos
Use Kerberos authentication (AUTH_KERB RPC) for NFS transactions. This option requires an ONC+™ site license.

noac Suppresses attribute caching.

`nocto` Suppresses fresh attributes when a file is opened. This argument can improve read performance for files that are read frequently and are not expected to change.

`nolock` Suppresses access to the Network Lock Manager `lockd`(8) daemon for this file system. Advisory and mandatory lock requests are rejected for files on NFS file systems mounted with this option. If the remote NFS server does not run the `lockd` and `statd`(8) daemons, the system will hang.

`nosuid` Does not allow the execution of `setuid` calls.

`nreadah = n`
Sets the number of asynchronous readaheads to n. The default value is 1.

`port=`*n* Sets the server IP port number to *n*.

`retrans=`*n*
Sets the number of NFS retransmissions to *n*.

`rsize=`*n* Sets the read buffer size to *n* bytes; the default size is 8192 bytes for standard NFS file systems and 32,768 bytes when the `cray` argument is used.

`soft` Returns an error if the server does not respond; the default is `hard`, indicating that `mount` continues to try the request until the server responds.

`spongy` Uses `soft` semantics for `stat`, `lookup`, `fsstat`, `readlink`, and `readdir` file system operations and `hard` semantics for others. This option is meant to be similar to `hard`, except that processes will not be hung forever when they try to access mount points to inactive servers.

`sync` Disables NFS V3 asynchronous writes.

`timeo=`*n* Sets the initial NFS time-out to *n* tenths of a second. The default timeout value is 7 tenths of a second.

`wsize=`*n* Sets the write buffer size to *n* bytes; the default size is 8192 bytes for standard NFS file systems and 32,768 bytes when the `cray` argument is used.

If the BDS option is installed, the following options are valid for NFS filesystems that have BDS service enabled:

`bds` Turn on bulk data service for this file system.

`bdsauto=`*size*
For all read/write requests sized greater or equal to size bytes, do BDS I/O instead of NFS I/O.

`bdswindow=`*size*
Set the TCP protocol send and receive windows to size bytes instead of the default of 4Mbytes.

special Indicates the block special file that will be mounted on *directory*. If the file system type is NFS, *special* must be of the form *hostname*:*pathname*.

directory Indicates the directory mount point for *special*. The directory must already exist.

The umount command announces to the system that the file system that is previously mounted (*special* or *directory*) must be made unavailable. The following options are available for use with the umount command:

-v Displays a list of the arguments passed to the umount(2) system call, including those determined by the /etc/fstab file.

-a Unmounts all file systems, except root, that are currently mounted.

-t *type* Unmounts all file systems of the specified type. If used with the -h option, the file system type must be NFS.

-h *host* Unmounts all NFS file systems mounted on the specified host.

special Indicates the block special file that will be mounted on *directory*. If the file system type is NFS, *special* must be of the form *hostname*:*pathname*.

directory Indicates the directory mount point for *special*.

NOTES

Any user can use the mount command to list mounted file systems and resources. Only an appropriately authorized user can mount and unmount file systems.

If mounting an NFS file system using the NFS v3 protocol, asychronous writes will only be done on hard or spongy-mounted file systems.

On a UNICOS system with device enforcement enabled, the security label range of the file system must fall within the security label range of the device.

If these commands are installed with privilege assignment lists (PALs), a user with one of the following active categories may perform the action shown:

Active Category	Action
system, secadm, sysadm	May mount and unmount any file system

If the PRIV_SU configuration option is enabled, the super user or a user who has the mount permbit may mount and unmount any file system. Users who are not the super user are subject to access restrictions on the mount point path.

WARNINGS

Although checks are in the system, you should not mount a block special file or remote resource that is not a recognized file system. Similarly, any physical unmounting, such as disconnecting the network or powering down the disk, must follow the umount command to ensure that the device is not being accessed.

MESSAGES

If the mount(2) system call fails, the mount command prints an appropriate message. If the file system to be mounted is currently mounted under another name, the mount command issues a warning. If the resource is not available, a remote resource mount fails.

If *special* is not mounted or if it is busy, the umount command fails. The *special* directory is considered busy if it contains an open file or a user's working directory. Then, you can use fuser(8) to list and kill processes that are using *special*.

EXAMPLES

The following example mounts an NFS file system:

```
mount -t NFS -o soft mach:/usr/mach /nfs/mach
```

FILES

/etc/fstab	File system table
/etc/mnttab	Mount table

SEE ALSO

bds(8), fuser(8), mountd(8), nfsd(8), quadmin(8)

mount(2), quotactl(2), umount(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

libudb(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

fstab(5), mnttab(5), quota(5), udb(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

mountd – Performs NFS mount request server function

SYNOPSIS

/etc/mountd [-n]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The mountd command is a Remote Procedure Call (RPC) server that answers file system mount requests. It reads the /etc/xtab file, described in exports(5), to determine the file systems that are available to each machine and user. It also identifies the clients that have file systems mounted. The exportfs(8) command places entries in /etc/xtab.

NOTES

If your system is licensed for ONC+™, mountd will also register for version 3 of the nfs/mount protocol. This protocol is necessary for mounting network file system (NFS) version 3 file systems.

The configuration parameter NFS_REMOTE_RW_OK determines whether a remote file system can be mounted. If NFS_REMOTE_RW_OK is nonzero, a remote file system can be NFS-mounted in read-write mode. If NFS_REMOTE_RW_OK is 0, read-only mode is the only mode in which the remote file system can be mounted. If NFS_SECURE_EXPORT_OK is nonzero, the system can export file systems (which a remote host can then mount). If NFS_SECURE_EXPORT_OK is 0, file systems cannot be exported (or mounted from remote hosts).

The mountd command catches the SIGHUP signal and reregisters itself with portmap(8) when it receives the signal. This enables mountd to continue running properly when portmap must be restarted.

By default, the mountd command performs port checking to ensure that mount requests originate from processes running with root privileges. It rejects requests received from non-privileged ports. This port monitoring can be disabled using the -n option. This option should not be used unless it is absolutely necessary. Some NFS client implementations may not make mount requests from privileged ports. Port monitoring prevents receipt of imitations of valid NFS requests sent from unauthorized user processes. Sending unauthorized requests over NFS is known as spoofing.

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

SEE ALSO

exportfs(8), mount(8), portmap(8), umount(8)

privtext(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

exports(5), services(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

mppview – Displays massively parallel processing (MPP) system activity

SYNOPSIS

/usr/bin/mppview [-h *host*] [-L] [-r *refresh*] [-T *timeout*] [-t]

IMPLEMENTATION

Cray MPP systems

DESCRIPTION

The mppview command displays a map of active partitions running on the Cray MPP system. It uses the curses(3) library to drive the terminal display so that many terminal types may be supported. To specify the system to be monitored, use the *host* option. If you do not specify *host*, the local system is monitored. The screen refreshes every *interval* seconds until you type q to quit mppview.

The mppview command communicates through rpc(3C) with the system activity monitoring (sam) server, samdaemon(8), running on the host system to obtain the information that you request.

The mppview command accepts the following options:

[-h *host*] Specifies the network name of the host to be monitored. The default *host* is the local system.

[-L] Line-mode display. Don't enter refresh mode.

[-r *refresh*] The refresh rate in seconds. The default is 3 seconds.

[-T *timeout*] For text mode only. The default is 60 seconds.

[-t] Text mode. The default is screen mode. In text mode, you can use the following commands:

REGION DISABLED	Disabled Nodes
REGION PARTITION	Active Partitions
REGION POOL	Active Pools
REPORT CONFIG	MPP Configuration
REPORT RUNQUEUE	Running MPP Applications
REPORT WAITQUEUE	Waiting MPP Applications

Interactive Input

After you enter the mppview command, an interactive screen appears. At the top of the screen, the following option menu is displayed:

```
help   summary   torus   page   refresh   clear   quit
```

To move between the options, use the <TAB> key. When the desired option is highlighted, press the <RETURN> key to execute it. For the help option, first you must press the <?> key to enable help mode. For the other options, you can type just the first letter to execute the option directly.

Option	Description
help	Displays help information. You first must press the <?> key to enable help mode; then, other menu options can bring up their help displays.
summary	Displays summary information for all of the MPP partitions.
torus	Graphically displays each node of the Cray MPP system. To select the information displayed in a node, use the torus submenu options:
	user — User name of the program on that node
	group — Group ID of the program
	pid — Process ID of the program
	prog — Program name
page	Moves you to other pages. If the display does not fit on one screen page, enter one of the following commands to select a page:
	+ — Forward one page
	- — Backward one page
	# — Select a page number
refresh	Sets the rate at which the display is refreshed. The refresh is expressed in tenths of a second. Refresh rates smaller than the rate at which the samdaemon(8) is collecting data do not take affect.
clear	Clears the screen and repaints the entire display.
quit	Quits the program.

After the csam(8) utility is running on your terminal, you can use the following keys to change displays:

Key	Description
TAB	Changes the selected (highlighted) option.
?	Enables help mode, in which other menu options can bring up their help displays.

NOTES

If mppview is not compiled on a system with UNICOS MAX software, you will get a message, "No MPP system present", when you try to execute the mppview command.

SEE ALSO

csam(8), samdaemon(8), xsam(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`mrinfo` – Obtains routing information from a multicast router

SYNOPSIS

`mrinfo` [`-d` [*debuglevel*]] [`-r` *retries*] [`-t` *timeout*] *router*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `mrinfo` command obtains multicast information from the specified router (which should be a host name or a host address).

The `mrinfo` command accepts the following options and arguments:

`-d` [*debuglevel*] If the `-d` option is specified with no argument, the debug level defaults to 2. Debug levels have the following effects:

- Level 1 Prints all logged messages to `stderr`.
- Level 2 Prints all level 1 messages plus notifications of significant events to `stderr`.
- Level 3 Prints all level 2 messages plus notifications of all packet arrivals and departures to `stderr`.

`-r` *retries* Sets the number of retries when requesting information from a router. Default is 3 retries.

`-t` *timeout* Sets the time out in seconds when waiting for a response from a router. Default is 4 seconds.

router Specifies the router using either a host name or a host address.

The `mrinfo` command prints a description of the router, then prints the virtual interface list in the following form:

```
local_address -> remote_address (remote_name) [metric/threshold/type]
```

SEE ALSO

`map_mbone`(8), `mrouted`(8)

NAME

`mrouted` – Forwards an Internet Protocol (IP) multicast datagram using Truncated Reverse Path Broadcasting (TRPB)

SYNOPSIS

`/etc/mrouted` [`-c` *config_file*] [`-d` [*debug_level*]]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `mrouted` daemon is an implementation of the Distance-Vector Multicast Routing Protocol (DVMRP), an earlier version of which is specified in RFC 1075. It maintains topological knowledge using a distance-vector routing protocol (like RIP, described in RFC 1058), upon which it implements a multicast forwarding algorithm called Truncated Reverse Path Broadcasting (TRPB).

The `mrouted` daemon forwards a multicast datagram along the shortest reverse path tree rooted at the subnet on which the datagram originates. The tree is a *broadcast* tree, which means it includes all subnets reachable by a cooperating set of `mrouted` routers. However, the datagram is not forwarded onto leaf subnets of the tree if those subnets do not have members of the destination group. Furthermore, the Internet Protocol (IP) time-to-live (TTL) of a multicast datagram may prevent it from being forwarded along the entire tree.

In order to support multicasting among subnets that are separated by unicast routers that do not support IP multicasting, `mrouted` includes support for "tunnels," which are virtual point-to-point links between pairs of `mrouted` daemons located anywhere in an internet. IP multicast packets are encapsulated for transmission through tunnels, so that they look like normal unicast datagrams to intervening routers and subnets. The encapsulation occurs when a packet enters a tunnel, and is stripped out when it exits from a tunnel. By default, the packets are encapsulated using the IP-in-IP protocol (IP protocol number 4). Older versions of `mrouted` encapsulate using IP source routing, which puts a heavy load on some types of routers. This version supports IP source route encapsulation only for backward compatibility.

The tunnel mechanism allows `mrouted` to establish a virtual internet for the purpose of multicasting only, which is independent of the physical internet, and which may span multiple Autonomous Systems (AS). This capability is intended for experimental support of internet multicasting only, pending widespread support for multicast routing by the regular unicast routers. The `mrouted` daemon suffers from the well-known scaling problems of any distance-vector routing protocol, and does not support hierarchical multicast routing or interoperation with other multicast routing protocols.

The `mrouted` daemon handles multicast routing only; there may or may not be a unicast router running on the same host as `mrouted`. With the use of tunnels, it is not necessary for `mrouted` to have access to more than one physical subnet in order to perform multicast forwarding.

The `mrouted` command accepts the following options:

`-c` *config_file* Specifies an alternative configuration file. Default is `/etc/mrouted.conf`.

`-d` [*debug_level*] If the `-d` option is given with no argument, the debug level defaults to 2. Debug levels have the following effects:

- Level 1 Prints all logged messages to `stderr`.
- Level 2 Prints all level 1 messages plus notifications of significant events to `stderr`.
- Level 3 Prints all level 2 messages plus notification of all packet arrivals and departures to `stderr`.

If no `-d` option is specified, or if the debug level is specified as 0, `mrouted` detaches from the invoking terminal. Otherwise, it remains attached to the invoking terminal and is responsive to signals from that terminal. Regardless of the debug level, `mrouted` always writes warning and error messages to the system log demon.

Configuration

The `mrouted` daemon automatically configures itself to forward on all multicast-capable interfaces, that is, interfaces that have the `IFF_MULTICAST` flag set (excluding the loopback interface), and it finds other `mrouted` daemons directly reachable which use those interfaces. To override the default configuration, or to add tunnel links to other `mrouted` daemons, configuration commands may be placed in the file `/etc/mrouted.conf` (or an alternative file, specified by the `-c` option). There are two types of configuration commands:

`phyint` *local-addr* [`disable`] [`metric` *m*] [`threshold` *t*]
`tunnel` *local-addr remote-addr* [`metric` *m*] [`threshold` *t*] [`srcrt`]

The `phyint` command disables multicast routing on the physical interface identified by a local IP address, *local-addr*, or associates a non-default metric or threshold with the specified physical interface. `phyint` commands must precede `tunnel` commands.

The `tunnel` command establishes a tunnel link between a local IP address, *local-addr*, and remote IP address, *remote-addr*, and associates a non-default metric or threshold with that tunnel. The tunnel must be set up in the `mrouted.conf` files of both ends before it can be used. For backward compatibility with older `mrouted` daemons, the `srcrt` keyword specifies encapsulation using IP source routing.

The metric is the "cost" associated with sending a datagram on the given interface or tunnel; it may be used to influence the choice of routes. The metric defaults to 1. Metrics should be kept as small as possible, because `mrouted` cannot route along paths with a sum of metrics greater than 31. The following metrics are recommended:

Metric	Description
1	Local area network (LAN), or tunnel across a single LAN
2	Serial link, or tunnel across a single serial link
3	Multi-hop tunnel

The threshold is the minimum IP TTL required for a multicast datagram to be forwarded to the given interface or tunnel. It controls the scope of multicast datagrams. (The TTL of forwarded packets is only compared to the threshold; it is not decremented by the threshold. Every multicast router decrements the TTL by 1.) The default threshold is 1. Suggested thresholds are as follows:

Metric	Description
32	For links that separate sites
64	For links that separate regions
128	For links that separate continents

In general, all `mrouted` daemons connected to a particular subnet or tunnel should use the same metric and threshold for that subnet or tunnel.

The `mrouted` daemon does not initiate execution if it has fewer than two enabled virtual interfaces (VIF), where a VIF is either a physical multicast-capable interface or a tunnel. The daemon logs a warning if all of its VIFs are tunnels; such a `mrouted` configuration would be better replaced by more direct tunnels.

Signals

The `mrouted` daemon responds to the following signals:

Signal	Description
`HUP`	Terminates execution gracefully (sends termination messages to all neighboring routers).
`TERM`	Terminates execution gracefully (sends termination messages to all neighboring routers).
`INT`	Terminates execution gracefully (sends termination messages to all neighboring routers).
`USR1`	Dumps the internal routing tables to `/usr/tmp/mrouted.dump`.
`QUIT`	Dumps the internal routing tables to `stderr` (only if `mrouted` was invoked with a nonzero debug level).

EXAMPLES

The following shows an example of a routing table:

```
Virtual Interface Table
Vif  Local-Address                    Metric  Thresh  Flags
 0   36.2.0.8      subnet: 36.2          1       1    querier
                   groups: 224.0.2.1
                           224.0.0.4
 1   36.11.0.1     subnet: 36.11         1       1    querier
                   groups: 224.0.2.1
                           224.0.1.0
                           224.0.0.4
 2   36.2.0.8      tunnel: 36.8.0.77     3       1
                   peers : 36.8.0.77
 3   36.2.0.8      tunnel: 36.8.0.110    3       1

Multicast Routing Table
Origin-Subnet    From-Gateway   Metric   In-VIF  Out-VIFs
36.2                               1        0    1* 2  3*
36.8             36.8.0.77         4        2    0* 1* 3*
36.11                              1        1    0* 2  3*
```

In this example, there are four VIFs connecting to two subnets and two tunnels. The VIF 3 tunnel is not in use (no peer address). The VIF 0 and VIF 1 subnets have some groups present; tunnels never have any groups. This instance of `mrouted` is the one responsible for sending periodic group membership queries on the VIF 0 and VIF 1 subnets, as indicated by the `querier` flags.

Associated with each subnet from which a multicast datagram can originate is the address of the previous hop gateway (unless the subnet is directly connected), the metric of the path back to the origin, the incoming VIF for multicasts from that origin, and a list of outgoing VIFs. An asterisk (`*`) means that the outgoing VIF is connected to a leaf of the broadcast tree rooted at the origin, and a multicast datagram from that origin will be forwarded on that outgoing VIF only if there are members of the destination group on that leaf.

FILES

`/etc/mrouted.conf`

SEE ALSO

Deering, S., "Multicast Routing in Internetworks and Extended LANs" in the *Proceedings of the ACM SIGCOMM ´88 Conference* for information on TRPB, along with other multicast routing algorithms.

NAME

msgd – Allows operators to display action messages

SYNOPSIS

/usr/lib/msg/msgd

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The msgd command allows the operator to display action messages, such as tape mount messages. Action messages consist of the following: a message number, the time the message was sent, and message text. An action message requires a reply from the operator unless it is canceled by the sender. msgd displays messages in order of receipt and can be used by anyone with a special operator group ID.

NOTES

If this command is installed with a privilege assignment list (PAL), a user who is assigned the following privilege text upon execution of this command is allowed to perform the actions shown:

Privilege Text	Action
showall	Allowed to see all messages.
both	Allowed to see all messages.

If this command is installed with a PAL, a user with one of the following active categories is allowed to perform the action shown:

Active Category	Action
system, secadm, sysadm, sysops	Allowed to see all messages.

If the PRIV_SU configuration option is enabled, the super user is allowed to see all messages.

SEE ALSO

infd(8), msgdaemon(8), msgdstop(8), rep(8)

msgi(1), msgr(1), privtext(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

NAME

msgdaemon – Starts the message daemon

SYNOPSIS

/usr/lib/msg/msgdaemon [-l]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The msgdaemon command starts the message daemon, which handles the communication between users and operators. Most commonly, the message daemon starts automatically when the system comes up.

The msgdaemon command accepts the following option:

-l Create and use a linear message log file, rather than a circular log.

The message daemon lets users send action or informative messages to the operator. When a user sends an action message, the operator sees the message and replies to it. The daemon returns the operator reply to the message sender. When a user sends an informative message, the operator sees the message but does not reply.

The message daemon keeps track of user messages and operator replies, and transports them to their appropriate destinations. The message daemon also logs all informative and action messages in order of receipt.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to start the message daemon.

If the PRIV_SU configuration option is enabled, the super user is allowed to start the message daemon.

MESSAGES

msgdaemon returns an error if the message daemon is already running.

FILES

/usr/spool/msg/msglog.log	Log file
/usr/spool/msg/mdlock	Locking file

SEE ALSO

`infd`(8), `msgd`(8), `msgdstop`(8), `oper`(8), `rep`(8)

`msgi`(1), `msgr`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

msgdstop – Stops the message daemon

SYNOPSIS

/etc/msgdstop

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The msgdstop command causes the message daemon to stop executing. The message daemon can also be stopped by the following command:

```
kill -2 pid
```

The *pid* operand is the process ID of the message daemon.

If the message daemon still refuses to terminate, the super user may kill it with the following command:

```
kill -9 pid
```

The *pid* operand is the process ID of the message daemon.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

SEE ALSO

infd(8), msgd(8), msgdaemon(8), oper(8), rep(8)

kill(1), msgi(1), msgr(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

`mvdir` – Moves a directory

SYNOPSIS

`/etc/mvdir` *dirname name*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `mvdir` command moves directories within a file system.

The `mvdir` command accepts the following operands:

dirname Must be a directory.

name Must not already exist.

Neither name may be a subset of the other; `/x/y` cannot be moved to `/x/y/z`, and vice versa.

NOTES

The results of `mvdir` are unpredictable if you are not the super user.

SEE ALSO

`mkdir`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

NAME

mverr – Moves the errlog file

SYNOPSIS

/etc/mverr

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The mverr command terminates errdemon(8), moves /usr/adm/errfile to /usr/adm/errfile[*lastfile* + 1], and restarts errdemon.

FILES

/usr/adm/errfile	Default error file

SEE ALSO

errdemon(8) for information on invoking the error-logging daemon
errstop(8) for information on stopping errdemon(8)

NAME

mvfiles – Manages AIR log files

SYNOPSIS

/usr/air/bin/mvfiles

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The mvfiles shell script manages the size of the automated incident reporting (AIR) log files in the /usr/spool/air/logs directory by checking them for errors, moving them into a subdirectory named the current date in the /usr/spool/air/data directory, and then reopening them.

The aird(8) process must be running on the system for mvfiles to execute correctly; mvfiles sends a signal to aird(8) in order to reopen the log files.

NOTES

The destination directory must be on the same logical disk device as the source because the mv(1) command does not link across devices and because, in order not to lose data, mvfiles depends on links when moving the files.

If you want to move log files located in a directory other than /usr/spool/air/logs, or if you wish to move log files into a directory other than /usr/spool/air/data/*date*, you can set the AIRSPOOL and DESTDIR environment variables in mvfiles.sh to different directories.

SEE ALSO

aird(8), airexist(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

named – Specifies Internet domain name server

SYNOPSIS

/etc/named [-d [*debuglevel*]] [-p *port#*[/*localport#*]] [-b *bootfile*] [-q] [-r] [-S[t|u] *tos*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The named daemon is the Internet domain name server. See RFCs 1033, 1034, and 1035 for more information on the Internet name-domain system. Without any arguments, named reads the default boot file /etc/named.boot (see named.boot(5)), reads any initial data, and listens for queries.

The named program accepts the following options:

-d [*debuglevel*] Prints debugging information. The *debuglevel* is a number that determines the level of messages that are printed. The default *debuglevel* is 1.

-p *port#*[/*localport#*]
Use nonstandard port numbers. The default is the standard port number as returned by getservbyname(3) for service "domain." The argument can specify two port numbers separated by a slash ("/") in which case the first port number is that used when contacting remote servers, and the second one is the service port bound by the local instance of named. This is used mostly for debugging purposes.

-b *bootfile* Specifies an alternate boot file for named to use. The -b is optional, which allows you to specify a file by using a leading dash.

-q Traces all incoming queries if named has been compiled with QRYLOG defined. The functionality of this option can also be invoked by the boot file directive options query-log.

-r Turns recursion off in the server. Answers can come only from local (primary or secondary) zones. This can be used on root servers. The functionality of this option can also be invoked by the boot file directive options no-recursion.

-S *tos* Sets the Type-of-Service (IP TOS) option on all connections to the value *tos*, which may be a numeric TOS value or a symbolic TOS name found in the /etc/iptos file.

-St *tos* Sets the IP TOS option on stream (TCP-based) connections to the value *tos*, which may be a numeric TOS value or a symbolic TOS name found in the /etc/iptos file.

-Su *tos* Sets the IP TOS option on datagram (UDP-based) connections to the value *tos*, which may be a numeric TOS value or a symbolic TOS name found in the /etc/iptos file.

Any additional argument is taken as the name of the boot file. If multiple boot files are specified, only the last is used.

The boot file contains information about where the `name` server is to get its initial data. Lines in the boot file cannot be continued on subsequent lines. See `named.boot`(5) for additional information on the `named.boot` file.

NOTES

The following signals produce the specified results when sent to the server process by using the `kill`(1) command.

Signal	Description
`SIGHUP`	Causes the server to read `named.boot` and reload the database. If the server is built with the `FORCED_RELOAD` compile-time option, then `SIGHUP` also causes the server to check the serial number on all secondary zones. Typically, the serial numbers are checked only at the SOA-specified intervals.
`SIGINT`	Dumps the current database and cache into `/usr/tmp/named_dump.db`.
`SIGIOT`	Dumps the statistics data into `/usr/tmp/named.stats` if the server is compiled as `-DSTATS`. Statistics data is appended to the file.
`SIGSYS`	Dumps the profiling data into `/usr/tmp/gmon.out` if the server is compiled with profiling (server `forks`, `chdirs`, and `exits`).
`SIGTERM`	Dumps the primary and secondary database files. `SIGTERM` is used to save modified data on shutdown if the server is compiled with dynamic updating enabled.
`SIGUSR1`	Turns on debugging. Each `SIGUSR1` increments by one debug level.
`SIGUSR2`	Turns off debugging.
`SIGWINCH`	Toggles off logging of all incoming queries by using `syslog`(3C). This signal requires the server to be built by using the `QRYLOG` option.

FILES

`/etc/named.boot`	Name server configuration boot file
`/etc/named.pid`	Process ID
`/usr/tmp/named_dump.db`	Dump of the name server database
`/usr/tmp/named.run`	Debug output
`/usr/tmp/named.stats`	Name server statistics data

SEE ALSO

`named-xfer`(8)

`hostname`(1), `kill`(1), `nslookup`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`gethost`(3C), `getservbyname`(3C) (see `getserv`(3C), `resolver`(3C), `signal`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`named.boot`(5), `masterfile`(5), `resolv.conf`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

RFCs 882, 883, 974, 1033, 1034, 1035, and 1123

Appendix D, *UNICOS Networking Facilities Administrator's Guide*, Cray Research publication SG–2304

NAME

named-xfer – Performs a domain system zone transfer

SYNOPSIS

/etc/named-xfer -z *zone* -f *db_file* [-s *serial*] [-d *debug_level*] [-l *debug_log_file*] [-t *trace_file*] [-p *port*] [-P *port*] [-q] *servers*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The named-xfer command performs a domain-name zone transfer. named(8) uses this facility.

The named-xfer program accepts the following options:

-z *zone* Specifies the zone for which domain-name information is fetched. The zone being updated is assumed to have a current serial number of *serial*; named-xfer transfers the zone information only if the information being transferred has a serial number greater than *serial*.

-f *db_file* Specifies the database file into which the domain-name information is put. The database file is in the named(8) .db file format. A temporary file, called *db_file.XXXXXX* (in which *XXXXXX* is a unique integer), is created to hold the partial data base while the program is running. This file is renamed *db_file* after a successful zone transfer, and unlinked (except if debugging is enabled) after an unsuccessful zone transfer. If the zone transfer fails, the *db_file* file is not disturbed.

-s *serial* Specifies the current serial number of the zone being updated. The default value is 0.

-d *debug_level* Enables debugging and specifies the debug level. *debug_level* must be a nonnegative number.

-l *debug_log_file* Specifies the file to receive the debug output. You must specify the -d option with this option. The *debug_log_file* has a mktemp style ID appended to it (for example, *debug_log_file* is changed to *debug_log_file.XXXXXX* where .*XXXXXX* is the mktemp style ID). The default *debug_log_file* name is */usr/tmp/xfer.XXXXXX*.

-t *trace_file* Specifies the file to which domain system messages are written. You must specify the -d option with the -t option.

-p *port* Specifies a port number in host-byte order. *port* is expected to have a socket already attached, presumably by named(8). This option is identical to the -P option and is provided for compatibility.

-P *port* Specifies a port number in network-byte order. *port* is expected to have a socket already attached, presumably by named(8). This option is identical to the -p option and is provided for compatibility.

`-q` Inhibits error logging.

servers Specifies the servers to be contacted. These servers subsequently fetch the domain-name information for the specified zone into the specified database file. The *servers* arguments should be a list of one or more server Internet addresses. Each server is contacted once, in turn, until one answers, or until the list of servers is depleted.

EXIT STATUS

0 The `db_file` is already up-to-date.

1 The zone transfer completes successfully and `db_file` is updated.

2 The zone transfer fails because of unreachable servers or transfer time-out.

3 Any other error.

EXAMPLES

Example 1: The following example dumps zone `cmu.edu` into `/usr/named.cmu.db`, first querying address 128.251.2.21, then address 128.251.222.173. The temporary file `/usr/named/cmu.db.10402` might be used.

```
named-xfer -z cmu.edu -f /usr/named/cmu.db -s 0128.251.2.21 128.251.222.173
```

Example 2: The following example performs the same operation, but with debugging enabled and a log file specified.

```
named-xfer -f/usr/named/cmu.db -l /tmp/xfer.log -d 4 -z cmu.edu -s 0128.251.2.21
```

FILES

db_file.XXXXXX Temporary file

SEE ALSO

`named`(8)

`nslookup`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

RFC 1034, Domain Names–Concepts and Facilities, Mockapetris, November 1987

RFC 1035, Domain Names–Implementation and Specification, Mockapetris, November 1987

NAME

nconf – Display information about the IOS model E network driver

SYNOPSIS

/etc/nconf [-z *device*] [*command*]

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

DESCRIPTION

The nconf utility gets information from the network driver tables, which controls low-speed network input and output in an IOS model E. The information is obtained by using an ioctl request and is displayed on the caller's screen. The nconf utility accepts the following options:

-z *device* The device special file for nconf to communicate with the np(4) driver. The default is /dev/comm/npctl.

command If a command is specified on the command line, nconf will execute that command and exit. If no command is specified, nconf will enter screen mode and provide a refreshed display. The first display is the help screen. At this point, you can enter display and configuration commands at the prompt. nconf uses the $TERM environment variable or $term to set its display characteristics. Your screen or window should be at least 80 columns wide. Keywords are shown in uppercase for commands, but they can be entered in either uppercase or lowercase. Most numbers may be entered in octal, decimal, or hexadecimal; a leading 0 indicates octal, and a leading 0x indicates hexadecimal. You can abbreviate commands if the string entered is unique (for example, DEVICE can be abbreviated as DEV or dev, but not D).

The displays are as follows:

Display	Description
ADAPTER	Displays A130 adapter status
DEVICE	Displays device configuration
HELP	Lists the displays and commands
IOSSTAT	Displays the ios statistics
LPATH	Displays the logical path status
STATUS	Displays the device status

The configuration commands are as follows:

Command	Description
CMODE *dev* 6MB|12MB|12LP	Configures cca1 mode for a network device: *dev* Device ordinal

`CONFIG` *dev* `ON|UP|OFF|DOWN` Configures a network device. `ON` and `UP` are synonymous, as are `OFF` and `DOWN`.

dev Device ordinal

`DBE` *dev* `ON|OFF` Configures the double-bit error interrupt:

dev Device ordinal

`DFUNC` *dev code* Configures the device function code:

dev Device ordinal

code Function code

`DMODE` *dev mode* Configures the device mode:

dev Device ordinal

mode Device mode

```
DTYPE dev FEI3 | FEI3FY | FEI4  | FEICN
        FEIDS | FEIUC  | FEIVA | FEIVB
        FEIVM | CRAY   | A130  | N130
        EN643 | DX4130 | VAXBI | ULTRA
```

Configures the device type:

dev Device ordinal

`IOPATH` *dev ioc iop channel* Sets the I/O path for a device:

dev Device ordinal

ioc I/O cluster 0 through 7

iop I/O processor 0 through 3

channel I/O channel 030, 032, 034, or 036

`TIMEOUT` *dev* `IN|OUT` *time* Configures I/O channel time-out for a network device:

dev Device ordinal

time Time-out in 0.1 seconds

`TYPE` *dev* `RAW|MP|PB|A130|LCP|FY|ULTRA`

Configures driver type for a network device:

dev Device ordinal

`WATCHDOG` *dev* `ON|OFF` Enables or disables the messages produced by the watchdog function for the specified network device:

dev Device ordinal

Other commands are as follows:

Command	Description
`refresh` *n*	Sets the refresh time to *n* seconds
`scroll` *n*	Sets the size of the command scroll window to *n* lines
`end`, `exit`, `quit`	Exits screen mode

NOTES

In a PAL-only system where this command is installed with a privilege assignment list (PAL), a user who runs this command must have one of the following categories active to open and manipulate the network driver: `system`, `secadm`, `sysadm`. In a `PRIV_SU` + `PAL` or `PRIV_SU`-only system, the super user can also open and manipulate the network driver.

SEE ALSO

`iocstat`(8)

NAME

netperf – Displays X Window System TCP/IP network and UNICOS NFS statistics

SYNOPSIS

/etc/netperf -tcp -nfs [*options*] [*arguments*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The netperf command continuously displays several TCP/IP and/or network file system (NFS) statistics as a set of parallel line graphs. The name of the host is displayed in the upper left-hand corner of the window. The information is scaled so that it fills up the entire window.

The netperf command accepts the following options and arguments:

-tcp	Displays TCP/IP statistics.
-nfs	Displays NFS statistics.
-bd *color* -color *color*	Specifies the border color.
-bg *color* -background *color*	Specifies the background color.
-bitmap_color *color*	Specifies the background bit map color.
-bitmap *pathname*	Specifies the background bit map.
-bw *pixels* -border *pixels*	Specifies the border width in *pixels*. The default is 3.
-fg *color* -foreground *color*	Specifies the foreground color.
-fn *fontname* -font *fontname*	Specifies the font that the host name and labels will be displayed. The default is 6X10.
-fw -forward	Forces colors to be as specified (rather than reversed).
-geometry	The netperf window is created with a size and location determined by the specified geometry specification. The format is the standard X Window System geometry specification.

host:*display*
Runs `netperf` on a specified *host* with a specified *display*. The default is `unix:0`. The `DISPLAY` environment variable sets your default host and display number.

`-n` *stat stat stat ...*
`-not` *stat stat stat ...*
Specifies a list of statistics not to be displayed. This option supersedes any statistics specified up to that time. Omits all subsequent words on the command line that are statistics from the display.

`-rv`
`-reverse`
Reverses the screen colors black and white.

stat stat stat ...
A list of statistics to be displayed. If none are listed, all statistics are displayed. If any are listed, only those listed are displayed, unless you specify the `-n` or `-not` option. The following statistics are possible for TCP/IP:

`cksum`	Checksum errors
`dropped`	Dropped packets
`ftpc`	FTP connections
`input`	Input packets
`mbuf`	Mbufs in use
`mbufd`	Mbufs denied
`otherc`	Other connections
`output`	Output packets
`rcmdsc`	`rlogin/rcp/rsh` connections
`tcpc`	TCP connections
`tcpd`	TCP dropped connections
`tcpr`	TCP retransmitted packets
`telnetc`	`telnet` connections

The following statistics are possible for NFS:

`bytes_in`	Client bytes in
`bytes_out`	Client bytes out
`creq`	Client requests
`-help`	Invokes a help facility that lists all command-line options and
`sread`	Server reads
`swrite`	Server writes
`sreq`	Server requests

`sdup`	Server duplicate requests X-resource database formats
`timo`	Client timeouts

`-u` *seconds*
`-update` *seconds* Specifies the update interval for the graph in seconds. The default is 1.

`-st` *pixels*
`-stepsize` *pixels* Specifies the step size of the graph. The default is 1 pixel.

While `netperf` is running, you can perform certain tasks by pressing keys over the window, as follows:

Key	Description
`Q` `q`	Quits.
`R`	Resets graph and timer.
`s`	Decreases (slows) update interval by a small amount.
`S`	Decreases (slows) update interval by a large amount.
`f`	Increases update interval by a small amount.
`F`	Increases update interval by a large amount.
`?`	Help.

`-v` *verbose* Specifies verbose mode. The default is nonverbose.

NOTES

You must not invoke verbose mode to report error messages to the `stderr` of the parent process, to return exit status returned from the child processes, for the help key to display the help menu to `stdout`.

See the X Window System documentation for your frontend for more information on using X Window Systems.

BUGS

Occasionally, when the `netperf` window size changed too many times in rapid succession, it is not updated correctly.

NAME

netstart – Starts networking software

SYNOPSIS

/etc/netstart

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The netstart script is called from the rc(8) command to initialize the system's networking software. The netstart script should not need any modification, relying instead on configuration files of the commands it calls that start the networking software to accomplish site-specific configuration.

The netstart script performs the following functions:

- Executes the local script /etc/netstart.pre to perform any local initialization configured by the system administrator.
- Executes the /etc/nwmstart script to initialize the underlying network media.
- Executes the /etc/tcpstart script to initialize the TCP/IP software.
- Executes the /etc/unetup script to initialize the UltraNet software.
- Executes the /etc/nfsstart script to initialize the network file system (NFS) software.
- Executes the /etc/ypstart script to initialize the network information service (NIS) software (formerly called yellow pages).
- Executes the local script /etc/netstart.pst to perform any local initialization configured by the system administrator.

In all cases, netstart executes the indicated command or script only if it exists and is executable, allowing the disabling of a specific network feature on startup by removing, or turning off the execution bit for the appropriate start-up command or script.

SEE ALSO

brc(8), nfsstart(8), nwmstart(8), tcpstart(8), ypstart(8)

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

nettest, nettestd – Performs client and server functions for timing data throughput

SYNOPSIS

/etc/nettest [-c] [-C] [-d] [-f] [-F] [-h] [-b *bufsize*] [-S *tos*] [-n *conns*] [-p tcp|udp] [-s*n*] [-m] [-w] [*host* [*count* [*size* [*port*]]]]
/etc/nettest [-c] [-C] C-d] [-f] [-h] [-b *bufsize*] -p unix|unixd|pipe [-n *conns*] [-w] [*count* [*size* [*filename*]]]
/etc/nettest [-c] [-C] [-d] [-f] [-h] [-b *bufsize*] -p file *writefile readfile* [*count* [*size*]]
/etc/nettest -V

/etc/nettestd [-d] [-b] [-p tcp|udp] [*port*]
/etc/nettestd [-d] [-b] -p unix|unixd|pipe [*filename*]
/etc/nettestd [-d] [-b] -p file *readfile writefile*
/etc/nettestd -V

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nettest and nettestd commands invoke client and server programs that are used for timing data throughput of various methods of interprocess communication. For Transmission Control Protocol (TCP) connections, the nettest program establishes a connection with the nettestd program, and then it does *count* writes of *size* bytes, followed by *count* reads of *size* bytes. For user datagram protocol (UDP), the nettest program performs only writes; reads are not performed. The nettestd program, if used with UDP connections, reads the data packets and prints a message for each data packet it receives. The number and size of the reads and writes may not correlate with the number and size of the actual data packets that are transferred; it depends on the protocol that is chosen. If you append an optional k (or K) to the *size*, *count*, or *bufsize* value, the number specified is multiplied by 1024.

The nettest and nettestd commands accept the following options and arguments:

-c Specifies that the data must be checked to verify its accuracy. Because this is done by comparing one character at a time, using the -c option can cause noticeable throughput degradation. The data is verified by filling up the data buffer with a 32-byte repeating pattern of all the lowercase letters and the first six uppercase letters of the alphabet. This option is useful for detecting data that has been corrupted. If there is a problem with lost or duplicated data, this option might generate many error messages.

`-C` Similar to `-c`, except that the data is written out as sequential 64-bit numbers in network byte order. Because this is done by comparing one word at a time, it is slightly faster than the `-c` option, but it can still cause noticeable throughput degradation. This option is useful for detecting data that has been lost or duplicated, as it resynchronizes itself when an error is encountered. However, if the size of the lost or duplicated data is not an even multiple of 8, it may not resynchronize properly.

`-d` Turns on the socket-level debugging flag for TCP and UDP connections.

`-f` Indicates that a full-size read must be issued. Usually, when a read returns a short count, both `nettest` and `nettestd` issue a read for the remaining data for that buffer, whether or not a short count was received. (The total number of bytes is not changed.)

`-F` Turns on the `TCP_NODELAY` socket option for TCP connections. The TCP code in the kernel usually tries to send only full-sized packets over the network; this is accomplished by delaying some writes until a full packet size accumulates. The `-F` option disables this algorithm.

`-h` Indicates that hash marks will be printed. Each time a complete buffer is written or read, a hash mark is printed. If a read returns a partial count and the `-f` option is not specified, a period is printed. If the `-f` option is specified, a hash mark is printed each time a read completes, regardless of the amount of data read.

`-b` *bufsize* Specifies the amount of kernel buffering allowed for TCP and UDP connections. This option applies only to `nettest`.

`-b` Directs the daemon to detach itself from its controlling terminal and put itself into the background. This option, run as a background daemon, applies only to `nettestd`. All error messages are logged through `syslog`(3C), instead of through `perror`(3C). If both the `-C` and `-v` options of `nettest` are used in conjunction with this `-b` option, any errors that are detected in the data stream will not be reported.

`-S` *tos* Specifies the Type-of-Service (TOS) value for TCP and UDP connections. A check for the symbolic name *tos* in `/etc/iptos` determines the actual order. (The `-t` *tos* option is a valid synonym, for historical compatibility.)

`-n` *conns* Specifies the number of simultaneous connections to be opened for TCP and UNIX connections. For each connection, a subprocess is created. Each subprocess, after establishing a connection to the server and negotiating the options, suspends itself. When all of the connections have been established, a continue signal is sent to all subprocesses to start them running at the same time. As each subprocess completes, it returns its timing results, and returns that information to the main process, which then prints out the individual timing information. After all subprocesses have completed, aggregate timing results are given. The aggregate timings are based on the total amount of data transferred by all subprocesses, the start time of the first subprocess to begin writing its data to its server, and the end time of the last subprocess to complete reading its data from its server. The synchronization information shows when each subprocess began running, the duration of the data transfer for each subprocess, and the ending time of each subprocess. These times are relative to the start time of the first subprocess to begin running.

`-p` *protocol* Specifies the protocol in use. The valid values for *protocol* are `tcp`, `udp`, `unix`, `unixd`, `pipe`, and `file`.

If the `-p` option is not specified, `tcp` is the default.

The `unix` protocol uses UNIX domain stream sockets; *filename* can be specified to override the default file name `nt_socket`.

The `unixd` protocol uses UNIX domain datagram sockets; *filename* can be specified to override the default file name `nt_dsocket`.

For `pipe` protocol connections, two named pipes are created when you specify *filename*, one for reading and one for writing. The `nettest` program creates the names of these files by appending `R` and `W` to *filename*. The default names are `nt_pipeR` and `nt_pipeW`.

For `file` protocols, *writefile* is the name of the special file to which information is written; *readfile* is the name of the special file that is read. The order of *writefile* and *readfile* is reversed between `nettest` and `nettestd`. This allows the same file names to be specified in the same order for both `nettest` and `nettestd`, because the file to which `nettest` writes is the file from which `nettestd` reads, and vice versa. The intent of this option is to allow `nettest` to be run across arbitrary devices that have a character-device interface that can be accessed just by opening up a special character file for reading or writing. It is not intended for reading or writing to a regular file.

`-s`*n* Increases the maximum TCP window by a factor of 2^*n*; $1 \leq n \leq 14$.

`-m` Indicates that for datagram connections (`-p udp` and `-p unixd`), `nettest` should use the `sendmsg` system call instead of the `sendto` system call (see `send`(2)), and that `nettestd` should use the `recvmsg` system call instead of the `recvfrom` system call (see `recv`(2)). For other protocols, this option is ignored.

`-w` Specifies that the `MSG_WAITALL` flag must be used when `recv`(2) is called. This allows the kernel to accumulate incoming data so that the read buffer is filled before it returns control to the application. You do not need the `-f` option when you use this option.

`-V` Prints information about the version of the program.

host Specifies the name of the machine on which the server is running for TCP and UDP connections. If *host* is omitted or specified as `-`, the name that `gethostname`(2) returns is used.

count Specifies the number of read or write operations. A value of `-` indicates that the default value must be used. The default value is 100.

size Specifies the number of bytes to be read or written. A value of `-` indicates that the default value must be used. The default value is 4096.

port Specifies an alternative port number for TCP and UDP connections. *port* must be a decimal number.

readfile Specifies the name of the special file that is read for `file` protocols.

writefile Specifies the name of the special file to which information is written for `file` protocols.

The output from `nettest` is timing information and a histogram of the various sizes that the read operations returned. System load affects the results because all throughput times are calculated from wall-clock times. The percentages listed for system and user times are percentages of wall-clock time.

The write time is measured from the time at which the application starts its first write until the time it completes its last write. The read time begins when the last write is complete and ends when the last read is complete. Because the kernel may buffer outgoing data, if everything on the network is working correctly, it is typical for the write times to be slightly faster than the read times. This difference in throughput represents the amount of buffering in the kernel and the network round-trip time. The read and write time is measured from the time the first write is started to the time the last read is completed; thus, if the speed of the network is the same in both directions and both machines have the same processing power and load, the read and write times are the most accurate.

The histogram output shows the sizes that the read system calls return. These may not have any correlation to the size and number of packets that are actually sent and received over the network. This is especially true for TCP connections.

BUGS

The `-p pipe` option creates named pipes; the `-p unix` and `-p unixd` options create UNIX domain sockets. The named pipes and UNIX domain sockets remain after the programs exit.

If `-p pipe` *filename* is specified and *filename* is either a relative or absolute path name, neither `nettest` nor `nettestd` insert the `W` and `R` before the final component of the path name; they are always prepended to the entire file name.

FILES

`/etc/iptos`	IP (TOS) database
`nt_pipeW`, `nt_pipeR`	Default names for named pipes
`nt_socket` and `nt_dsocket`	Default name for stream and datagram UNIX domain sockets

SEE ALSO

`gethostname`(2), `recv`(2), `send`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

NAME

netvar – Displays and alters network configuration variables

SYNOPSIS

/etc/netvar [-h] [-i] [-o] [-*flagvalue*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The netvar command displays and modifies parameters of the TCP/IP network software. These parameters affect the operation of the network. This command is invoked at system startup by /etc/tcpstart.

Changes that are made take effect immediately, except for sockets that are currently open. Sockets that are opened before the execution of netvar are not affected by the change. Daemons that inetd(8) starts are also unaffected, unless inetd(8) is killed and restarted. Using netvar with no options displays the current values of these parameters. The -i option lets you modify variables in interactive mode. The -*flagvalue* option lets you modify variables on the command line.

Only the super user can change the network configuration variables.

The netvar command accepts the following options:

-h Displays help information. With the -h option, netvar displays the command-line flags used to set configuration variables on the command line.

-i Changes variables in interactive mode. netvar prompts with the current value of each variable; press <RETURN> to accept that value, or enter a new value to change the variable.

-o Displays the values of the variables. This is useful for checking the values that are set when using command-line or interactive mode to change the variables. Usually, a change is made without any further comment.

-*flagvalue*
In addition to interactive configuration mode, netvar has a simple command-line mode of operation for changing configuration variables. The flag and the new value for the variable to be changed are provided on the command line; netvar makes the changes without further prompting. You can check the valid flags by using the -h option.

The valid flags and their meanings are as follows:

-a Turns on or off the administrator maximum transmission unit (mtu) override. The value specified can be on, off, yes, no, 1 (on/yes), or 0 (off/no). The administrator mtu is set by using the route(8) command; if administrator override is on, the administrator mtu overrides a dynamically discovered mtu.

`-f` Turns on or off the IP forwarding. The value specified can be `on`, `off`, `yes`, `no`, `1` (on/yes), or `0` (off/no).

`-k` Changes the Transmission Control Protocol (TCP) `keepalive` delay. The value specified must be an integer.

`-l` Changes the default IP time-to-live for TCP sockets. The value specified must be an integer.

`-L` Changes the default IP time-to-live for user datagram protocol (UDP) sockets. The value specified must be an integer.

`-m` Turns on or off dynamic network mtu discovery. The value specified may be `on`, `off`, `yes`, `no`, `1` (on/yes), or `0` (off/no).

`-n` Limits the number of mbufs allocated as socket structures. There is only one mbuf allocated per socket structure. Once the limit is reached, only processes owned by user `root` are able to create new sockets. The value specified must be an integer.

`-p` Changes the default minimum interval between operator messages of the same type. The value specified must be an integer.

`-r` Turns Internet protocol (IP) sending of Internet control message protocol (ICMP) redirects on or off. The value specified may be `on`, `off`, `yes`, `no`, `1` (on/yes), or `0` (off/no).

`-s` If `on`, `yes`, or `1`, treats subnets as local; if `off`, `no`, or `0`, treats subnets as not local.

`-S` Changes the system-wide maximum send and receive space for socket buffering. This flag indicates the maximum number of bytes that can be set on the `SO_SNDBUF` and `SO_RCVBUF` socket options. It is the maximum high-water mark for the socket and is enforced on all sockets. The value specified must be an integer.

`-t` Changes the default TCP send space for socket output buffering. This flag changes the default high-water mark for individual TCP `send` windows when `SO_SNDBUF` for TCP socket sessions is not used. The value specified must be an integer; units are in bytes.

`-T` Changes the default TCP receive space for socket input buffering. This flag changes the default high-water mark for individual TCP `receive` windows when `SO_RCVBUF` for TCP socket sessions is not used. The value specified must be an integer; units are in bytes.

`-u` Changes the default UDP send space for socket output buffering. The value specified must be an integer.

`-U` Change the default UDP receive space for socket input buffering. The value specified must be an integer.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

For changes made by netvar to affect connections established to daemons that inetd(8) has started, inetd(8) must be killed and restarted. This includes connections for users who use telnet(1B) or ftp(1B) to access the Cray Research system.

EXAMPLES

The following example, using the netvar command-line mode, turns off IP forwarding and sets the TCP send and receive space to 64 Kbytes:

```
netvar -f off -t 65536 -T 65536
```

The following example changes the TCP send and receive socket buffers so that the default is 128 Kbytes (131,072 bytes), and it then displays the values of all netvar variables.

```
netvar -t 131072 -T 131072 -o
```

SEE ALSO

inetd(8), route(8)

ftp(1B), privtext(1), telnet(1B) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

newkey – Creates a new key in the publickey database

SYNOPSIS

/etc/newkey [-h *hostname*] [-u *user*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The newkey command usually is run by the network administrator on the network information service (NIS) master machine to create public/private key pairs for users and super users on the network. These keys are needed for using secure Remote Procedure Call (RPC).

The newkey command prompts for the login password of the specified user and then creates a new public/secret key pair in /etc/publickey, encrypted with the login password of the specified user.

The newkey command accepts the following options:

-h *hostname* Creates a new public key for the super user at the specified host. Prompts for the root password of the specified host.

-u *user* Creates a new public key for the specified user. Prompts for the NIS password of the specified user.

SEE ALSO

keyserv(8)

keylogin(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

newmsglog – Saves the latest versions of the message log file

SYNOPSIS

/etc/newmsglog

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The newmsglog shell script saves the latest versions of the message log file (/usr/spool/msg/msglog.log), as files msglog.log.0, msglog.log.1, and so on, with msglog.log.0 being the most recent. newmsglog also signals the message daemon to reopen the log file. It should be run with the crontab(1) command.

FILES

/usr/spool/msg/msglog.log Message log file

SEE ALSO

crontab(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

NAME

newsys – Starts syslogd(8) and renames its log files

SYNOPSIS

/etc/newsys [-s]

/etc/newsys [*scriptname*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The newsys shell script has two modes of operation: in one mode, it starts the syslogd(8) daemon, in the other mode, it renames the log files created by syslogd(8) and executes an optional shell script. This operation is necessary because the syslogd(8) daemon appends the messages it receives onto log files and never truncates those files, even when restarted. If the daemon were restarted without renaming the files, they would eventually grow to fill the file systems on which they reside.

The newsys command accepts the following option:

-s Restarts the syslogd(8) daemon. In this mode, the newsys script starts the syslogd(8) daemon but does not rename the log files.

When executed without the -s option, the newsys command renames the log files created by syslogd(8). In this mode, newsys accepts the following argument:

scriptname Specifies a script for newsys to execute after renaming the log files. This argument is ignored if the -s option is used.

When renaming files, newsys saves the last 20 copies of the daylog and debug log files, renaming them by appending a unique number to the original name of the log file (for example, daylog.1, daylog.2, daylog.3, and so on). The newsys command also saves the last 30 days of the kern and auth log files, renaming them by prepending the month and day plus a unique number to the file's original name. For example, a copy of the file kern being moved on March 21 would become 03-21.0.kern. If you executed newsys once more on March 21, the next name would be 03-21.1.kern.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

If the PRIV_SU configuration option is enabled, the user must be the super user to use this command.

SEE ALSO

`syslogd`(8)

`logger`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`airlog`(3C), `syslog`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`log`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

nfsaddhost – Adds host addresses to ID mapping domains

SYNOPSIS

```
nfsaddhost [-d map_name -c [-s] [-A] [-C]] -l addr [-u addr] [-m mask]
nfsaddhost [-d map_name -c [-s] [-A] [-C]] -u addr [-l addr] [-m mask]
nfsaddhost [-d map_name -c [-s] [-M] [-C]] -l addr [-u addr] [-m mask]
nfsaddhost [-d map_name -c [-s] [-M] [-C]] -u addr [-l addr] [-m mask]
nfsaddhost [-d map_name -s [-c] [-A] [-C]] -l addr [-u addr] [-m mask]
nfsaddhost [-d map_name -s [-c] [-A] [-C]] -u addr [-l addr] [-m mask]
nfsaddhost [-d map_name -s [-c] [-M] [-C]] -l addr [-u addr] [-m mask]
nfsaddhost [-d map_name -s [-c] [-M] [-C]] -u addr [-l addr] [-m mask]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nfsaddhost command initializes an ID mapping domain. It accepts the name of a previously defined ID map and a network address specification. You also may specify a range of network addresses. The range of addresses may be further refined by the specification of a mask to use against the network addresses. You must specify at least one map name and one address on the command line.

For client-side mapping, any outgoing request to an address in the ID mapping domains uses the ID maps associated with that address to map the user and group IDs attached to that request.

For server-side mapping, any incoming request from an address in the ID mapping domains uses the ID maps associated with that address to map the user and group IDs attached to that request.

If ID mapping is being used, network file system (NFS) accepts only requests from network addresses that are contained by NFS ID mapping domains that have been to the kernel through the use of the nfsaddhost command. This can be used to control NFS access to and from the local system. Any NFS request with a network address not defined in the ID mapping domains fails with an authentication error.

IDs may pass through the ID mapping code without modification. This is called *mapping through* or *a MAPTHRU domain*. If you do not specify a map name by using the -d option, IDs are mapped through on both client and server sides.

You must specify the -l or -u option. If you specify the -d option, either the -c or -s option also must be specified. nfsaddhost accepts the following options:

-d *map_name*

Associates this ID mapping domain with the specified ID maps. The specified map should have been previously added to the system through use of the nfsaddmap(8) command. If you omit this option, this ID mapping domain defaults to MAPTHRU for both the client and server sides.

`-c` Specifies that the user and group ID maps specified with the `-d` option will be used on the client side of NFS. If you omit this option, this ID mapping domain defaults to `MAPTHRU` for the client side.

`-s` Specifies that the user and group ID maps specified with the `-d` option will be used on the server side of NFS. If you omit this option, this ID mapping domain defaults to `MAPTHRU` for the server side.

`-A` Indicates that if an ID is not found in a map, it is mapped to "nobody" (–2). By default, it is mapped to "baduid" (–1).

`-C` Indicates that the remote machine or machines are Cray Research systems capable of NFS ID mapping. This information is used internally to handle NFS access checking and the mapping of file attributes more correctly.

`-M` Indicates that if an ID is not found in a map, it is mapped through. The default is to map the ID to "baduid" (–1).

`-l` *addr* Specifies a lower-bound network address that is associated with the specified map names. If you omit this option, it defaults to the upper-bound address specified with the `-u` option, which must be present if you omit the `-l` option.

`-u` *addr* Specifies an upper-bound network address that is associated with the specified map names. If you omit this option, it defaults to the lower-bound address specified with the `-l` option, which must be present if you omit the `-u` option.

`-m` *mask* Specifies a byte-ordered ASCII representation of a bit mask that is used in conjunction with the range of addresses specified on the command line to distinguish further the valid network addresses for this ID mapping domain. The default mask is derived from the class of Internet address that is being used by either the `-l` or `-u` option previously described. For a description of the classes of Internet addresses, see `inet`(3C).

A map name is an arbitrary ASCII name, significant to 8 characters, that is used to define a user or group map to the system through the `nfsaddmap`(8) command.

A network address is a host name or host alias from the `/etc/hosts` file (`hosts`(5)), a network name or network alias from the `/etc/networks` file (`networks`(5)), or a byte-ordered ASCII representation of a network address as described in `inet`(3C).

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

Example 1: The following command line specifies server-side ID mapping using previously defined group and user ID maps named Cray-Z for the network address associated with the host name Cray-Z-inet. Client-side requests to that network address are defined to be mapped through; that is, no ID mapping occurs on outgoing requests from the local system.

```
nfsaddhost -d Cray-Z -s -l Cray-Z-inet
```

Example 2: The following command line specifies mapping through (leaving unchanged) all IDs for both client and server side operations to and from all even numbered network addresses from the range aaa-inet to zzz-inet:

```
nfsaddhost -l aaa-inet -u zzz-inet -m 0xfffffffe
```

FILES

/etc/uidmaps/Set.domains	An administrative shell script that initializes the ID maps and ID mapping domains for the system

SEE ALSO

nfsaddmap(8), nfsadduser(8), nfsckhash(8), nfsclear(8), nfsgid(8), nfsidmap(8), nfsidmem(8), nfslist(8), nfsmerge(8), nfsrmhost(8), nfsrmmap(8), nfsrmuser(8), nfsuid(8)

privtext(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

inet(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

hosts(5), intro(4), networks(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

nfsaddmap – Defines a user ID and/or group ID map for use with NFS

SYNOPSIS

```
nfsaddmap [-v] [-r] [-u map_file] -g map_file map_name
nfsaddmap [-v] [-r] -M map_file
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nfsaddmap command reads a user ID map file and/or a group ID map file previously created by using the nfsmerge(8) command, and it defines in the kernel a user ID map and a group ID map, respectively. The kernel user ID map and/or the kernel group ID map that is defined is assigned the name *map_name*. The map name is significant to 8 characters; that is, the first 8 characters must be unique among the names of the currently defined kernel maps. Generally, the maps are named for the remote administrative domain for which the map will be used.

The nfsaddmap command always creates a kernel user ID map and a group ID map. If the user map file is not specified on the command line, an empty kernel user map is created. The nfsadduser(8) command can then be used to add user entries to the user map.

The nfsaddmap command accepts the following options:

`-v` Prints information on command-line options and on each user entry added to the user map.

`-r` Replaces the kernel ID map if it already exists; otherwise, it creates the kernel ID map(s).

`-u` *map_file* Reads the user ID map in *map_file* and defines the kernel user ID map with the information contained in *map_file*.

`-g` *map_file* Reads the group ID map in *map_file* and defines the kernel group ID map with the information contained in *map_file*. You must specify this option.

`-M` *map_file* Reads the user ID map in *map_file* and defines the special mapthru kernel map. This special mapthru map uses the reserved map name MAP_THRU. When defined, all mapthru ID mapping domains use this map. This special map is required for Kerberos validation. It is also required with Internet Protocol Security Options (IPSO) protocol for security label and compartment information. It can be used to obtain disk accounting information. See the *UNICOS Networking Facilities Administrator's Guide*, Cray Research publication SG–2304, for more information on this special mapthru map.

map_name Specifies the kernel ID maps that are being defined.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

The following command line defines both a group ID map and a user ID map for the kernel. It expects the user ID map to be contained within the file u.local.remote and the group ID map to be contained within the file g.local.remote. It is expected that the nfsaddhost(8) command will use these maps subsequently to describe an ID mapping domain between the local system and the remote system.

```
nfsaddmap -u u.local.remote -g g.local.remote remote
```

FILES

/etc/uidmaps/nfsaddmap	Defines a user or group ID map in the kernel

SEE ALSO

nfsaddhost(8), nfsadduser(8), nfsckhash(8), nfsclear(8), nfsidmap(8), nfsidmem(8), nfslist(8), nfsmerge(8), nfsrmhost(8), nfsrmmap(8), nfsrmuser(8), nfsuid(8), udbgen(8)

privtext(1), setucat(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

nfsadduser – Adds entry from user ID map file to kernel user ID map

SYNOPSIS

```
nfsadduser [-v] [-r] user_mapfile map_name user1 [user2 user3 ...]
nfsadduser -a [-v] [-r] user_mapfile map_name
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nfsadduser command reads a user ID map file (previously created by using the nfsmerge(8) command), finds the entries for the specified users, and adds them to the kernel user ID map named *map_name*. The kernel user ID map must have been previously created by using the nfsaddmap(8) command.

The nfsadduser command accepts the following options and operands:

-v Prints the contents of the user ID map entry before adding it to the kernel (verbose option).

-r Replaces the entries for the users specified on the command line in the kernel ID map if they already exist; otherwise, adds them to the kernel ID map.

-a Adds all entries from the user map file to the specified kernel map.

user_mapfile
Specifies user ID map file to search for the user map entries.

map_name
Specifies kernel map to which the entries will be added.

user1 [*user2 user3* ...]
Identifies users whose entries from the map file will be added to the specified kernel map. If the -a option is used, a list of users is ignored.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

Example 1: The following command searches the user map file `u.local.remote` for entries that belong to users on the local Cray Research system whose login names are `local_user1` and `local_user2`. Each time it finds an entry, it adds the entry to the kernel map named `remote`.

```
nfsadduser u.local.remote remote local_user1 local_user2
```

Example 2: The following command adds all user entries from the map file `u.local.remote` to the kernel map named `remote`.

```
nfsadduser -a u.local.remote remote
```

FILES

`/etc/uidmaps/nfsadduser` File that adds users to the kernel user ID map

SEE ALSO

`nfsaddhost`(8), `nfsaddmap`(8), `nfsckhash`(8), `nfsclear`(8), `nfsgid`(8), `nfsidmap`(8), `nfsidmem`(8), `nfslist`(8), `nfsmerge`(8), `nfsrmhost`(8), `nfsrmmap`(8), `nfsrmuser`(8), `nfsuid`(8), `udbgen`(8)

`privtext`(1), `setucat`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

nfsckhash – Checks consistency of NFS ID mapping hash tables in a kernel or a kernel dump

SYNOPSIS

/etc/uidmaps/nfsckhash [-g] [-u] [-v] [-h] [-d *dump*] [-s *system*] [*list_of_map_names*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nfsckhash command is primarily an administrator diagnostic or debugging tool. It examines each network file system (NFS) ID map and its hash table to ensure that each entry is hashed properly for both local and remote IDs. It also checks to ensure that the information in the map header is consistent with the actual map entries.

If any map names are listed on the command line, this command checks only the specified maps. If no map names are specified, this command checks all maps defined in the kernel.

The nfsckhash command accepts the following options:

-g Checks the consistency of the kernel group ID maps.

-u Checks the consistency of the kernel user ID maps.

-v (Verbose) Prints information about each entry in a kernel ID map it finds.

-h Prints only kernel ID map header information. This option effectively negates the -v option, because no entries are examined.

-d *dump* Reads the kernel dump file *dump* instead of /dev/kmem.

-s *system* Uses the UNICOS kernel binary file *system* instead of /unicos to obtain symbol information.

list_of_map_names
If any map names are listed on the command line, nfsckhash checks only the specified maps. If no map names are specified, this command checks all maps defined in the kernel.

If neither the -g nor the -u option is specified, the consistency of both the user ID maps and group ID maps in the kernel are checked.

To obtain a list of all currently defined kernel ID maps and to check their consistency, use the following command:

```
nfsckhash -v
```

This command also effectively reports the names of all group ID maps because for every user ID map there is a group ID map of the same name, and for every group ID map there is a user ID map of the same name.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

SEE ALSO

`nfsaddhost`(8), `nfsaddmap`(8), `nfsadduser`(8), `nfsclear`(8), `nfsidmap`(8), `nfsidmem`(8), `nfslist`(8), `nfsmerge`(8), `nfsrmhost`(8), `nfsrmmap`(8), `nfsrmuser`(8), `nfssid`(8), `nfsuid`(8)

`privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

nfsclear – Removes user and group mapping tables and ID mapping domains from the kernel

SYNOPSIS

nfsclear [-F]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nfsclear command removes from the kernel all user and group map tables that had been defined by using the nfsaddmap(8) command, as well as all ID mapping domains that had been defined by using the nfsaddhost(8) command. It releases any memory space allocated to these tables and domains. This command is typically the first of a set of commands used to initialize (or reinitialize) the user and group mapping tables and ID mapping domains.

This command accepts the following option:

-F Forces clearing of all kernel ID maps even if Kerberos-validated addresses are associated with any of the user ID maps. This option wipes out all Kerberos-validated addresses for all users that had them. This process requires remote users to revalidate with this network file system (NFS) server after ID maps have been readded to the kernel.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

FILES

/etc/uidmaps/nfsclear	Removes mapping tables and ID mapping domains from the kernel

SEE ALSO

`nfsaddhost`(8), `nfsaddmap`(8), `nfsadduser`(8), `nfsckhash`(8), `nfsgid`(8), `nfsidmap`(8) `nfsidmem`(8), `nfslist`(8), `nfsmerge`(8), `nfsrmmap`(8), `nfsrmhost`(8), `nfsrmuser`(8), `nfsuid`(8), `udbgen`(8)

`privtext`(1), `setucat`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

nfsd, biod, cnfsd, pcnfsd – Starts NFS daemons and services requests

SYNOPSIS

/etc/nfsd [*nservers*]
/etc/biod [*ndaemons*]
/etc/cnfsd [*nservers*]
/etc/pcnfsd [-m *ID mapname* -u *user ID map file*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nfsd command starts the network file system (NFS) server daemons that handle client file system requests. The *nservers* operand is the number of file system request daemons to start. This number should be based on the load expected on this server.

The biod command starts NFS client asynchronous block I/O daemons. The *ndaemons* operand is the number of daemons to start.

The cnfsd command starts NFS server daemons that use a modified NFS protocol that reduces system overhead. The *nservers* operand is the number of daemons to start. This modified protocol is valid only between cooperating UNICOS systems. See the mountd(8) for more information.

The pcnfsd command runs continuously on a server system, servicing PC-NFS requests for user authentication and print spooling. pcnfsd accepts the following options:

nservers	Number of file system request daemons to start for the nfsd and cnfsd commands.
ndaemons	Number of daemons to start for the biod command.
-m *ID map name*	*ID map name* is the name of the kernel map into which the entry from the map file will be placed.
-u *user ID map file*	Used only in conjunction with NFS ID mapping. *user ID map file* is the file from which user map entries are obtained for anyone logging in to pcnfsd.

If using the -u and -m options, the ID map specified by *ID map name* must be present in the kernel prior to a reference to pcnfsd by a PC user. The *user ID map file* must be present prior to starting pcnfsd. See the *UNICOS Networking Facilities Administrator's Guide*, Cray Research publication SG–2304, for more details.

The nfsd, cnfsd, and pcnfsd daemons catch the SIGHUP signal and reregister themselves with portmap(8) when they receive the signal. This enables these commands to continue running properly when portmap(8) must be restarted.

NOTES

If your system is licensed for ONC+™, `nfsd`() will also register for version 3 of the `nfs/mount` protocol. This protocol is necessary for mounting NFS version 3 file systems.

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

SEE ALSO

`mountd`(8), `portmap`(8)

`privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`exports`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

nfsgid – Prints a group ID map entry

SYNOPSIS

```
nfsgid -g map_file group [group ...]
nfsgid -m map_name [-d dump] [-s system] group [group ...]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nfsgid command prints in a readable format the specified group entries from either the specified group ID map file, or the specified group ID map in the kernel or kernel dump. This command is useful mainly for diagnostic purposes.

The nfsgid command accepts the following options:

-g *map_file*	Specifies the name of the group ID map file in which to find the entries to print.
-m *map_name*	Specifies the name of the kernel group ID map in which to find the entries to print.
-d *dump*	Reads the kernel dump file *dump* instead of /dev/kmem.
-s *system*	Uses the UNICOS kernel binary file *system* instead of /unicos to obtain symbol information.
group [*group* ...]	Specifies names of groups for which to find and print entries.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

The following command finds the group ID map entries from the kernel group ID map named remote for the local login names local_group1 and local_group2, and it prints the contents of each entry if it is in the map.

```
nfsgid -m remote local_group1 local_group2
```

FILES

`/etc/gidmaps/nfsgid`	Prints a group ID map entry

SEE ALSO

`nfsaddhost`(8), `nfsaddmap`(8), `nfsadduser`(8), `nfsckhash`(8), `nfsclear`(8), `nfsidmap`(8), `nfsidmem`(8), `nfslist`(8), `nfsmerge`(8), `nfsrmhost`(8), `nfsrmmap`(8), `nfsrmuser`(8), `nfsuid`(8)

`privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

nfsidmap – Reports or changes the state of NFS ID mapping in the kernel

SYNOPSIS

```
/etc/uidmaps/nfsidmap [-d]
/etc/uidmaps/nfsidmap [-e]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

If no options are specified, the nfsidmap command reports the state of network file system (NFS) ID mapping (enabled or disabled). Any user can use this command to determine the current state of NFS ID mapping.

Only an appropriately authorized user may change the state of NFS ID mapping. The nfsidmap command accepts following options:

-d Disables NFS ID mapping in the kernel. If mapping is already disabled, the disabled state is reported.

-e Enables NFS ID mapping in the kernel. If mapping is already enabled, the enabled state is reported.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Allowed to change the state of NFS ID mapping.

If the PRIV_SU configuration option is enabled, the super user is allowed to change the state of NFS ID mapping.

SEE ALSO

nfsaddhost(8), nfsaddmap(8), nfsadduser(8), nfsckhash(8), nfsclear(8), nfsgid(8), nfsidmem(8), nfslist(8), nfsmerge(8), nfsrmhost(8), nfsrmmap(8), nfsrmuser(8), nfsuid(8)

privtext(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

nfsidmem – Reports the amount of kernel memory that NFS ID mapping is using

SYNOPSIS

```
/etc/uidmaps/nfsidmem [-g] [-i] [-u] [-d dump] [-s system]
/etc/uidmaps/nfsidmem [-v] [-d dump] [-s system]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nfsidmem command reports the number of network file system (NFS) ID mapping domains, the number of user maps, and the number of group maps that have been inserted in the kernel. It also reports the number of bytes of kernel memory that have been allocated and the number of bytes of kernel memory that NFS ID mapping is actually using.

The nfsidmem command also reports any ID maps in the kernel that are not referenced by an ID mapping domain. Unreferenced maps generally indicate an administrative error: either the unreferenced maps should not have been added to the kernel, or an ID mapping domain that references these maps should be added.

The nfsidmem command accepts the following options:

-g	Gives detailed information on the kernel group ID maps. In addition to the totals described previously, the size of each group ID map is reported.
-i	Gives detailed information about memory usage of NFS ID mapping domains.
-u	Gives detailed information on the kernel user ID maps. In addition to the totals described previously, the size of each user ID map is reported.
-d *dump*	Reads the kernel dump file *dump* instead of /dev/kmem.
-s *system*	Uses the UNICOS kernel binary file *system* instead of /unicos to obtain symbol information.
-v	Gives more detailed information on each kernel ID map and the memory used by the NFS ID mapping domains. In addition to the totals described previously, the size of each ID map is reported. Using this option is the same as using the -g, -i, and -u options.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

SEE ALSO

nfsaddhost(8), nfsaddmap(8), nfsadduser(8), nfsckhash(8), nfsclear(8), nfsidmap(8), nfslist(8), nfsmerge(8), nfsrmhost(8), nfsrmmap(8), nfsrmuser(8), nfsgid(8), nfsuid(8)

privtext(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

nfslist – Lists all kernel ID mapping domains

SYNOPSIS

nfslist [-f] [-b] [-r] [-u] [-z] [-d *dump*] [-s *system*] [*map_names*]
nfslist [-f] [-b] [-r] [-d *dump*] [-s *system*] [-a *address*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nfslist command lists all currently defined network file system (NFS) ID mapping domains in the kernel. For each ID mapping domain, nfslist prints the name of the map, mapping flags, the address range, and the address mask. It then prints out all addresses within the address range covered by the mask. For each matching address, the following information is printed: a byte-ordered decimal address, followed by a byte-ordered hexadecimal address, followed by the name from the /etc/hosts or /etc/networks file associated with that address. (See the hosts(5) and networks(5) man pages.) Map names are generally named for the remote administrative domain associated with the remote address.

The nfslist command accepts the following options:

-f — Lists all ID mapping domain information, even if NFS ID mapping is currently disabled. If ID mapping is turned off, the listing of any domains defined in the kernel is suppressed unless the -f option was specified.

-b — Lists the short form of the ID mapping domain information. The list of names for each address in the address range is not displayed.

-r — Displays the depth in the radix tree, the reference count, and the usage count of each ID mapping domain structure. This option is used primarily for problem diagnosis.

-u — Displays only ID mapping domains with nonzero usage counts.

-z — Displays only ID mapping domains with zero usage counts.

-d *dump* — Reads the kernel dump file *dump* instead of /dev/kmem.

-s *system* — Uses the UNICOS kernel binary file *system* instead of /unicos to obtain symbol information.

map_names — Specifies names of maps to be listed.

-a *address* — For the specified Internet address, lists the ID mapping domains that contain the address. The address can be a host name, a network name, or a byte-ordered Internet address. This option is mainly used for diagnosing ID mapping set-up problems.

The map flags are as follows:

BAD_ID, NOBODY, or MAP_THRU
: Indicates how a user ID not in the user map will be treated. It can be mapped to a bad ID (-1), mapped to nobody (-2), or mapped through without change.

CLIENT
: Indicates that the specified map is used to map IDs on the client side of the NFS protocol.

CRAY
: Indicates that all of the machines specified by the address range can map IDs in the same manner as that of the Cray Research system. Currently, they must be Cray Research machines. This is useful for correct access checking when mapping IDs.

SERVER
: Indicates that the specified map is used to map IDs on the server side of the NFS protocol.

If one or more map names are specified on the command line, only the ID mapping domains associated with those maps are printed. The special name MAP_THRU is used to list domains for which IDs are the same as the current host.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

The following example shows sample output from nfslist. There are three ID mapping domains listed.

The first output references the MAP_THRU map using the loopback address and maps IDs for both the NFS client and NFS server. IDs not found in the user map are automatically mapped to BAD_ID (-1).

The second output references a map named absurd using an internet host address for a host known as absurd. IDs are only mapped on the NFS client and the host known as absurd appears to be another Cray machine using Cray style ID mapping. IDs not found in the user map are automatically mapped to NOBODY (-2).

The third output references a map named peanut using an internet host address for a host known as peanut. IDs are only mapped on the NFS server and the host known as peanut appears to be another Cray machine using Cray style ID mapping. IDs not found in the user map are automatically mapped to BAD_ID (-1).

```
NFS ID Mapping is : ENABLED

NFS ID    NFS ID
Map       Mapping                                    Address
Name      Flags                          Address     Mask

MAP_THRU  CLIENT  SERVER  BAD_ID         7f.00.00.00 [ff.00.00.00]
```

```
                          Addr(dec)          Addr(hex)    Host Name
                          127.0.0.0          7f.00.00.00  loopback

     absurd    CLIENT           NOBODY  CRAY  ff.aa.07.01  [ff.ff.ff.ff]
                          Addr(dec)          Addr(hex)    Host Name
                          255.170.7.1        ff.aa.07.01  absurd

     peanut             SERVER  BAD_ID  CRAY  ee.bb.70.01  [ff.ff.ff.ff]
                          Addr(dec)          Addr(hex)    Host Name
                          238.187.112.1      ee.bb.70.01  peanut
```

FILES

`/etc/uidmaps/nfslist`	Lists all kernel mapping domains

SEE ALSO

`nfsaddhost`(8), `nfsaddmap`(8), `nfsadduser`(8), `nfsckhash`(8), `nfsclear`(8), `nfsgid`(8), `nfsidmap`(8), `nfsidmem`(8), `nfsmerge`(8), `nfsrmhost`(8), `nfsrmmap`(8), `nfsrmuser`(8), `nfsuid`(8)

`privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`hosts`(5), `intro`(4), `networks`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

nfsmerge – Creates a user or group mapping between the local Cray Research administrative environment and a remote administrative environment

SYNOPSIS

```
nfsmerge [-l pef] [-e gef] [-u umf] [-g gmf] lpswd rpswd lgrp rgrp
nfsmerge [-L pef] [-e gef] [-u umf] [-g gmf] lpswd rpswd lgrp rgrp
nfsmerge [-L pef] [-E gef] [-u umf] [-g gmf] lpswd rpswd lgrp rgrp
nfsmerge [-l pef] [-E gef] [-u umf] [-g gmf] lpswd rpswd lgrp rgrp
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nfsmerge command creates both a user ID mapping file and a group ID mapping file for use between the local Cray Research systems and remote administrative environments. network file system (NFS) ID mapping is necessary when users and/or groups do not have the same numerical IDs in two different administrative environments. The ID mapping files contain binary information suitable for use by the NFS routines in the UNICOS kernel.

To produce ID mapping files, the nfsmerge command uses the name fields from files in passwd(5) and group(5) formats, assuming that a common name in the passwd files of two administrations refers to the same user. Similarly, nfsmerge assumes that a common name in the group files of the two administrations refers to the same group. IDs that correspond to names that match exactly, or are specified as equivalent through the use of an exceptions file, are put into the map file. Exceptions files are explained in the following paragraph.

Some users and groups do not necessarily have common names between the local and remote administrations. For example, a user may not be able to have the same login name for the two administrations; someone else may already be using the name on one of the systems. Another example is that of two different users, each with an account in only one of the two administrations, and having the same login name. (The EXAMPLES section provides more explicit examples of these situations.) You can resolve these situations with the use of an exceptions file.

An *exceptions file* contains exceptions between names in the local and remote passwd or group files. For each entry in a login name exceptions file, there is a login name for a user in the local administration and an optional equivalent login name for the same user in the remote administration, separated by white space on a single line. If the name for the user in the remote administration is not present, the remote name is assumed to be the same as the one in the local administration. This is useful if you are forcing the use of exceptions by using either the -L or the -E option. Forcing the use of the exceptions file means that IDs for the names that appear in the exceptions file are the only ones put into the ID mapping file. A group name exceptions file has the same format and produces similar results for the group ID mapping.

The nfsmerge command produces ID mapping files in formats suitable for use by the nfsaddmap(8) command. The -u option allows you to specify a name for the user ID map file that is produced. The -g option allows you to specify a name for the group ID map file that is produced. If the names of the ID mapping files are not specified, nfsmerge generates a unique name, using the mktemp(3C) library routine. A template of usrmap.*XXXXXX* is used for the user ID map file. A template of grpmap.*XXXXXX* is used for the group ID map file. It is strongly recommended, however, that you specify on the command line a meaningful name for the ID mapping files.

The names of the local and remote password files, the names of the local and remote group files, the name of the exceptions file(s) (if being used), a list of user names for which IDs have been mapped, and a list of group names for which group IDs have been mapped are all written to standard output. Name exceptions are noted in the standard output by separating the local name from the remote name with an arrow symbol (==>).

See the *UNICOS Networking Facilities Administrator's Guide*, Cray Research publication SG–2304, for more detailed information on configuring and administrating NFS with ID mapping.

The nfsmerge command accepts the following options and operands (if no options are specified, unique names for the user and group map files are chosen by the mktemp(3C) library routine):

-l *pef* Specifies the login name exceptions file.

-L *pef* Maps only the login names in the login name exceptions file. This option forces the use of the login name exceptions file.

-e *gef* Specifies the group name exceptions file.

-E *gef* Maps only the group names in the group name exceptions file. This option forces the use of the group name exceptions file.

-u *umf* Specifies the name of the user ID map file.

-g *gmf* Specifies the name of the group map file.

The following arguments must appear on the command line in the order specified:

lpswd Specifies the name of a passwd(5) format file that contains user information for the local Cray Research system.

rpswd Specifies the name of a passwd(5) format file that contains user information for the remote administrative environment.

lgrp Specifies the name of a group(5) format file that contains group information for the local Cray Research system.

rgrp Specifies the name of a group(5) format file that contains group information for the remote administrative environment.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

Example 1: The following command line creates a user ID map file named u.local.remote and a group ID map file named g.local.remote from the specified password and group files (the password and group files need not be sorted). Any name from the local password file, passwd.local, that appears in the exceptions file, l.local.remote, also is mapped if the remote name from the exceptions file is found in the remote password file, passwd.remote. All output created by this command is redirected into the LOG.local.remote file; that is, LOG.local.remote functions as a log file.

```
nfsmerge -l l.local.remote -u u.local.remote -e e.local.remote \
       -g g.local.remote passwd.local passwd.remote \
          group.local group.remote > LOG.local.remote
```

Example 2: Shows the contents of l.local.remote (the login name exceptions file in the previous command line):

```
jdoe        johnd
lvp         linus
lucy        lvp
bin         nobody
sys         nobody
dtm         dennis
fsmith      freds
```

The information in this file indicates that the IDs for the name jdoe on the local Cray Research system are mapped to the IDs for the name johnd on the remote system. This file also indicates that user lvp on the local Cray Research system is user linus on the remote system. However, user lvp on the remote system has a user name of lucy on the local Cray Research system.

SEE ALSO

`nfsaddhost`(8), `nfsaddmap`(8), `nfsadduser`(8), `nfsckhash`(8), `nfsclear`(8), `nfsidmap`(8), `nfsidmem`(8), `nfslist`(8), `nfsgid`(8), `nfsuid`(8), `nfsrmuser`(8), `nfsrmhost`(8), `nfsrmmap`(8)

`privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`mktemp`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`group`(5), `intro`(4), `passwd`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

nfsrmhost – Removes an ID mapping domain host address or a range of host addresses

SYNOPSIS

```
nfsrmhost -l addr [-u addr] [-m mask]
nfsrmhost -u addr [-l addr] [-m mask]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nfsrmhost command removes network file system (NFS) ID mapping domains from the kernel based on the range of network addresses associated with the domain. The range of network addresses should be equivalent to the addresses used when defined with the nfsaddhost(8) command. If the address specification does not exactly match a currently defined domain, an error is returned. Use the nfslist(8) command to list the ID mapping domains currently defined in the kernel.

You must specify either the -l or the -u option. nfsrmhost accepts the following options:

-l *addr* Specifies a lower-bound network address that is associated with the specified map names. If you omit this option, the address becomes, by default, the upper-bound address specified by the -u option.

-u *addr* Specifies an upper-bound network address that is associated with the specified map names. If you omit this option, the address becomes, by default, the lower-bound address specified by the -l option.

-m *mask* Specifies a byte-ordered ASCII representation of a bit mask that is used in conjunction with the range of addresses specified on the command line to specify further the valid network addresses for this ID mapping domain.

A *network address* is a host name or host alias from the /etc/hosts file (see hosts(5)), a network name or network alias from the /etc/networks file (see networks(5)), or a byte-ordered ASCII representation of a network address, as described in inet(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

The following command lines remove two separate domains that were previously defined in the kernel by using the nfsaddhost(8) command (see the examples in the nfsaddhost(8) man page to see how those domains were added to the kernel):

```
nfsrmhost -l Cray-N-inet
nfsrmhost -l aaa-inet -u zzz-inet -m 0xfffffffe
```

FILES

/etc/uidmaps/nfsrmhost	Remove NFS ID mapping domains

SEE ALSO

nfsaddhost(8), nfsaddmap(8), nfsadduser(8), nfsckhash(8), nfsclear(8), nfsgid(8), nfsidmap(8), nfsidmem(8), nfslist(8), nfsmerge(8), nfsrmmap(8), nfsrmuser(8), nfsstat(8), nfsuid(8)

privtext(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

inet(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

hosts(5), intro(4), networks(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

nfsrmmap – Removes a previously defined user and/or group ID map

SYNOPSIS

```
nfsrmmap [-F] map_name
[map_name map_name ...]
nfsrmmap [-F] -M [map_name map_name ...]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nfsrmmap command removes the specified user ID map and/or group ID map file(s), which were previously added to the kernel by using the nfsaddmap(8) command. You may remove the maps only if they are not being referenced by an ID mapping domain. If the specified map is not currently defined in the kernel, an error is returned.

This command accepts the following options and operand:

-F Forces the removal of a map even when entries are in a user map with Kerberos validated addresses. This effectively prevents those users from accessing and exporting file systems that require kerberos validated addresses.

-M Removes the special mapthru map if it is defined in the kernel. If secure UNICOS or kerberized network file system (NFS) is running, this special mapthru map may be required. This effectively prevents those users from accessing and exporting file systems that require Kerberos validated addresses. See the *UNICOS Networking Facilities Administrator's Guide*, Cray Research publication SG–2304, for detailed information on this special ID map.

map_name Specifies the name of the kernel ID map to delete. A list of these may exist on the command line.

By default, both the user and group maps that have the name *map_name* are removed from the kernel.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

The following command line deletes from the kernel the user ID maps and group ID maps named `remote` and `other`:

```
nfsrmmap remote other
```

FILES

`/etc/uidmaps/nfsrmmap`	Removes a map

SEE ALSO

`nfsaddhost`(8), `nfsaddmap`(8), `nfsadduser`(8), `nfsckhash`(8), `nfsclear`(8), `nfsgid`(8), `nfsidmap`(8), `nfsidmem`(8), `nfslist`(8), `nfsmerge`(8), `nfsrmhost`(8), `nfsrmuser`(8), `nfsstat`(8), `nfsuid`(8)

`privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

nfsrmuser – Removes a user ID map entry from a kernel ID map

SYNOPSIS

```
nfsrmuser [-F] map_name user1 [user2 user3 ...]
nfsrmuser [-F] -M user1 [user2 user3 ...]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nfsrmuser command removes a user ID map entry from a kernel ID map.

This command accepts the following options and operands:

-F Forces the removal of the specified user ID map entries even if Kerberos validated addresses are associated with any of the entries. This effectively prevents those users from accessing and exporting file systems that require Kerberos-validated addressses.

-M Removes user ID map entries from the special mapthru map if it is defined in the kernel. This effectively prevents those users from accessing and exporting file systems that require Kerberos-validated addresses. If secure UNICOS or kerberized network file system (NFS) is running, this special mapthru map may be required. See the *UNICOS Networking Facilities Administrator's Guide*, Cray Research publication SG–2304, for detailed information on this special ID map.

map_name The kernel map from which to remove the entries.

user1 [*user2 user3* ...]
The local login names of the user ID map entries will be removed from the specified kernel map.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

The following command gets the user ID (UID) for each user name specified on the command line (`local_user1` and `local_user2`), looks for the user entry in the kernel map named `map_name`, and removes the entry if it is found.

```
nfsrmuser remote local_user1 local_user2
```

FILES

`/etc/uidmaps/nfsrmuser`	Removes a user ID map entry from a kernel ID map

SEE ALSO

`nfsaddhost`(8), `nfsaddmap`(8), `nfsadduser`(8), `nfsckhash`(8), `nfsclear`(8), `nfsgid`(8), `nfsidmap`(8), `nfsidmem`(8), `nfslist`(8), `nfsmerge`(8), `nfsrmhost`(8), `nfsrmmap`(8), `nfsuid`(8), `udbgen`(8)

`privtext`(1), `setucat`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

nfsstart – Starts the network file system (NFS)

SYNOPSIS

/etc/nfsstart

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nfsstart script starts the software necessary for the network file system (NFS) at system startup when executed by the netstart(8) script.

The nfsstart script performs the following functions:

- Starts NFS daemons (see sdaemon(8)). These typically include nfsd(8), mountd(8), and automount(8).
- Initializes NFS user ID mapping by calling the /etc/uidmaps/Set.domains script if it exists.
- Calls the /etc/mountnfs script if it exists, to mount remote NFS file systems explicitly.

SEE ALSO

automount(8), mountd(8), netstart(8), nfsd(8), sdaemon(8)

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

nfsstat – Displays NFS statistics

SYNOPSIS

```
/etc/nfsstat [-csnrt]
/etc/nfsstat [-mtx]
/etc/nfsstat [-a]
/etc/nfsstat [-C]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nfsstat command displays statistical information about the UNICOS network file system (NFS) and Remote Procedure Call (RPC) interfaces to the kernel. You also can use it to reinitialize this information.

If you omit all options, nfsstat prints everything and reinitializes nothing, as if you specified the -c, -n, -r, and -s options. You must group multiple options that are specified with nfsstat after a single hyphen on the command line (as shown in the SYNOPSIS section).

The nfsstat command accepts the following options:

-c Displays client information. Prints only the client-side NFS and RPC information. To print only client NFS or client RPC information, combine this option with the -n and -r options.

-s Displays server information. Works like the -c option.

-n Displays NFS information. Prints NFS information for both the client and the server. To print only client or server NFS information, combine this option with the -c and -s options.

-r Displays RPC information. Works like the -n option.

-t Displays timing information. Combine this option with any of the previous options or with the -m and -x options.

-m Displays memory allocation statistics.

-x Displays data transfer statistics.

-a Displays all information.

-C Clears all NFS statistics. Only root can use this option.

FILES

`/dev/kmem` Kernel memory

SEE ALSO

`nfs`(4P) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

nfsuid – Prints entries from a user ID map file, or a user ID map in the kernel or kernel dump

SYNOPSIS

```
nfsuid -u map_file user [user ...]
nfsuid -m map_name [-d dump] [-s system] user [user ...]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nfsuid command prints, in a readable format, the specified user entries from either the specified user ID map file, or the specified kernel or kernel dump user ID map. This command is primarily useful for diagnostic purposes.

The nfsuid command accepts the following options and operands:

-u *map_file*	Specifies the name of the user ID map file that contains the entries to print.
-m *map_name*	Specifies the name of the kernel user ID map that contains the entries to print.
-d *dump*	Reads the kernel dump file *dump* instead of /dev/kmem.
-s *system*	Uses the UNICOS kernel binary file *system* instead of /unicos to obtain symbol information.
user [*user* ...]	Specifies names of users for whom to find and print entries.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

The following command finds the user ID map entries from the kernel user ID map named remote for the local login names local_user1 and local_user2, and it prints out the contents of each entry if it is in the map:

```
nfsuid -m remote local_user1 local_user2
```

FILES

`/etc/uidmaps/nfsuid` Prints entries from a user ID map file

SEE ALSO

`nfsaddhost`(8), `nfsaddmap`(8), `nfsadduser`(8), `nfsckhash`(8), `nfsclear`(8), `nfsgid`(8), `nfsidmap`(8), `nfsidmem`(8), `nfslist`(8), `nfsmerge`(8), `nfsrmhost`(8), `nfsrmmap`(8), `nfsrmuser`(8)

`privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

nisaddcred – Creates NIS+ credentials

SYNOPSIS

nisaddcred [-p *principal*] [-P *nis_principal*] [-l *login_password*] *auth_type* [*domain_name*]

nisaddcred -r [*nis_principal*]

nisaddcred -r [*domain_name*]

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nisaddcred command creates security credentials for NIS+ principals. NIS+ credentials serve two purposes. The first is to provide authentication information to various services; the second is to map the authentication service name into an NIS+ principal name.

The nisaddcred command accepts the following options:

-p *principal* Uses the principal name *principal* to fill the auth_name field for this entry. For LOCAL credentials, the name supplied with this option should be a string specifying a user ID (UID). For Data Encryption Standard (DES) credentials, the name should be a secure Remote Procedure Call (RPC) network name of the form *unix.id@domain*. If the -p option is not specified, the auth_name field is constructed from the UID of the current process and the name of the local domain.

-P *nis_principal* Specifies the NIS+ principal name. Creates LOCAL credentials for users whose home domain is different than the local machine's default domain.

When the -P option is not specified, nisaddcred constructs a principal name for the entry as follows. When it is not creating an entry of type LOCAL, nisaddcred calls nis_local_principal, which looks for an existing LOCAL entry for the effective UID of the current process in the cred.org_dir table and uses the associated principal name for the new entry. When creating an entry of authentication type LOCAL, nisaddcred constructs a default NIS+ principal name by taking the login name of the effective UID for its own process, and appending a dot (.) symbol to it followed by the local machine's default domain. If the caller is a super user, the machine name is used instead of the login name.

`-l` *login_password* — Specifies a password to encrypt the secret key for the credential entry. This overrides the prompting for a password from the shell. This option is intended for administration scripts only. Prompting guarantees that no one can see your password on the command line by using `ps`(1), and that you have not made any mistakes.

The value of *login_password* does not have to be the user's password, but it simplifies logging in if it is.

`-r` *nis_principal* — Removes all credentials associated with the principal *nis_principal* from the `cred.org_dir` table. If *nis_principal* is not specified, the default is to remove credentials for the current user.

`-r` *domain_name* — Removes a client or user from the system. If *domain_name* is not specified, the operation is executed in the default NIS+ domain.

When the `nisaddcred` command is run, these credentials are created and stored in a table named `cred.org_dir` in the default NIS+ domain. If *domain_name* is specified, the entries are stored in the `cred.org_dir` table of the specified domain. The credentials of users must be stored in the same domain as their passwords.

domain_name — The name of a domain.

NIS+ Principal Names

NIS+ principal names are used to specify clients that have access rights to NIS+ objects. See `nischmod`(8), `nischown`(8), `nis_objects`(3N), and `nis_groups`(3N) for more information. Various other services can also implement access control based on these principal names.

The `cred.org_dir` table is organized as follows:

cname	*auth_type*	*auth_name*	*public_data*	*private_data*
`fred.foo.com.`	`LOCAL`	`2990`	`10,102,44`	
`fred.foo.com.`	`DES`	`unix.2990@foo.com`	`098...819`	`3b8...ab2`

The `cname` column in this table contains a canonical representation of the NIS+ principal name. By convention, this name is the login name of a user, or the host name of a machine, followed by a dot (`.`) followed by the fully qualified home domain of that principal. For users, the home domain is defined to be the domain where their DES credentials are kept. For hosts, their home domain is defined as the *domain_name* returned by the `domainname`(1) command executed on that host.

Auth_type Entries

There are two types of *auth_type* entries found in the `cred.org_dir` table: the authentication type `LOCAL`, and the authentication type DES. The authentication type should be specified on the command line, in uppercase or lowercase, as either `local` or `des`.

Entries of type LOCAL are used by the NIS+ service to determine the correspondence between fully qualified NIS+ principal names and users identified by UIDs in the domain containing the cred.org_dir table. This correspondence is required when associating requests made using the AUTH_SYS RPC authentication flavor to an NIS+ principal name. It is also required for mapping a UID in one domain to its fully qualified NIS+ principal name whose home domain may be elsewhere.

The principal's credentials for any authentication flavor are found the cred.org_dir table in the principal's home domain (extracted from the principal name). The same NIS+ principal may have LOCAL credential entries in more than one domain. Only users, and not machines, have LOCAL credentials. In their home domain, NIS+ users should have both LOCAL and DES types of credentials.

The *auth_name* associated with the LOCAL type entry is a UID that is valid for the principal in the domain containing the cred.org_dir table. This may differ from that in the principal's home domain.

The public information stored in the *public_data* column for a LOCAL entry type contains a list of group IDs (GIDs) for groups in which the user is a member. The GIDs also apply to the domain in which the table resides. There is no private data associated with this type. Neither a UID nor a principal name should appear more than once among the LOCAL entries in any one cred.org_dir table.

The DES authentication type is used for secure RPC authentication.

The authentication name associated with the DES *auth_type* is a secure RPC network name. A secure RPC network name has the form *unix.id@domain*, where *domain* must be the same as the domain of the principal. For principals that are users, the UNIX ID UID is the principal in the principal's home domain. For principals that are hosts, the UNIX ID is the host's name. In secure RPC, processes running under an effective UID of 0 (root) are identified with the host principal. Unlike LOCAL, there cannot be more than one DES credential entry for one NIS+ principal in the NIS+ namespace.

The public information in a DES authentication type entry is the public key for the principal. The private information in this entry is the private key of the principal encrypted by the principal's network password.

NIS+ users should have both types of credentials in their home domain. In addition, a principal must have a LOCAL entry in the cred.org_dir table of each domain from which the principal wants to make authenticated requests. An NIS+ client that makes a request from a domain in which it does not have a LOCAL entry will be unable to acquire DES credentials. An NIS+ service running at security level 2 considers such users unauthenticated and assigns them the name nobody for access rights.

This command is run only by NIS+ principals who are authorized to add or delete the entries in the cred.org_dir table.

If credentials are being added for the caller itself, nisaddcred automatically executes the keylogin(1) command for the caller.

EXIT STATUS

The nisaddcred command exits with one of the following values:

Value	Description
0	Success.

1 Failure, an error occurred.

SEE ALSO

nischmod(8), nischown(8), nismatch(8), nistbladm(8)

chkey(1), domainname(1), keylogin(1), ps(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

nis_groups(3N), nis_local_names(3N), nis_objects(3N) in *ONC+ Technology for the UNICOS Operating System*, Cray Research publication SG–2169

NAME

nisaddent – Creates NIS+ tables from their corresponding /etc files and NIS maps

SYNOPSIS

nisaddent [-a] [-P] [-r] [-v] [-D *defaults*] [-t *table*] *type* [*nisdomain*]

nisaddent [-a] [-m] [-P] [-r] [-v] [-D *defaults*] -f *file* [-t *table*] *type* [*nisdomain*]

nisaddent [-a] [-P] [-r] [-v] [-D *defaults*] [-t *table*] -y *ypdomain* [-Y *map*] *type* [*nisdomain*]

nisaddent -d [-A] [-M] [-t *table*] *type* [*nisdomain*]

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nisaddent command creates NIS+ tables from their corresponding /etc files and NIS maps. This operation is customized for each of the standard tables that are used in the administration of UNICOS systems.

The nisaddent command accepts the following options and operands:

-a Adds the *file* or *map* to the NIS+ table without deleting any existing entries. This option is the default. This mode propagates only additions and modifications, not deletions.

-A Specifies that all the data within the table and all of the data in tables in the initial table's concatenation path be returned.

-d Dumps the NIS+ table to the standard output in the appropriate format for the given *type*.

-f *file* Specifies that *file* should be used as the source of input instead of the standard input.

-m Merges the *file* or *map* with the NIS+ table. This is the most efficient way to bring an NIS+ table up to date with a *file* or NIS *map* when there are only a small number of changes. This option adds entries that are not already in the database, modifies entries that already exist (if changed), and deletes any entries that are not in the source. Use the -m option whenever the database is large and replicated, and the *map* being loaded differs only in a few entries. This option reduces the number of update messages that have to be sent to the replicas. Also see the -r option.

-M Specifies that lookups should be sent only to the master server. This guarantees that the most up-to-date information is seen, but has the disadvantage that it may make the master server busy and therefore unavailable.

-P Follows the concatenation path. Specifies that lookups should follow the concatenation path of a table if the initial search is unsuccessful.

`-r` Replaces the *file* or *map* in the existing NIS+ table by deleting any existing entries and then adding the entries from the source (`/etc` files or NIS+ maps). This option has the same effect as the `-m` option. The use of this option is strongly discouraged, unless there are a large number of changes, due to its adverse impact on performance.

`-t` *table* Specifies the NIS+ table to use for this operation. The *table* argument should be a relative name as compared to either your default domain or the *domainname* if it has been specified.

`-v` Specifies verbose mode.

`-y` *ypdomain* Uses the `dbm` files for the appropriate NIS map, from the NIS domain *ypdomain*, as the source of input. The files are expected to be on the local machine in the `/etc/yp/`*ypdomain* directory. If the machine is not an NIS server, use `ypxfr`(8) to get a copy of the `dbm` files for the appropriate map.

`-Y` *map* Uses the `dbm` files for *map* as the source of input.

`-D` *defaults* Specifies to use a different set of *defaults* during this operation. The *defaults* string is a series of tokens separated by colons. These tokens represent the default values to be used for the generic object properties. Following are descriptions of the legal tokens.

`ttl=`*time*
Sets the default time-to-live for objects that are created by this command. The value *time* is specified in the format as defined by the `nischttl`(8) command. The default is 12 hours.

`owner=`*ownername*
Specifies that the NIS+ principal *ownername* should own the created object. The default for this value is the principal who is executing the command.

`group=`*groupname*
Specifies that the group *groupname* should be the group owner for the object that is created. The default value is `null`.

`access=`*rights*
Specifies the set of `access` rights that are to be granted for the given object. The value of *rights* is specified in the format as defined by the `nischmod`(8) command. The default is `----rmcdr----r---`.

type Specifies the type of data being processed. Legal values for *type* are one of the following aliases:

- `bootparams`
- `ethers`
- `group`
- `hosts`
- `netid`

- netmasks
- networks
- passwd
- protocols
- publickey
- rpc
- services

nisdomain The name of the NIS+ domain.

By default, nisaddent reads from the standard input file and adds this data to the associated NIS+ table by using the type specified on the command line. An alternate NIS+ table may be specified by using the -t option. For a key-value type table, a table specification is required.

Note that the data type can be different than the table name. For example, the automounter tables have key-value as the table type.

Files may be processed by using the -f option, and NIS version 2 (YP) maps may be processed by using the -y option. The -m option is not available when reading data from standard input.

If a *ypdomain* argument is specified, the nisaddent command takes its input from the dbm files for the appropriate NIS map group.byname, passwd.byname, or publickey.byname, An alternate NIS map may be specified by using the -Y option. If key-value is specified for the *type* operand, a map specification is required. The map must be in the /etc/yp/*ypdomain* directory on the local machine. Note that *ypdomain* is case sensitive.

If *nisdomain* is specified, nisaddent operates on the NIS+ table in that NIS+ domain, otherwise the default domain is used.

In terms of performance, loading up the tables is fastest when you use the -y option, followed by the -f *files* option and the standard input.

ENVIRONMENT VARIABLES

Following are environment variables that affect the defaults associated with a process.

NIS_DEFAULTS This variable contains a default string that overrides the NIS+ standard defaults. If the -D switch is used, those values override both the NIS_DEFAULTS variable and the standard defaults.

NIS_PATH If this variable is set and neither the *nisdomain* nor the table are fully qualified, each directory specified in NIS_PATH is searched until the table is found. See nisdefaults(8) for more information.

EXIT STATUS

The `nisaddent` command exits with one of the following values:

Value	Description
0	Success.
1	Failure, an error occurred.

SEE ALSO

`nischmod`(8), `nisdefaults`(8), `nissetup`(8), `nistbladm`(8), `ypxfr`(8)

`passwd`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`hosts`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

nis_cachemgr – Maintains a cache of location information about NIS+ servers

SYNOPSIS

nis_cachemgr [-d *level*] [-i] [-m *maxsize*] [-n] [-s *size*] [-v]

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nis_cachemgr daemon maintains a cache of the NIS+ directory objects. It is recommended that this daemon is run on all machines that are using NIS+.

The cache contains location information necessary to contact the NIS+ servers that serve the various directories in the namespace. This includes transport addresses, information needed to authenticate the server, and a time-to-live field that indicates how long the directory object can be cached. Although the cache helps to improve the performance of the clients that are traversing the NIS+ namespace, NIS+ requests are serviced whether or not nis_cachemgr is running.

The cache maintained by this program is shared by all of the processes that access NIS+ on a machine. At startup, nis_cachemgr initializes the cache from the coldstart file and preserves unexpired entries that already exist in the cache file. See nisinit(8) for more information. The cache survives machine reboots because the cache is maintained in a file that is memory mapped by all the processes.

The nis_cachemgr program is usually started from a system start-up script.

The nis_cachemgr command makes NIS+ requests under the NIS+ principal name of the host on which it runs. Before running nis_cachemgr, security credentials for the host should be added to the cred.org_dir table in the host's domain by using the nisaddcred(8) command. Data Encryption Standard (DES)-type credentials are required if the NIS+ service is operating at security level 2. See nisd(8) for more information. See the WARNINGS section for more information on security and performance.

The nisshowcache(8) command is used to look at cached objects.

The nis_cachemgr command accepts the following options:

-d *level* Turns debugging mode on. In insecure mode, the nis_cachemgr daemon displays information on the terminal. The amount of information depends on the level (1 to 3) selected.

-i Forces nis_cachemgr to ignore the previous cache file and reinitialize the cache from the coldstart file. By default, the cache manager initializes itself from both the coldstart file and the old cache file, thereby maintaining the entries in the cache across machine reboots.

-m *maxsize* Determines the maximum cache size. When the limit is reached no more information can be cached. *maxsize* specifies the number of 4-kbyte chunks. The default is 8 * 4 kbytes.

`-n` Executes `nis_cachemgr` in an insecure mode. By default, before adding a directory object to the shared cache, on the request of another process on the machine, it checks the encrypted signature on the request to make sure that the directory object is a valid one and is sent by an authorized server. In insecure mode, `nis_cachemgr` adds the directory object to the shared cache without making this check.

`-s` *size* Determines the starting size of the cache. *size* specifies the number of 4-kbyte chunks. The default and minimum is 2 * 4Kbytes

`-v` Sets verbose mode. In this mode, the `nis_cachemgr` program logs not only errors and warnings, but also additional status messages. The additional messages are logged using `syslog`(3C) with a priority of `LOGINFO`.

WARNINGS

If the host principal does not have the proper security credentials in the `cred.org_dir` table for its domain, running this program without the `-n` insecure mode option may significantly degrade the performance of processes issuing NIS+ requests.

MESSAGES

The `nis_cachemgr` daemon logs error messages and warnings using `syslog`(3C). See `syslog`(3C) for more information.

FILES

`/etc/nis/NIS_SHARED_DIRCACHE`	The shared cache file
`/etc/nis/COLD_START_FILE`	The coldstart file
`/etc/init.d/rpc`	Initialization scripts for NIS+

SEE ALSO

`nisaddcred`(8), `nisd`(8), `nisinit`(8), `nisshowcache`(8)

`syslog`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`nisfiles`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

`niscat` – Displays NIS+ tables and objects

SYNOPSIS

`niscat` [`-A`] [`-h`] [`-L`] [`-M`] [`-v`] *tablename...*

`niscat` [`-A`] [`-L`] [`-M`] [`-P`] `-o` *name...*

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The `niscat` command displays the contents of the NIS+ tables specified by *tablename*. The command also displays the internal representation of the NIS+ objects specified by *name*, and sends it to the standard output.

The `niscat` command accepts the following options and operands:

`-A` Displays the data within the table and all of the data in tables when the initial table's concatenation path is returned.

`-h` Displays a header line prior to displaying the table. The header consists of the hash (#) symbol followed by the name of each column. The column names are separated by the table separator character.

`-L` Follows links. If *tablename* or *name* specifies a `LINK` type object, the link is followed and the object or table named by the link is displayed.

`-M` Sends the request to only the master server of the specified data. This guarantees that the most current information is seen. This option increases the load on the master server and increases the possibility that the NIS+ server will be unavailable for updates.

`-P` Specifies that the request should follow the concatenation path of a table if the initial search is unsuccessful. This option is useful only when using an indexed name for the `-o` *name* option.

`-v` Displays columns containing binary data on the standard output. Without this option, binary data is displayed as the string `*BINARY*`.

`-o` *name*
Displays the internal representation of the named NIS+ objects. If *name...* is an indexed name, each of the matching entry objects is displayed. See the `nismatch`(8) man page for more information. This option is used to display access rights and other attributes of individual columns.

tablename
The specified NIS+ table.

NOTES

Columns without values in the table are displayed by two adjacent table separator characters.

ENVIRONMENT VARIABLES

Following is an environment variable that affects the defaults associated with a process.

`NIS_PATH`
: If this variable is set and the NIS+ table name is not fully qualified, each directory specified is searched until the table is found. See `nisdefaults`(8) for more information.

EXIT STATUS

The `niscat` command exits with one of the following values:

Value	Description
0	Success.
1	Failure, an error occurred.

SEE ALSO

`nisdefaults`(8), `nismatch`(8), `nistbladm`(8)

`nis_objects`(3N), `nis_tables`(3N) in *ONC+ Technology for the UNICOS Operating System*, Cray Research publication SG–2169

NAME

nischgrp – Changes the group owner of an NIS+ object

SYNOPSIS

nischgrp [-A] [-f] [-L] [-P] *group name...*

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nischgrp command changes the group owner of the NIS+ objects or entries specified by *name* to the specified NIS+ group. Entries are specified using indexed names. See nismatch(8) for more information. If *group* is not a fully qualified NIS+ group name, it will be resolved using the directory search path. See nisdefaults(8) for more information.

The only restriction on changing an object's group owner is that you must have modify permissions for the object.

This command fails if the master NIS+ server is not running.

The nischgrp command accepts the following options:

-A Modifies all entries in all tables in the concatenation path that match the search criterion specified in *name*. This option requires that you also use the -P switch.

-f Forces the operation and fails silently if it does not succeed.

-L Follows links and changes the group owner of the linked object or entries rather than the group owner of the link itself.

-P Follows the concatenation path within a named table. The *name* is either an indexed name or the -L switch is specified and the named object is a link pointing to entries.

group The NIS+ group name.

name The NIS+ object name specified for a change.

NOTES

The NIS+ server checks the validity of the group name before making a change.

ENVIRONMENT VARIABLES

Following is an environment variable that affects the defaults associated with a process.

NIS_PATH If this variable is set and the NIS+ name is not fully qualified, each directory specified is searched until the object is found. See nisdefaults(8) for more information.

EXIT STATUS

The `nischgrp` command exits with one of the following values:

Value	Description
0	Success.
1	Failure, an error occurred.

SEE ALSO

`nischmod`(8), `nischown`(8), `nisdefaults`(8), `nisgrpadm`(8), `nismatch`(8)

`nis_objects`(3N) in *ONC+ Technology for the UNICOS Operating System*, Cray Research publication SG–2169

NAME

nischmod – Changes access rights on an NIS+ object

SYNOPSIS

nischmod [-A] [-f] [-L] [-P] *mode name...*

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nischmod command changes the access rights (mode) of the NIS+ objects or entries specified by *name* to *mode*. Entries are specified using indexed names. See nismatch(8) for more information. Only principals with modify access to an object may change its mode.

The nischmod command accepts the following options and operands:

-A Modifies all entries in all tables in the concatenation path that match the search criteria specified in *name*. This option requires that you also use the -P option.

-f Forces the operation and fails silently if it does not succeed.

-L Follows links and changes the permission of the linked object or entries rather than the permission of the link itself.

-P Follows the concatenation path within a named table. The *name* is either an indexed name or the -L switch is specified and the specified object is a link pointing to an entry.

mode The access rights of NIS+ objects.

name The name of an NIS+ object.

The variable *mode* has the following form:

rights [,*rights*]...

The variable *rights* has the following form:

[*who*] *op permission* [*op permission*]...

The *who* variable is a combination of the following values:

n Nobody's permissions

o Owner's permissions

g Group's permissions

w World's permissions

a All, or owg

If *who* is omitted, the default is `a`.

The *op* variable is one of the following values:

`+` To grant the permission.

`-` To revoke the permission.

`=` To set the permissions explicitly.

The *permission* variable is any combination of the following variables:

`r` Read

`m` Modify

`c` Create

`d` Destroy

NOTES

Unlike the system `chmod`(1) command, this command does not accept an octal notation.

ENVIRONMENT VARIABLES

Following is an environment variable that affects the defaults associated with a process.

`NIS_PATH` If this variable is set and the NIS+ name is not fully qualified, each directory specified is searched until the object is found. See `nisdefaults`(8) for more information.

EXIT STATUS

The `nischmod` command exits with one of the following values:

Value	Description
0	Success.
1	Failure, an error occurred.

SEE ALSO

`nischgrp`(8), `nischown`(8), `nisdefaults`(8), `nismatch`(8)

`chmod`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`nis_objects`(3N) in *ONC+ Technology for the UNICOS Operating System*, Cray Research publication SG–2169

`nis`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

nischown – Changes the owner of an NIS+ object

SYNOPSIS

nischown [-A] [-f] [-L] [-P] *owner name...*

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nischown command is used to change the owner of the NIS+ objects or entries specified by the operands *name* and *owner*. Entries are specified using indexed names. See nismatch(8) for more information. If the operand *owner* is not a fully qualified NIS+ principal name, the default domain is appended to it. See nisaddcred(8) and domainname(1) for more information.

The only restriction on changing an object's owner is that you must have modify permissions for the object. If you are the current owner of an object and you change ownership, you may not be able to regain ownership unless you have modify access to the new object.

The nischown command accepts the following options and operands:

-A Modifies all entries in all tables in the concatenation path that match the search criteria specified in name. It also requires the -P option.

-f Forces the operation and fails silently if it does not succeed.

-L Follows links and changes the owner of the linked object or entries rather than the owner of the link itself.

-P Follows the concatenation path within a specified table. Use this option only when either *name* is an indexedname or the -L option is specified and the specified object is a link pointing to other entries.

owner The new owner of an NIS+ object.

name The name of an NIS+ object or an entry in an NIS+ table.

NOTES

The NIS+ server checks the validity of the name before making the modification.

ENVIRONMENT VARIABLES

Following is an environment variable that affects the defaults associated with a process.

NIS_PATH If this variable is set and the NIS+ name is not fully qualified, each directory specified is searched until the object is found. See nisdefaults(8) for more information.

EXIT STATUS

The nischown command exits with one of the following values:

Value	Description
0	Success.
1	Failure, an error occurred.

SEE ALSO

nisaddcred(8), nischgrp(8), nischmod(8), nischttl(8), nisdefaults(8), nismatch(8)

domainname(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

nis_objects(3N) in *ONC+ Technology for the UNICOS Operating System*, Cray Research publication SG–2169

NAME

`nischttl` – Changes the time-to-live value of an NIS+ object

SYNOPSIS

`nischttl` [`-A`] [`-f`] [`-L`] [`-P`] *time name...*

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The `nischttl` command is used to change the time-to-live value (*ttl*) of the NIS+ objects or entries specified by *name* to the value specified by *time*. Entries are specified by using indexed names. See `nismatch`(8) for more information.

The time-to-live value is used by object caches to expire objects within their cache. When an object is read into the cache, this value is added to the current time in seconds, yielding an expiration time. The object may be returned from the cache until the current time is earlier than the calculated expiration time. When the expiration time is reached, the object is flushed from the cache.

The time-to-live value may be specified in seconds or in days, hours, minutes, seconds format. The latter format uses a suffix letter of `d`, `h`, `m`, or `s` to identify the units of time.

The `nischttl` command accepts the following options and operands:

`-A` Modifies all tables in the concatenation path that match the search criteria specified in *name*. This option implies the `-P` switch.

`-f` Forces the operation and fails silently if it does not succeed.

`-L` Follows links and change the time-to-live of the linked object or entries rather than the time-to-live of the link itself.

`-P` Follows the concatenation path within a named table. The *name* is either an indexed name, or the `-L` switch is specified and the named object is a link pointing to entries.

time Specified time value of NIS+ object.

name Specified entry by *indexedname*.

NOTES

Setting a high *ttl* value allows objects to stay persistent in caches for a longer period of time and can improve performance. However, when an object changes, in the worst case, the number of seconds in this attribute must pass before that change is visible to all clients. Setting a *ttl* value of 0 means that the object should not be cached at all.

A high *ttl* value is a week, a low value is less than a minute. Password entries should have *ttl* values of about 12 hours. This allows easily one password change per day. Entries in the Remote Procedure Call (RPC) table can have *ttl* values of several weeks because this information is rarely changes.

ENVIRONMENT VARIABLES

Following is an environment variable that affects the defaults associated with a process.

`NIS_PATH` If this variable is set and the NIS+ name is not fully qualified, each directory specified is searched until the object is found. See `nisdefaults`(8) for more information.

EXIT STATUS

The `nischttl` command exits with one of the following values:

Value	Description
0	Success.
1	Failure, an error occurred.

EXAMPLES

The following example shows how to represent the *ttl* of an object by using the seconds format and the days, hours, minutes, seconds format. The second example shows that the *ttl* of the object is set to 1 day and 12 hours.

```
example% nischttl 184000  object
example% nischttl 1d12h object
```

SEE ALSO

`nischgrp`(8), `nischmod`(8), `nischown`(8), `nisdefaults`(8), `nismatch`(8)

`nis_objects`(3N) in *ONC+ Technology for the UNICOS Operating System*, Cray Research publication SG–2169

NAME

`nisctl` – Controls the operation of NIS+ servers

SYNOPSIS

`nisctl` [`-M`] [`-v 1`] [`-f` *flag*] [`-H` *hostname*] *domain*

`nisctl` [`-M`] [`-v 0`] [`-f` *flag*] [`-H` *hostname*] *domain*

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The `nisctl` command controls the operation of NIS+ servers. It is used to turn debugging on and off, to flush caches, and to print statistics.

The `nisctl` command accepts the following options:

`-M` Modifies the data segment space allocation for the NIS+ daemon `nisd` that resides on all servers that serve the local domain if no domain or server is specified. See the `-H` option. Modifies the data segment space allocation for the NIS+ daemon `nisd` on all servers that serve a specified domain if no server is specified. Modifies the data segment space allocation for the NIS+ daemon `nisd` that resides on a specific server that serves a specified domain if the `-H` option is used.

`-v` Turns debugging on (`1`) or off(`0`) on all the servers or just a specific server that serve(s) the local or a specified domain.

`-f` *flag* Causes NIS+ servers to release cached information about the namespace. The `-f` option accepts the following *flag* arguments:

- `-o` Flushes all cached information about NIS+ object.
- `-g` Flushes all cached information about NIS+ groups.
- `-d` Flushes all cached information about NIS+ directory.

`-H` *hostname*
Specifies a server in an NIS+ domain.

domain Specifies a particular domain in the namespace. If no domain is specified, the local domain is selected.

EXIT STATUS

The `nisctl` command exits with one of the following values:

Value	Description
0	Success.
1	Failure, an error occurred.

SEE ALSO

nisd(8)

NAME

nisd – NIS+ service daemons

SYNOPSIS

nisd [-A] [-C] [-D] [-F] [-f] [-h] [-r] [-v] [-Y] [-c *seconds*] [-d *dictionary*] [-L *load*] [-S *level*]

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nisd daemon is an Open Network Computing (ONC) Remote Procedure Call (RPC) service that implements the NIS+ service. This daemon must be running on all machines that serve a portion of the NIS+ namespace.

The nisd daemon is usually started from a system startup script.

The nisd daemon accepts the following options:

-A Logs all the authentication related activities in verbose mode to syslogd(8) with LOGINFO priority.

-C Opens a log file that contains all the messages displayed on the console.

-D Turns debugging mode on.

-F Forces the server to do a checkpoint of the database when it starts. Forced checkpoints may be required when the server is low on disk space. This option removes updates from the transaction log that have propagated to all of the replicas.

-f Forces registration even if the program number is already in use.

-h Prints all the nisd options.

-r Tells the daemon that it is running as a root server. The primary difference between the root server and a nonroot server is that root servers are members of the same domain they serve, whereas nonroot servers are always members of a domain above the domain they serve.

-v Sends a running verbose narration of what the daemon is doing to the syslog daemon with LOGINFO priority. See syslogd(8) for more information. This option is most useful for debugging problems with the service. (See also the -A option.)

-Y Puts a server into NIS (YP) compatibility mode. When operating in this mode, the NIS+ server responds to NIS version 2 requests using the NIS version 2 protocol. Because the YP protocol is not authenticated, only those items that have read access to nobody (an unauthenticated request) are visible through the version 2 protocol. The server supports only the standard NIS version 2 maps in this mode.

`-c` *seconds* Sets the number of seconds between sending out updates to the replicas. By default this is set to 120 seconds or 2 minutes.

`-d` *dictionary* Specifies an alternate dictionary for the NIS+ database. The primary use of this option is for testing. The string is not interpreted, it is simply passed to the `db_initialize` function. See `nis_db`(3N) for more information.

`-L` *load* Specifies the load that the NIS+ service is allowed to place on the server. The load is specified in terms of the number of child processes that the server may spawn. The number of child processes must be at least 1 for the callback functions to work correctly. The default is 128.

`-S` *level* Sets the authentication security level of the service. By default, the daemon runs at security level 2.

The value of *level* is a number between 0 and 2:

0 Security level 0 is designed to be used for testing and the initial setup of an NIS+ namespace. When running at level 0, the daemon does not enforce any access controls. Any client is allowed to perform any operation.

1 At security level 1, the daemon accepts both `AUTH_SYS` and `AUTH_DES` credentials for authenticating clients and authorizing them to perform NIS+ operations. This is not a secure mode of operation since `AUTH_SYS` credentials are easily forged. `AUTH_SYS` authentication is not recommended on networks where there is the potential for untrusted user access.

2 At security level 2, the daemon accepts only `AUTH_DES` credentials for authentication and authorization. This is the highest level of security currently provided by the NIS+ service. This is the default security level if the `-S` option is not used.

ENVIRONMENT VARIABLES

Following is an environment variable that affects the defaults associated with a process.

`NETPATH` The transports that the NIS+ service uses can be limited by setting this environment variable. See `netconfig`(4) for more information.

FILES

`/etc/nis/parent.object`
This file contains an external Data Representation (XDR)-encoded NIS+ object that describes the namespace above a root server. This parent namespace may be another NIS+ namespace or a foreign namespace such as one served by the domain name service. It is present only on servers that are serving the root of the namespace.

`/etc/nis/root.object`
This file contains an XDR-encoded NIS+ object that describes the root of the namespace. It is present only on servers that are serving the root of the namespace.

`/etc/nis/COLD_START_FILE`
This file contains the necessary information for the NIS+ client libraries to locate a server for the default domain.

`/etc/nis/NIS_SHARED_DIRCACHE`
This file contains a set of cached objects describing servers that serve other portions of the NIS+ namespace. It is created and maintained by the `nis_cachemgr`(8).

`/etc/init.d/rpc`
This file is an initialization script for NIS+.

`/etc/nisl.log`
This file combines all the messages displayed on the console if the `-C` option is used.

SEE ALSO

`nis_cachemgr`(8), `nisinit`(8), `nissetup`(8), `syslogd`(8)

`nis_db`(3N) in *ONC+ Technology for the UNICOS Operating System*, Cray Research publication SG–2169

`netconfig`(4), `nisfiles`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

nisdefaults – Displays NIS+ default values

SYNOPSIS

nisdefaults [-a] [-d] [-g] [-h] [-p] [-r] [-s] [-t] [-v]

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nisdefaults command prints the default values that are returned by calls to the NIS+ local name functions. See nis_local_names(3N) for more information. With no options specified, all defaults are printed in verbose mode. When options are specified, options are displayed only in a terse mode suitable for shell scripts.

The nisdefaults command accepts the following options:

-a Prints all defaults in a terse format.

-d Prints the default domain name.

-g Prints the default group name.

-h Prints the default host name.

-p Prints the default principal name.

-r Prints the default access rights.

-s Prints the default directory search path.

-t Prints the default time-to-live value.

-v Prints the defaults in a verbose format. This prepends an identifying string to the output.

ENVIRONMENT VARIABLES

The following environment variables affect the defaults associated with a process.

NIS_DEFAULTS This variable contains a defaults string that overrides the NIS+ standard defaults. The defaults string is a series of tokens separated by colons. These tokens represent the default values to be used for the generic object properties. Following are descriptions of all the legal tokens.

ttl=*time*

Sets the default time-to-live for objects that are created. The value *time* is specified in the format as defined by the nischttl(8) command. The default value is 12 hours.

owner=*ownername*
: Specifies that the NIS+ principal *ownername* should own created objects. The default for this value is the principal who is executing the command.

group=*groupname*
: Specifies that the group *groupname* should be the group owner for created objects. The default is NULL.

access=*rights*
: Specifies the set of access rights that are to be granted for created objects. The value of *rights* is specified in the format as defined by the nischmod(8) command. The default value is ----rmcdr--r---.

NIS_GROUP
: This variable contains the name of the local NIS+ group. If the name is not fully qualified, the default domain is appended to it.

NIS_PATH
: This variable overrides the default NIS+ directory search path. It contains an ordered list of directories separated by colon (:) symbols. The dollar sign ($) symbol is treated specially. Directory names that end in the dollar sign have the default domain appended to them, and a dollar sign by itself is replaced by the list of directories between the default domain and the global root that are at least two levels deep. The default NIS+ directory search path is $.

SEE ALSO

nischmod(8), nischttl(8)

nis_local_names(3N) in *ONC+ Technology for the UNICOS Operating System*, Cray Research publication SG–2169

NAME

`niserror` – Displays NIS+ error messages

SYNOPSIS

`niserror` *error-num*

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The `niserror` command displays the NIS+ error message associated with the status value *error-num* in standard output. It is used to translate NIS+ error numbers that are returned into text messages for shell scripts.

The `niserror` command accepts the following operand:

error-num Specifies the status number attached to an error message.

EXIT STATUS

The `niserror` command exits with one of the following values:

Value	Description
0	Success.
1	Failure, an error occurred.

SEE ALSO

`nis_error`(3N) in *ONC+ Technology for the UNICOS Operating System*, Cray Research publication SG–2169

NAME

nisgrpadm – Administers NIS+ groups

SYNOPSIS

nisgrpadm -a [-s] *group principal...*

nisgrpadm -r [-s] *group principal...*

nisgrpadm -t [-s] *group principal...*

nisgrpadm [-s] *group principal...*

nisgrpadm -c [-M] [-s] *group*

nisgrpadm -d [-M] [-s] *group*

nisgrpadm -l [-M] [-s] *group*

nisgrpadm [-M] [-s] *group*

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nisgrpadm command administers both NIS+ groups and group membership lists. The nisgrpadm command is used to create, destroy, or list NIS+ groups. The nisgrpadm command is also used to administer a group's membership list. It can add or delete principals to the group, or test principals for membership in the group.

The names of NIS+ groups are syntactically similar to names of NIS+ objects, but they occupy a separate namespace. A group named a.b.c.d. is represented by an NIS+ group object named a.groups_dir.b.c.d.; the functions described here all expect the name of the group, not the name of the corresponding group object. There are three types of group members:

- An explicit member is just an NIS+ principal name, such as wickedwitch.west.oz..
- An implicit domain member, such as *.west.oz., means that all principals in the given domain belong to this member. No other forms of wildcards are allowed. Both wickedwitch.*.oz. and wickedwitch.west.*. are not valid. The principals in subdomains of the given domain are not included.
- A recursive group member, such as @cowards.oz., refers to another group. All principals that belong to that group are considered to belong here.

Any type of group member may be made negative by prefixing it with a minus sign (-). A group may thus contain explicit, implicit, recursive, negative explicit, negative implicit, and negative recursive members.

A principal is considered to belong to a group if it belongs to at least one nonnegative group member of the group and belongs to no negative group members.

The `nisgrpadm` command accepts the following options and operands:

`-a` Adds the list of NIS+ principals specified to *group*. The *principal* name must be fully qualified.

`-c` Creates a group in the NIS+ namespace. The NIS+ group name must be fully qualified.

`-d` Destroys (removes) a group from the namespace.

`-l` Lists the membership list of the specified group. (See the `-M` option.)

`-M` Master server only. Sends the lookup to only the master server of the specified data. This guarantees that the most up-to-date information is seen, but has the disadvantage that it may make the master server busy and therefore unavailable. The `-M` option is applicable only when used with with the `-l` option.

`-r` Removes the list of principals specified from *group*. The principal name must be fully qualified.

`-s` Returns using the exit status of the command silently. This status can be translated into a text string by using the `niserror`(8) command.

`-t` Displays whether the principals specified are members in *group*.

group Specifies the group added or removed.

principal Specifies the principal added or removed.

NOTES

Principal names must be fully qualified, but group names can be abbreviated on all operations except `create`.

ENVIRONMENT VARIABLES

Following is an environment variable that affects the defaults associated with a process.

`NIS_PATH` If this variable is set and the NIS+ group name is not fully qualified, each directory specified is searched until the group is found. See `nisdefaults`(8) for more information.

MESSAGES

`NIS_SUCCESS` The `nisgrpadm` command returns 0 on success.

`NIS_PERMISSION` You do not have the needed access right to change the group.

`NIS_NOTFOUND` The specified group does not exist.

`NIS_TRYAGAIN` The server for the group's domain is currently checkpointing or in a read-only state. Retry the command at a later date.

`NIS_MODERROR` The group was modified by someone else during the execution of the command. Reissue the command and, optionally, check the group's membership list again.

SEE ALSO

`nischgrp`(8), `nisdefaults`(8)

`nis_error`(3N), `nis_groups`(3N) in *ONC+ Technology for the UNICOS Operating System*, Cray Research publication SG–2169

`nis`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

nisinit – Initializes an NIS+ client and server

SYNOPSIS

```
nisinit -r

nisinit -p Y parent_domain host...
nisinit -p D parent_domain host...
nisinit -p N parent_domain host...

nisinit -c -C coldstart

nisinit -c -H hostname

nisinit -c -B
```

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The `nisinit` command initializes a machine to be an NIS+ client or an NIS+ root server.

The `nisinit` command accepts the following options and operands:

`-r` Initializes a machine as an NIS+ root server. This option creates the file `/etc/nis/root.object` and initializes it to contain information about this machine.

`-p` Initializes an `/etc/nis/parent.object` file to make a domain part of the namespace above it. Only root servers can have parent objects. A parent object describes the namespace above the NIS+ root. If this is an isolated domain, this option should not be used. The *argument* to this option tells the command what type of name server is serving the domain above the NIS+ domain. This option takes a `Y`, `D`, or `N` argument. When clients attempt to resolve a name that is outside of the NIS+ namespace, this object is returned with the error `NIS_FOREIGNNS`, indicating that a namespace boundary has been reached. It is up to the client to continue the name resolution process.

The parameter *parent_domain* is the name of the parent domain in a syntax that is native to that type of domain. The list of host names that follow the domain parameter are the names of hosts that serve the parent domain. If there is more than one server for a parent domain, the first *host...* specified should be the master server for that domain.

`Y` Specifies that the parent directory is an NIS version 2 domain.

`D` Specifies that the parent directory is a Domain Naming Service (DNS).

`N` Specifies that the parent directory is another NIS+ domain. This option is useful for connecting a pre-existing NIS+ subtree into the global namespace.

Currently, the NIS+ clients do not use the `-p` option. This parent object is currently not replicated on root replica servers.

`-c` Initializes the machine to be an NIS+ client. The three initialization options available are *broadcast*, *coldstart*, and *hostname*. The most secure option is to initialize a client from a trusted *coldstart* file. The second option is to initialize a client using a *hostname* that you specify as a trusted host. The third option is to initialize a client by *broadcast*. The third option is the least secure.

Using the `-c` option enables only the navigation of the NIS+ namespace from a specific client. To make NIS+ your name service, modify the `/etc/nsswitch.conf` file. See `nsswitch`(4) for more information.

`-C` *coldstart*
Uses the file *coldstart* as a prototype *coldstart* file when initializing an NIS+ client. This *coldstart* file may be copied from a machine that is already a client of the NIS+ namespace. For maximum security, an administrator can encrypt and encode the *coldstart* file and mail it to an administrator who is starting up a new machine. The new administrator can decode, decrypt, and use this file with the `nisinit` command to initialize the machine as an NIS+ client. If the *coldstart* file is from another client in the same domain, the `nisinit` command may be safely skipped and the file copied into the `/etc/nis` directory as `/etc/nis/COLD_START_FILE`.

`-H` *hostname*
Specifies that the host *hostname* should be contacted as a trusted NIS+ server. The `nisinit` command iterates over each transport in the `NETPATH` environment variable and attempts to contact `rpcbind`(8) on that machine. This *hostname* must be reachable from the client without the name service running. For Internal Protocol (IP) networks there must be an entry in `/etc/hosts` for the specified host when `nisinit` is invoked.

`-B` Specifies that the `nisinit` command should use an IP broadcast to locate an NIS+ server on the local subnetwork. Any machine that is running the NIS+ service may answer. No guarantees are made that the server that answers is a server of the organization's namespace. If this option is used, we recommend checking with your system administrator if the server and domain served are valid. The binding information can be dumped to standard output using the `nisshowcache`(8) command.

ENVIRONMENT VARIABLES

Following is an environment variable that affects the defaults associated with a process.

`NETPATH` This environment variable may be set to the transports to try when contacting the NIS+ server. See `netconfig`(4) for more information.

EXIT STATUS

The `nisinit` command exits with one of the following values:

Value	Description
0	Success.

1 Failure, an error occurred.

FILES

`/etc/nis/COLD_START_FILE`
This file contains a list of servers, their transport addresses, and their secure RPC public keys that serve the machine's default domain.

`/etc/nis/hostname/root.object`
This file describes the root object of the NIS+ namespace. It is a standard XDR-encoded NIS+ directory object that can be modified by authorized clients using the `nis_modify`(3N) interface.

`/etc/nis/hostname/parent.object`
This file describes the namespace that is logically above the NIS+ namespace. The most common type of parent object is a DNS object. This object contains contact information for a server of that domain.

SEE ALSO

`rpcbind`(8), `nisshowcache`(8)

`uuencode`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`netconfig`(4), `nisfiles`(4), `nsswitch`(4), `hosts`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

nisln – Symbolically links NIS+ objects

SYNOPSIS

nisln [-L] [-D *defaults*] *name linkname*

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nisln command links an NIS+ object specified by *name* to an NIS+ name specified by *linkname*. If *name* is an indexed name, the link points to entries within an NIS+ table. See nismatch(8) for more information. Clients who want to look up information in the name service can use the FOLLOW_LINKS flag to force the client library to follow links to the name they point to. All NIS+ administration commands accept the -L switch, indicating that they should follow links. See nis_names(3N) for a description of the FOLLOW_LINKS flag.

The nisln command accepts the following options and operands:

-L
: Specifies that this command should follow links. If *name* is itself a link, then this command will follow it to the linked object that it points to. The new link will point to that linked object rather than to *name*.

-D *defaults*
: Specifies to use a different set of defaults for the creation of the link object. The *defaults* string is a series of tokens separated by colons. These tokens represent the default values to be used for the generic object properties. The legal tokens are as follows:

  ttl=*time*
  : Sets the default time-to-live for objects that are created by this command. The value *time* is specified in the format as defined by the nischttl(8) command. The default is 12 hours.

  owner=*ownername*
  : Specifies that the NIS+ principal *ownername* should own the created object. The default for this value is the principal who is executing the command.

  group=*groupname*
  : Specifies that the group *groupname* should be the group owner for the object that is created. The default value is NULL.

  access=*rights*
  : Specifies the set of access rights that are to be granted for the given object. The value of *rights* is specified in the format as defined by the nischmod(8) command. The default value is ----rmcdr--r---.

name
: Specifies the name of an NIS+ object.

linkname A specified name that identifies a link between an NIS+ object and and an NIS+ name.

NOTES

When creating the link, `nisln` verifies that the linked object exists. Once created, the linked object may be deleted or replaced and the link will not be affected. If the linked object is deleted, the link is not valid and attempts to return the error message `NIS_LINKNAMEERROR` to the client. When the path attribute in tables specifies a link rather than another table, the link is followed if the flag `FOLLOW_LINKS` was present in the call to the `nis_list` function. If the flag is not present, the link is ignored. See `nis_tables`(3N) for more information. If the flag is present and the link is no longer valid, a warning is sent to the system logger and the link is ignored.

ENVIRONMENT VARIABLES

Following is an environment variable that affects the defaults associated with a process.

`NIS_PATH` If this variable is set and the NIS+ name is not fully qualified, each directory specified is searched until the object is found. See `nisdefaults`(8) for more information.

EXIT STATUS

The `nisln` command exits with one of the following values:

Value	Description
0	Success.
1	Failure, an error occurred.

SEE ALSO

`nischmod`(8), `nischttl`(8), `nisdefaults`(8), `nismatch`(8), `nisrm`(8), `nistbladm`(8)

`nis_names`(3N), `nis_tables`(3N) in *ONC+ Technology for the UNICOS Operating System*, Cray Research publication SG–2169

NAME

nislog – Displays the contents of the NIS+ transaction log

SYNOPSIS

nislog [-h *num*] [-v] [*directory...*]

nislog [-t *num*] [-v] [*directory...*]

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nislog command displays the contents of the NIS+ server transaction log to standard output. This command is used to track changes in the namespace. The /etc/nis/*hostname*.log file contains the transaction log maintained by the NIS+ server. The *hostname* variable is a string returned by uname -n. When updates occur, they are logged to this file and then propagated to replicas as log transactions. When the log is checkpointed, updates that have been propagated to the replicas are removed. The nislog command must be executed by root on an NIS+ server. It displays the log entries for that server only. If *directory* is not specified, the entire log is searched. If *directory* is specified, only log entries that correspond to the specified directories are displayed.

The nislog command accepts the following options and operands:

-h [*num*] Displays *num* transactions from the "head" of the log. If the numeric parameter is omitted, it is assumed to be 1. If the numeric parameter is 0, only the log header is displayed.

-t [*num*] Displays *num* transactions from the "tail" of the log. If the numeric parameter is omitted, it is assumed to be 1. If the numeric parameter is 0, only the log header is displayed.

-v Specifies verbose mode.

directory Specifies the log for a specified directory.

FILES

/etc/nis/*hostname*.log *transaction log*

SEE ALSO

nisd(8)

uname(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

nis(4), nisfiles(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

`nisls` – Lists the contents of an NIS+ directory

SYNOPSIS

`nisls` [`-d`] [`-g`] [`-l`] [`-L`] [`-m`] [`-M`] [`-R`] [*name...*]

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The `nisls` command lists the contents of the NIS+ directory that is specified by *name*. If *name* specifies an NIS+ object other than a directory, `nisls` simply echoes the name. If no *name* is specified, the first directory in the search path is listed. See `nisdefaults`(8) for more information.

The `nisls` command accepts the following options and operand:

`-d` Treats NIS+ directories like other NIS+ objects, rather than listing their contents.

`-g` Displays the group owner instead of the owner when listing in long format.

`-l` Displays additional information in long format about the objects such as their type, creation time, owner, and access rights. The access rights are listed in the following order in long mode: nobody, owner, group owner, and world.

`-L` Specifies that links are to be followed. If the *name* variable points to a link, the link is followed to the linked object.

`-m` Displays modification time instead of creation time when listing in long format.

`-M` Returns information from only the master server of the specified object. This option guarantees that the most up-to-date information is seen, but has the disadvantage that it may make the master server busy and therefore unavailable.

`-R` Lists directories recursively by repeating the list for each subdirectory found in the process of listing each name.

name Specifies the name of the NIS+ directory you want to list.

ENVIRONMENT VARIABLES

Following is an environment variable that affects the defaults associated with a process.

`NIS_PATH` If this variable is set and the NIS+ name is not fully qualified, each directory specified is searched until the object is found. See `nisdefaults`(8)) for more information.

EXIT STATUS

The `nisls` command exits with one of the following values:

Value	Description
0	Success.
1	Failure, an error occurred.

SEE ALSO

`nisdefaults`(8), `nisgrpadm`(8), `nismatch`(8), `nistbladm`(8)

`nis_objects`(3N) in *ONC+ Technology for the UNICOS Operating System*, Cray Research publication SG–2169

NAME

nismatch, nisgrep – Searches NIS+ tables

SYNOPSIS

nismatch [-A] [-c] [-h] [-M] [-o] [-P] [-v] *key tablename*

nismatch [-A] [-c] [-h] [-M] [-o] [-v] *colname=key... tablename*

nismatch [-A] [-c] [-h] [-M] [-o] [-P] [-v] *indexedname*

nisgrep [-A] [-c] [-h] [-M] [-o] [-v] *key tablename*

nisgrep [-A] [-c] [-h] [-M] [-o] [-v]

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nismatch and nisgrep commands are used to search NIS+ tables. The nisgrep command differs from the nismatch command by the ability to accept regular *keypat* expressions for the search criteria rather than simple text matches.

For both commands, the parameter *tablename* is the NIS+ name of the table to be searched. If only one *key* or *keypat* is specified, then it is applied and the first column is searched. Specific named columns can be searched by using the *colname=* syntax. When multiple columns are searched, only entries that match in all columns are returned. This is the equivalent of a logical join operation.

Because nisgrep uses a callback function, it is not constrained to searching only those columns that are specifically made searchable at the time of table creation. Thus, it is more flexible, but slower, than nismatch.

The nismatch command accepts an additional form of search criteria, *indexedname*, which is an NIS+ indexed name of the form: [*colname=value,...*],*tablename*

The nismatch and nisgrep commands accept the following options and operands:

-A Returns all the data within a table and all of the data in tables in the initial table's concatenation path.

-c Prints only a count of the number of entries that matched the search criteria.

-h Displays a header line before the matching entries that contains the names of the table's columns.

-M Sends the lookup to only the master server of the specified data. This guarantees that the most up-to-date information is seen, but has the disadvantage that the master server may be busy.

-o Displays the internal representation of the matching NIS+ objects.

`-P`	Specifies that the lookup should follow the concatenation path of a table if the initial search is unsuccessful. This option is used only with the `nismatch` command.
`-v`	Specifies verbose mode.
key	Pattern of key strokes.
tablename	Name of a specified table.
colname=key	Specified column and key pattern to be searched.
indexedname	Indexed name of the column name and table name.

ENVIRONMENT VARIABLES

Following is an environment variable that affects the defaults associated with a process.

`NIS_PATH`
: If this variable is set and the NIS+ table name is not fully qualified, each directory specified is searched until the table is found. See `nisdefaults`(8) for more information.

EXIT STATUS

The `nismatch` command exits with one of the following values:

Value	Description
0	Returns 0 when it successfully matches entries.
1	Returns 1 when it successfully searches a table and no matches are found.
2	Returns 2 when an error condition occurs. An error message is also printed.

MESSAGES

`No memory`	An attempt to allocate some memory for the search failed.
tablename `is not a table`	The object with the name *tablename* was not a table object.
`Can't compile regular expression`	The regular expression in *keypat* was malformed.
`column not found:` *colname*	The column named *colname* does not exist in the table named *tablename*.

SEE ALSO

`niscat`(8), `nisdefaults`(8), `nisls`(8), `nistbladm`(8)

`nis_objects`(3N) in *ONC+ Technology for the UNICOS Operating System*, Cray Research publication SG–2169

NAME

nismkdir – Creates NIS+ subdirectories

SYNOPSIS

nismkdir [-D *defaults*] [-m *hostname*] *dirname*

nismkdir [-D *defaults*] [-s *hostname*] *dirname*

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nismkdir command creates new NIS+ subdirectories within an existing domain. It has the option of creating replicated or nonreplicated directories.

The two primary aspects that are controlled when making a directory are its access rights and its degree of replication. Without options, this command will create a subdirectory with the same master and replicas as the parent directory.

The nismkdir command accepts the following options, tokens, and operands:

-D *defaults* Specifies to use a different set of defaults when creating new directories. The *defaults* string is a series of tokens separated by colons. These tokens represent the default values that will be used for the generic object properties. All of the legal tokens are as follows:

ttl=*time*
: Sets the default time-to-live for objects that are created by this command. The value of *time* is specified in the format as defined by the nischttl(8) command. The default value is 12 hours.

owner=*ownername*
: Specifies that the NIS+ principal *ownername* should own the created object. The default for this value is the principal who is executing the command.

group=*groupname*
: Specifies that the group *groupname* should be the group owner for the object that is created. The default value is NULL.

access=*rights*
: Specifies the set of access rights that are to be granted for the given object. The value of *rights* is specified in the format as defined by the nischmod(8) command. The default value is ----rmcdr--r---.

-m *hostname* If the directory specified by *dirname* does not exist, then a new directory that is not replicated is created with host *hostname* as its master server. If the directory name specified by *dirname* does exist, then the host specified by *hostname* is made its master server.

`-s` *hostname* Specifies that the host *hostname* will be a replica for the existing directory that is specified by *dirname*.

dirname Specifies the fully qualified NIS+ name of the directory that will be created.

NOTES

A host that serves an NIS+ directory must be an NIS+ client in a directory above the one it is serving. Root NIS+ servers, which are both clients and servers of the same NIS+ directory, are exceptions to this rule.

When the host's default domain is different from the default domain on the client where the command is executed, the hostname supplied as an argument to the `-s` or `-m` options must be fully qualified.

EXIT STATUS

The `nismkdir` command exits with one of the following values:

Value	Description
0	Success.
1	Failure; an error occurred.

ENVIRONMENT VARIABLES

Following are environment variables that affect the defaults associated with a process.

`NIS_DEFAULTS` This variable contains a defaults string that overrides the NIS+ standard defaults. If the `-D` option is used those values override both the `NIS_DEFAULTS` variable and the standard defaults.

`NIS_PATH` If this variable is set and the NIS+ directory name is not fully qualified, each directory specified is searched until the directory is found. See `nisdefaults`(8) for more information.

SEE ALSO

`nischmod`(8), `nischttl`(8), `nisdefaults`(8), `nisls`(8), `nisrmdir`(8)

NAME

nispasswd – Changes NIS+ password information

SYNOPSIS

nispasswd [-g] [-h] [-s] [-D *domainname*] [*username*]

nispasswd -a

nispasswd [-D *domainname*] [-d [*username*]]

nispasswd [-l] [-f] [-n *min*] [-x *max*] [-w *warn*] [-D *domainname*] *username*

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nispasswd command changes a password, gecos field, home directory, or login shell associated with a user name in the NIS+ passwd table.

The command is also used to view or modify aging information associated with the user specified if the invoker has the right NIS+ privileges.

The nispasswd command accepts the following options:

-g Changes the gecos field (finger(1)) information.

-h Changes the home directory.

-s Changes the login shell.

-D *domainname*
Consults the passwd.org_dir table in *domainname*. If this option is not specified, the default *domainname* returned by nis_local_directory() is used. This *domainname* is the same as that returned by domainname(1).

-a Shows the password attributes for all entries. This will show only the entries in the NIS+ passwd table in the local domain that the invoker is authorized to read.

-d *username*
Displays password attributes for the caller or the user specified if the invoker has the right privileges.

-l Locks the password entry for the user specified by *username*. Subsequent attempts to log in by using login(1) and this NIS+ password entry would not be allowed.

-f Forces users to change their password at the next login by expiring the password for *username*.

`-n` *min* Sets the minimum field for *username*. The *min* field contains the minimum number of days between password changes for *username*. If *min* is greater than *max*, the user may not change the password. Always use this option with the `-x` option, unless *max* is set to –1 (aging turned off). In that case, *min* need not be set.

`-x` *max* Sets the maximum field for *username*. The *max* field contains the number of days that the password is valid for *username*. The aging for *username* will be turned off immediately if *max* is set to –1. If it is set to 0, users are forced to change their password at the next login session, and aging is turned off.

`-w` *warn* Sets the warning field for *username*. The warning field contains the number of days before the password expires that users will be warned whenever they attempt to login.

Users may also use the `-n`, `-x`, and `-w` options to set or modify the minimum, maximum and warning attributes for their password information.

Users may also use the `-d` option to display password attributes for their login name. The format of the display is as follows:

username status mm/dd/yy min max warn

If password aging information is not present the format is as follows:

username status

The *username* specifies the login ID of the user.

The following variables are displayed:

Variable	**Description**
status	Displays the password status of *username*. `PS` stands for password exists or locked, `LK` stands for locked, and `NP` stands for no password.
mm/dd/yy	Displays the date the password was last changed for *username*. All password aging dates are determined using Greenwich Mean Time and, therefore, may differ by as much as a day in other time zones.
min	Displays the minimum number of days required between password changes for *username*.
max	Displays the maximum number of days the password is valid for *username*.
warn	Displays the number of days relative to *max* before the password expires and the user will be warned.

The `nispasswd` command uses Secure RPC to communicate with the NIS+ server, and never sends unencrypted passwords over the communication medium.

The `nispasswd` command does not read or modify the local password information stored in the `/etc/passwd` file.

When used to change a password, `nispasswd` prompts nonprivileged users for their old password. It then prompts for the new password twice to forestall typing mistakes. When the old password is entered, `nispasswd` checks to see if it has aged sufficiently. If aging is insufficient, `nispasswd` terminates.

The old password is used to decrypt the *username* secret key. If the password does not decrypt the secret key, `nispasswd` prompts for the old secure-RPC password. It uses this password to decrypt the secret key. If this fails, it gives the user one more chance, and if the user input fails again, `nispasswd` terminates. The old password also is used to ensure that the new password differs from the old by at least 3 characters.

Assuming that password aging is sufficient, a check is made to ensure that the new password meets specific construction requirements as described in the following. When the new password is entered a second time, the two copies of the new password are compared. If the two copies are not identical, the cycle of prompting for the new password is repeated twice. The new password is used to re-encrypt the user's secret key. Hence, it also becomes the user's secure-RPC password.

Passwords must meet the following requirements:

- Each password must have at least 6 characters. Only the first 8 characters are significant.
- Each password must contain at least 2 alphabetic characters and at least 1 numeric or special character. Alphabetic characters include all uppercase and lowercase letters.
- Each password must differ from the user's login *username* and any reverse or circular shift of that login *username*. For comparison purposes, an upper case letter and its corresponding lowercase letter are equivalent.
- New passwords must differ from the old by at least 3 characters. For comparison purposes, an uppercase letter and its corresponding lowercase letter are equivalent.

Network administrators own the NIS+ password table. They may change any password attributes if they establish their credentials before invoking `nispasswd`. The `nispasswd` command does not prompt privileged users for the old password and they are not forced to comply with password aging and password construction requirements. See `keylogin`(1) for more information.

NOTES

The login program, file access display programs such as `ls -l` and network programs such as `rlogin`(1B) and `ftp`(1B)) that require user passwords use the standard `getpwent`(3C) interface to get password information. These programs get the NIS+ password information, which is modified by `nispasswd`, only if the `passwd`: entry in the `/etc/nsswitch.conf` file refers to `nisplus`. See `nsswitch`(4) for more information.

EXIT STATUS

The `nispasswd` command exits with one of the following values and messages:

Value	Messages
0	`SUCCESS.`
1	Permission denied.

2 Invalid combination of options.

3 Unexpected failure. NIS+ `passwd` table unchanged.

4 NIS+ `passwd` table missing.

5 NIS+ is busy. Try again later.

6 Invalid argument to option.

7 Aging is disabled.

SEE ALSO

`nistbladm`(8)

`domainname`(1), `finger`(1B), `ftp`(1B), `keylogin`(1), `login`(1), `passwd`(1), `rlogin`(1B) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`getpwent`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`nsswitch`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

nispath – Prints the search path of a specified NIS+ name

SYNOPSIS

nispath [-v] *name*

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nispath command displays the list of candidate names for the name specified in *name*. If this name is not fully qualified, nispath generates a list of names using the default NIS+ directory search path, or the environment variable NIS_PATH if it is set.

The nispath command accepts the following options:

-v Displays a more detailed message about the search path.

name The specified name for your search.

ENVIRONMENT VARIABLES

Following is an environment variable that affects the defaults associated with a process.

NIS_PATH If this variable is set and the NIS+ name is not fully qualified, each directory specified is searched until the object is found. This variable contains an ordered list of directories separated by colon (:) symbols.

NAME

nisping – Sends a ping to NIS+ servers

SYNOPSIS

nisping [-u] [-f] [-H *hostname*] [-r]

nisping [-u] [-f] [-H *hostname*] [*directory*]

nisping -C [-H *hostname*] [*directory*]

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nisping command sends a ping to all replicas of an NIS+ directory. Once a replica receives a ping, it will check with the master server for the directory to get updates. Prior to pinging the replicas, this command attempts to determine the last update received by a replica and the last update logged by the master. If these two time stamps are the same, the ping is not sent. The -f option overrides this feature.

Usually, NIS+ replica servers get new information from the master NIS+ server within a short time. Therefore, there should not be any need to use the nisping command.

The nisping command with the -C option is used to send a checkpoint request to the servers. Upon receiving a checkpoint request, the servers commit all the updates from the table log files to the database files. If they are out of date, updates to replicas are sent. This option is necessary because the database log files for NIS+ are not automatically checkpointed. nisping should be used at frequent intervals to checkpoint the NIS+ database log files. It is recommended that the nisping command with the -C option be used at least once a day. This command may be added to a crontab(1) file. If the database log files are not checkpointed, their sizes will continue to grow.

The nisping command accepts the following options and operands:

-u Displays the time of the last update; no servers are sent a ping.

-f Forces a ping, even though the time stamps indicate there is no reason to do so. This option is useful for debugging.

-H *hostname* Sends the ping, checks for an update time, or checkpoints only the host *hostname*.

-r Retrieves status about the root object from the root servers, especially when new root replicas are added or deleted from the list. If used without the -u option, the -r option is used to send a ping request to the servers serving the root domain. When the replicas receive a ping, they update their root object if needed. The -r option can be used with all other options except with the -C option; the root object does not need be checkpointed.

directory Specifies the directory that is checkpointed or sent a ping.

`-C` Sends a request to checkpoint, rather than a ping, to each server. The servers schedule to commit all the transactions to stable storage.

NOTES

If the server specified by the `-H` option does not serve the directory, then no ping is sent.

ENVIRONMENT VARIABLES

Following is an environment variable that affects the defaults associated with a process.

`NIS_PATH` If this variable is set and the NIS+ directory name is not fully qualified, each directory specified is searched until the directory is found.

EXIT STATUS

The `nisping` command exits with one of the following values:

Value	Description
–1	No servers were contacted, or the server specified by the `-H` switch could not be contacted.
0	Success.
1	Some, but not all, servers were successfully contacted.

SEE ALSO

`nislog`(8)

`crontab`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`nis_admin`(3N) in *ONC+ Technology for the UNICOS Operating System*, Cray Research publication SG–2169

`nisfiles`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

nisrm – Removes NIS+ objects from the namespace

SYNOPSIS

```
nisrm [-i] name...
nisrm [-f] name...
```

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nisrm command removes specified NIS+ objects from the NIS+ namespace.

The nisrm command accepts the following options and operands:

-i Interactive mode. Removes NIS+ objects with request and confirmation interaction. Similar to the rm(1) command, the nisrm command asks for confirmation prior to removing an object. If the name specified by *name* is not a fully qualified name this option is forced on. This prevents the removal of unexpected objects.

-f Forces the removal of the NIS+ object, and if it fails for permission reasons, a nischmod(8) is attempted and the removal retried. If the command fails, it fails silently.

NOTES

This command does not remove directories or nonempty tables. See nisrmdir(8) and nistbladm(8) for more information.

ENVIRONMENT VARIABLES

Following is an environment variable that affects the defaults associated with a process.

NIS_PATH If this variable is set and the NIS+ name is not fully qualified, each directory specified is searched until the object is found. See nisdefaults(8) for more information.

EXIT STATUS

The nisrm command exits with one of the following values:

Value	Description
0	Success.
1	Failure; an error occurred.

SEE ALSO

nischmod(8), nisdefaults(8), nisrmdir(8), nistbladm(8)

rm(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

NAME

`nisrmdir` – Removes NIS+ directories

SYNOPSIS

`nisrmdir` [`-f`] [`-i`] [`-s` *hostname*] *dirname*

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The `nisrmdir` command deletes existing NIS+ directories. It can remove a directory outright, or simply remove replicas from serving a directory.

This command modifies the object that describes the directory specified by *dirname*, and then notifies each replica to remove the directory named *dirname*. If the notification of any of the affected replicas fails, the directory object is returned to its original state unless the `-f` option is present.

The `nisrmdir` command accepts the following options and operands:

`-f` Forces the command to succeed, even though it may not be able to contact the affected replicas. This option is used when a replica is down and does not respond to the removal notification. When the replica is rebooted, it will read the updated directory object (it is no longer a replica for that directory), and stop responding to lookups on that directory. Cleanup of the files that held the now removed directory can be accomplished manually by removing the appropriate files in the `/etc/nis` directory. See `nisfiles`(4) for more information.

`-i` Specifies interactive mode. Asks for confirmation prior to removing a directory specified by *dirname*.

`-s` *hostname* Specifies that the host *hostname* should be removed as a replica for the directory specified *dirname*. If this option is not present all replicas and the master server for a directory are removed and the directory is removed from the namespace.

dirname Specifies the directory name.

ENVIRONMENT VARIABLES

Following is an environment variable that affects the defaults associated with a process.

`NIS_PATH` If this variable is set and the NIS+ directory name is not fully qualified, each directory specified is searched until the directory is found. See `nisdefaults`(8) for more information.

EXIT STATUS

The `nisrmdir` command exits with one of the following values:

Value	Description
0	Success.
1	Failure; an error occurred.

SEE ALSO

`nisdefaults`(8), `nisrm`(8)

`nisfiles`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

nissetup – Initializes an NIS+ domain

SYNOPSIS

nissetup *domain*

nissetup [-Y] *domain*

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nissetup command sets up an NIS+ domain to service clients that want to store system administration information in a domain named *domain*. This domain should already exist prior to executing this command. See nismkdir(8) and nisinit(8) for more information.

An NIS+ domain consists of an NIS+ directory and its subdirectories: org_dir and groups_dir. org_dir stores system administration information and groups_dir stores information for group access control. The nissetup command creates the subdirectories org_dir and groups_dir in the local domain. Both subdirectories will be replicated on the same servers as the parent domain. After the subdirectories are created, nissetup creates the default tables that NIS+ serves. These are auto_master, auto_home, bootparams, cred, ethers, group, hosts, netmasks, networks, passwd, protocols, rpc, services, timezone, and netgroups.

The nissetup command uses the nistbladm(8) command to create these tables. The command can be customized easily to add site-specific tables that should be created at setup time. This command is usually executed just once per domain.

The nissetup command accepts the following option and operand:

domain Specifies the name of an NIS+ domain.

-Y *domain* Specifies that the domain will be served as both an NIS+ domain as well as an NIS domain using the backward compatibility flag. This set up is less secure because all the system tables will be readable by unauthenticated clients.

NOTES

This command creates the default NIS+ tables without data. The nisaddent(8) command is used to populate NIS+ tables with data.

SEE ALSO

nisaddent(8), nisinit(8), nismkdir(8), nistbladm(8)

NAME

nisshowcache – Prints the contents of the shared cache file

SYNOPSIS

nisshowcache [-v]

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nisshowcache command prints the contents of the per-machine NIS+ directory cache that is shared by all processes accessing NIS+ on the machine. By default, nisshowcache prints the directory names in the cache along with the cache header. The shared cache is maintained by nis_cachemgr(8).

The nisshowcache command accepts the following option:

-v Prints the contents of each directory object, including information on the server name and its universal addresses in verbose mode.

MESSAGES

Error messages are sent to the syslogd(8) daemon.

FILES

/etc/nis/NIS_SHARED_DIRCACHE

SEE ALSO

nis_cachemgr(8), syslogd(8)

nisfiles(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

nisstat – Reports NIS+ server statistics

SYNOPSIS

nisstat [-H *host*] [*directory*]

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nisstat command queries an NIS+ server for various statistics on its operation. These statistics may vary between implementations and from release to release. Not all statistics are available from all servers. Requesting a statistic from a server that does not support that statistic is never fatal, it simply returns the message unknown statistic.

By default, statistics are fetched from the server(s) of the NIS+ directory for the default domain. If the *directory* variable is specified, servers for that directory are queried.

Supported statistics for this release are as follows:

Operations This statistic returns results in the form:

OP=*opname*:C=*calls*:E=*errors*:T=*micros*

Where *opname* is replaced by the Remote Procedure Call (RPC) procedure name or operation, *calls* is the number of calls to this procedure that have been made since the server started running. *errors* is the number of errors that have occurred while processing a call, and *micros* is the average time in microseconds to complete the last 16 calls.

Directory Cache This statistic reports the number of *calls* to the internal directory object cache, the number of hits on that cache, the number of misses, and the hit rate percentage.

Group Cache This statistic reports the number of *calls* to the internal NIS+ group object cache, the number of hits on that cache, the number of misses, and the hit rate percentage.

Uptime This statistic reports how long the service has been running.

The nisstat command accepts the following option and operands:

-H *host* Queries a specified *host*.

directory Specifies a directory. If *directory* is specified, servers for that directory are queried.

ENVIRONMENT VARIABLES

Following is an environment variable that affects the defaults associated with a process.

`NIS_PATH` If this variable is set, and the NIS+ directory name is not fully qualified, each directory specified is searched until the directory is found. See `nisdefaults`(8) for more information.

EXIT STATUS

The `nisstat` command exits with one of the following values:

Value	Description
0	Success.
1	Failure; an error occurred.

SEE ALSO

`nisdefaults`(8)

NAME

`nistbladm` – Administers NIS+ tables

SYNOPSIS

`nistbladm -a` [`-D` *defaults*] *colname=value...* *tablename*
`nistbladm -A` [`-D` *defaults*] *colname=value...* *tablename*

`nistbladm -a` [`-D` *defaults*] *indexedname*
`nistbladm -A` [`-D` *defaults*] *indexedname*

`nistbladm -c` [`-D` *defaults*] [`-p` *path*] [`-s` *sep*] *type* *colname*=[*flags*][,*access*]... *tablename*

`nistbladm -d` *tablename*

`nistbladm -m` *colname=value...* *indexedname*

`nistbladm -r` [*colname=value...*] *tablename*
`nistbladm -R` [*colname=value...*] *tablename*

`nistbladm -r` *indexedname*
`nistbladm -R` *indexedname*

`nistbladm -u` [`-p` *path*] [`-s` *sep*] [`-t` *type*] [*colname=access...*] *tablename*

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The `nistbladm` command is used to administer NIS+ tables. There are five primary administration operations that `nistabladm` performs:

- Creates and deletes NIS+ tables
- Deletes NIS+ tables
- Adds entries to NIS+ tables
- Modifies NIS+ table entries
- Removes entries from NIS+ tables

Though NIS+ does not place restrictions on the size of tables or entries, the size of data has an impact on the performance and the disk space requirements of the NIS+ server. NIS+ is not designed to store huge data files. To avoid performance problems, it is recommended that pointer files to large data files be stored in NIS+. NIS+ is a naming service and not a directory service.

NIS+ design is optimized to support 10,000 objects with a total size of 10 megabytes. If your requirements exceed this limit, it is suggested that your domains be broken up into small segments, or the data stored in the tables be pointers to the actual data, instead of the data itself.

When creating tables, a table type, *type*, and a list of column definitions must be provided. *type* is a string, stored in the NIS+ table, which is used to verify the type of new entries that are being added to that table.

The syntax for column definitions is:

colname=[*flags*][,*access*]

For the variable *flags*, specify one or more of the following options:

S Specifies that searches can be done on the column's values. See `nismatch`(8) for more information.

I Specifies that searches are case insensitive and to search for uppercase and lowercase. This flag is used with the `S` flag).

C Specifies that the column's values are encrypted.

B Binary data. This flag is not used in combination with the `S` flag. If not set, the column's values are expected to be null-terminated ASCII strings.

X External Data Representation (XDR) encoded data. This flag is used in combination with the `B` flag.

The variable *access* is specified in the format defined by the `nischmod`(8) command. When manipulating entries, `nistbladm` takes two forms of entry name. The first uses a series of space separated *colname=value* pairs that specify column values in the entry. The second is an NIS+ indexed name, *indexedname*, of the form:

[*colname=value...*],*tablename*

The `nistbladm` command accepts the following options and operands:

`-a | -A` Adds entries to an NIS+ table. The difference between the lowercase `-a` and the uppercase `-A` options is in the treatment of preexisting entries. The entry's contents are specified by the *colname=value* pairs on the command line. The values for all columns must be specified when adding entries to a table.

Usually, NIS+ reports an error if an attempt is made to add an entry to a table that would overwrite an entry that already exists. This prevents multiple parties from adding duplicate entries and overwriting entries. If you wish to force the add, the uppercase `-A` option specifies that the entry is to be added, even if it already exists. This is analogous to a modify operation on the entry.

`-c` Creates a table named *tablename* in the namespace. The table that is created must have at least one column and at least one column must be searchable.

`-d` Destroys the table named *tablename*. The table must be empty. The table's contents can be deleted by using the `-R` option.

`-m` Modifies an entry in the table that is specified by *indexedname*. Since it is possible to modify the value in a column that would change *indexedname* for an entry, both the column value pair and *indexedname* are required. The user must specify the *indexedname* to look up the entry, modify it, and save it with the new value. The *indexedname* must uniquely identify a single entry.

`-r | -R` Remove entries from a table. The entry is specified by either a series of *colname=value* pairs on the command line, or an *indexedname* that is specified as entryname. The difference between the lowercase `-r` and the uppercase `-R` option is in the treatment of nonunique entry specifications. Usually, the NIS+ server disallows an attempt to remove an entry when the search criterion specified for that entry resolves to more than one entry in the table. However, it is sometimes desirable to remove more than one entry, or all of the entries from a table. In this case, using the uppercase `-R` option forces the NIS+ server to remove all entries matching the passed search criterion. If that criterion is `NULL` and no column values are specified, all entries in the table are removed.

`-u` Updates attributes of a table. This allows the concatenation path (`-p`), separation character (`-s`), column access rights, and table type string (`-t`) of a table to be changed. Neither the number of columns, nor the columns that are searchable may be changed.

`-D` *defaults* Specifies a different set of defaults to be used during this operation. The *default* string is a series of tokens separated by colons. These tokens represent the default values to be used for the generic object properties. Following are descriptions of all the legal tokens.

`ttl=`*time*
: Sets the default time-to-live for objects that are created by this command. The value *time* is specified in the format as defined by the `nischttl`(8) command. The default value is 12 hours.

`owner=`*ownername*
: Specifies that the NIS+ principal *ownername* should own the created object. The default value is the principal who is executing the command.

`group=`*groupname*
: Specifies that the group *groupname* should be the group owner for the object that is created. The default value is `NULL`.

`access=`*rights*
: Specifies the set of access rights that are to be granted for the given object. The value of *rights* is specified in the format as defined by the `nischmod`(8) command. The default value is `----rmcdr--r---.`

`-p` *path* Specifies the table search path to use when creating or updating a table. When an `nis_list` function is invoked, the user can specify the flag `FOLLOW_PATH` to tell the client library to continue searching tables in the table's path if the search criteria used does not yield any entries. The *path* argument consists of an ordered list of table names, separated by colons (`:`). The names in *path* must be fully qualified.

`-s` *sep* Specifies the table's separator character. The separator character is used by `niscat`(8) when displaying tables on the standard output. Its purpose is to separate column data when the table is in ASCII form. The default value is a space.

`-t` *type* Specifies the table's type string.

colname=value The column entry value.

indexedname A specified entry name.

tablename The name of an NIS+ table.

WARNINGS

The symbols open bracket ([) and close bracket (]) are interpreted by the shell. When typing entry names in the form of NIS+ indexed names, the name must be protected by using single quotes. It is possible to specify a set of defaults such that you cannot read or modify the table object later.

ENVIRONMENT VARIABLES

Following are environment variables that affect the defaults associated with a process.

`NIS_DEFAULTS` This variable contains a defaults string that overrides the NIS+ standard defaults. If the `-D` option is used, those values override both the `NIS_DEFAULTS` variable and the standard defaults.

`NIS_PATH` If this variable is set and the NIS+ table name is not fully qualified, each directory specified is searched until the table is found. See `nisdefaults`(8) for more information.

EXIT STATUS

The `nistbladm` command exits with one of the following values:

Value	Description
0	Success.
1	Failure; an error occurred.

SEE ALSO

`niscat`(8), `nischmod`(8), `nischown`(8), `nischttl`(8), `nisdefaults`(8), `nismatch`(8), `nissetup`(8)

`nis`(4) in the *ONC+ Technology for the UNICOS Operating System*, Cray Research publication SG–2169

NAME

nistest – Returns the state of the NIS+ namespace

SYNOPSIS

nistest [-A] [-t *type*] [-a *rights*]

nistest [-L] [-t *type*] [-a *rights*]

nistest [-M] [-t *type*] [-a *rights*]

nistest [-P] [-t *type*] [-a *rights*]

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nistest command returns the state of the NIS+ namespace.

The nistest command accepts the following options:

-A Specifies that the data within the table and in all the tables in the first table's concatenation path is returned. This option is only valid when using indexed names or following links.

-L Specifies that if the object or the tablename component of an *indexedname* specifies a LINK type object, the link is followed when this option is present.

-M Specifies that the lookup should be sent only to the master server of the specified data. Although this guarantees that the most up-to-date information is seen it may make the master server busy and therefore unavailable.

-P Specifies that the lookup should follow the concatenation path of a table if the initial search is unsuccessful. This option is only valid when using indexed names or following links.

-t *type* Tests the type of object. The value of *type* can be one of the following:

- G Return true if the object is a group object.
- D Return true if the object is a directory object.
- L Return true if the object is a table object.
- T Return true if the object is a link object.
- E Return true if the object is an entry object.
- P Return true if the object is a private object.

-a *rights* Verifies that the current process has the desired or required access rights on the specified object or entries. The access rights are specified in the same way as the nischmod(8) command. See nischmod(8) for more information.

ENVIRONMENT VARIABLES

Following is an environment variable that affects the defaults associated with a process.

`NIS_PATH` If this variable is set, and the NIS+ directory name is not fully qualified, each directory specified is searched until the directory is found. See `nisdefaults`(8) for more information.

EXIT STATUS

The `nistest` command exits with one of the following values:

Value	Description
0	Success.
1	Failure due to object not present, not of specified type, and/or no such access.
2	Failure due to illegal usage.

SEE ALSO

`nischmod`(8), `nisdefaults`(8), `nismatch`(8)

NAME

nisupdkeys – Updates the public keys in an NIS+ directory object

SYNOPSIS

```
nisupdkeys [-a] [-H host] [directory]
nisupdkeys [-C] [-H host] [directory]
```

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The nisupdkeys command updates the public keys in an NIS+ directory object. When the public key for an NIS+ server is changed, the new key must be propagated to all directory objects that reference that server.

The nisupdkeys command reads a directory object and attempts to get the public key for each server of that directory. These keys are placed in the directory object and the object is then modified to reflect the new keys.

If *directory* is specified, the directory object for that directory is updated. If *directory* is not specified, the directory object for the default domain is updated.

Before the nisupdkeys command is executed, make sure that the new address/public key has been propagated to all replicas.

The nisupdkeys command accepts the following options and operands:

-a Updates the universal addresses of the NIS+ servers in the directory object. Currently, this works for the Transmission Control Protocol, Internet Protocol (TCP/IP) family of transports. This option should be used when the IP address of the server is changed. The server's new address is resolved using gethost(3C). The /etc/nsswitch.conf file must point to the correct source for the host's entry.

-C Specifies to clear rather than set the public key. Communication with a server that has no public key does not require the use of Secure RPC.

-H *host* Limits key changes to the server specified by *host*. If the host name is not a fully qualified NIS+ name, then it is assumed to be a host in the default domain. If the specified host does not serve the directory, no action is taken.

directory Specified NIS+ directory object.

NOTES

The user executing this command must have modify access to the directory object. The existing directory object is displayed by using the niscat(8) command with the -o option.

This command does not update the directory objects stored in the COLD_START_FILE file on NIS+ clients.

SEE ALSO

nisaddcred(8), niscat(8)

chkey(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

gethost(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

nis_objects(3N) in *ONC+ Technology for the UNICOS Operating System*, Cray Research publication SG–2169

NAME

npecho – Utility to loop data to the IOS model E low-speed network driver

SYNOPSIS

/etc/npecho [-D] [-n *count*] [-t] [-z *device*]

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

DESCRIPTION

The npecho utility sends "echo" N-packets to the IOS-E communications driver, receives the replies, and prints timing information if desired. The npecho utility accepts the following options:

-D Debug flag. Prints debugging statements.

-n *count* Number of echo requests to make. Default is 1.

-t Prints the average time for an echo packet to be returned.

-z *device* Device special file for npecho to communicate with the np driver. The default is /dev/comm/0030/lp00.

NOTES

In a PAL-only system where this command is installed with a privilege assignment list (PAL), a user who runs this command must have one of the following categories active to open and manipulate the network driver: system, secadm, sysadm. In a PRIV_SU + PAL or PRIV_SU-only system, the super user can also open and manipulate the network driver.

SEE ALSO

np(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

nqsresp – Attempts to open the named pipe that nqsdaemon is reading

SYNOPSIS

/usr/air/bin/nqsresp

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nqsresp command attempts to open the named pipe that the nqsdaemon process reads. If the open function fails, either the local Network Queuing System (NQS) daemon is not running or the local NQS daemon is attempting to shutdown and has not completed. In the shutdown state, group and world write access on the pipe are disabled; however, nqsresp does not recognize this as a temporary situation and indicates only that it was unable to open the pipe.

FILES

/usr/spool/nqs/private/requests/FIFO The nqsdaemon pipe

SEE ALSO

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

nschedv – Sets memory scheduling parameters

SYNOPSIS

/etc/nschedv [-B *thrash_blocks*] [-C *compress_interval*] [-c *cpuhog*] [-d] [-f *fit_boost*] [-G *guarantee_in*] [-g *guarantee_out*] [-H *hog_max_mem*] [-h *memhog*] [-i *interactive_preferred*] [-K *constant_in*] [-k *constant_out*] [-L *big_proc*] [-M *mfactor_in*] [-m *mfactor_out*] [-N *nfactor_in*] [-n *nfactor_out*] [-P *pfactor_in*] [-p *pfactor_out*] [-R *thrash_interval*] [-r *cpu_factor*] [-T *tfactor_in*] [-t *tfactor_out*] [-V *max_runs*] [-v] [-X *MPX_scheduling*] [-x *max_outage*] [-Y *interactive_interval*] [-y *small_proc*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The nschedv command sets and displays memory scheduling parameters using the schedv(2) system call. Only an appropriately authorized user can set parameters.

The nschedv command accepts the following options:

-B *thrash_blocks* Specifies the number of blocks as the swap threshold. If this value is exceeded within a specified time period, the scheduler attempts to limit swapping to SWAPDEV. You must use -B with -R, which sets the interval in which *thrash_blocks* blocks can be swapped out. Swapping caused by user memory requests is not counted towards the total limit set by -B. The default for this option is 0, which specifies no limit (no swap threshold is enforced).

-C *compress_interval*
Sets the compression interval, which is the minimum time between memory repacks caused by memory fragmentation. (Memory repacks are also done to move processes into low core as the result of a plock system call, or when real-time processes are swapped in; the *compress_int* parameter has no effect in these cases.) Never set -C to 0; this could cause severe performance problems in some circumstances by allowing the swapper to repack memory continuously. The default for -C is in the range from 4 to 30, based on the number of megawords of memory.

-c *cpuhog* Specifies threshold, in CPU seconds, at which a process becomes a hog process and therefore comes under hog restrictions. System processes (such as init), processes spawned by root, and plocked processes are never considered CPU hogs. The -h option (*memhog*) defines the size of a hog process.

-d Displays all parameters. This option should be used in conjunction with other options. If no options are specified with nschedv, this is the default behavior.

`-f` *fit_boost* Specifies a negative floating-point value that is added to the priority of processes being considered for swap-out if they are equal to or greater in size than the process attempting to swap in. The default for this option is –2.0.

`-G` *guarantee_in* Specifies a comma-separated pair of floating-point values. The first value is the minimum time a process is guaranteed residency in memory. The second value is a size-proportional value that is calculated based on the normalized size of the process and the `-G` parameter value. The normalized memory size is a value between 0 and 1, based on the size of the process as a percentage of *hog_max_mem* (if used) or of the memory available to user processes if *hog_max_mem* is 0. The *guarantee_in* value is multiplied by the normalized memory size and added to the guaranteed residency time for the process. (For sample values of `-G`, see the EXAMPLES section.)

`-g` *guarantee_out* Specifies a comma-separated pair of floating-point values. The first value is the minimum time a process is guaranteed residency on `SWAPDEV`. The second value is a size-proportional value that is calculated based on the normalized size of the process and the `-g` parameter value. The normalized memory size is a value between 0 and 1, based on the size of the process as a percentage of *hog_max_mem* (if used) or of the memory available to user processes if *hog_max_mem* is 0. The *guarantee_out* value is multiplied by the normalized memory size and added to the guaranteed residency time for the process. (For sample values of `-g`, see the EXAMPLES section.)

`-H` *hog_max_mem* Specifies an integer value that is the total memory in 512-word blocks (also called *clicks*) that is available for processes greater than or equal in size to *memhog*, or that have used more CPU time than *cpuhog*. Typical values for *hog_memory* are 60% to 90% of a machine's memory available to user processes, which reserves 10% to 40% for processes that are less than the *memhog* and *cpuhog* definitions. If 0, the hog feature is disabled, which specifies that no space is available for hog processes.

`-h` *memhog* Specifies the size in clicks at which a process becomes a *hog* process; a process of this size and greater comes under the hog restrictions. System processes (such as `init`) and plocked processes are never considered memory hogs.

`-i` *interactive_preferred*

If *interactive_preferred* is 0, this option is disabled. If *interactive_preferred* is 1, it indicates that nonhog interactive processes should be queued for swap-in before noninteractive processes of the same category. For example, hog interactive processes are queued before hog batch processes, but not before nonhog batch processes. The following category values are defined and used by the scheduler:

0 Interactive kernel process

1 Batch kernel process

2 Interactive realtime process

3 Batch realtime process

4 Interactive root process

5 Batch root process

6 Interactive nonhog process

7 Batch nonhog process

8 Interactive hog process

9 Batch hog process

`-K` *constant_in* Specifies a constant floating-point value that is added to the priority of an incore process being considered for swap-out.

`-k` *constant_out* Specifies a constant floating-point value that is added to the priority of a process on `SWAPDEV` being considered for swap-in.

`-L` *big_proc* Specifies the size, in clicks, at which a process is locked into core. If a process is greater than or equal to `bigproc` in size, it never swaps from memory to the `SWAPDEV`, unless it makes a memory request or is suspended. If this parameter is set to values less than or equal to one half of memory available to user processes (if *hog_memory* is set to memory available to user processes), the system may lock until one job completes. A value of 0 disables this behavior.

`-M` *mfactor_in* Specifies a positive floating-point value that defines a process' incore priority based on the memory size of that process. The calculation of incore priority, which must be a positive floating-point value, is based on *hog_memory*, if set (with the `-H` option), or the memory available to user processes if *hog_memory* is 0. The system derives this portion of the incore priority using the following equation:

float(*process size*)/float(*hog_memory*)***mfactor_in*

`-m` *mfactor_out* Specifies a negative floating-point value that defines the swap priority (the order in which processes move onto the swap device (`SWAPDEV`)), based on the memory size of the process. The calculation of swap priority, which must be a negative floating point value, is based on *hog_memory*, if set (with the `-H` option), or on the memory available to user processes if *hog_memory* is set to 0. The system derives this portion of the swap priority using the following equation:

float`(`*process size*`)/`float`(`*hog_memory*`)*`*mfactor_out*

`-N` *nfactor_in* Specifies a positive floating-point value that provides a process that swaps into memory a priority based on its nice value. The system derives this portion of the incore priority using the following equation (*p_nice* is the nice value of the process):

`(1.0 - (`float`(`*p_nice*`)/`float`(NZERO*2))*`*nfactor_in*

`-n` *nfactor_out* Specifies a negative floating-point value that provides a process that swaps to `SWAPDEV` a priority based on its nice value. The system derives this portion of the swap priority using the following equation (*p_nice* is the nice value of the process):

(float(*p_nice*)/float(NZERO*2))**nfactor_out*

-P *pfactor_in* Specifies a positive floating-point value that is part of the priority calculation when a process swaps into memory. Each process is given a percentage of this value as the process relates to the maximum priority on the system plus 1.0. The system derives the process priority using *pfactor_in* and the priority of the process swapping into memory (*p_upri*) in the following equation:

(1.0-(float(*p_upri*)/float(MAXPRI+1)))**pfactor_in*

-p *pfactor_out* Specifies a negative floating point value that is part of the priority calculation when a process swaps onto SWAPDEV. Each process is given a percentage of this value as the process relates to the maximum priority on the system plus 1.0. The system derives the process priority using *pfactor_out* and the priority of the process swapping into memory (*p_upri*) in the following equation:

(float(*p_upri*)/float(MAXPRI+1))**pfactor_out*

-R *thrash_interval*

Specifies an integer value that is the interval of time, in seconds, in which *thrash_blocks* (set with -B) blocks can be swapped to SWAPDEV. The default is 0, which disables this option.

-r *cpu_factor* Specifies an integer value that is the number of running processes that the scheduler attempts to retain in memory. When the scheduler has this number of processes in memory under certain conditions, it does not swap any more processes out of memory. If the value is set too low for your site, idle CPU cycles may result and large processes (that take a large percentage of memory) may swap into memory. This parameter must never be set to 0.

-T *tfactor_in* Specifies a floating-point value less than or equal to 0 that is added to the priority of an incore process each second. If both *tfactor_in* and *tfactor_out* (set with -t) are set to 0, no priority aging or growth occurs.

-t *tfactor_out* Specifies a floating-point value greater than or equal to 0 that is added to the priority of a process on SWAPDEV each second. If both *tfactor_in* and *tfactor_out* (set with -t) are set to 0, no priority aging or growth occurs.

-V *max_runs* Specifies the maximum number of times the scheduler is allowed to run per second. The default is 0, which allows the scheduler to run an unlimited number of times per second.

`-v` Verification option. Computes and displays swap-in and swap-out priorities for all combinations of priority-based parameters using current `nschedv` settings. The combinations are determined by several tables of values in the `nschedv` program (`pritab`, `upritab`, `nictab`, and `siztab`); if your site has source, you may change these tables to suit your system's environment. You can use the `-v` option to check the range of priorities for the current settings, as well as to compare the priorities of competing processes.

`-X` *MPX_scheduling*
Toggles multiplexed scheduling (MPX); not enabled by default. The value 1 enables MPX scheduling, and the value 0 disables it. This feature performs asynchronous swap-out operations and overlaps swap I/Os. This can reduce the overall time that processes wait to swap in, and should improve interactive response time on systems that have a mix of large (usually batch) and small (usually interactive) processes. To enable the MPX scheduling feature, the swap device must be configured with more than one partition; for best performance, the path to each partition should be independent of the others. To improve interactive response, the `-X` option should be used with interactive-preferred scheduling, specified by the `-i` option.

`-x` *max_outage* Specifies an integer value used to express the maximum swapped-out time of a process. A 0 value disables this option.

[`-Y` *interactive_interval*]
Sets the interactive interval, in seconds, which is the maximum time that a small process (*small_proc*) will not be considered for a swap-out operation if there has been no intervening input from a terminal. You must use the `-Y` option with the the `-y` option, which sets the size of a small process.

[`-y` *small_proc*] Specifies the size, in clicks, of a small process. If a process is smaller than *small_proc*, it will not be swapped out for at least *interactive_interval* seconds (set with the `-Y` option). For each interaction from a terminal, the time interval for a process is reset.

Swap priorities are used for establishing both the order in which processes are swapped in and the order in which processes are considered for swapping out to make room for a process to swap in.

The swap priorities are established by the addition of process variables that are multiplied against weighting factors established by `nschedv`. The higher the numeric value of the priority, the higher an out-of-memory process is queued for swap-in; the lower the value, the higher an in-memory process is queued for swap-out.

The priority of a swapped-out process being considered for swap-in is computed as follows:

constant_out + "normalized-size"* *mfactor_out* + "normalized-nice"* *nfactor_out* + "normalized-priority"* *pfactor_out* + "time-swapped"* *tfactor_out*

The priority of an incore process being considered for swap-out is computed as follows:

constant_in + "normalized-size"* *mfactor_in* + "normalized-nice"* *nfactor_in* + "normalized-priority"* *pfactor_in* + "time-swapped"* *tfactor_in*

If the process is greater than or equal in size to the process attempting to swap in, *fit_boost* (see the `-f` option) is added to the calculated priority.

Default values for all of the `nschedv` parameters are calculated at boot time.

`nschedv` does not check parameter settings beyond the checking provided by `schedv`(2).

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to set parameters.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to set parameters.

EXAMPLES

Examples of `-G` option usage are as follows:

`-G 60` : size proportional guaranteed residence time of 60 seconds.
`-G 10,0` : constant guaranteed residence time of 10 seconds.
`-G 5,60` : constant of 5 seconds plus size proportional of up to 60 seconds.

Examples of `-g` option usage are as follows:

`-g 60` : size proportional guaranteed residence time of 60 seconds.
`-g 10,0` : constant guaranteed residence time of 10 seconds.
`-g 5,60` : constant of 5 seconds plus size proportional of up to 60 seconds.

SEE ALSO

`limit`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`limit`(2), `schedv`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

NAME

ntalkd – Listens at ports and logs debugging information

SYNOPSIS

/etc/ntalkd [-d]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The ntalkd program is the remote server used by the UNIX talk(1B) command. It listens at the port indicated in the talk service description (see services(5)). The actual conversation occurs on a TCP connection that is established by negotiation between the two machines involved.

The ntalkd program accepts the following option:

-d Through syslog(3C), ntalkd logs general debugging information concerning talk requests and responses handled by ntalkd.

BUGS

The protocol is architecture-dependent; the current implementation works between Sun computer systems and Cray Research systems, but not between DEC VAX computer systems and Cray Research systems.

SEE ALSO

talk(1B) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

syslog(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

services(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

ntp – Queries a network time protocol (NTP) clock

SYNOPSIS

/etc/ntp [-v] [-s] [-f] [*hosts*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The ntp command sends an NTP packet to the NTP daemon that runns on each of the specified hosts. The daemon fills in fields of the NTP packet per RFC 1059 and sends the packet back. ntp then formats and prints the result on the standard output.

The default output shows the delay, offset, and remote date and time in ctime(3C) format.

You can use options to reset the time of the local system clock.

The ntp command accepts the following options:

-v Verbose output. Shows the full contents of received NTP packets, plus calculated offset, displacement, and so on.

-s Sets system time-of-day clock. Happens only if time offset is less than compiled-in constant WAYTOBIG (currently 1000 seconds). If remote host is unsynchronized, does not happen. Only root is allowed to use this option.

-f Forces the setting of the system clock regardless of offset. Must be used with -s option. If the remote system is unsynchronized, the clock is not reset.

hosts Specifies host names.

NTP Results

The default output for each *host* appears as follows:

```
128.8.10.1: delay:1.845207 offset:-0.358460  Mon Mar 20 08:05:44 1989
```

The verbose output for each *host* appears as follows:

```
Packet from: [128.8.10.1]
Leap 0, version 1, mode Server, poll 6, precision -10 stratum 1 (WWVB)
Synch Distance is 0000.1999  0.099991
Synch Dispersion is 0000.0000  0.000000
Reference Timestamp is a7bea6c3.88b40000 Tue Mar  7 14:06:43 1989
Originate Timestamp is a7bea6d7.d7e6e652 Tue Mar  7 14:07:03 1989
Receive Timestamp is   a7bea6d7.cf1a0000 Tue Mar  7 14:07:03 1989
Transmit Timestamp is  a7bea6d8.0ccc0000 Tue Mar  7 14:07:04 1989
Input Timestamp is     a7bea6d8.1a77e5ea Tue Mar  7 14:07:04 1989
umd1: delay:0.019028 offset:-0.043890  Tue Mar  7 14:07:04 1989
```

The various fields are interpreted as follows:

Packet from: [*address*]
: The address from which this NTP packet was received.

Leap indicator: *n*
: The leap second indicator. If a leap second will be added or subtracted at the new year, *n* is nonzero.

version: The version of NTP used by this implementation. The version currently ported to UNICOS is version 1.

mode: May be Server, Client, Broadcast, Symmetric Active, Symmetric Passive, Unspecified, or Reserved. The value depends on the relationship between the local host and the remote host, as specified in the configuration file.

poll = *n* The desired poll rate of the peer (in log base 2 seconds).

precision = *2^exponent* (dec)
: The claimed precision of the clock (in log base 2 seconds).

stratum: *n* (*source*)
: The stratum of the clock in the NTP hierarchy, along with the source of the clock, which is either the name of a reference standard (such as WWVB or GOES) or the Internet address of the clock from which this clock is derived. Servers with physically attached reference clocks are usually stratum 1. Servers that synchronize with stratum *n* servers are, themselves, stratum (*n*+1) servers.

Synch Distance is *hex-fixedpoint double*
: The minimum total round-trip path delay between each server and the primary reference source to which it is synchronized.

Synch Dispersion is *hex-fixedpoint double*
: A measure of the variability of the measured round-trip packet delays between each server and the primary reference source to which it is synchronized.

The next five time stamps are given as NTP fixed-point values, in both hexadecimal and `ctime`(3C) format. These are set either by this NTP process, or by the server you are querying.

Reference Timestamp is *hex-timestamp ctime string*
: The last time the server clock was adjusted (remote time).

Originate Timestamp is *hex-timestamp ctime string*
: The time at which the NTP request was transmitted by you to the server (local time).

Receive Timestamp is *hex-timestamp ctime string*
: The time at which the NTP request was received at the server (remote time).

Transmit Timestamp is *hex-timestamp ctime string*
: The time at which the NTP response was transmitted by the server (remote time).

Input Timestamp is *hex-timestamp ctime string*
: The time at which the NTP response was received by you (local time).

hostname: delay:*time* offset:*time*
: The summary of the results of the query, giving the host name of the responding clock (from the command line), the round-trip delay, and the offset between the two clocks (assuming symmetric round-trip times).

BUGS

Using `ntp` with the local host shows inaccurate results.

The `ntp` command uses version 1 of the NTP protocol. Version 2, as defined by RFC 1119, may be supported in a future UNICOS release.

SEE ALSO

`ntpd`(8), `ntpdc`(8), `ntptime`(8)

`ctime`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

RFC 1059, *Network Time Protocol (Version 1) Specification and Implementation*, Revised 15 April 1988, David L. Mills, University of Delaware.

NAME

ntpd – Performs NTP time synchronization daemon function

SYNOPSIS

/etc/ntpd [-a *threshold*] [-n] [-c *config-file*] [-d] [-D *level*] [-l] [-r *time*] [-s]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The ntpd command is the network time synchronization daemon, and it is usually invoked at boot time from the /etc/netstart file. It implements a version of the network time protocol (NTP) that is described in RFC 1059. It synchronizes the host's time with a set of distributed time servers, which each have varying accuracy and reliability. Multiple time server masters can exist, but no requirement exists to elect a single master.

The ntpd command uses the adjtime(2) and settimeofday(2) system calls to adjust the clock of the host by small amounts to keep the clock synchronized.

When ntpd is started on the machine, it reads configuration information from /etc/ntp.conf, which contains information about other NTP time servers and host-specific information. Configuration information is listed one entry per line, with fields separated by white space. Lines that begin with a # character are treated as comments. The following is a sample configuration file:

```
#
#               Local clock parameters
#
#       Precision of the local clock to the nearest power of 2
#               ex.
#                       60-HZ   = 2**-6
#                       100-HZ  = 2**-7
#                       1000-HZ = 2**-10
precision -10
#
peer       foo.umd.edu
peer       192.5.39.94
peer       bar.arpa
server     bogon.umd.edu
```

Two major types of information are specified in the configuration file: local host information and remote time server specification.

The local host information describes the intrinsic properties of the local host's timekeeping machinery. The keyword in this group is `precision`, which takes a number that describes the resolution of the local clock as a power of 2. On Cray Research systems, optimal performance is achieved with a `precision` value of –10.

The `driftfile` keyword specifies the name of the file from which the drift compensation register is loaded at initialization time and into which updated values are written. The drift compensation value describes your host's clock intrinsic drift. By default, the `/etc/ntp.drift` file is used.

The time server specifications currently supported are `peer`, `server`, and `passive`. Each keyword takes either a standard Internet dot address or a host name. Each host that is specified with any one of the three keywords is eligible for synchronization; random hosts that set up a peer relationship are not eligible. The `peer` and `server` keywords create an active polling situation; for `peer`, the NTP packets are sent in symmetric-active mode; using `server` causes the packets to be sent in client mode. When accessibility is lost with a configured host in either of these two cases, the daemon continues to poll to reacquire that host. However, a host that is specified with the `passive` keyword does not continue to be polled. If that host begins to poll, it is eligible to be synchronized but is not polled if accessibility is lost.

You must specify most of the configured peers by using the `server` keyword; this minimizes resource usage on the remote NTP server. If your host is serving as a redistribution point for a cluster of hosts, you must set up `peer` relationships with higher-quality clocks (lower stratums) and other equal stratum clocks. Thus, when you are not redistributing time to others, you do not have to configure any `peers` in your NTP configuration, because `server` specifications are used more appropriately.

The `ntpd` command accepts the following options:

`-a` *threshold* Sets the threshold that limits the amount *ntpd* changes the system clock. It is used as an ultimate sanity check to prevent your system time from being changed often. By default, the threshold is 1000 seconds. To defeat the sanity check, specify *threshold* in units of seconds or use the string `any`.

`-n` Specifies that `ntpd` must lock itself into UNICOS memory and not allow itself to be swapped out.

`-c` *config-file* Specifies the location of the `ntpd` configuration file. By default, `/etc/ntp.conf` is used.

`-d` Bumps the debug level by 1. You can specify this option more than once to increment the debug level by 1 each time. If `ntpd` is not compiled with `DEBUG` defined, this option has no effect.

`-D` *level* Sets the debug level to the specified value. If `ntpd` is not compiled with `DEBUG` defined, this option has no effect.

`-l` Logs a message each time the logical clock is changed. Do not specify this option unless you wanted to gather statistical information to analyze the logical clock behavior. When you specify the `-l` option, `ntpd` logs a message approximately every 2 minutes.

`-r` *time* Generates a detailed NTP report every *time* hours and writes the report to the system log file.

`-s` Specifies that the local clock must never be adjusted.

NOTES

Choose your NTP peers and servers carefully; for assistance, send mail to `ntp@TRANTOR.UMD.EDU`.

BUGS

To prevent a major time warp when running the UNICOS operating system, you must run the `ntp`(8) command with the `-s` and `-f` options before `ntpd` is started. This causes the local time to be the network time during system startup, before UNICOS user processes begin. Cray Research recommends that you use the `ntpstart.sh` script when you start NTP.

Note: To set the time, `root` must invoke the `ntp -s -f` command.

Although the current implementation of `ntpd` uses the NTP version 1 protocol, the NTP protocols are designed to be upwardly compatible. Thus, your Cray Research `ntpd` that is running the version 1 protocol can synchronize to a foreign host that is running NTP version 2 software. However, an NTP version 1 time server does not respond to NTP version 2 requests. Thus, this particular `ntpd` does not synchronize version 2 clients unless the clients disguise their requests as being from version 1 hosts. However, most version 2 implementations allow configuration parameters to be set to allow version 2 clients to disguise their requests to prespecified servers.

FILES

`/etc/ntp.conf` NTP daemon configuration file

`/etc/ntp.drift` NTP daemon drift history file

SEE ALSO

`ntp`(8), `ntpdc`(8), `ntptime`(8)

`gettimeofday`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

RFC 1059, *Network Time Protocol (Version 1) Specification and Implementation*, Revised 15 April 1988, David L. Mills, University of Delaware. This implementation is based primarily on the version of `ntpd` written by Louis A. Mamakos and Michael G. Petry of The University of Maryland Computer Science Center.

NAME

ntpdc – Monitors operation of network time protocol (NTP) daemons

SYNOPSIS

/etc/ntpdc [-n] [-v] *hosts...*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The ntpdc command sends an INFO_QUERY packet to an NTP daemon that runs on the specified hosts. Each daemon responds with information about each of its peers, which ntpdc formats on the standard output.

The ntpdc command accepts the following options:

-n Disables the printing of the name of the responding host and its peers, printing only the Internet addresses. The default is to print the names.

-v Generates a verbose report. The default is a terse, table-style report.

hosts Specifies hosts on which the NTP daemon is running.

Terse Report

A typical terse report looks as follows:

```
 (rem)Address      (lcl)    Strat  Poll  Reach  Delay  Offset   Disp
==============================================================================
-umd1           128.8.10.14   1      64   266     3.0   -65.0    0.0
*DCN1.ARPA      128.8.10.14   1     256   332   155.0    -4.0    0.0
128.8.251.92    128.8.10.14   2      64   367    16.0   -61.0    0.0
idunno.Princeto 128.8.10.14   3      64   252    60.0   -53.0    0.0
leo             128.8.10.14   2      64   275     4.0  -273.0 1536.2
```

The fields in this report are interpreted as follows:

- or * The - character indicates a preconfigured peer (mentioned in /etc/ntp.conf). The * character identifies the preconfigured peer (if any) currently being used for synchronization.

(rem) Address The remote host name or Internet address of a peer.

(lcl) The local host as specified as an argument to ntpdc.

Strat The stratum level of the peer (as perceived by the local host). Hosts with physically attached reference clocks are usually of stratum 1. Servers that synchronize with stratum *n* servers are stratum *(n+1)* servers.

`Poll`	Current polling interval in seconds for this peer.
`Reach`	Octal value of a shift register that indicates the responses received from the previous eight polls to this peer (see RFC 1059).
`Delay`	Round-trip delay (in milliseconds) for this peer, as of the latest poll.
`Offset`	The difference between the local time and the times reported by each host (in milliseconds).
`Disp`	Current value of the packet dispersion (see RFC 1059) for this peer.

Verbose Reports

When you use the `-v` option, a series of verbose reports is presented. A typical report looks as follows:

```
Neighbor address 128.4.0.6 port:123  local address 192.35.201.47
Reach: 0376 stratum: 1, precision: -10
dispersion: 0.014000 flags: 0 leap: 0
Reference clock ID: WWV  timestamp: a7c2832e.6f9d0000
hpoll: 10, ppoll: 10, timer: 1024 sent: 266 received: 192
Delay(ms)  1144.00 1296.00 1118.00 1115.00 1225.00 1129.00 1086.00 1087.00
Offset(ms)   19.00   92.00  -17.00   12.00   41.00    4.00   -1.00  -14.00

        delay: 1086.000000 offset: -1.000000 dsp 0.014000
```

The fields in this report are interpreted as follows:

`Neighbor address`	The address and port number of this neighbor, followed by the local address.
`Reach:` *nn*	Reachability in response to last eight polls (octal value of shift register).
`stratum:` *n*	Stratum level. Servers with physically attached reference clocks are usually stratum 1. Servers that synchronize with stratum *n* servers are stratum *(n+1)* servers.
`precision:` *nn*	The precision of this clock (in log base 2 seconds). For example, a clock derived from the power line frequency (60 Hz) has a precision of 1/60 second (about 2^−6) and would be indicated by a precision of–6.
`dispersion:` *nn*	A measure of the variation of packet round-trip times to the specified host.
`flags:` *nn*	The flags field is a bit mask, whose bits are defined as follows:

```
#define PEER_FL_CONFIG       0x0001  /* peer is in configuration file */
#define PEER_FL_AUTHENABLE   0x0002  /* peer supports authentication */
#define PEER_FL_SANE         0x0100  /* a "sane" peer */
#define PEER_FL_CANDIDATE    0x0200  /* a candidate peer */
#define PEER_FL_SYNC         0x1000  /* peer can be sync'd to */
#define PEER_FL_BCAST        0x2000  /* broadcast peer */
#define PEER_FL_REFCLOCK     0x4000  /* peer has a local reference clock */
#define PEER_FL_SELECTED     0x8000  /* the "best" peer available */
```

`leap:` *flag* The leap second indicator. If a leap second will be added or subtracted at the new year, value is nonzero.

`Reference clock ID:` [*string*]
Specifies the type of clock used by the reference server. Examples of common types are Geosynchronous Orbit Environmental Satellite (GOES), Atomic Clock (ATOM), often Cesium-133 based, and WWV, WWVB, WWVL, or WWVH. The last four types represent radio clocks, which are tuned to one of the stations that broadcast time information. The letters are the actual FCC call letters assigned to those stations.

`timestamp:` *nn* Fixed-point representation of the time stamp sent by the specified server.

`hpoll:` *time* The log base 2 of the number of seconds proposed between polls to this host.

`ppoll:` *time* The log base 2 of the number of seconds currently between polls to this host.

`timer:` *nn* The number of seconds currently between polls to this host.

`sent:` *nn* The number of NTP packets sent to this neighbor.

`received:` *nn* The number of NTP packets received from this neighbor.

`Delay and Offset` The round-trip delay and clock offset for the last eight NTP packet exchanges. If fewer than eight valid samples exist, the delay field will be 0.

`delay:` *avg-delay* `offset:` *avg-offset* `dsp` *dispersion*
Average delay, offset, and dispersion calculated from the preceding samples.

SEE ALSO

`ntp`(8), `ntpd`(8), `ntptime`(8)

RFC 1059, *Network Time Protocol Specification and Implementation (Version 1)*, Dave Mills, University of Delaware.

NAME

`ntpstart` – Starts the network time protocol (NTP) daemon

SYNOPSIS

`/etc/ntpstart` [*ntpd_arguments*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `ntpstart` script sets the system time and starts the necessary software for the network time protocol (NTP) at system startup. It is primarily intended to be used as an entry in the `/etc/config/daemons` file that the `sdaemon`(8) script uses.

The `ntpstart` script accepts the following argument:

ntpd_arguments Specifies argument list to be passed to the `ntpd` daemon.

The `ntpstart` script performs the following functions:

- Scans its argument list (*ntpd_arguments*) for any `-c` arguments, which indicate specification of an alternate NTP configuration file for `ntpd`.
- Calls the `ntp`(8) program with any peers or servers that are listed in the NTP configuration file to set the system time.
- Calls the `ntpd`(8) daemon itself and passes its argument list to the daemon.

SEE ALSO

`ntp`(8), `ntpd`(8), `sdaemon`(8)

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

ntptime – Displays offset between local time and real time

SYNOPSIS

/etc/ntptime [-v]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The ntptime command sends an INFO_QUERY packet to all network time protocol (NTP) daemons that run on the hosts found in the ntp.conf configuration file. Each daemon responds with information about its time, which ntptime formats on the standard output.

The default output specifies only the amount of time the local clock varies from the time reported by the best server.

The ntptime command accepts the following option:

-v Generates a verbose report, specifying the offsets from all hosts in the configuration file.

EXAMPLES

A typical terse output appears as follows:

```
Your clock is running 0.002 seconds slow.
```

When you use -v option, a more verbose report is presented. A typical report is as follows:

```
[ *130.126.174.40   8/8  (0.008888/-0.002000) ]
[  128.4.0.1        5/4  (7.507498/0.024999) ]
[  128.8.10.1       4/3  (15.505128/0.013001) ]
Your clock is running 0.002 seconds fast
```

The verbose lines have the following form:

[**Host* *sent*/*recv* (*disp*/*offset*)]

The fields in the output line have the following meanings:

* Indicates the host used for the actual offset calculation.

Host Host name or host Internet address obtained from the configuration file.

sent Number of packets sent to this host.

recv Number of packets received from this host.

disp Dispersion, a measure of the variance of the packet round-trip times to this host. Higher numbers indicate greater variance in round-trip times.

offset Difference between the local time and the reference time offered by this host.

SEE ALSO

`ntp`(8), `ntpd`(8), `ntpdc`(8)

RFC 1059, *Network Time Protocol Specification and Implementation (Version 1)*, David Mills, University of Delaware.

NAME

nu – Manages user login accounts

SYNOPSIS

```
/etc/nu -a [-p path] [-c file] [-D] [-n] [-h]
/etc/nu -m [-p path] [-c file] [-D] [-n] [-h]
/etc/nu -d [-p path] [-c file] [-D]
/etc/nu -g [-p path] [-c file] [-D] [-h]
/etc/nu [-p path] [-c file] [-D] -k users
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nu command helps a UNICOS system administrator create, modify, delete, and destroy user accounts. Although everything accomplished by nu can be done manually by editing files and issuing shell commands or running udbgen(8), nu can direct you through the details correctly by checking file locking, typographical errors, and so on. Alternatively, all the functionality of nu is available through a graphical user interface (GUI) by using the xadmin(8) command.

The nu command accepts the following options (the -a, -m, -d, -g, and -k options are mutually exclusive):

-a Adds new accounts. The program prompts you for the login ID, password, name, and other information about each new user. It then creates the account, creates its directories, initializes the contents of the directories, and makes an entry in the log file.

-m Modifies existing accounts. The program asks for account names (login IDs) and instructions for the changes that you want to make to those accounts, until you indicate that you are finished.

-d Disables accounts (login IDs) while still leaving an entry in /etc/udb (see udb(5)) for the deleted account. This prevents the specific user ID from being reused, and it lets you access the accounting data after deleting an account. The program repeatedly asks you for the names of accounts to be deleted and the amount of deleting you want to do to those accounts. WARNING: When you use this option, user files are deleted.

-g Displays and allows editing of the global tape name map and the user database (UDB) default fields. Each field to be changed is selected and its new value assigned. When all of the changes have been made, the new information is written to the UDB. All records created after the changes to the default table will assume the new values. See udbgen(8) for more information.

`-k` *users* Deletes from the system almost all information that pertains to the specified user IDs, *users*. Specifically, `nu` removes the entry from `/etc/udb`, deletes the login directory and all of its contents, and deletes the mailbox. It does not remove that user from any mailing lists in `/usr/lib/aliases`. The `-k` option is not interactive. It must be the last option on the command line. You should provide the complete list of users to be deleted as arguments to the option on the command line; you must separate items in this list with spaces.
WARNING: When you use this option, user files are deleted.

`-p` *path* Specifies the path name to access the `nu` and `udb` files. This option allows private or test versions of these files to be maintained. The path name must end with a directory name. The default path name is `/etc`.

`-c` *file* Specifies the configuration file that `nu` will use. If you specify the `-p` option, the default file is *path*`/nu.cf60`; otherwise, it is `/etc/nu.cf60`.

`-D` Causes `nu` to run in debug mode. Debug mode helps you remove bugs from your shell scripts before you run them as `root`. You do not have to be logged in as `root` to run `nu` in debug mode. Typically, this option is used with the `-p` option.

`-n` Specifies that the user entry data will be processed and displayed in the new format.

`-h` Specifies that section headers will be output when possible. Typically, this option is used with the `-n` option.

Configuration File

When `nu` is started, it reads configuration commands from the `nu.cf60` file, which contains information needed to create new accounts on your machine. By default, this file is `/etc/nu.cf60`. However, if you specify the `-p` option, this file is *path*`/nu.cf60`; if you use the `-c` option, this file is specified in the `-c` option.

To update the `/etc/nu.cf60` file, use the UNICOS Installation Menu System (installation tool). The system administrator must use the installation tool to make site-specific modifications to `nu`. In particular, the `GroupHome` declarations must be updated to reflect the file systems that hold the login directories for members of different groups. You also should change the default values to match the UDB defaults. If they do not agree, these defaults will be used in preference to those in the UDB, and messages indicating this will be issued whenever `nu` is initiated.

When `nu` is required to create a new directory, it runs a shell script specified in `nu.cf60`. Similarly, when it is required to initialize the files in a newly created directory, it runs another shell script, the name of which it finds in `nu.cf60`. To change configuration and/or customize, you can edit these shell scripts to conform to local practices. If you do edit the scripts, remember that `nu` runs as root and that the shell scripts contain possibly irreversible statements, such as `rm -rf *`. Caution is advised.

The nu.cf60 configuration file is a text file that contains a series of statements, one statement per line. A semicolon that is not inside a quoted string causes the rest of that line to be treated as a comment. Each line in the file that is nonblank, after stripping comments, is treated as an assignment statement. Each statement assigns a value to one variable. All integers are nonnegative integers, and all strings must be delimited with double quotation marks ("). You may use the words none and unlimited without quotation marks in place of a numeric value in the context of resource limits. These special values are translated into the appropriate internal form to specify that none of a resource or an unlimited amount of a resource may be used. No quoting or doubling convention exists for a double-quotation character inside a string. The configuration variables are case-sensitive.

The configuration file contains two types of information:

- System information
- User default information, which is organized in the following categories:
  - General user information
  - User security information
  - User general limits
  - User file limits
  - User MPP limits
  - User tape limits

System (administration) Information

The system variables in the nu.cf60 configuration file include the following:

Variable	Description
CreateDir	A string variable that identifies the shell script that is run whenever nu must create a new directory. This shell script must be executable. It is called with the following arguments: 1 The integer user ID of the desired user. 2 The integer group ID of the desired user. 3 The name of the user's actual home directory. 4 Same function as argument 3 (included for historical reasons). 5 An integer that is nonzero if it is possible to overwrite an existing directory with the same name as argument 4. 6 An integer that is nonzero if nu is running in debug mode. 7 The integer account ID of the desired user. The standard value for CreateDir is /etc/nulib/nu1.sh.

`CreateFiles` A string variable that identifies the shell script that is run when `nu` must initialize a directory (newly created or otherwise) with some standard files. The system administrator must create the `/usr/skel/.[a-z]*` directory and it must be initialized with files appropriate to the particular site. The `nu2.sh` script copies `/usr/skel/.[a-z]*` into a new login directory. This shell script must be executable. It is called with the following arguments:

1 The name of the login directory to be initialized.

2 The integer user ID of the user.

3 The integer group ID of the user.

4 An integer that is nonzero if setup for the Cray station will be performed.

5 The account name of the desired user.

6 An integer that is nonzero if `nu` is running in debug mode.

The standard value for `CreateFiles` is `/etc/nulib/nu2.sh`.

`DeleteAccts` A string variable that identifies the shell script that is run whenever `nu` must delete a user's account that was created in some earlier session with `nu`. Deleting accounts involves changing the password so that the user cannot log in, deleting all of the user's files and directories, and deleting the user's mailbox. This shell script must be executable. It is called with the following arguments:

1 The login ID of the account to be deleted.

2 The login directory for that account.

3 The name given in `/etc/udb` for the login directory (which might possibly be a symbolic link to item 2 and therefore it must be specified separately).

4 An integer that is nonzero if `nu` is running in debug mode.

The standard value for `DeleteAccts` is `/etc/nulib/nu4.sh`.

`DestroyAccts` A string variable that identifies the shell script that is run whenever `nu` must destroy a user's account that was created in some earlier session with `nu`. Destroying accounts involves removing the user from the user database, deleting all of the user's files and directories, and deleting the user's mailbox. For a less drastic account removal action, see `DeleteAccts`.

This shell script must be executable. It is called with the following arguments:

1 The login ID of the account to be deleted.

2 The login directory for that account.

3 The name given in `/etc/udb` for the login directory (which might be a symbolic link to item 2 and therefore it must be specified separately).

4 An integer that is nonzero only if `nu` is running in debug mode.

The standard value for DestroyAccts is /etc/nulib/nu3.sh.

Logfile	A string variable that specifies the file in which all nu transactions are logged. The standard value of Logfile is /usr/adm/nu.log.
MaxGroups	An integer variable that defines the maximum number of GroupHome symbols contained in the configuration file. Released as 100.
MaxGroupAdm	An integer variable that defines the maximum number of GroupAdm symbols contained in the configuration file. Released as 100.
MaxNameLength	An integer variable that gives the maximum number of characters permitted in a login name; for unmodified UNICOS systems, it should be set to 8. It must never exceed 15; more than 8 is not recommended because many standard UNIX command output formats expect up to 8 characters.
MaxSymbols	An integer variable that defines the maximum number of general symbols contained in the configuration file. Released as 100.
WantCSsetup	An integer variable that should be set to 1, if you want nu to set up for the Cray station. nu just passes the value of WantCSsetup through to the shell script specified in CreateFiles, which does the actual initialization. The standard value is 1.

User Default Information

The user default variables in the nu.cf60 configuration file are organized by category.

The general user information category includes the following:

Variable	Description
DefaultAcids	Default account IDs. Typical value is 0.
DefaultAge	Default password aging. Four fields are provided to define the aging characteristics that the UNICOS operating system maintains. The fields are written from left to right, separated by commas. You may omit fields to the right if you do not need default values for those fields.

```
DefaultAge = "[flags [maxage [minage [age]]]" "prompt"; comment
```

These fields have the following meanings:

Field [Values]	Meaning
flags [`superuser`, `force`, `none`]	`superuser` means only an administrator changes passwords. `force` means the user must change the password on the next login. `none` means neither of these is enabled.
maxage [`0w` through `64w`]	Maximum age of the password. An optional suffix to specify the unit may be provided. `w` means weeks, `d` means days, and `s` means seconds. The default unit is weeks.
minage [`0w` through `64w`]	Minimum age of the password. An optional suffix to specify the unit may be provided. `w` means weeks, `d` means days, and `s` means seconds. The default unit is weeks.
age [`0w` < *value*]	Initial age of the password. The value 0 means the present date and time. An optional suffix to specify the unit may be provided. `w` means weeks, `d` means days, and `s` means seconds. The default unit is weeks. No specific upper bound to the age is imposed, although an age that would translate to a date more than 64 weeks in the past may not be honored and makes no sense.
prompt [`on` or `off`]	You must turn this default either `on` or `off`. If it is off, the specified default is used; you are not prompted for these values when you are adding new users. If it is on, you are prompted for information.

Examples:

The following example shows how to set the default to force changes at login with a maximum age of 21 days and a minimum age of 2 days. An initial password is unnecessary because `force` makes the password appear to be expired. The default is `on`; thus, the administrator is given the opportunity to change the values.

```
DefaultAge "force, 3w, 2d" "on"
```

You also can force the password to be changed by setting the initial age older than the maximum age:

```
DefaultAge "none, 3w, 2d, 22d" "on"
```
or
```
DefaultAge "3w, 2d, 22d" "on"
```

If the flag is not wanted, it is correct either to specify `none` or to omit it because, for the default, no flags are predefined.

`DefaultAl` Default Data Migration Facility (DMF) archive limit (in disk blocks). Released as 0.

`DefaultAm` Default Data Migration Facility (DMF) media type. Typical value is 0.

`DefaultAs` Default allocation shares. Typical value is 100.

`DefaultDr` Default login root string. Typical value is `/`.

`DefaultGids` Default group IDs. Typical value is 0.

`DefaultPm` Default permission name string. Typical value is `none`.

`DefaultRg` Default resource group ID. Typical value is 0.

`DefaultPb` Default permbits. Typical value is `none`.

`DefaultHome` A string variable that is set to the file system or top-level directory that will be used to hold the login directory for accounts in groups not explicitly set up to have their login directories elsewhere. When you are creating a new account, `nu` asks you for the group number of the group in which you would like the account held. If that group number is mentioned in a `GroupHome` declaration, the home directory for the group is the one specified in that `GroupHome` declaration. If the group number is not mentioned in a `GroupHome` declaration, login accounts created in that group will have their login directories put into `DefaultHome`. There is no standard value for `DefaultHome`.

`GroupHome` A pseudo variable that is a name by which the configuration code can load entries into a directory location table. In particular, if you provide two `GroupHome` declarations, they are both processed; if you provide two of any other declaration, only the latest one has any effect. For example, a typical set of `GroupHome` declarations might look like the following:

```
GroupHome= 10 "/usr"
GroupHome= 20 "/mnt"
GroupHome= 25 "/usr/cis"
GroupHome= 31 "/usr/guest"
```

The `GroupHome` declarations serve as default login directory location information for new accounts. You can put any account anywhere you want; however, the `GroupHome` information makes the defaults appear in the correct places.

Consequently, to create a new account, press the <`RETURN`> key to accept the defaults. In the preceding example, the declarations would cause group 10 to default to `/usr` (for example, `/usr/smith` or `/usr/jones`). Group 31 would default to `/usr/guest` (for example, `/usr/guest/smith` or `/usr/guest/jones`). If the login group is not mentioned in a `GroupHome` declaration, the `DefaultHome` variable is used.

`DefaultShell` A string variable that is set to the name of the shell file to be used by default. The standard value for `DefaultShell` is `/bin/csh`.

The user security information category includes the following:

Variable	Description
`DefaultDl`	Default default security level name or number. Typical value is 0.
`DefaultMx`	Default maximum security level name or number. Typical value is 0.
`DefaultMn`	Default minimum security level name or number. Typical value is 0.
`DefaultCm`	Default valid compartment name string. Typical value is `none`.
`DefaultDc`	Default active compartment name string. Typical value is `none`.
`DefaultDcat`	Default default category name string. Typical value is `none`.
`DefaultVcat`	Default valid category name string. Typical value is `none`.
`DefaultDl_class`	Default default integrity class. A typical value is 0. The ues of integrity class values is not supported.
`DefaultMx_class`	Default maximum integrity class. A typical value is 0. The ues of integrity class values is not supported.

Each of the following defaults sets a limit for the user. These limits have the format:

variable = *value* "*queue*" "*prompt*"

They can be any of the following:

Variable	Meaning
variable	The variable name, `DefaultAge`.
value	The default value for the variable: `none`, `unlimited`, or an integer number.
queue	The run mode for the session: `batch` or `integer`.
prompt	The state of the prompt (the same as `DefaultAge`): `on` or `off`.

The general user limits category also includes the following variables:

Variable	Description
`DefaultCj`	Default CPU time per session. Released as `unlimited`.
`DefaultCp`	Default CPU time per process. Released as `unlimited`.
`DefaultMj`	Default memory per-session limit. Released as `unlimited`.
`DefaultMp`	Default memory per-process limit. Released as `unlimited`.
`DefaultNc`	Default session nice value adjustment. Released as `none`.
`DefaultSc`	Default per-session socket buffer limit. Released as `unlimited`.

DefaultSj	Default per-session shared memory segments. Released as none.
DefaultSm	Default per-session shared memory size limit. Released as none.

The file limits category includes the following variables:

Variable	Description
DefaultFd	Default per-process open file size limit. Released as unlimited.
DefaultJl	Default per-session file size limit. Released as unlimited.
DefaultJs	Default per-session SDS limit. Released as none.
DefaultPc	Default per-process core file size limit (512-word blocks). Released as unlimited.
DefaultPs	Default per-process SDS limit. Released as none.
DefaultUl	Default per-process file size limit. Released as unlimited.

The MPP limits category includes the following variables:

Variable	Description
DefaultMb	Default per-session MPP barrier limit. Released as none.
DefaultMt	Default per-session MPP time limit. Released as none.
DefaultPe	Default per-session MPP processing element (PE) limit. Released as none.
DefaultPt	Default per-process MPP time limit. Released as none.

The tape limits category includes the following variable:

Variable	Description
DefaultTp	Default per-session tape limit. Released as none. You must define this default for each of the eight tape types; also, each of the tape types must specify a default for both batch and inter and must be either on or off. (See EXAMPLES.)

NOTES

The nu command traps interrupt signals (such as <CONTROL-c> and does not end if you try to stop it in the middle of a critical section. Critical sections are primarily the updates of the UDB. A list of changes is recorded in a log file (usually /usr/adm/nu.log).

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Can modify all UDB fields.
sysadm	Can modify all UDB fields, except sensitive fields.

If the PRIV_SU configuration option is enabled, the administrator running as the super user may modify all UDB fields.

BUGS

The extensive use of shell scripts for doing sensitive duties such as deleting and changing accounts means that someone may be able to make nu fail in destructive ways without having access to the source code.

EXAMPLES

The following example shows how to select a user's password if the machine-generated password feature is enabled:

```
New password is:  lempamdo
Re-enter new password or (CR) to get another:
```

The following are examples of the Default.*xx* directives:

General user limits category

```
DefaultCj = unlimited     "batch" "off" ; default CPU time per session
DefaultCj = unlimited     "inter" "off" ; default CPU time per session
DefaultCp = unlimited     "batch" "off" ; default CPU time per process
DefaultCp = unlimited     "inter" "off" ; default CPU time per process
DefaultMj = unlimited     "batch" "off" ; default memory per-session limit
DefaultMj = unlimited     "inter" "off" ; default memory per-session limit
DefaultMp = unlimited     "batch" "off" ; default memory per-process limit
DefaultMp = unlimited     "inter" "off" ; default memory per-process limit
DefaultNc = none          "batch" "on"  ; per-session nice value
DefaultNc = none          "inter" "on"  ; per-session nice value
DefaultSj = none          "batch" "on"  ; shared memory limit per session
DefaultSj = none          "inter" "on"  ; shared memory limit per session
DefaultSm = none          "batch" "on"  ; shared memory segments per session
DefaultSm = none          "batch" "on"  ; shared memory segments per session
```

User file limits category

```
DefaultFd = unlimited     "batch" "on"  ; default per-process open file limit
DefaultFd = unlimited     "inter" "on"  ; default per-process open file limit
DefaultJl = unlimited     "batch" "on"  ; default per-session file size limit
DefaultJl = unlimited     "inter" "on"  ; default per-session file size limit
DefaultUl = unlimited     "batch" "off" ; default file size limit
DefaultUl = unlimited     "inter" "off" ; default file size limit
DefaultJs = none          "batch" "off" ; default per-session SDS limit
DefaultJs = none          "inter" "off" ; default per-session SDS limit
DefaultPs = none          "batch" "off" ; default per-process SDS limit
DefaultPs = none          "inter" "off" ; default per-process SDS limit
DefaultPc = unlimited     "batch" "on"  ; default per-process core file size limit
DefaultPc = unlimited     "inter" "on"  ; default per-process core file size limit
```

User MPP limits category

```
DefaultPe = none          "batch" "on"  ; default per-session MPP PE limit
DefaultPe = none          "inter" "on"  ; default per-session MPP PE limit
DefaultMt = none          "batch" "on"  ; default per-session MPP time limit
DefaultMt = none          "inter" "on"  ; default per-session MPP time limit
DefaultMb = none          "batch" "on"  ; default per-session MPP barrier limit
DefaultMb = none          "inter" "on"  ; default per-session MPP barrier limit
DefaultPt = none          "batch" "on"  ; default per-process MPP time limit
DefaultPt = none          "inter" "on"  ; default per-process MPP time limit
```

User tape limits category

```
DefaultTp = none  "type0" "batch" "on"  ; default per-session tape limit
DefaultTp = none  "type0" "inter" "off" ; default per-session tape limit
DefaultTp = none  "type1" "batch" "off" ; default per-session tape limit
DefaultTp = none  "type1" "inter" "off" ; default per-session tape limit
DefaultTp = none  "type2" "batch" "off" ; default per-session tape limit
DefaultTp = none  "type2" "inter" "off" ; default per-session tape limit
DefaultTp = none  "type3" "batch" "off" ; default per-session tape limit
DefaultTp = none  "type3" "inter" "off" ; default per-session tape limit
DefaultTp = none  "type4" "batch" "off" ; default per-session tape limit
DefaultTp = none  "type4" "inter" "off" ; default per-session tape limit
DefaultTp = none  "type5" "batch" "off" ; default per-session tape limit
DefaultTp = none  "type5" "inter" "off" ; default per-session tape limit
DefaultTp = none  "type6" "batch" "off" ; default per-session tape limit
```

You can replace the tape names (such as "`type0`") by the names defined in the global tape map after it has been defined in the UDB.

FILES

The default directory for the following files is `/etc`; to change it, use the `-p` or `-c` option.

`/etc/acid`	Account file
`/etc/group`	System group file
`/usr/include/udb.h`	User database structure
`/etc/nu.cf60`	Configuration file
`/etc/nulib/*.sh`	Shell scripts to perform the work
`/etc/nulib/nu.cf60`	Release default configuration file
`/etc/udb`	User validation file that contains user control limits
`/etc/udb.public`	Public version of the user database with security sensitive information removed

SEE ALSO

udbgen(8), xadmin(8)

spset(1), udbsee(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

getgrent(3C), getpwent(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

group(5), udb(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

nulladm – Creates an empty file owned by adm

SYNOPSIS

/usr/lib/acct/nulladm *file*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The nulladm command creates a file with mode 664 and ensures that the owner and group are adm. nulladm is included as part of many accounting scripts, and it rarely is invoked alone.

NOTES

Sites may allow users in group adm to run Cray Research system accounting (CSA), if they have the permission bit acct set in their user database (UDB) entries. However, such users cannot run accounting after a super user has done so, because the group ID and permissions of the files will change. In this case, execute the csaperm(8) command to reset group IDs and permissions so that nonsuper users can run accounting again.

SEE ALSO

acct(8), acctsh(8), csa(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

nwmstart – Initializes networking media

SYNOPSIS

/etc/nwmstart

IMPLEMENTATION

Cray Research systems with I/O subsystem model E

DESCRIPTION

The nwmstart script initializes the underlying networking media necessary for communication over networking interfaces, according to information contained in a set of tabular configuration files. On IOS model E systems, this consists of executing fddiload(8) to initialize the FDDI channels that are used for networking.

nwmstart is intended to be used primarily by the netstart(8) command during system startup.

File Format

The nwmstart configuration file for the IOS model E systems is /etc/config/fddi_config. It consists of a series of lines with the following format:

interface ordinal *action*

The elements in the entry have the following meanings:

interface ordinal
: Ordinal of the FDDI interface channel being referenced. This ordinal should correspond to the fddev ordinal in the param file for the same channel.

action
: Indication of whether to initialize the channel. An *action* value of YES or ON (case is insignificant) indicates that nwmstart will initialize FDDI device number *interface ordinal* on the appropriate attached IOS system by executing fddiload with appropriate arguments.

EXAMPLES

The following example /etc/config/fddi_config file initializes one FDDI channel on the attached IOS system:

```
#
# Example fddi_config file to initialize one FDDI channel
#
0 YES
1 NO
```

FILES

`/etc/config/fddi_config`	FDDI channel configuration file

SEE ALSO

`fddiload`(8), `netstart`(8)

NAME

olcfpt, olcm, olcrit, olcsvc, olibuf, olsbt – Online diagnostic CPU confidence tests

SYNOPSIS

/etc/diag/olcfpt [chkpnt *mode*] [cpu *clist*] [cputime *h*:*m*:s] [+/-getseed] [getseed *file*] [help] [maxerr *n*] [maxp *n*] [+/-modes] [+/-waitded] [+/-parcel] [time *h*:*m*:*s*] [+/-verbose] [disable *ilist*] [enable *ilist*] [+/-isolate] [isop *n*] [numins *n*] [+/-repeat] [seed *n*] [vl *n*] [+/-vload] [+/-fpbits] [+/-fprand] [+/-random] [+/-fpadd] [+/-fpmult] [+/-fprecip] [+/-scalar] [+/-vector]

/etc/diag/olcm [chkpnt *mode*] [cpu *clist*] [cputime *h*:*m*:s] [+/-getseed] [getseed *file*] [help] [maxerr *n*] [maxp *n*] [+/-modes] [+/-waitded] [+/-parcel] [time *h*:*m*:*s*] [+/-verbose] [+/-repeat] [section *slist*] [seed *n*] [stride *n*] [words *n*]

/etc/diag/olcrit [chkpnt *mode*] [cpu *clist*] [cputime *h*:*m*:s] [+/-getseed] [getseed *file*] [help] [maxerr *n*] [maxp *n*] [+/-modes] [+/-waitded] [+/-parcel] [time *h*:*m*:*s*] [+/-verbose] [+/-ranclus] [cluster *n*] [disable *ilist*] [+/-eis] [enable *ilist*] [+/-hwbmm] [+/-isolate] [isop *n*] [numins *n*] [+/-repeat] [seed *n*] [vl *n*] [+/-vload] [+/-bits] [+/-onezero] [+/-random] [+/-address] [+/-ci] [+/-cm] [+/-fpadd] [+/-fpmult] [+/-fprecip] [+/-int] [+/-jump] [+/-logical] [+/-pop] [+/-scalar] [+/-shift] [+/-shr] [+/-vector]

/etc/diag/olcsvc [chkpnt *mode*] [cpu *clist*] [cputime *h*:*m*:s] [+/-getseed] [getseed *file*] [help] [maxerr *n*] [maxp *n*] [+/-modes] [+/-waitded] [+/-parcel] [time *h*:*m*:*s*] [+/-verbose] [disable *ilist*] [enable *ilist*] [+/-eis] [+/-isolate] [isop *n*] [numpar *n*] [+/-repeat] [seed *n*] [+/-sgci] [vl *n*] [+/-vuse *vrlist*] [+/-onezero] [+/-random] [+/-slide] [+/-cm] [+/-fpadd] [+/-fpmult] [+/-fprecip] [+/-int] [+/-logical] [+/-pop] [+/-shift] [+/-lz]

/etc/diag/olibuf [chkpnt *mode*] [cpu *clist*] [cputime *h*:*m*:s] [+/-getseed] [getseed *file*] [help] [maxerr *n*] [maxp *n*] [+/-modes] [+/-waitded] [+/-parcel] [time *h*:*m*:*s*] [+/-verbose] [+/-repeat] [seed *n*] [+/-bits] [+/-onezero] [+/-ozpar] [+/-random] [+/-solid]

/etc/diag/olsbt [chkpnt *mode*] [cpu *clist*] [cputime *h*:*m*:s] [+/-getseed] [getseed *file*] [help] [maxerr *n*] [maxp *n*] [+/-modes] [+/-waitded] [+/-parcel] [time *h*:*m*:*s*] [+/-verbose] [+/-anycpus] [cluster *n*] [numins *n*] [+/-repeat] [seed *n*] [+/-bits] [+/-onezero] [+/-random]

IMPLEMENTATION

Cray PVP systems. The olsbt test does not run on CRAY T90 systems.

DESCRIPTION

Online diagnostic CPU confidence tests provide a comprehensive performance check of the system hardware. The CPU confidence tests are as follows:

`olcfpt` Comprehensive floating-point test

`olcm` Central memory test

`olcrit` Comprehensive random instruction test

`olcsvc` Comprehensive scalar and vector comparison test

`olibuf` Instruction buffer test

`olsbt` Semaphore, shared B and shared T register test

Each confidence test's command synopsis includes monitor options, which are listed first in the synopsis, and test-specific command options. The monitor options are common to all confidence tests. You can enter the test options in any order. If an option is omitted, the default value is used. The following options provide different methods of specifying the starting seed value (specify only one for each test executed): `+/-getseed`, `getseed` *file*, or `seed` *n*. Test results are sent to `stdout` by default or to a file when UNICOS output redirection is specified on the command line.

Monitor Options

The monitor options are as follows:

`chkpnt` *mode*

Indicates whether restart files are to be generated. *mode* is one of the following arguments:

`first` Generates a restart file for the first failure detected (default)

`all` Generates a restart file for each failure detected, including failures detected during error isolation

`none` Does not generate restart files

Restart file names follow the format:

pid-master_cpu_digit diagnostic_name

For example, if `olcrit` is run in CPU c with a process ID of 87865, the restart file will be named `87865-2olcrit`

`cpu` *clist* Selects the CPUs to be tested. Enter *clist* in the following format:

x,x,...,x

x can be a, b, c, ..., and so on, or 0, 1, 2, ..., and so on, up to the maximum number of CPUs. The range of CPU selection using characters is from a through p. Digits must be used to select CPUS above 16. The first CPU selected is the master CPU. The default is cpu a. (For olsbt, the default is cpu a,b on multiple-CPU systems and cpu a on single-CPU systems.) The test output always specifies the CPU(s) as a digit.

The recommended usage of this option is cpu *x*, where *x* is a single CPU designator. Specifying more than one CPU at a time with this option causes increased wall-clock completion times. The increase in time could be substantial on heavily loaded systems. The exception to this recommendation is the olsbt test, which will complete fastest when two CPUs are selected. The olsbt test does not run efficiently on single-CPU systems. It is recommended that olcrit be used to test the shared B and T register instructions and the set and clear semaphore instructions.

The olcm test cannot be executed in multiple-CPU mode (the cpu *clist* option can contain only a single CPU designator for *clist*; for example, cpu b).

cputime *h*:*m*:*s*
Sets the execution time in CPU time. The time is specified in hours (*h*), minutes (*m*), and seconds (*s*); minutes and seconds; or just seconds.

Generally, actual execution time is within 1 second of the specified CPU time. If cputime is allowed to default or is set to 0, the test uses the maxp value. However, if set to a value other than 0, cputime overrides maxp.

+/-getseed
Enables or disables the option that reads the file *test*.seed to obtain a starting seed. If the test terminates because the maximum pass or error limit is reached, the seed from the last pass is saved in the file *test*.seed. If there are any problems reading the seed from this file, the test uses the default seed (O'33). If you select +getseed, do not select seed *n*. The default is -getseed.

getseed *file*
Gets a starting seed from *file*. *file* can contain a dump from a previous failure or a single seed value. If allowed to default, the test uses the seed value specified by +getseed or seed *n*.

help Generates an online help display that contains a command synopsis.

maxerr *n*
Sets the maximum number of errors (in octal). The default for *n* is O'1.

maxp *n* Sets the maximum number of passes (in octal). The default for *n* is O'1000. If cputime or time is set to a value other than 0, the specified option overrides maxp.

`+/-modes`

Enables or disables the option that tests exchange package mode bits. Only the BDM, AVL, FPE, ORE, and IUM mode bits are tested. Before the CPU test is run in the selected CPU the mode bits are tested. If the mode bits do not set and clear properly, the test will abort and report the failing mode bits. The mode bit test will not run when more than one CPU is selected. The default is `-modes`.

`+/-waitded`

Enables or disables the wait for a dedicated CPU. When this option is enabled, the diagnostic waits for the dedicated CPU to be freed before running. When this option is disabled, the diagnostic does not attempt to run in the dedicated CPU; instead, it exits normally and prints an informative message that the CPU was dedicated. The default is `-waitded`. See the `cpu`(8) man page for more information on dedicating a CPU.

`+/-parcel`

Enables or disables the option that forces dumped data to parcel format. `+parcel` forces data that would otherwise be in word format (64 bits in octal, with leading 0's) to parcel format (four groups of 16 bits in octal). Parcel format displays 2 words (8 parcels) per line. Word format displays 4 words per line. The default is `-parcel`.

`time` *h*:*m*:*s*

Sets the test execution time in elapsed (wall-clock) time. The time is specified in hours (*h*), minutes (*m*), and seconds (*s*); minutes and seconds; or just seconds.

Generally, actual execution time is within 1 second of the specified elapsed time. If `time` is allowed to default or is set to 0, the test uses the `maxp` value. However, if set to a value other than 0, `time` overrides `maxp`.

`+/-verbose`

Enables or disables the generation of informational messages. `+verbose` causes a line of output to be generated after each pass of the test. The default is `-verbose`.

`olcfpt` Test-specific Options

The test-specific command options for `olcfpt` are as follows:

`disable` *ilist*

Deselects specific instructions. Enter *ilist* in the following format:

n,*n*,...,*n*

n is the octal value in the *gh* field of the specific instruction. This option overrides the `enable` *ilist* option and any selected (+) or deselected (–) instruction options.

`enable` *ilist*

Selects specific instructions. Enter *ilist* in the following format:

n,*n*,...,*n*

n is the octal value in the *gh* field of the specific instruction. This option overrides any selected (+) or deselected (–) instruction options. When a test is run with default values for the +/– instruction options, and the `enable` *ilist* option is selected, only the instructions specified by the `enable` *ilist* option are run.

`+/-isolate`
Enables or disables the error isolation option. The default is `+isolate`.

`isop` *n* Sets the isolation pass limit to *n* (octal). During isolation, the test repeatedly executes the suspected failing sequence. If the sequence fails, the loop terminates and the test attempts to isolate the sequence further. If the sequence does not fail, the loop terminates after *n* passes, and the test assumes that the error is not in the tested sequence. The default for *n* is O'1000.

`numins` *n*
Sets the number of instructions to be generated. *n* can be any octal value within the range 1 through 20. The default for *n* is O'20.

`+/-repeat`
Enables or disables the option that repeats the first pass until the test terminates. This option is useful for recreating an error and is usually used with one of the following options: `seed` *n*, `+getseed`, or `getseed` *file*. The default is `-repeat` (the program generates new instructions and data after each pass).

`seed` *n* Sets the random seed to *n*. *n* can be any 64-bit octal value. If *n* is 0, the test reads the real-time clock and uses the value for the initial seed. The default for *n* is O'33. If this option is selected, do not select `+getseed` or `getseed` *file*.

`vl` *n* Sets the vector length to *n*. *n* can be any octal value in the range 0 through 100 (*n* can be up to O'200 for the CRAY C90 series and CRAY T90 series). If *n* is set to 0, a random value is used to initialize the test. The default for *n* is O'100 (O'200 for the CRAY C90 series and CRAY T90 series).

`+/-vload`
Selects or deselects vector instructions for the instruction buffer and, in the case of `-vload`, does not allow loading or saving of the vector registers. `-vload` overrides vector instructions selected by `+vector` and `enable` *ilist*. The default is `+vload`.

`+/-fpbits`, `+/-fprand`, `+/-random`
Selects or deselects specific data patterns. If allowed to default, all data patterns are run. If the `vl` option is 0, the vector length register is initialized with 6 bits of random data. The data patterns are as follows:

`fpbits`	Random number of consecutive 1-bits in the coefficient. Exponent data depends on the floating-point instruction.
`fprand`	Random bit generation in the coefficient. Exponent data depends on the floating-point instruction.
`random`	Random bit generation in a word.

`+/-fpadd`, `+/-fpmult`, `+/-fprecip`, `+/-scalar`, `+/-vector`

Selects or deselects specific instruction groups for the following options:

`fpadd`	Floating-point addition
`fpmult`	Floating-point multiply
`fprecip`	Floating-point reciprocal
`scalar`	Scalar instruction (destination)
`vector`	Vector instruction (destination)

If allowed to default, all instruction groups are run. The groups are as follows:

`fpadd`	`062`, `063`, `170` through `173`
`fpmult`	`064` through `067`, `160` through `167`
`fprecip`	`070`, `174`
`scalar`	`062`, `063`, `064` through `067`, `070`
`vector`	`160` through `167`, `170` through `174`

`olcm` Test-specific Options

The test-specific command options for `olcm` are as follows:

`+/-repeat`

Enables or disables the option that repeats the first pass until the test terminates. This option is useful for recreating an error and is usually used with one of the following options: `seed` *n*, `+getseed`, or `getseed` *file*. The default is `-repeat` (the program generates new instructions and data after each pass). This option is valid only for test section 6.

`section` *slist*

Selects the test sections to be executed. Enter *slist* in the following format:

n,n,...,n

n can be any of the following test sections, entered in any order (if allowed to default, all test sections are executed):

1 Central memory storage and scalar path test.

2 Central memory storage and T-register path test.

3 Central memory storage and B-register path test.

4 Central memory storage and vector register path test. Simplified algorithm for quick memory address testing.

5 Central memory storage and vector register path test using both vector logical units.

6 Central memory random-data test.

7 Central memory conflict test.

`seed` *n* Sets the random seed to *n*. *n* can be any 64-bit octal value. If *n* is 0, the test reads the real-time clock and uses the value for the initial seed. The default for *n* is O'33. If this option is selected, do not select `+getseed` or `getseed` *file*.

`stride` *n*
Indicates the stride used to go through memory. *n* is a positive value in the range O'1 through the maximum allowable memory. `stride` *n* must be less than or equal to `words` *n*. This option is applicable only to test section 1. The default for *n* is O'1.

`words` *n* Sets the number of words to be tested in central memory. *n* is a value in the range O'1000 through the maximum allowable memory. All values are rounded down to the nearest O'1000 words. The default for *n* is O'3000.

`olcrit` Test-specific Options

The test-specific command options for `olcrit` are as follows:

`+/-ranclus`
Enables or disables random cluster selection. The default is `-ranclus`.

`cluster` *n*
Selects a specific cluster. The value for *n* can range from 2 through the maximum number of clusters (the maximum number of clusters available is the value *numclstr* returned by the `target`(1) command). The default for *n* is 0, which selects a cluster number randomly. This option must be used to recreate a failure. If `cluster` *n* is selected, do not select the `+ranclus` option.

`disable` *ilist*
Deselects specific instructions. Enter *ilist* in the following format:

n,*n*,...,*n*

n is the octal value in the *gh* field or *ghijk* field of the specific instruction. If the *gh* field does not specify a unique instruction, use the *ijk* field to deselect a specific instruction. This option overrides the `enable` *ilist* option and any selected (+) or deselected (–) instruction options.

`+/-eis` This option applies only to the CRAY C90 series and CRAY T90 series. Enables or disables the 005400 CRAY C90 series and CRAY T90 series system instructions.

`enable` *ilist*
Selects specific instructions. Enter *ilist* in the following format:

n,n,...,n

n is the octal value in the *gh* field or *ghijk* field of the specific instruction. If the *gh* field does not specify a unique instruction, use the *ijk* field to deselect a specific instruction. This option overrides any selected (`+`) or deselected (`-`) instruction options. When a test is run with default values for the +/– instruction options, and the `enable` *ilist* option is selected, only the instructions specified by the `enable` *ilist* option are run.

When using the `enable` option to select any of the following instructions, `numins` *n* should be greater than 1: `34` through `37`, `56`, `57`, `76`, `77`, `100` through `130`, `150` through `153`, `176`, `177`.

`+/-hwbmm`
This option applies only to Cray PVP systems with bit matrix multiply (BMM) hardware. Enables or disables testing of BMM hardware. The default is that the test determines whether or not BMM hardware is present using the result from the `sysconf`(2) system call and performs testing as appropriate.

`+/-isolate`
Enables or disables the error isolation option. The default is `+isolate`.

`isop` *n* Sets the isolation pass limit to *n* (octal). During isolation, the test repeatedly executes the suspected failing sequence. If the sequence fails, the loop terminates and the test attempts to isolate the sequence further. If the sequence does not fail, the loop terminates after *n* passes, and the test assumes that the error is not in the tested sequence. The default for *n* is O'1000.

`numins` *n*
Sets the number of instructions to be generated. For `olcrit`, *n* can be any octal value within the range 1 through 2000. The default for *n* is O'200.

`+/-repeat`
Enables or disables the option that repeats the first pass until the test terminates. This option is useful for recreating an error and is usually used with one of the following options: `seed` *n*, `+getseed`, or `getseed` *file* (or `+cluster` together with `cluster` *n*). The default is `-repeat` (the program generates new instructions and data after each pass).

`seed` *n* Sets the random seed to *n*. *n* can be any 64-bit octal value. If *n* is 0, the test reads the real-time clock and uses the value for the initial seed. The default for *n* is O'33. If this option is selected, do not select `+getseed` or `getseed` *file*.

`vl` *n* Sets the vector length to *n*. *n* can be any octal value in the range 0 through 100 (*n* can be up to O'200 for the CRAY C90 series and CRAY T90 series). If *n* is set to 0, a random value is used to initialize the test. The default for *n* is O'0.

`+/-vload`
: Selects or deselects vector instructions for the instruction buffer and, in the case of `-vload`, does not allow loading or saving of the vector registers. `-vload` overrides vector instructions selected by `+vector` and `enable` *ilist*. The default is `+vload`.

`+/-bits`, `+/-onezero`, `+/-random`
: Selects or deselects specific data patterns. The default is all data patterns are run. The data patterns are as follows:

`bits`	Random number of consecutive 1-bits in a word
`onezero`	Random selection of all 1's or all 0's in a word
`random`	Random bit generation in a word

`+/-address`, `+/-ci`, `+/-cm`, `+/-fpadd`, `+/-fpmult`, `+/-fprecip`, `+/-int`, `+/-jump`, `+/-logical`, `+/-pop`, `+/-scalar`, `+/-shift`, `+/-shr`, `+/-vector`
: Selects or deselects specific instruction groups. The default is all instruction groups are selected.

`olcsvc` Test-specific Options

The test-specific command options for `olcsvc` are as follows:

`disable` *ilist*
: Deselects specific instructions. Enter *ilist* in the following format:

n,*n*,...,*n*

n is the octal value in the *gh* field of the specific instruction. This option overrides the `enable` *ilist* option and any selected (`+`) or deselected (`-`) instruction options.

`+/-eis` This option applies only to the CRAY C90 series and CRAY T90 series. Enables or disables the 005400 CRAY C90 series and CRAY T90 series system instructions.

`enable` *ilist*
: Selects specific instructions. Enter *ilist* in the following format:

n,*n*,...,*n*

n is the octal value in the *gh* field of the specific instruction. This option overrides any selected (+) or deselected (–) instruction options. When a test is run with default values for the +/– instruction options, and the `enable` *ilist* option is selected, only the instructions specified by the `enable` *ilist* option are run.

`+/-isolate`
: Enables or disables the error isolation option. The default is `+isolate`.

`isop` *n* Sets the isolation pass limit to *n* (octal). During isolation, the test repeatedly executes the suspected failing sequence. If the sequence fails, the loop terminates and the test attempts to isolate the sequence further. If the sequence does not fail, the loop terminates after *n* passes, and the test assumes that the error is not in the tested sequence. The default for *n* is O'1000.

`numpar` *n*

Sets the minimum number of parcels of vector instructions to be generated on each pass. *n* can be any octal value in the range 1 through 200. The default for *n* is O'100.

`+/-repeat`

Enables or disables the option that repeats the first pass until the test terminates. This option is useful for recreating an error and is usually used with one of the following options: `seed` *n*, `+getseed`, or `getseed` *file*. The default is `-repeat` (the program generates new instructions and data after each pass).

`seed` *n* Sets the random seed to *n*. *n* can be any 64-bit octal value. If *n* is 0, the test reads the real-time clock and uses the value for the initial seed. The default for *n* is O'33. If this option is selected, do not select `+getseed` or `getseed` *file*.

`+/-sgci`

Enables or disables testing of the scatter/gather/compressed index hardware. When allowed to default, the test determines the type of hardware configuration and sets the default value accordingly.

`vl` *n* Sets the vector length to *n*. *n* can be any octal value in the range 0 through 100 (*n* can be up to O'200 for the CRAY C90 series and CRAY T90 series). If *n* is set to 0, a random value is used to initialize the test. The default for *n* is O'0.

`vuse` *vrlist*

Lets you specify a vector register to be used in place of those usually generated. This option provides a mechanism for allowing the same instruction and data to be used with different registers and can help you determine whether a failure is related to a register or functional unit.

vrlist is a comma-separated list of up to eight vector register designators. The position of the element determines which vector register is being reassigned.

`+/-onezero`, `+/-random`, `+/-slide`

Selects or deselects specific data patterns. The default is all data patterns are run. The data patterns are as follows:

`onezero`	Random selection of all 1's or all 0's in a word.
`random`	Random bit generation in a word.
`slide`	Random number of consecutive 1's (0's) that slide in either direction through a filed of 0's (1's). Consecutive words contain the sliding pattern.

`+/-cm`, `+/-fpadd`, `+/-fpmult`, `+/-fprecip`, `+/-int`, `+/-logical`, `+/-pop`, `+/-shift`, `+/-lz`

Selects or deselects specific instruction groups. `lz` applies only to the CRAY C90 series and CRAY T90 series. The default is all instruction groups are run.

`olibuf` Test-specific Options

The test-specific command options for `olibuf` are as follows:

`+/-repeat`

Enables or disables the option that repeats the first pass until the test terminates. This option is useful for recreating an error and is usually used with one of the following options: `seed` *n*, `+getseed`, or `getseed` *file*. The default is `-repeat` (the program generates new instructions and data after each pass).

`seed` *n* Sets the random seed to *n*. *n* can be any 64-bit octal value. If *n* is 0, the test reads the real-time clock and uses the value for the initial seed. The default for *n* is O'33. If this option is selected, do not select `+getseed` or `getseed` *file*.

`+/-bits`, `+/-onezero`, `+/-ozpar`, `+/-random`, `+/-solid`

Selects or deselects specific data patterns. The default is all data patterns are run. The data patterns are as follows:

`bits`	Random number of consecutive 1-bits are run through the test area
`onezero`	Random patterns of all 1's or all 0's are run through the test area
`ozpar`	Random selection of all 1's or all 0's in each parcel are run through the test area
`random`	Random bit patterns are run through the test area
`solid`	A random pattern of either all 1's or all 0's is run through each instruction buffer area of the test area with one complement pattern

`olsbt` Test-specific Options

The test-specific command options for `olsbt` are as follows:

`+/-anycpus`

Overrides the `olsbt` restriction on the number of CPUs that can be selected. The default is `-anycpus` (from 2 to 4 CPUs can be selected). It is recommended that the `+anycpus` option be used only on lightly loaded dedicated systems or Cray Research STCO systems.

`cluster` *n*

Selects a specific cluster. The value for *n* can range from 2 through the maximum number of clusters (the maximum number of clusters available is the value *numclstr* returned by the `target`(1) command). The default for *n* is 0, which selects a cluster number randomly. This option must be used to recreate a failure.

`numins` *n*

Sets the number of instructions to be generated. *n* can be any octal value within the range 1 through 20. The default for *n* is O'20.

`+/-repeat`

Enables or disables the option that repeats the first pass until the test terminates. This option is useful for recreating an error and is usually used with one of the following options: `seed` *n*, `+getseed`, or `getseed` *file* (or `cluster` *n* for `olsbt`). The default is `-repeat` (the program generates new instructions and data after each pass).

`seed` *n* Sets the random seed to *n*. *n* can be any 64-bit octal value. If *n* is 0, the test reads the real-time clock and uses the value for the initial seed. The default for *n* is O'33. If this option is selected, do not select `+getseed` or `getseed` *file*.

`+/-bits`, `+/-onezero`, `+/-random`

Selects or deselects specific data patterns. The default is all data patterns are run. The data patterns are as follows:

`bits`	Random number of consecutive 1-bits in a word
`onezero`	Random selection of all 1's or all 0's in a word
`random`	Random bit generation in a word

EXAMPLES

The following example executes `olcrit` in CPU c. (To run a sequence of diagnostic tests, use the `runsequence` utility.) See the *Online Maintenance Tools Guide for Cray PVP Systems*, Cray Research publication SD–1012, for information on `runsequence`.

```
olcrit cpu c +getseed cputime 3 > critC.fail
```

The following example executes `olcsvc` in CPU a. This example would be used if `olcsvc` had failed previously without completing instruction isolation. The `getseed` option obtains the seed from the previous `olcsvc` failure, which is stored in `csvcA.fail`. The `+repeat` option will repeat the first pass until the test terminates. The `isop` option tries to isolate a failing instruction sequence up to 10,000 times. The ampersand (&) causes the entire command to execute in the background so that another prompt is immediately displayed and you can continue to use the system.

```
olcsvc cpu a cputime 2:00 getseed csvcA.fail +repeat isop 10000 > csvcA.iso &
```

The following examples illustrate the use of the `+/-waitded` option. In the first example, assume CPU b is dedicated. Because the `+waitded` option is enabled, `olcrit` waits until the dedicated CPU is freed (for however long it takes) and then runs in the CPU.

```
voyager$ olcrit cpu b +waitded
olcrit cpu b +waitded
olcrit started in cpu 1 on Tue Sep 17 17:47:23 1996
CRAY TS MODE
Bit matrix multiply detected.
olcrit reached maximum pass limit with 1000 passes and 0 errors on Tue Sep 17 17:47:25 1996
```

In this example, CPU b is dedicated, but `+waitded` is not specified.

```
voyager$ olcrit cpu b
olcrit cpu b
olcrit not run, dedicated cpu 1 Tue Sep 17 17:46:51 1996
```

SEE ALSO

`oldmon`(8) for information on running the online confidence tests in a down CPU environment under the `oldmon` down CPU monitor

`sysconf`(2) for information on the `sysconf` system call, which the `olcrit`(8) test uses to retrieve system implementation information

`target`(1) for information on the maximum number of clusters available (the value *numclstr*) for use with the `olcrit` or `olsbt cluster` *n* option in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

Online Maintenance Tools Guide for Cray PVP Systems, Cray Research publication SD–1012 (This document contains information private to Cray Research, Inc. It can be distributed to non-CRI personnel only with approval of the appropriate Cray manager.)

OLNET Online Diagnostic Network Communications Program Maintenance Manual for UNICOS, Cray Research publication SD–1021 (This document contains information private to Cray Research, Inc. It can be distributed to non-CRI personnel only with approval of the appropriate Cray manager.)

Disk Drive Offload User Guide (Version 2.0), publication CDM–1028–000 (This manual is Cray Research Proprietary; dissemination of this documentation 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

oldmon – Initiates, controls, and monitors the down CPU diagnostic tests

SYNOPSIS

/etc/oldmon [-d *cpulist* [-b *batchfile*]] [-q] [-u *cpulist*]

IMPLEMENTATION

Cray PVP systems (except CRAY T90 series and CRAY J90 series)

DESCRIPTION

The oldmon down CPU diagnostic monitor initiates, controls, and monitors the down CPU diagnostic tests. Two types of down CPU tests run under oldmon: confidence tests and maintenance tests. The down CPU confidence tests are online confidence tests that have been converted to run under oldmon (offline). The down CPU maintenance tests are derived from the offline diagnostic release (except for the CRAY C90 series).

By default, oldmon provides an interactive display that lets you load and execute diagnostic tests and examine the results. The diagnostic tests that oldmon can execute are in the /ce/oldmon directory.

The following options can be entered in any order.

-d *cpulist* Provides a list of CPUs to be downed. The list of CPUs has the following format:

n,*n*,...,*n*

n is a value in one of the following ranges:

0,1,2,...,*n*

a,b,c,...,*x*

By default, no CPUs are downed.

-b *batchfile* Provides the name of the batch input file. This file contains a list of diagnostic tests (and associated test options) to be executed. The tests will be executed in the first CPU specified in the -d *cpulist* option; therefore, the -d option is required with the -b option. If *batchfile* is '-', stdin will be read as the batch input. In batch mode, the interactive interface is not entered.

Batch input has the following format:

test [*param*[=]*val*] ... [+/-*option*] ... [>[>]*outfile*]

test is the name of the diagnostic test to be executed. For a list of available tests, see the `/ce/oldmon` directory. *param* is a parameter to be set with the corresponding *val*, which can be `cputime`, `maxp`, `time`, or a diagnostic DIB entry. (For a list of diagnostic DIB entries, see the help display for the online version of the offline test you want to run. For example, enter `olcrit help` to obtain the DIB entries for `offcrit`.)

If a test fails when executing in batch mode, the entire memory image will be saved to *outfile*. This file can be loaded into `oldmon` interactively to examine the results at a later time.

`-q` Exits `oldmon` after processing the command line. This option is useful for using `oldmon` to bring CPUs up or down without entering the interactive user interface.

`-u` *cpulist* Provides a list of CPUs that have been returned to normal system operations. The list of CPUs has the following format:

n,*n*,...,*n*

n is a value in one of the following ranges:

0,1,2,...,*n*

a,b,c,...,*x*

EXAMPLES

Example 1: Shows how to start the interactive user interface.

```
/etc/oldmon
```

Example 2: Shows two different ways to bring down CPU `c` and start the interactive user interface.

```
/etc/oldmon -d c
```

```
/etc/oldmon -d 2
```

Example 3: Brings down CPUs `a` and `c` without entering the interactive user interface.

```
/etc/oldmon -d a,c -q
```

Example 4: Executes the diagnostic tests in the `batch` file as batch input to CPU `d`.

```
/etc/oldmon -d d -b batch
```

Example 5: Again executes the diagnostic tests in the `batch` file as batch input to CPU `d`. When execution is complete, CPU `d` will be returned to normal system operations.

```
/etc/oldmon -d 3 -b batch -u 3
```

EXIT STATUS

When executing the interactive user interface, the exit code is undefined. In batch mode or with the `-q` option, the following exit codes are defined:

0 All diagnostic tests executed successfully.

1 An option on the command line or in the batch input file was not valid.

2 A diagnostic test in the batch input file failed.

3 A diagnostic test in the batch input file terminated abnormally. The termination could be caused by an operand range error, error exit, register parity error, and so on.

SEE ALSO

Online Maintenance Tools Guide for Cray PVP Systems, Cray Research publication SD–1012. (This document contains information private to Cray Research, Inc. It can be distributed to non-CRI personnel only with approval of the appropriate Cray manager.)

NAME

oldt – Provides online confidence testing of Cray Research disk drives

SYNOPSIS

/etc/diag/oldt -d *devicename* | -f *filename* [-D *dskname*] [-F *fsize*] [-n] [-P *fs_prcnt*] [-b *bsize*] [-S] [-m *maxpass*] [-p *patterns*] [-u *userpat*] [-s *seed*] [-t *tests*] [-C *hh:mm:ss*] [-T *hh:mm:ss*] [-a] [-v] [-q] [-h]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The oldt command tests Cray Research disk drives by first creating a new file or using an existing user-specified file (-f *filename*) in a specified directory and then performing reads and writes to that file, which resides on the specified device (-d *devicename*). If a miscompare is detected, oldt stops execution and displays error information including expected, actual, and difference for the word in error and 20 words before and after the error. Also, the physical location on the device is displayed.

-d *devicename* Defines the name of the device to be tested (required, unless -f *filename* is used). For example, 0235.4 on an IOS model E (IOS-E) or ESDI_100 on the CRAY EL series. You can find device names on an IOS-E system or the CRAY EL series by using the /etc/pddstat command. Specifying -d sds will test the secondary data segment (SDS) area of the SSD, but no actual file is created. The sds argument uses the ssbreak(2), sswrite(2), and ssread(2) system calls. If SSD is a defined device, it can also be tested, as long as an SSD-resident file system is mounted. By default, the file name created is /path/oldt_*devicename*. If the -d option is used with the -f option, you can specify the name for the file. (If you also specify a device with the -d option, specify only the file name with the -f *filename* option; do not specify a full path name). (You must specify the -S option along with the -d option to save the file.)

-f *filename* Specifies the name of the existing file to use for all reads and writes (required, unless -d *devicename* is used). This file will reside on the device to be tested. If this option is not specified, oldt creates a file with the default name /*path*/oldt_*devicename*. When using the -f option, specify the full path name (for example, /tmp/oldt_0235.4). The first time you run oldt to a disk, use the -d *devicename* option. The test creates a file with the above-mentioned type name and tells you where the file was created. If you also specify a device with the -d *devicename* option, specify only the file name with the -f *filename* option (do not specify a full path name). (You must specify the -S option with the -f option to save the file.)

`-D` *dskname* Defines the file system name where you want the work file created. */dev/dsk/name* can be found by issuing a `/etc/dmap -p` *device* command for the device to which you want to run `oldt`. The `-D` option must be used with the `-d` *devicename* option. This option lets you create the work file on a particular area of the disk. `oldt` will attempt to create a file only in the file system selected with `-D`. (Do not specify the `/dev/dsk` path for *dskname*.)

`-F` *fsize* Defines the file size of the work file. *fsize* can be a decimal block value between 1 and the maximum file size. The default is 1 sector. Do not specify `-F` with `-P`.

`-n` Indicates that the `-F` *fsize* value should be in decimal tracks rather than decimal blocks. Do not specify `-n` with `-P`.

`-P` *fs_prcnt* Defines the percentage of the file system on the selected device to create the work file. The maximum value is 98, which causes the work file being created to be the same size as 98% of the available free space of the partition on the selected device. Do not specify `-P` with `-F` or `-n`.

`-b` *bsize* Defines the size of the buffer to use for reads and writes to the work file. Specify the value of *bsize* in sectors. The default is 1 sector. The maximum value that may be specified is the number of sectors in 1 track. The larger the value of *bsize*, the more memory will be allocated by `oldt` (3 buffers of *bsize* are allocated during execution).

`-S` Saves the created work file. By default, the file is removed at test completion. If the device specified by the `-d` *devicename* option is `sds`, the `-S` option is not valid.

`-m` *maxpass* Defines the maximum number of passes to run the test. The default is 10 passes. If you specify `-m 0`, `oldt` creates a work file, but it does not run any tests. This feature is useful if you want to create a file on a particular disk (for example, `oldt -d 0235.4 -m 0 -S`).

`-p` *patterns* Defines the test patterns to run. Available patterns are as follows: `zeros`, `ones`, `chkbrd`, `chkbrdc`, `alt2`, `alt2c`, `alt4`, `alt4c`, and `rdm`. The default is all patterns are run. *patterns* is a comma-separated list (for example, `-p zeros,alt2,alt2c`). The pattern descriptions are as follows:

`zeros`	00000000000000000000
`ones`	01777777777777777777
`chkbrd`	01252525252525252525
`chkbrdc`	00525252525252525525
`alt2`	01777777777740000000
`alt2c`	00000000000037777777
`alt4`	01777770000037777600000
`alt4c`	00000077777740000177777

`rdm` Set from the real-time clock value or seed (`-s`)

`-u` *userpat* Specifies a user-defined octal data pattern (up to 64 bits) to run.

`-s` *seed* Specifies a user-defined decimal seed used for generating random data patterns and read/write tests. The default is the real-time clock value.

`-t` *tests* Defines the test sections to run. Available tests are as follows: `sw`, `sr`, `rw`, `rr`, `ss`, `rs`. The default is all test sections are run. (The seek tests are not performed if the device specified by the `-d` *devicename* option is `sds`.) *tests* is a comma-separated list (for example, `-t sw,sr`). The test descriptions are as follows:

`sw` Sequential write test

`sr` Sequential read test

`rw` Random write test

`rr` Random read test

`ss` Sequential seek test

`rs` Random seek test

`-C` *hh:mm:ss* Defines the amount of CPU time the test will run. If this option is used in conjunction with `-T`, the first option to be satisfied will stop the test. The `-m` *maxpass* value will be overridden.

`-T` *hh:mm:ss* Defines the amount of wall-clock time the test will run. If this option is used in conjunction with `-C`, the first option to be satisfied will stop the test. The `-m` *maxpass* value will be overridden.

`-a` Performs asynchronous reads and writes rather than the default synchronous reads and writes.

`-v` (Verbose) Echoes all file creation information, device information, and a pass counter to `stdout` and `stderr` as the test is running. The default is to echo only error information.

`-q` Performs a quick test. If you use the `-q` option, you need only specify the device (`-d`) or the file (`-f`) with `-q` (for example, `oldt -d 0235.4 -q`). When `-q` is specified, only the `sw` and `sr` tests, the `ones` pattern, and 1 pass are run.

`-h` Displays the help screen.

NOTES

If this command is installed with the default privilege assignment list (PAL), you must have an active `secadm`, `sysadm`, or `diagadm` category to use this command.

`oldt` depends on a character special device node existing in the `/dev/ddd` or `dev/ssdd` (SSD) directory in order to work.

EXAMPLES

Example 1: Creates a work file on device 0235.4 in the file system `/dev/dsk/tmp_40` (`-D tmp_40`) with the file name of `myfile` (`-f myfile`) and a default size of 1 sector. The test runs for 100 passes (`-m 100`) of the default tests and patterns (`all`), deletes the file (no `-S`), and then exits.

```
oldt -d 0235.4 -D tmp_40 -f myfile -m 100
```

Example 2: Creates a work file with a size of 100 blocks (`-F 100`) on device 0233.2, runs the default (10) passes of the default tests and patterns (all), saves the work file (`-S`), and then exits. All information is displayed during test execution (`-v`).

```
oldt -d 0233.2 -F 100 -S -v
```

Example 3: Uses the work file created in the previous example (`-f /tmp/oldt_0233.2`), increases the file size to 100 tracks (`-F 100 -n`), and runs the sequential write and read tests (`-t sw,sr`) using only the one/zero, one/zero-complement and user-selected patterns (`-p chkbrd,chkbrdc -u 1234`). After execution is completed, the work file is deleted (no `-S`).

```
oldt -f /tmp/oldt_0233.0 -F 100 -n -t sw,sr -p chkbrd,chkbrdc -u 1234
```

Example 4: Creates a work file named `test1` (`-f test1`) on device 2214.6. `test1` has a size of 20 blocks (`-F 20`), uses the default values for all other options, and saves the file at test completion (`-S`).

```
oldt -d 2214.6 -f test1 -F 20 -S
```

Example 5: Creates a work file with the default size of 1 sector on device 3226.0, runs the `sw` and `sr` tests using the `ones` pattern for 1 pass only, deletes the work file, and then exits.

```
oldt -d 3226 -q
```

Example 6: Tests the SDS space of the SSD. It allocates 2 units of 128 blocks each (`-F 200`), runs all tests and patterns for 30 passes (`-m 30`), and then exits.

```
oldt -d sds -F 200 -m 30 -v
```

SEE ALSO

`olhpa`(8) for information on the system error log formatter

Online Maintenance Tools Guide for Cray PVP Systems, Cray Research publication SD–1012. (This document contains information private to Cray Research, Inc. It can be distributed to non-CRI personnel only with approval of the appropriate Cray manager.)

NAME

olhpa – Formats UNICOS hardware errors logged by errdemon(8)

SYNOPSIS

/etc/diag/olhpa [-h] [-l] [-q] [-g [d] [m] *n*] [-cdnimptv] [-C *arg*] [-D *arg*] [-N *arg*] [-I *arg*] [-M *arg*] [-P *arg*] [-T *arg*] [-V *arg*] [-s *start*] [-e *end*] [-r] [-F *file*] [-A] [-z] [*errfile(s)*]

IMPLEMENTATION

All Model E based Cray Research systems

DESCRIPTION

The olhpa command formats hardware errors generated by UNICOS device drivers and logged by errdemon(8). Error types supported are local disk, network disk, tape, SSD, central memory, register parity, IOS communication (both COMM and HIPPI errors), and Cray MPP local memory. (SSD, register parity, and Cray MPP reports are not supported on the CRAY EL series.)

NOTE: olhpa is not supported on GigaRing based Cray Research systems. Instead, error information can be viewed with errpt(8) and the sysmon(8) GUI interface on the Cray system workstation (SWS). For more information, see the errpt(8) and sysmon(8) man pages and the *SWS-ION Administration and Operations Guide*, publication SG–2204.

Memory location on the CRAY J90 series is reported as three values (for example, 01/01/05) corresponding to CRAY J90s with a 2X2, 4X4, and 8X8 backplane (respectively). The correct memory location value should be selected based on the hardware configuration of the machine.

Errors are grouped by hardware type and are displayed in the format specified. Possible formats include quick, regular, long, custom, and graph. The quick report provides a summary of errors for each hardware type. The regular report displays several key fields of the error record on one line. The long report displays all available error fields and is formatted in a paragraph form. The custom report prints all user-specified fields on the same line. The graph report displays a bar graph of errors in intervals specified by hours, days, or minutes.

By default, olhpa reads error records from /usr/adm/errfile. You can override this by specifying error file(s) on the command line. By default, olhpa uses the current execution time for the ending time and 30 days minus the ending time for the start time. Only errors within the given time interval are displayed.

The olhpa command accepts the following options:

-h Displays the command synopsis.

-l Displays a long version of an error report, which contains the information as it was found in errpt(8). A long report created with the -l option should look just like an errpt report.

-q Displays only the summary information.

`-g [d] [m] n`
Displays a bar graph in which *n* is the interval in hours or (optionally) days or minutes (`d` or `m`).

`-c` Displays a report of all register parity errors in the specified *errfile(s)* and between the specified times. This option is not valid for the CRAY EL series.

`-d` Displays a report of all local disk errors in the specified *errfile(s)* and between the specified times.

`-n` Displays a report of all network disk errors in the specified *errfile(s)* and between the specified times.

`-i` Displays a report of all IOS communication errors in the specified *errfile(s)* and between the specified times.

`-m` Displays a report of all memory errors in the specified *errfile(s)* and between the specified times.

`-p` This option is available at Cray MPP sites only. Calls the `mpphpa`(8) command, which displays a report of all Cray MPP errors in the specified *errfile(s)* and between the specified times. This option is not valid for the CRAY EL series.

`-t` Displays a report of all tape errors in the specified *errfile(s)* and between the specified times.

`-v` Displays a report of all SSD errors in the specified *errfile(s)* and between the specified times. This option is not valid for the CRAY EL series.

`-C` *arg*, `-D` *arg*, `-N` *arg*, `-I` *arg*, `-M` *arg*, `-P` *arg*, `-T` *arg*, `-V` *arg*
Displays register parity, local disk, network disk, IOS communication, memory, Cray MPP local memory, tape, and SSD errors, respectively. (SSD, register parity, and Cray MPP local memory reports are not supported on the CRAY EL series.) The `-P` option is available at Cray MPP sites only. This option calls the `mpphpa`(8) command, which displays a report of all Cray MPP errors in the specified *errfile(s)* and between the specified times. The required argument can be one of the following:

`S,`*field*`[!]=`*value*`[,`*field*`[!]=`*value*`]`	Searches.
`P[,+],`*field*`[,`*field*`]`	Prints fields.
`H`	Displays an associated help menu.

`-s` *start* *start* can be one of the following: *hh*`:`*mm*`,`*MM/DD/YY*, *hh*`:`*mm*, or *MM/DD/YY*.

`-e` *end* *end* can be one of the following: *hh*`:`*mm*`,`*MM/DD/YY*, *hh*`:`*mm*, or *MM/DD/YY*.

`-r` Displays the error records in raw mode.

`-F` *file* Stores all binary records that are displayed to the file specified by *file*. *file* is in the standard `errfile` format and can be read by `olhpa` or `errpt`(8).

`-A` Abbreviates the IOS-E disk summary to display only recovered, corrected, unrecovered, and recovery off disk error totals.

`-z` Suppresses printing of header and summary information if the graph (`-g`) or quick (`-q`) option is not specified.

errfile(s) Specifies the *errfile(s)* to be read. *errfile* can be any file that was produced by `errdemon`(8) and read by `errpt`(8).

NOTES

If this command is installed with the default privilege assignment list (PAL), you must have an active `secadm`, `sysadm`, or `diagadm` category to use this command.

EXAMPLES

Example 1: Displays in the regular format all tape errors in the `/usr/adm/errfile` error file from the past 30 days.

```
olhpa -t
```

Example 2: Displays in the long report format all disk, tape, and IOS communication errors from a starting date of 2/3/93 until the present date and time. All error records are read from the `/usr/adm/errfile1` error file.

```
olhpa -s 2/3/93 -ldti /usr/adm/errfile1
```

Example 3: Displays for the past 30 days and in the long format all disk errors for which the device type is DD-49. Error records are read from all files in the format `/usr/adm/errfile*` (that is, `errfile1`, `errfile2`, and so on).

```
olhpa -lDS,dt=dd49 /usr/adm/errfile*
```

Example 4: Displays for the past 30 days and in the regular format all disk errors for which the device type is `dd60` and the cylinder value is not 1572. Error records are read from the `/usr/adm/errfile` error file.

```
olhpa -DS,dt=dd60,cyl!=1572
```

SEE ALSO

`errdemon`(8) for information on the UNICOS error-logging daemon
`errpt`(8) for information on the `errpt` command, which also processes data logged by `errdemon`

UNICOS MAX:
`mpphpa`(8) for information on formatting hardware errors logged by `mppd`(8), the Cray MPP daemon

Online Maintenance Tools Guide for Cray PVP Systems, Cray Research publication SD–1012. (This document contains information private to Cray Research, Inc. It can be distributed to non-CRI personnel only with approval of the appropriate Cray manager.)

CRAY T3D Diagnostic Reference Manual, publication CDM–0601–000 (This manual is Cray Research Proprietary; dissemination of this documentation 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

`olnet` – Detects and isolates problems in networks associated with Cray Research systems

SYNOPSIS

`/etc/diag/olnet` [`-a` *input-channel-timeout*] [`-b` *input-short-burst-flag*] [`-c` *input-channel-state-flag*] [`-d` *input-drop-connection-flag*] [`-e` *input-send-i-field-flag*] [`-f` *input-i-field-mask*] [`-g` *input-software-timeout*] [`-h` *input-header-usage-flag*] [`-i` *input-i-field-value*] [`-l` *message-length*] [`-m` *messages-per-pass*] [`-n`] [`-p` *pass-count*] [`-r` *remote-address*] [`-A` *output-channel-timeout*] [`-B` *output-short-burst-flag*] [`-C` *output-channel-state-flag*] [`-D` *output-drop-connection-flag*] [`-E` *output-send-i-field-flag*] [`-F` *output-i-field-mask*] [`-G` *output-software-timeout*] [`-H` *output-header-usage-flag*] [`-I` *output-i-field-value*] [`-L`] [`-M` *test-mode*] [`-N` *errorfile*] [`-P` *pattern-type*] [`-R` *messages/ack-ratio*] [`-S` *SSD-address*] [`-T` *tracefile*] *test-name input-device-name* [*output-device-name*]

`/etc/diag/olnet` [*comma-separated list of OLNET menu choices*]

IMPLEMENTATION

All Cray Research Model E systems except CRAY J90 series and CRAY EL series

DESCRIPTION

The OLNET online diagnostic network communications program provides diagnostic testing for the following systems and hardware associated with Cray Research systems:

System/OS	Hardware	Test Configuration(s)
UNICOS IOS-E	Cray LOSP (CCA1)	Cable loopback
	IBM FEI-1 (standard and data-streaming)	End-to-end (OLNET active/passive) with IBM MVS and VM Data-streaming FEI internal and external loopback
	FEI-3 (VME)	Cray cable loopback End-to-end (OLNET active/passive)
	HIPPI	Cable loopback Software (PS32) loopback Hardware loopback through Ultra device End-to-end (OLNET active/passive)
	NSC A series adapters	Remote adapter loopback End-to-end (OLNET active/passive)

System/OS	Hardware	Test Configuration(s)
	NSC N (DX) series HYPERchannel or FDDI adapters	Local adapter loopback Remote adapter loopback End-to-end (OLNET active/passive) Remote adapter statistics
	FDDI (FCA-1)	End-to-end (OLNET active/passive) Ring loopback IOS software echo
	FDR-4	Local FDR loopback Remote FDR loopback Address loopback Data error injection Address/block length error injection Control error injection Synchronization error injection Framing pulse error injection Block length error injection High bias error injection Low lamp error injection SSD loopback
Cray MPP	n/a	Software loopback Write to MPP Read from MPP Type statistics IOG statistics IOG echo
IBM/MVS IBM/VM	FEI-1 (standard and data-streaming)	External FEI loopback End-to-end (OLNET active/passive)
	NSC A series adapters	Remote adapter loopback End-to-end (OLNET active/passive)
	NSC N (DX) series HYPERchannel or FDDI adapters	Local adapter loopback Remote adapter loopback End-to-end (OLNET active/passive)
Sun Workstation	NSC A or NSC N (DX) series HYPERchannel or FDDI adapters	Local adapter loopback Remote adapter loopback End-to-end (OLNET active/passive)

System/OS	Hardware	Test Configuration(s)
SUN OWS/MWS Workstation	VME FEI-3	End-to-end (OLNET active/passive) Cable loopback

For a detailed description of supported systems and OLNET test configurations, see the "Supported Configurations" section in the *OLNET Online Diagnostic Network Communications Program Maintenance Manual for UNICOS*, Cray Research publication SD–1021.

For information on accessing and building non-CRI source files, see the "OLNET Build Procedures" section in the *OLNET Online Diagnostic Network Communications Program Maintenance Manual for UNICOS*, Cray Research publication SD–1021.

The OLNET executables are located as follows:

System	Directory/Location
UNICOS	`/ce/bin`
Sun/UNIX OWS/MWS	`/home/`*ows*`/cri/smarte/bin/olnet`, where *ows* is the name of the OWS, for example, `/home/ows1601/cri/smarte/bin/olnet`
Sun/UNIX Workstation	Determined by installer
IBM/MVS IBM/VM	Determined by installer

OPTIONS

The `olnet` program accepts the following options:

`-a` *input-channel-timeout*
: Specifies the channel time-out value for the input HIPPI channel. HIPPI test only.

`-b` *input-short-burst-flag*
: Specifies whether the short burst flag should be turned on. Valid values are `YES` or `NO`. HIPPI test only. IOS model E only.

`-c` *input-channel-state-flag*
: Specifies the channel state of the input HIPPI channel. Valid values are `UP` or `DOWN`. HIPPI test only.

`-d` *input-drop-connection-flag*
Specifies if connections should be dropped between data transfers for the input HIPPI channel. If enabled, the connection will be broken down and rebuilt for each data transfer. If disabled, the connection will stay active until the channel is closed. Valid values are `ENABLED` or `DISABLED`. HIPPI test only.

`-e` *input-send-i-field-flag*
Valid values are `ENABLED` or `DISABLED`. HIPPI test only.

`-f` *input-i-field-mask*
The value should be a 32-bit hexadecimal number. HIPPI test only.

`-g` *input-software-timeout*
Specifies the software time-out value for the input HIPPI channel. HIPPI test only.

`-h` *input-header-usage-flag*
Valid values are `ENABLED` or `DISABLED`. HIPPI test only.

`-i` *input-i-field-value*
Specifies the value to be used for the I-field for the input HIPPI channel. The value should be a 32-bit hexadecimal number. HIPPI test only.

`-l` *message-length*
Specifies the length of data to be sent or received. The *message-length* value is test-dependent but can be set to `RN` (random). The default is 100.

`-m` *messages-per-pass*
Specifies how many messages are to be sent for each pass of the chosen test. The *messages-per-pass* value can range from 1 through 1000000. The default is 10.

`-n` Specifies that `olnet` should continue on error.

`-p` *pass-count*
Specifies how many passes are to be performed of the chosen test. The *pass-count* value can range from 1 through 1000000. The default is 1.

`-r` *remote-address*
Specifies the source address (the machine on which OLNET is currently running). The *remote-address* value is dependent on *test-name* and is valid only for the following tests:

`FDT` The value should be either the 48-bit IEEE Ethernet-style address or a machine name from the `/etc/ethers` file. The default is 0:0:0:0:0:0.

`HI` The value should be a decimal number for the remote logical path.

`NT` The value should be the 16-bit hexadecimal HYPERchannel adapter address. The default is 0000.

`VT` The value should be the 8-bit hexadecimal path number. The default is 0000.

`-A` *output-channel-timeout*
Specifies the channel time-out value for the output HIPPI channel. HIPPI test only.

`-B` *output-short-burst-flag*
Specifies whether the short burst flag should be turned on for the output HIPPI channel. Valid values are `YES` or `NO`. HIPPI test only. IOS model E only.

`-C` *output-channel-state-flag*
Specifies the channel state of the input HIPPI channel. Valid values are `UP` or `DOWN`. HIPPI test only.

`-D` *output-drop-connection-flag*
Specifies if connections should be dropped between data transfers for the output HIPPI channel. If enabled, the connection will be broken down and rebuilt for each data transfer. If disabled, the connection will stay active until the channel is closed. Valid values are `ENABLED` or `DISABLED`. HIPPI test only.

`-E` *output-send-i-field-flag*
Valid values are `ENABLED` or `DISABLED`. HIPPI test only.

`-F` *output-i-field-mask*
The value should be a 32-bit hexadecimal number. HIPPI test only.

`-G` *output-software-timeout*
Specifies the software time-out value for the output HIPPI channel. HIPPI test only.

`-H` *output-header-usage-flag*
Valid values are `ENABLED` or `DISABLED`. HIPPI test only.

`-I` *output-i-field-value*
Specifies the value to be used for the I-field for the output HIPPI channel data transfer. The value should be a 32-bit hexadecimal number. HIPPI test only.

`-L` Specifies that OLNET should be locked into memory.

`-M` *test-mode*
Specifies the test mode to use. Valid arguments are as follows:

`AA`	Asynchronous active mode
`AEI`	Address/block length error injection; FDR test only
`AL`	Address loopback; FDR test only
`AM`	Active mode
`AP`	Asynchronous passive mode
`CC`	Control cable test; VME test only
`CL`	LOSP cable loopback; NSC, VME, FEI, and FEI-station tests only
`BLEI`	Block length error injection; FDR test only
`CEI`	Control error injection; FDR test only
`DC`	Disable disconnect interrupt; VME test only

DEI	Data error injection; FDR test only
DISEDB	DS FEI external double-byte loopback; FEI, and FEI-station tests only
DISESB	DS FEI external single-byte loopback; FEI, and FEI-station tests only
DISIDB	DS FEI internal double-byte loopback; FEI, and FEI-station tests only
DISISB	DS FEI internal single-byte loopback; FEI, and FEI-station tests only
DL	Display driver statistics; VME and NSC tests only
ECHO	IOS software loopback; FDDI test only
FPEI	Framing pulse error injection; FDR test only
FY	fymc test; VME test only
HBEI	High bias error injection; FDR test only
HM	HYPERchannel map; NSC test only
IO	I/O master clear; VME test only
IOGE	IOG echo; MPP test only
IS	Read MPP IOG statistics; MPP test only
LBK	Loopback; FDDI, HIPPI, FDR, and MPP tests only
LL	Local loopback; NSC and FDR tests only
LLEI	Low lamp error injection; FDR test only
MC	Master clear; VME test only
MPPR	Read from the MPP; MPP test only
MPPW	Write to the MPP; MPP test only
PL	Program loop
PM	Passive mode
RL	Remote loopback; NSC and FDR tests only
RP	Read single adapter profile; NSC test only
RPM	Map adapter profiles; NSC test only
RS	Remote statistics; NSC test only
S	Local statistics; NSC test only
SE	IOS software echo; VME test only
SEI	Synchronization error injection; FDR test only
VR	Dump VME registers; VME test only
WE	WIN echo; VME test only

XM Xmapping routine; NSC test only

YS Read MPP YPE statistics; MPP test only

The default is AM.

-N *errorfile*

Specifies the name of the file to which olnet will write error output.

-P *pattern-type*

Specifies the data pattern type to use for the chosen test. Valid arguments are as follows:

AD Address

AO All ones

AP All patterns

AZ All zeros

BT Bits

RN Random

SO Sliding ones

SZ Sliding zeros

The default is AD. The *pattern-type* value is not case-sensitive.

-R *messages/ack-ratio*

When -M *test-mode* is set to AA or AP only, specifies the ratio of messages sent to acknowledgments received. The *messages/ack-ratio* value can be of the form *mm*:1, *mm*:0, or *mm*:RN. The default is 3:1.

-S *SSD-address*

Specifies the SSD address to be used with the address loopback and SSD loopback tests. The default is 0.

-T *tracefile*

Specifies the name of the file to be created that will contain OLNET trace information. The default is not to create a trace file.

test-name

Specifies the kind of device being tested. Valid values are as follows:

FDR4 FDR-4 test

FDT FDDI test

FT FEI test

HI HIPPI test

MPP Cray MPP test

NT NSC test

VT VME test

The *test-name* value is not case-sensitive.

input-device-name
Specifies the name of the character-special file to be used for the test. If *output-device-name* is used, *input-device-name* specifies the name of the character-special file to be used for input (HI only). The *input-device-name* value is case-sensitive.

output-device-name
If specified, *output-device-name* specifies the name of the character-special file to be used for output (HI only). The *output-device-name* value is case-sensitive.

NOTES

If this command is installed with the default privilege assignment list (PAL), you must have an active secadm, sysadm, or diagadm category to use this command. If the PRIV_SU configuration option is enabled, you must be root to use this command.

SPECIAL NOTES ON THE FDR-4 TESTS

The FDR-4 test suite was designed to diagnose the components of the shared file system configuration. Those components are: the local and remote FDR-4 fiber optic VHISP channel adapters, the MPX-24 VHISP channel multiplexer, and one or more SSDs.

When an SSD is configured into a system, it is sliced up in much the same way as a traditional disk device. Nodes in the /dev/pdd directory can access the entire SSD address range and are referred to as *control slices*. Nodes in the /dev/ssdd directory are configured to access only specific ranges of SSD addresses and are referred to as *noncontrol slices.*

Kernel support for diagnostic access to the SSD VHISP channels makes no attempt to differentiate a diagnostic request from a traditional SSD I/O operation. The only validation the kernel does is to make sure the request does not attempt to access addresses outside the valid range for the slice. However, the error injection tests are set up by turning on some high-order bits of the SSD address register. The outcome is that, depending on the size of slice chosen, the kernel may not allow some of the error injection tests to be executed. For that reason, the error injection tests are restricted to control slices that will always have a large enough address range to accomodate SSD addresses set by OLNET.

With the exception of the SSD loopback test, the hardware forces all transfers to be four SSD blocks in size. An SSD block is sixty-four 64-bit words. For consistency, the SSD loopback test also uses this size. Also, even though the address loopback test allows you to specify a block length value, the transfer will be for four SSD blocks.

Following is a brief description of each test that OLNET is capable of performing:

Local FDR loopback
Available to either kind of slice. Data is written to, and then read from, the local FDR module and compared.

Remote FDR loopback
: Available to either kind of slice. Data is written to, and then read from, the remote FDR module and compared.

Address loopback
: Available to either kind of slice. Data is read from the MPX module and compared. The data returned is a combination of the SSD address and block length you specify. By specifying various SSD addresses and block lengths, you can test all bit combinations for those two fields.

Address/block length error injection
: Available only when using a control slice. You can choose which fiber the error will be injected into as well as whether the error is injected at the local or remote FDR. The error will be detected by the maintenance workstation (MWS).

Data error injection
: Available only when using a control slice. You can choose which fiber the error will be injected into as well as whether the error is injected at the local or remote FDR. The error will be detected by the MWS.

Control error injection
: Available only when using a control slice. You can choose which fiber the error will be injected into as well as whether the error is injected at the local or remote FDR. The error will be detected by the MWS.

Synchronization error injection
: Available only when using a control slice. You can choose which fiber the error will be injected into as well as whether the error is injected at the local or remote FDR. The error will be detected by the MWS.

Framing pulse error injection
: Available only when using a control slice. You can choose which fiber the error will be injected into as well as whether the error is injected at the local or remote FDR. The error will be detected by the MWS.

Block length error injection
: Available only when using a control slice. You can choose which fiber the error will be injected into as well as whether the error is injected at the local or remote FDR. The error will be detected by the MWS.

High bias error injection
: Available only when using a control slice. You can choose which fiber the error will be injected into as well as whether the error is injected at the local or remote FDR. The error will be detected by the MWS.

Low lamp error injection
: Available only when using a control slice. You can choose which fiber the error will be injected into as well as whether the error is injected at the local or remote FDR. The error will be detected by the MWS.

SSD loopback

Available only when using a noncontrol slice. Data is written to, then read from, the SSD address specified. This is potentially the most dangerous test because SSD memory is directly manipulated. Because of the risk of corrupting customer data, this test is not made available when using a control slice. Typically, a noncontrol slice is configured for diagnostic use.

EXAMPLES

OLNET is intended to be used as an interactive menu-driven utility, but it can also be executed from the command line.

To initiate OLNET execution, enter `/ce/bin/olnet` from the command line. The main menu is displayed as follows (this example shows a menu from a Cray PVP system; the menu may vary, depending on system type):

```
                 OLNET A.1 MAIN MENU
                 ---------------

     YOUR SYSTEM:  NAME = sn1703c, NODE NAME = cool, RELEASE = 8.1.0ac,
                   VERSION = d81.13, MACHINE = CRAY Y-MP

         COMMAND     DESCRIPTION
         -------     -----------
            FDT - Call the FDDI test.
            FT - Call the FEI test.
            HI - Call the HIPPI test.
            NT - Call the NSC test.
            VT - Call the VME test.
            FDR4 - Call the FDR-4 test.
            MPP - Call the MPP test.
            HELP  - Get HELP information about this menu.
            PLK - Enable/disable PROGRAM LOCK.  Lock is: DISABLED
            QT - Quit OLNET.

          OLNET IS A PROPRIETARY PRODUCT OF CRAY RESEARCH INC.

        Enter a command:
```

In a typical OLNET menu, the leftmost column contains a command, followed by a brief command description, followed by an associated value (if applicable).

From the preceding menu, you can select the NSC test by entering the NT command, followed by pressing the <RETURN> key. The NSC menu is displayed as follows:

```
          ********** NSC TEST INITIAL MENU **********

        NSC INITIAL MENU COMMANDS          Current Value(if applicable)
        -------------------------          ----------------------------
         DV - Device path ----------------->  undefined

         DPM - NSC Device Path Menu.

         HELP - Get HELP information about this menu.

         TMM - Select the OLNET NSC Test Mode Menu.

         RT - Return to the OLNET Main menu.

      Enter a command:
```

You are now ready to execute the OLNET NSC test.

In the following example, OLNET is executed without using the built-in menu system.

The configuration of machine A is as follows: the device /dev/comm/n00-32/lp04 is an NSC HYPERchannel adapter with HYPERchannel address BF00. The logical path is 04, by convention. Machine A will be the passive mode side.

The configuration of machine B is as follows: the device /dev/comm/n00-32/lp0a is an NSC HYPERchannel adapter with HYPERchannel address 6A00. The logical path is 0a, by convention. Machine B will be the active mode side.

Command line for machine A:

```
olnet -p 100 -l 1024 -M pm 6A0A nt /dev/comm/n00-32/lp04
```

Command line for machine B:

```
olnet -p 100 -l 1024 -M am BF04 nt /dev/comm/n00-32/lp0a
```

For detailed information on test execution and command descriptions, see the *OLNET Online Diagnostic Network Communications Program Maintenance Manual for UNICOS*, Cray Research publication SD–1021.

SEE ALSO

`fddi`(4), `hippi`(4), `hy`(4), `np`(4), `vme`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

OLNET Online Diagnostic Network Communications Program Maintenance Manual for UNICOS, Cray Research publication SD–1021. (This document contains information private to Cray Research, Inc. It can be distributed to non-CRI personnel only with approval of the appropriate Cray manager.)

FDR-4/MPX-24 Engineering Note, publication PRN–0917 (This manual is Cray Research Proprietary; dissemination of this documentation 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

oper, screen – Invokes the operator display manager

SYNOPSIS

/usr/lib/msg/oper [-r *rate*] [*command* [*arguments*]]

/etc/screen [-r *rate*] [*command* [*arguments*]]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

oper and screen are functionally identical, except where noted.

The oper command displays informative messages and action messages and runs other commands as refresh displays. Display commands, such as who(1), ps(1), infd(8), and msgd(8), are defined in a configuration file, as are commands such as vi(1) and ex(1), which require full control of the screen.

The screen is divided into two parts. The bottom three lines are used for input and for running non-display commands. The remainder of the screen is used for refresh displays. When in oper, you can use the top line of the display to notify the operator of something that requires attention.

The valid command-line options and arguments are as follows:

-r *rate* Refresh rate in seconds. The default is 10 seconds.

command [*arguments*]
Name of a *command* and any *arguments* to the *command*.

The following commands affect the oper screen:

ref *time* Sets the refresh rate for the refresh display to a specified length of *time* (in seconds). The minimum refresh rate is 1 second.

redraw Redraws the screen.

snap *file* Sends a copy of the screen to *file*.

exit
ex Exits from oper.

? Help.

- Displays the previous page.

+
space bar Displays the next page.

The following commands display informative and action messages when in `oper`:

`infd` Displays informative messages. See `infd`(8) and `msgd`(8) for more information.

`msgd` Displays action messages, such as tape mount messages. See `msgd`(8) for more information.

The following command lets you respond to an action message when in `oper`:

`rep` *msg_number* [*reply_string*]
Allows an operator to respond to an action message, such as a tape mount message. See `rep`(8) for more information.

NOTES

If this command is installed with a privilege assignment list (PAL), a user who is assigned the following privilege text upon execution of this command is allowed to perform the action shown:

Privilege Text	Action
`both`	Messages are not subject to security label restrictions.

If this command is installed with a PAL, a user with one of the following active categories is allowed to perform the action shown:

Active Category	Action
`system`, `secadm`, `sysadm`, `sysops`	Allowed to use this command to view all messages.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command to view all messages.

FILES

`$HOME/.operrc`	Optional configuration file
`/usr/lib/oper.rc`	Default configuration file

SEE ALSO

`infd`(8), `msgd`(8), `msgdaemon`(8), `msgdstop`(8), `rep`(8)

`msgi`(1), `msgr`(1), `privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

ospf_monitor – Monitors OSPF gateways

SYNOPSIS

ospf_monitor *mon_db_file*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The ospf_monitor command queries open shortest path first (OSPF) routers. The ospf_monitor command operates in interactive mode. It allows the user to query the various OSPF routers to provide detailed information on input/output (I/O) statistics, error logs, link-state databases, autonomous system (AS) external databases, the OSPF routing table, configured OSPF interfaces, and OSPF neighbors.

The ospf_monitor command accepts the following option:

mon_db_file Specifies the complete path name of a database composed of records configuring destinations for ospf_monitor remote commands. Each destination record is a single-line entry that lists the destination Internet Protocol (IP) address, the destination host name, and an OSPF authentication key (if authentication is activated by the destination). Since authentication keys may be present in the destination records, it is recommended that general access to this database be restricted.

See RFC 1583 (OSPF Specification, version 2) for details about OSPF database and packet formats.

Commands

When entering interactive mode, ospf_monitor presents the following prompt:

```
[ # ] dest command params >
```

You can enter any of the ospf_monitor interactive commands at this prompt. The command line length must be less than 200 characters. Interactive commands can be interrupted at any time using a keyboard interrupt. ospf_monitor uses both local and remote commands.

Local commands

The following list describes the ospf_monitor local commands:

? Displays all local commands and their functions.

?R Displays all remote commands and their functions.

d Displays all configured destinations. This command displays *dest_index*, the IP address, and the host name of all potential ospf_monitor command destinations configured in *mon_db_file*.

h Displays the command history buffer showing the last 30 interactive commands.

x Exits the ospf_monitor program.

@ *remote_command*
Sends *remote_command* to the same (previous) destination.

@*dest_index remote_command*
Sends *remote_command* to the configured destination *dest_index*.

F *filename* Sends all `ospf_monitor` output to *filename*.

S Sends all `ospf_monitor` output to `stdout`.

Remote commands

The following list describes the `ospf_monitor` remote commands:

a *area_id type ls_id adv_rtr*
Displays link-state advertisement. The *area_id* parameter specifies the OSPF area for which the query is directed. The `ls_id` parameter specifies the IP address of the link-state advertisement, either network or host, depending on the type of request. The *adv_rtr* parameter specifies the router ID of the router which originated this link-state advertisement. The *type* parameter specifies the type of advertisement to request and can have one of the following values:

1 Requests the router links advertisements. They describe the collected states of the router's interfaces. For this type of request, set the *ls_id* field to the router ID of the originating router.

2 Requests the network links advertisements. They describe the set of routers attached to the network. For this type of request, set the *ls_id* field to the IP interface address of the network's designated router.

3 Requests the summary link advertisements describing routes to networks. They describe inter-area routes, and enable the condensing of routing information at area borders. For this type of request, set the *ls_id* field to the IP address of the destination network.

4 Requests the summary link advertisements describing routes to AS boundary routers. They describe inter-area routes, and enable the condensing of routing information at area borders. For this type of request, set the *ls_id* field to the router ID of the described AS boundary router.

5 Requests the AS external (ASE) link advertisements. They describe routes to destinations external to the AS. For this type of request, set the *ls_id* field to the IP address of the destination network.

c Displays a cumulative log. This log includes I/O statistics for monitor request, hello, data base description, link-state request, link-state update, and link-state acknowledgment packets. Area statistics describe the total number of routing neighbors and number of active OSPF interfaces. Routing table statistics are summarized and reported as the number of intra-area routes, inter-area routes, and AS external database entries.

e Displays cumulative errors. This log reports the various error conditions that can occur between OSPF routing neighbors and shows the number of occurrences for each.

`l` *retrans* Displays the link-state database (except for ASE link advertisements). This table describes the routers and networks making up the AS. If *retrans* is specified and not 0, the retransmit list of neighbors present in the link-state database is displayed.

`A` *retrans* Displays the AS external database entries. This table reports the advertising router, forwarding address, age, length, sequence number, type, and metric for each AS external route. If *retrans* is specified and not 0, the retransmit list of neighbors present in the ASE database is displayed.

`o` Displays the OSPF routing table. This table reports the AS border routes, area border routes, summary AS border routes, and networks currently managed using OSPF.

`I` Displays all interfaces. This report shows all interfaces configured for OSPF. Information reported includes the area, interface IP address, interface type, interface state, cost, priority, and the IP address of the designated router (DR) and backup designated router (BDR) for the network.

`N` Displays all OSPF routing neighbors. Information reported includes the area, local interface address, router ID, neighbor IP address, state, and mode.

SEE ALSO

`gated`(8), `gdc`(8)

`gated-config`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

RFC 1583 (OSPF Specification, version 2)

COPYRIGHT INFORMATION

NAME

pac – Collects and displays printer/plotter accounting information

SYNOPSIS

/etc/pac [-c] [-m] [-P*printer*] [-p*price*] [-r] [-s] [*names*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The pac command reads the printer/plotter accounting files, accumulates the number of pages or feet (for raster devices) of paper consumed by each user, and prints out how much each user consumed in pages or feet and dollars. By default, statistics are printed for every user who has used any paper.

The pac command accepts the following options:

-c	Sorts the output by cost. By default, the output is sorted alphabetically by name.
-m	Ignores the host name in the accounting file. This lets a user on multiple machines have all of his or her printing charges grouped together.
-P*printer*	Causes accounting to be done for the specified *printer*. If you omit this option, accounting is done for the default printer (site-dependent) or for the printer that is the PRINTER environment variable value.
-p*price*	Causes the value *price* to be used for the cost in dollars, rather than the default value of 0.02 or the price specified in /etc/printcap.
-r	Reverses the sorting order.
-s	Causes the accounting information to be summarized on the summary accounting file. This summarization is necessary because on a busy system, the accounting file can grow by several lines per day.
names	Limits reported statistics to the user names specified.

FILES

/etc/printcap	Printer capability database
/usr/adm/?acct	Raw accounting files
/usr/adm/?_sum	Summary accounting files

SEE ALSO

printcap(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

panic – Sets or clears the system user panic flag

SYNOPSIS

```
/etc/panic  set
/etc/panic  clear
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The panic command sets or clears the system user panic flag according to what you specify on the command line. If you specify set, a upanic(2) system call from any user program crashes the system.

Only an appropriately authorized user can use this command.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Allowed to use this command.
sysadm, sysops	Allowed to use this command. Shell redirected I/O is subject to security label restrictions.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

Specify either set or clear on the command line for panic to work. Do not specify both.

SEE ALSO

upanic(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

NAME

pcache – Controls device cache allocation, deallocation and attributes

SYNOPSIS

/etc/pcache [-D *high*,[*low*]] [-d] [-h *hashlen*] [-m *members*] [-n *count*] [-s *size*] [-t *type*] [-T *max*,[*min*]] [-w] *device*

/etc/pcache [-f *infile*]

/etc/pcache [-z]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The pcache command controls device cache allocations and/or parameters for a device.

Space for device cache is allocated from main memory. The size of the memory pool is controlled through the boot config file or with the pcinit(8) command. Cache may be assigned to a device, or to a set of members comprising the device by using the -m option. If cache is being assigned to members of a device, each specified member will receive the amount of cache specified by the -n and -s options.

Cache may be assigned to any of the following device types: dev_pdd, dev_sdd, dev_mdd, dev_xdd, dev_hdd. Logical devices (dev_ldd) cannot have cache, due to the architecture of the UNICOS/mk operating system; however, a dev_ldd (filesystem) device can be specified on the pcache command line. In this case, the cache will be allocated to each member of the logical device (or each partition of the filesystem), as if the -m all option had been specified.

Cache is specified as a number of units where each unit consists of a specified number of 4096-byte blocks. The number of blocks in a cache unit must be a multiple of the largest I/O unit (disk sector) of the physical devices on which the partition (or partitions) reside.

Old cache units are periodically flushed according to an interval specified in the boot config file. The default interval is 120 seconds. This can cause large backlogs of I/O requests when this sync interval expires, so it is recommended that trickle-sync (-T option) be used to balance the the load of flusing old units.

The pcache command accepts the following options:

-D *high*,[*low*] Specifies threshold values for dirty units in cache. The *high* value specifies the maximum number of dirty blocks that may be in the cache of a slice at any one time. If the number of dirty units exceeds *high*, new requests to dirty units will sleep until the number falls below the threshhold. When the number of dirty units in cache exceeds *low*, the system automatically starts flushing the oldest dirty units. If *low* is not specified, it defaults to the same value as *high*. If *high* is larger than the number of cache units allocated, it is set to the number of cache units. If *low* is greater than *high*, *low* is set to *high*. To disable the threshhold parameters, set *high* to 0. The default is 0.

`-d` Enables debugging output.

`-f` *infile* Read cache allocation command lines from *infile*. Each line in *infile* is a pcache command line. Lines beginning with # are ignored. This option cannot be mixed with any other options, and cannot be nested.

`-h` *hashlen* Specifies the size of the cache unit header hash array. The minimum (and default) value is 64. The `-n` option must be used with this option.

`-m` *members* Acts on specified members of the named device. *Members* is a comma-separated list of tokens in one of the following formats:

name The device name of the member.

number A member number - slice 0 corresponds to the first member.

range A numeric member range having the format *min*:*max*. The range is inclusive.

`primary` If the named device is a formatted filesystem, `primary` refers to all members corresponding to primary partitions. Invalid if the device is not a filesystem.

`secondary` If the named device is a formatted filesystem, `secondary` refers to all members corresponding to secondary partitions. Invalid if the device is not a filesystem.

`all` Alias for all members. The `all` token is case-insensitive.

The following examples illustrate use of the `-m` option:

`-m 0` Indicates the first member of the device.

`-m 0,1` Indicates the first and second members of the device.

`-m 0:2` Indicates members 0 through 2 (inclusive).

`-m all` Indicates all members.

`-m primary`
Indicates all members corresponding to primary partitions.

`-m secondary`
Indicates all members corresponding to secondary partitions.

`-m primary, secondary`
In the case of file systems, functionally equivalent to `-m all`.

`-m 0:1,3` Indicates members 0, 1, and 3 (omitting 2).

`-n` *count* Number of cache units to allocate. A count of 0 indicates that cache should be deallocated. The `-s` option must also be used if count is greater than 0.

`-s` *size* Size in 4096-byte blocks of each cache unit. This option must be used when the `-n` option is specified, and the count is greater than 0.

[`-t` *type*]	Type of cache (ssd, or ram) to allocate. Currently only ram cache is supported on UNICOS/mk systems. *type* is case-insensitive. This option must be used with the `-n` option.
`-T` *max*,[*min*]	Defines device cache aging (trickle-sync) parameters. The *max* value specifies the maximum age in seconds that any unit in cache may reach before automatic flushing of dirty blocks begins. All units older than *min* are flushed. If *min* is not specified, it defaults to the same value as *max*. To disable the aging parameters, set *max* to 0. The default is 0.
`-w`	Indicates that the cache allocated be marked as write-through. The `-n` option must be used with this option.
`-z`	Release cache for all devices. This option cannot be used with any other options.
device	Specifies the device name.

DIAGNOSTICS

Error messages, warnings, and debug output is issued to `stderr`. Exits with 0 if command completed succesfully, non-zero otherwise.

SEE ALSO

`pcinit`(8), `pcstat`(8), `pcsync`(8)

`hdd`(4), `mdd`(4), `pdd`(4), `sdd`(4), `xdd`(4) in the *UNICOS/mk File Formats and Special Files Reference Manual*, Cray Research publication SR–2614

NAME

pcinit – Runtime initialization of pcache

SYNOPSIS

/etc/pcinit [-c *core*] [-h *hdrs*] [-d *maxdevs*] [-s *syncint*] [*ospe* ...]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The pcinit command is used to initialize pcache at runtime, rather than having it set up in the configuration file. The pcinit command may be run multiple times; however, all cache must be deallocated before running the pcinit command.

The pcinit command accepts the following options:

-c *core*	Number of 4096-byte memory blocks to set aside for the cache pool. There is no default.
-h *hdrs*	Number of cache units to allocate. Default is core / 16.
-d *maxdevs*	Maximum number of devices to be cached. Default is 256.
-s *syncint*	Interval to flush all dirty units to disk. Default is 900 seconds.
ospe	The OS PE on which to initialize pcache. By default, all OS PEs having a disk actor will be initialized.

SEE ALSO

pcache(8), pcstat(8), pcsync(8)

NAME

pcstat – Displays device cache statistics

SYNOPSIS

/etc/pcstat [-b] [-c] [-g] [-h] [-s] [*device* ...]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The pcstat command is used to examine device cache statistics. The stats may be restricted to a set of devices specified on the command line. By default, statistics for all cached devices are displayed.

The pcstat command accepts the following options:

-b Displays request size distribution statistics.

-c Displays cache configuration, including dirty unit numbers and ages.

-g Displays global cache information.

-h Displays cache hashing information.

-s Displays cache I/O summary.

device Specifies device name.

If no options are specified, the -s option is assumed. The display options are cumulative; for example, pcstat -cs displays configuration and summary information.

OUTPUT FORMATS

The cache I/O summary output (-s) has the following format:

```
snow$ pcstat -s
                          Read/Write stats - values in Blocks
Name                    Reads      Writes     Hits       Misses     Rate
----------------------- ---------- ---------- ---------- ---------- ------
/dev/xdd/root              88780      46682     122647      12815  90.54
/dev/xdd/tmp.s21           28691      91410     105617      14484  87.94
```

The global cache output (-g) has the following format:

```
snow$ pcstat -g
          RAM Cache  Cache Hdrs    Sync    Cached Devs
 OS PE   total avail total avail interval allowed cached
-------- ----- ----- ----- ----- -------- ------- -------
ospe_a    4096    16   512   342      300     256       2
```

Cache configuration (`-c`) has the following format:

```
snow$ pcstat -c
                                                Unit Age          Dirty
Name                  Unit Size T Hash  Max Flush Oldest Max  Min  #Dirty Flags
--------------------- ---- ---- - ---- ---- ----- ------ ---- ---- ------ -----
/dev/xdd/root          100   24 M   67    0     0     73    0    0      4  ---
/dev/xdd/tmp.s21        70   24 M   67    0     0     74    0    0      6  ---
```

Cache I/O bucket output (`-b`) has the following format:

```
snow$ pcstat -b
                  I/O Length Buckets (# blocks per I/O request)

             1     2-3     4-7    8-15   16-31   32-63  64-127 127-255    255+
       ------- ------- ------- ------- ------- ------- ------- ------- -------
Read      9666     183     317    1059    6303    8178    4844    1997       0
Write    42477     197     850     187     431       8     618       0       0
Total    52143     380    1167    1246    6734    8186    5462    1997       0
Percent  67.44    0.49    1.51    1.61    8.71   10.59    7.06    2.58    0.00
```

Cache hash length output (`-h`) has the following format:

```
snow$ pcstat -h
                      Hash Hash  Shortest   Longest
Name                  Size Used Hash Chain Hash Chain
--------------------- ---- ---- ---------- ----------
/dev/xdd/root           67   58          1          4
/dev/xdd/tmp.s21        67   15          1          1
```

SEE ALSO

pcache(8), pcinit(8), pcsync(8)

NAME

pcsync – Syncs device cache and sets cache sync interval

SYNOPSIS

/etc/pcsync [-s *interval*] *device* ...

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The pcsync command is used to sync (flush) dirty units from cache. Cache is flushed for each named device, or all cached devices if none are specified.

The kernel flushes all cache periodicly, based on an internal sync interval specified in the boot configuration file. This interval may be reset on a running system with the -s option. The new sync interval is set to *interval* seconds. If *interval* is 0, the automatic sync is disabled. This will not disable trickle-sync or threshold-based flushing if specified. See the pcache(8) man page for more information on the use of trickle-sync and threshold-based flushing.

-s *interval* Resets internal sync interval (in seconds).

device Specifies device name.

SEE ALSO

pcache(8), pcinit(8), pcstat(8)

NAME

pddconf – Controls the state of an IOS model E disk drive

SYNOPSIS

/etc/pddconf [-d *device*] [*function* [*parameter*]]

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

CRAY J90 series

DESCRIPTION

The pddconf command controls the state of an IOS model E (IOS-E) disk drive. If no device is specified, all open devices are displayed. ioctl requests are issued to both the logical and physical disk drivers to mark the file systems that are affected by the disk state change and to mark the disk driver tables that are necessary. If no options are given, a display of the current state of all drives is shown.

The pddconf command accepts the following options:

-d *device* Device *iopath* (for example, 0132 or 0230.3). If no *device* is specified, all open devices are shown. For disk queue sorting, specifying all enables or disables the global flag.

function The *function* argument can be one of the following:

- rw — Sets the physical device mode to read/write.
- ronly — Sets the physical device mode to read-only. Read requests are permitted, writes are returned with an errno of EIO. Any mounted file system slice(s) residing on the device are marked as not available for allocation.
- noall — Sets the physical device mode to nonallocatable. Any mounted file system slice(s) residing on the device are marked as not available for allocation.
- up — Sets the physical disk device state to up. Any previous setting of mode is still in effect.
- down — Sets the physical device state to down and terminates all queued I/O requests with errors. Any mounted file system slice(s) residing on the device are marked as not available for allocation.
- spinup — Spinup a disk drive. Valid only for DD-60 disk drives, spinup is also done during device open time. (Has no effect on CRAY EL series.)
- spindown — Spindown a disk drive. Valid only for DD-60 disk drives. (Has no effect on CRAY EL series.)
- pripath — Change disk device to primary path.

`altpath` Change disk device to alternate path if available. (Has no effect on CRAY EL series.)

`reset` Reset a given disk statistics.

`disable` *spindle 0-4*
Disable a given spindle of a disk array. A spindle number must be given. (Has no effect on CRAY EL series.)

`qsort` Turn on the disk queue sorting algorithm in the disk driver for the specified device; the device flag is on by default. Specifying `all` as the device turns on a global flag that enables queue sorting for all disks in a system for which the disk queue sorting algorithm is enabled; this global flag is off by default. See the EXAMPLES section.

`noqsort` Turn off the disk queue sorting algorithm for the indicated device; the device flag is on by default. Specifying `all` as the device turns off a global flag, disabling queue sorting for all disks in a system; this global flag is off by default. See the EXAMPLES section.

`autoswitch on|off`
Enable/disable autopath switching during error recovery. The system will not switch to the alternate path if the switch is disabled. The device node must be configured with an alternate path to work. (Has no effect on CRAY EL series.)

`racerron` Routes Recovered Disk Error messages to be printed on the console. This is the default.

`racerroff` Stops routing Recovered Disk Error messages to the console.

The display has the following format:

```
                        disk devices                          qsort ENABLED
 name  type  unit state mode  altp  flg  wstrm   rstrm  qon    q'ed
------ ----- ---- ----- ---- ------ --- ------- ------- ---   ------
 0230  DD60    0     up  rw   ----   0  ------- ------- yes        0
 0230  DD62    3     up  rw   ----   0  ------- ------- yes        0
 0230  DD60    4     up  rw   ----   1  ------- ------- yes        0
 0230  DD60    5     up  rw   ----   0  ------- ------- yes        0
 0232  DD60    0     up  rw   ----   1  ------- ------- yes       18
 0232  DD60    1     up  rw   ----   0  ------- ------- yes        0
 0232  DD60    2     up  rw   ----   0  ------- ------- yes        0
 0232  DD62    3     up  rw   ----   1  ------- ------- yes        0
 0234  DD60    0     up  rw   ----   1  ------- ------- yes        0
 0234  DD60    1     up  rw   ----   1  ------- ------- yes        0
 0234  DD60    2     up  rw   ----   1  ------- ------- yes        0
 0234  DD62    3     up  rw   ----   1  ------- ------- yes        0
 0236  DD62    2     up  rw   ----   0  ------- ------- yes        0
 0236  DD60    4     up  rw   ----   1  ------- ------- yes        0
 0236  DD60    5     up  rw   ----   1  ------- ------- yes        0
 0334  DA301   0     up  rw   ----   1  0100037 0100037 yes       66
```

The definitions of the headings in the preceding table are:

`name` Device `iopath`. A lower case "a" is appended to the name when a drive is using its alternate path.

`type` Device type.

`unit` Device unit number.

`state` Device state (`init`, `up`, or `down`).

`mode` Device mode (read-write (`rw`), read-only (`ro`), or no allocate (`na`)).

`altp` Device alternate path if available.

`flg` Path open flags (no path open = 0, primary open = 1, alternate open = 2, both open = 3).

`wstrm` Device write stream control flags (used by disk arrays).

`rstrm` Device read stream control flags (used by disk arrays).

`qsort ENABLED`
Indicates that the global `qsort` flag is enabled.

`qon` Indicates that sort is enabled for a particular device.

`q'ed` The total number of requests that have been sorted since the system was booted.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

Example 1: Use the following command to enable disk sorting for all disks with the disk queue sorting algoritm set to on:

```
# pddconf -d all qsort
pddconf: disk queue sorting ENABLED
```

Example 2: Use the following command to disable disk sorting for all disks:

```
# pddconf -d all noqsort
pddconf: disk queue sorting DISABLED
```

Example 3: Use the following command to switch the disk sorting algorithm to on for disk 1334.0:

```
# pddconf -d 1334.0 qsort
pddconf: disk 1334.0 queue sorting ON
```

Example 4: Use the following command to switch the disk sorting algorithm to off for disk 1334.0:

```
# pddconf -d 1334.0 noqsort
pddconf: disk 1334.0 queue sorting OFF
```

Example 5: Use the following command to control the routing of Recovered Disk Error messages to the console.

```
# pddconf recerron
pddconf: disk recovered error messages routed to Console

# pddconf recerroff
pddconf: disk recovered error messages not routed to Console
```

SEE ALSO

`ioctl`(2) in *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

NAME

pddstat – Displays information about the IOS model E

SYNOPSIS

/etc/pddstat [-d *device.unit*] [-e] [-r *rate*] [-l] [-q] [-s]

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

CRAY J90 series

CRAY EL series

DESCRIPTION

The pddstat command gets information from the disk table, which controls disk input/output (I/O) in an IOS model E (IOS-E). The information is attained with the tabread (see tabinfo(2)) system call and is displayed on the caller's screen.

The pddstat command accepts the following options:

-d *device.unit*
: Displays specific statistics for *device.unit*. *device.unit* is the I/O path (for example, 0130.0 is the device on cluster 0, IOP 1, channel 030, unit 0).

-e
: Displays error streams. This option displays error counts on a per-spindle basis for arrayed devices.

-r *rate*
: Repeats the display each *rate* second with new information from the MIOP table. If -r is not specified, the display is shown only one time.

-l
: Allows display of longer lines in the device (-d) display for screens with greater than 80 columns.

-q
: Displays development information about queue lengths on each device as it is sampled at the time of each display update. This option requires that -r be specified.

-s
: Displays scientific information about queue lengths on each device as it is sampled at the time of each display update. This option requires that -r be specified.

The default display has the following format:

```
          device                                  # of sectors         errors
iopath type unit state mode req  func    reads     writes  rd-ur wt-ur total
------ ---- ---- ----- ---- --- ------ --------- --------- ----- ----- -----
 0230  DD60   0     up   rw  0   write      2656       153     0     0     0
 0232  DD60   1     up   ro  1    read      5512      1189     0     0     0
 0234  DD60   2   down   rw  0  wrtbhd      1267       885     0     0     0
 0236  DD60   4     up   rw  1  wrtbhd        14         5     0     0     0
 0236  DD60   5     up   rw  3  wrtbhd      6374       205     0     0     0
```

The following are definitions of the headings in the preceding table:

`iopath` Device `iopath` (for example, 0230 means cluster 0, iop 2, channel 30)

`type` Device type

`unit` Device unit number

`state` Device state (`init`, `up`, or `down`)

`mode` Device mode (read-write (`rw`), read-only (`ro`), or no allocation allowed on this device (`na`))

`req` Number of outstanding requests

`func` Last function that was sent to the disk, unless closed. A full list is in `epackd.h`. The common functions are read, write, write behind, configured up or down, and closed.

`reads` Number of sectors read (size of sector depends on disk type).

`writes` Number of sectors written

`rd-ur` Number of unrecovered read errors

`wt-ur` Number of unrecovered write errors

`total` Total number of read and write errors (both recovered and unrecovered)

The `-d` Option

The display for an individual device has the following format:

Command line: `pddstat -r 2 -d 1336.0`

```
Command: pddstat   [interval: 2 sec]   Page:  1   Wed Oct  9 11:42:22 1991

          device                            # of sectors         errors
 iopath type unit state mode req  func    reads     writes  rd-ur wt-ur total
 ------ ---- ---- ----- ---- --- ------ --------- --------- ----- ----- -----

  1336  DD60   0     up  rw   0  read         718         0     0     0     0

  current requests           0         0 reads         0 writes  rd pct=   0
  total   requests         359       359 reads         0 writes  rd pct= 100.00
  I/O errors total           0         0 reads         0 writes
  avg recovery time       0.00      0.00 reads      0.00 writes  (milliseconds)

  avg. transfer size in sectors   read:    0.000        write:    0.000
  total avg transfer sz (sectrs)  read:    2.000        write:    0.000
  avg. I/O time in milliseconds   read:   0.0000        write:   0.0000
  total avg I/O time (millisec)   read:   2.1151        write:   0.0000


  avg. I/O rate in Mbytes/sec     read:   0.0000        write:   0.0000
  total avg I/O rate  (MB/sec)    read:   7.7461        write:   0.0000
  avg. wait time in milliseconds  read:   0.0000        write:   0.0000
  total avg wait time (millisecs) read:   0.0132        write:   0.0000

  avg. number cylinders crossed  rd/wt:   0.0000   last cylinder    15
  total avg number cyls crossed  rd/wt:   0.0418
  number of waiting requests: 00           partitions crossed        1
  number of spare hits: 000000             crossings per I/O     0.0028
  I/O queue high water mark: 01

               Seek Buckets (# cylinders crossed per I/O request)
             0       1    2-3     4-7     8-15   16-31   32-63   64-127 128-255
Current      0       0       0       0       0       0       0       0       0
Percent      0       0       0       0       0       0       0       0       0
Total      344      15       0       0       0       0       0       0       0
Percent  95.82    4.18    0.00    0.00    0.00    0.00    0.00    0.00    0.00
```

```
                Cylinder Position Buckets (# I/O to each range )
             0- 129 130- 260 261- 390 391- 521 522- 651 652- 782 783- 912 913-1043
  Current        0       0       0       0       0       0       0       0
  Percent     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
  Total        359       0       0       0       0       0       0       0
  Percent   100.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00

             =<1173  =<1304  =<1435  =<1565  =<1696  =<1826  =<1957  =<2087
  Current        0       0       0       0       0       0       0       0
  Percent     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00
  Total          0       0       0       0       0       0       0       0
  Percent     0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00

             =<2218  =<2348  =<2479  =<2610
  Current        0       0       0       0
  Percent     0.00    0.00    0.00    0.00
  Total          0       0       0       0
  Percent     0.00    0.00    0.00    0.00

                   Length Buckets (# sectors per I/O request)
              1    2-3     4-7     8-15   16-31   32-63   64-127    128+
  Current     0       0       0       0       0       0       0       0
  Percent     0       0       0       0       0       0       0       0
  Total       0     359       0       0       0       0       0       0
  Percent  0.00  100.00    0.00    0.00    0.00    0.00    0.00    0.00
```

The following is an explanation of the information in the `pddstat` display:

`current requests 0 0 reads 0 writes rd pct= 0`
: Requests since the last refresh of the display; total, number of reads, number of writes, and the percentage of the requests that were reads.

`total requests 359 359 reads 0 writes rd pct= 100.00`
: I/O requests for this unit since the last deadstart of the system; total, reads, writes, and percentage of the requests that were reads.

`I/O errors total 0 0 reads 0 writes`
: Count of number of I/O errors for this unit since the last deadstart of the system; total number, number on reads, and number on writes.

`avg recovery time 0.00 0.00 reads 0.00 writes (milliseconds)`
: Average time spent in recovery in milliseconds. Average is reported for total I/O errors, reads errors, and write errors.

`avg. transfer size in sectors read: 0.000 write: 0.000`
: Average transfer size of I/O requests to this unit since the last refresh of the display. The size is reported in number of disk sectors for this device (for example, for DD-60 the sector size is 4 blocks or 2048 words). Transfer size is reported for reads and writes.

`total avg transfer sz (sectrs) read: 2.000 write: 0.000`
Average transfer size of I/O requests to this unit since the last deadstart of the system. The size is reported in number of disk sectors for this device (for example, for DD-60 the sector size is 4 blocks or 2048 words). Transfer size is reported for reads and writes.

`avg. I/O time in milliseconds read: 0.0000 write: 0.0000`
Average time to complete an I/O request in the last refresh span of this display. The average is calculated for read and write requests and is reported in milliseconds per request.

`total avg I/O time (millisec) read: 2.1151 write: 0.0000`
Average time to complete an I/O request since the last deadstart of the system. The average is calculated for read and write requests and is reported in milliseconds per request.

`avg. I/O rate in Mbytes/sec read: 0.0000 write: 0.0000`
The I/O rate of the requests to this disk unit in the last refresh span of the display is calculated using the amount of time that the request took to complete divided by the number of requests. It is displayed for reads and writes in megabytes per second.

`total avg I/O rate (MB/sec) read: 7.7461 write: 0.0000`
The I/O rate of the requests to this disk unit in the period since the last deadstart of the system in megabytes per second; reads and writes.

`avg. wait time in milliseconds read: 0.0000 write: 0.0000`
The average time waiting on the I/O queue before submitting the request to the IOS is calculated for the time span since the last refresh of the display. The average is per I/O request and is an indication of the busy state of the unit. It is calculated for reads and writes.

`total avg wait time (millisecs) read: 0.0132 write: 0.0000`
The average wait time on I/O queue waiting for the IOS to accept the request is calculated for the time span since the last system deadstart.

`avg. number cylinders crossed rd/wt: 0.0000 last cylinder 15`
On each I/O request the number of cylinders that must be passed to be able to perform the I/O is accumulated. This line is the average number of cylinders that were crossed per I/O request in the time span since the last display refresh.

The last cylinder to which I/O was done is displayed.

`total avg number cyls crossed rd/wt: 0.0418`
This indicates the average number of cylinders crossed per I/O request on this unit since the last system deadstart.

`number of waiting requests: 00 partitions crossed 1`
Number of I/O requests outstanding in the IOS for this unit.

The number of file system partitions crossed on I/O requests is accumulated by the disk driver. This represents crossings since the last system deadstart.

`number of spare hits: 000000 crossings per I/O 0.0028`

The number of times I/O was done to spare cylinders for this device are accumulated by the driver. Spares are not contiguous to the location of the rest of the file.

Last deadstarted is displayed. The is an indication of the variety of file systems used on this unit.

`I/O queue high water mark: 01`

This represents the longest I/O queue on this unit since the last deadstart of the system.

`Seek Buckets (# cylinders crossed per I/O request)`

The number of seeks of different length is calculated and displayed for varying seek spans. The length of span is kept on an exponential increasing size (that is, the first bucket is 1 cylinder and each subsequent bucket is twice the length of the previous). There are 4 calculations: the number in the last refresh span, the percentage of the seeks in the last span that fall into each bucket, the total number of seeks of this length since the last system deadstart, and the percentage of the seeks since the last deadstart to be of the indicated length range.

`Cylinder Position Buckets (# I/O to each range )`

The cylinder position of each I/O is known at the time of I/O. The driver counts the number of each that fall within equal size ranges of cylinder numbers. The display shows four lines of information: The number of requests in each range during the last refresh span; the percentage of the requests in the last refresh span to fall within each cylinder range; the number of requests in each range during the life of the system; and the percentage of requests that fall within each range since the last system deadstart.

`Length Buckets (# sectors per I/O request)`

The length of each request to the displayed unit is accumulated by the disk driver during its operation. The lengths are counted within ranges of lengths which are not of equal size, but rather vary by powers of 2 in length. The display indicates the number and percentage of requests within each range for the last refresh cycle and the number and percentage of requests within each range for the time since the last system deadstart.

The `-e` Option

The `-e` option displays error counts for array devices only. As the following example shows, error counts are shown on a per-stream (per-spindle) basis. Columns 0 through 3 represent errors attributed to spindles 0 through 3. The P column displays errors attributed to the parity spindle. The ? column displays errors that cannot be attributed to any specific spindle, such as a high-speed channel error. A minus sign preceding the count on columns 0 through 3 or P signifies the drive has been logically removed from the array (see `pddconf`(8)).

This display identifies a spindle that may be a candidate for removal from the array because of errors (see `pddconf`(8)). It is a summary of current array errors only. More detailed information on specific errors is logged and can be extracted using the `errpt`(8) command.

The following example shows that spindle 2 of drive 0334.0 has a problem. The minus sign in front of the count indicates that spindle 2 has been logically removed fom the array using the `pddconf`(8) command. It also shows a read error on drive 0334.1 that was not attributable to any single spindle.

drive stream error counts

device	P	3	2	1	0	?	request
334.0	0	0	-89459	0	0	0	reads
	0	0	-4356	0	0	0	writes
334.1	0	0	0	0	0	1	reads
	0	0	0	0	0	0	writes

The -q Option

The `pddstat` command provides the `-q` option to display more information about each device and its I/O queues. The following is an example of the `-q` option:

Command line: `pddstat -r 10 -q`

```
Command: pddstat   [interval: 10 sec]   Page:  1   Mon Oct 28 16:03:27 1991
```

iopath	device type	unit	req	# of sectors reads	writes	more req	samp	I/O max	queue lengths 1- 4	<10	>=10
0234	DD61	2	0	85417	15655	0	17	0	0	0	0
0236	DD61	3	0	10359	0	0	17	0	0	0	0
0236	DD61	6	0	267219	7578	0	17	0	0	0	0
1030	DD60	0	0	840874	95655	147	17	0	0	0	0
1032	DD60	1	0	2935	20297	0	17	0	0	0	0
1034	DD60	2	0	1176338	1293458	4	17	0	0	0	0
1036	DD60	3	0	440665	448455	0	17	0	0	0	0
1130	DD60	0	0	11581	2506	0	17	0	0	0	0
1132	DD60	0	0	95454	24387	6	17	0	0	0	0
1132	DD60	1	0	15694	8443	0	17	0	0	0	0
1132	DD60	2	0	6326	4297	0	17	0	0	0	0
1134	DD60	4	0	275460	271289	189	17	0	0	0	0
1136	DD60	6	0	479087	522378	135	17	0	0	0	0
1230	DD60	0	0	282645	158304	243	17	0	0	0	0
1232	DD60	4	0	85185	92801	5	17	0	0	0	0
1234	DD60	6	3	1213315	569692	664	17	2	2	0	0

The headings in the preceding display are as follows:

`iopath` Path to the I/O device (cluster, iop, iop channel).

`type` Type of device on the channel (for example, DD-60, DD-61, DD-40).

`unit` The unit number of a device that may or may not be daisy chained.

`req` Number of requests for this device which are currently outstanding in the IOS-E.

`sectors read`
: Number of sectors which have been read from this device since the last system deadstart.

`sectors written`
: Number of sectors written to this device since the last system deadstart.

`more req`
: Number of read or write requests to this device since the last refresh cycle (last sample taken from device's statistics tables).

`samp`
: Number of samples or refresh cycles since this display was started.

`I/O queue lengths`
: The I/O queue lengths give an indication of how busy each device has been. The lengths are values determined at the time that samples are taken, that is, when the display is about to be updated. The I/O queue represents the requests that are waiting in the mainframe for the IOS to be ready to receive another request for the device in question.

`max`
: The longest I/O queue that was found for this device when the sample statistics were taken. The queue may have been longer between the times that the samples were taken.

`1-4`
: Times that the sampler found the queue of I/O requests (to go to the IOS) to be at least 1 and less than or equal to 4.

`<10`
: Times that the sampler found the queue of I/O requests to be greater than 4 and less than 10.

`>=10`
: Times that the sampler found the length of the I/O queue for this device to be greater than or equal to 10.

The `-r` Option

The `-r` option takes the preceding information and runs it through a refreshing screen display. When using the `-r` option for a refreshing display, the following commands control the screen:

`>` Increases refresh time

`<` Decreases refresh time

`R` Scrolls

`r` Ends scroll

`+` Displays next page of data

`-` Displays previous page of data

`n` Displays next device (if the `-d` option is used on the command line)

`p` Displays previous device (if the `-d` option is used on the command line)

`q` Exits screen mode

The -s Option

The pddstat command provides the -s option to display more information about each device and its I/O queues. The following is an example of the -s option:

Command line: pddstat -r 10 -s

```
Command: pddstat   [interval: 10 sec]   Page:  1   Wed Apr  1 16:41:13 1992
           device                      I/O  queue lengths          average   max
 iopath type unit   samp     1    2-3   4-7   8-15  16-31  >=32   queue   queue
 ------ ---- ----   ----   ----   ----  ----  ----  -----  ----   -----   -----

  0230  DD60   0     802     0      0     0     0     0     0     0.000       0
  0232  DD60   0     802     0      0     0     0     0     0     0.000       0
  0234  DD60   0     802     0      0     0     0     0     0     0.000       0
  0236  DD60   0     802     2      0     1     2     0     0     0.034       9
  1030  DD60   0     802     0      0     0     0     0     0     0.000       0
  1032  DD60   0     802     0      0     0     0     0     0     0.000       0
  1034  DD60   0     802     0      0     0     0     0     0     0.000       0
  1036  DD60   0     802     0      0     1     0     0     0     0.005       4
  1130  DD60   0     802     0      0     0     0     0     0     0.000       0
  1132  DD60   0     802     0      0     0     0     0     0     0.000       0
  1134  DD60   0     802     0      0     0     0     0     0     0.000       0
  1136  DD60   0     802     0      0     0     1     1     1     0.112      54
  1230  DD60   0     802     0      0     0     0     0     0     0.000       0
  1232  DD60   0     802     0      0     0     0     0     0     0.000       0
  1234  DD60   0     802     0      0     0     0     0     0     0.000       0
  1236  DD60   0     802     0      0     0     0     0     0     0.000       0
  1330  DD60   0     802     0      0     0     0     0     0     0.000       0
  1332  DD60   0     802     0      0     0     1     0     0     0.010       8
  1334  DD60   0     802     0      0     0     0     1     0     0.022      18
  1336  DD60   0     802     0      0     0     0     0     0     0.000       0
  2030  DD50   0     802     0      0     0     0     0     0     0.000       0
  2032  DD50   0     802     0      0     0     0     0     0     0.000       0
  2034  DD50   0     802     0      1     0     0     0     6     2.968     665
  2036  DD50   0     802     0      0     0     0     0     1     0.148     119
```

The headings in the preceding display are as follows:

iopath	Path to the I/O device (cluster, iop, iop channel).
type	Type of device on the channel (for example, DD-60, DD-61, DD-40).
unit	The unit number of a device that may or may not be daisy chained.
samp	Number of samples or refresh cycles since this display was started.

`I/O queue lengths`
: The I/O queue lengths give an indication of how busy each device has been. The lengths are values determined at the time that samples are taken, that is, when the display is about to be updated. The I/O queue represents the requests that are waiting in the mainframe for the IOS to be ready to receive another request for the device in question.

`1` Times that the sampler found the queue of I/O requests (to go to the IOS) to be 1.

`2-3` Times that the sampler found the queue of I/O requests (to go to the IOS) to be greater than 1 and less than 4.

`4-7` Times that the sampler found the queue of I/O requests (to go to the IOS) to be greater than 3 and less than 8.

`8-15` Times that the sampler found the queue of I/O requests (to go to the IOS) to be greater than 7 and less than 16.

`16-31` Times that the sampler found the queue of I/O requests (to go to the IOS) to be greater than 15 and less than 32.

`>=32` Times that the sampler found the length of the I/O queue for this device to be greater than or equal to 32.

`average queue`
: Total number of I/O requests found by the sampler divided by the number of samples.

`max queue` The longest I/O queue that was found for this device when the sample statistics were taken. The queue may have been longer between the times that the samples were taken.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	**Action**
`system`, `secadm`	Shell redirected I/O is not subject to security label restrictions.

If the `PRIV_SU` configuration option is enabled, for the super user, shell redirected I/O is not subject to security label restrictions.

SEE ALSO

`iocstat`(8)

NAME

pddtest – Physical disk device test

SYNOPSIS

pddtest [-cvwr] [-u *iou*] [-n *pas*] [-s *sec*] [-p *pat*] [-P *data*] *special*

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

CRAY J90 series

CRAY EL series

DESCRIPTION

The pddtest command provides a basic data integrity test of any logical or physical disk device referenced by the character or block special file *special*. By default, only reads are done, but writes can be done optionally as described below. Eight data patterns are tested, one per pass. The low-order 32 bits of each sector contains the block number for that sector.

-c Compares data written to data read. Data must have been sequentially written for compare to succeed.

-v Writes verbose description of what the test is doing to stdout.

-w Does writes as well as reads. This will destroy any previous data.

-r Runs on error. The default is to stop on error.

-u *iou* Does reads and writes in *iou* number of sectors. The default is usually 1 cylinder.

-n *pas* Runs *pas* number of passes through the test and stop.

-s *sec* Runs the sections corresponding to the bit mask *sec*. The default is to run all sections, although writes are done only if -w is specified.

Section	Bit	Mask	Action
0	2^^0	01	Sequential writes
1	2^^1	02	Random writes
2	2^^2	04	Sequential reads
3	2^^3	10	Random reads

-p *pat* Runs pattern number *pat*. By default all patterns are run, one per pass. Patterns are as follows:

pat	**data pattern**
0	00000000000000000000
1	01777777777777777777

2 01777770000037777600000
3 00000077777740000177777
4 01777777777740000000000
5 00000000000037777777777
6 01252525252525252525252
7 00525252525252525252525

`-P` *data* Runs the user specified data pattern *data*.

special Character or block special file.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`	Allowed to specify any file.
`sysadm`	Allowed to specify any file, subject to security label restrictions on the file's path. Shell redirected I/O is subject to security label restrictions.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to specify any file.

WARNINGS

The `pddtest` command should be run exclusively in read-only mode on a mounted file system. Do not run `pddtest` on a slice being used by a mounted file system with options selected that do writes and/or data compares. The `-w` option will cause filesystem data corruption and the `-cw` option may cause data miscompares, even if you use the `pddconf`(8) command to set the `NOALLOC` option.

EXAMPLES

Example 1: Runs all sections and all patterns doing writes and data compares:

```
pddtest –cvw /dev/pdd/scr0134.0
```

Example 2: Runs only sequential sections and all patterns doing writes and data compares:

```
pddtest –cvws 5 /dev/pdd/scr0134.0
```

Example 3: Runs all sections and pattern number 2 only doing writes and data compares:

```
pddtest –cvwp 1 /dev/pdd/scr0134.0
```

Example 4: Runs all sections and data patterns and does not stop on error:

```
pddtest –cvwr /dev/pdd/scr0134.0
```

SEE ALSO

`pddconf`(8)

`pdd`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

ping – Sends ICMP ECHO_REQUEST packets to network hosts

SYNOPSIS

ping [-c *count*] [-d] [-f] [-i *wait*] [-l *preload*] [-n] [-p *pattern*] [-q] [-r] [-R] [-s *packetsize*] [-v] *host*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The ping command uses the Internet control message protocol's (ICMP) mandatory ECHO_REQUEST datagram to elicit an ICMP ECHO_RESPONSE from a host or gateway. ECHO_REQUEST datagrams (also called pings) have an Internet Protocol (IP) and ICMP header, followed by a struct timeval; an arbitrary number of bytes are used to fill out the packet.

The ping command accepts the following options and operands:

-c *count*	Stops after sending (and receiving) *count* ECHO_RESPONSE packets.
-d	Sets socket-level debugging (SO_DEBUG) on the socket being used.
-f	Floods ping. Outputs packets as fast as they come back, or 100 times per second, whichever is greater. For every ECHO_REQUEST sent, a period is printed; for every ECHO_REPLY received, a backspace is printed. This provides a rapid display of the number of packets being dropped. Only the super user can use this option. The -f option can overload a network, and it and should be used with caution.
-i *wait*	Waits *wait* seconds between sending each packet. The default is 1 second. This option is incompatible with the -f option.
-l *preload*	Sends the number of packets specified by *preload* as fast as possible before falling into its normal mode of behavior.
-n	Specifies numeric output only. No attempt will be made to look up symbolic names for host addresses.
-p *pattern*	Pads the packet with bytes of a specified pattern. You may specify up to 16 bytes to fill out the packet you send. This is useful for diagnosing data-dependent problems in a network. For example, -p ff will cause the sent packet to be filled with all 1's.
-q	Specifies quiet output. Nothing is displayed except the summary lines at startup and completion.

`-r` Bypasses the normal routing tables (by setting flag `SO_DONTROUTE`) and sends requests directly to a host on an attached network. If the host is not on a directly attached network, an error is returned. You can use this option to reach a local host through an interface that has no established route.

`-R` Records the route. Includes the `RECORD_ROUTE` option in the `ECHO_REQUEST` packet and displays the route buffer on returned packets. The IP header is only large enough to display nine routes. Many hosts ignore or discard this option.

`-s` *packetsize* Specifies the number of data bytes to be sent. The default is 56, which translates to 64 ICMP data byes when combined with the 8 bytes of ICMP header data.

`-v` Enables verbose output. Lists ICMP packets, other than `ECHO_RESPONSE`, that are received.

host Specifies the network host. This operand is required.

When using `ping` for fault isolation, run `ping` to the local host to verify that the local network interface is running. Then, run `ping` to hosts and gateways increasingly farther away. `ping` sends one datagram per second, and it prints one line of output for each `ECHO_RESPONSE` returned. If there is no response, no output is produced. Round-trip times and packet loss statistics are computed. When all responses are received or the program times out (with a *count* specified), or if the program is terminated with a `SIGINT`, a brief summary is displayed.

NOTES

This program is intended for network testing, measurement, and management. It must be used primarily for manual fault isolation. Because of the load `ping` can impose on the network, you should not use `ping` during normal operations or from automated scripts.

ICMP Packet Details

An IP header without options is 20 bytes. An ICMP `ECHO_REQUEST` packet contains an additional 8 bytes worth of ICMP header, followed by an arbitrary amount of data. When a packet size is given, this indicates the size of this extra piece of data; the default is 56. Thus, the amount of data received inside of an IP packet of type `ICMP ECHO_REPLY` always will be 8 bytes more than the requested data space (the ICMP header). If the data space is at least 8 bytes in length, `ping` uses the first 8 bytes of this space to include a time stamp, which it uses in the computation of round trip times. If less than 8 bytes of padding are specified, no round trip times are given.

Duplicate and Damaged Packets

The `ping` command will report duplicate and damaged packets. Duplicate packets should never occur, and seem to be caused by inappropriate link-level retransmissions. Duplicates may occur in many situations and are rarely (if ever) a good sign, although the presence of low levels of duplicates may not always be cause for alarm. Damaged packets are obviously a cause for alarm and often indicate broken hardware somewhere in the `ping` packet's path (in the network or in the hosts).

Trying Different Data Patterns

The internetwork layer should never treat packets differently depending on the data contained in the data portion. Unfortunately, data-dependent problems do occur in networks and remain undetected for long periods of time. In many cases, the particular pattern that will have problems is something that does not have sufficient transitions (for example, all 1's or all 0's, or a pattern right at the edge, such as almost all 0's.) It is not necessarily enough to specify a data pattern of all 0's (for example) on the command line, because the pattern that is of interest is at the data link level, and the relationship between what you type and what the controllers transmit can be complicated. This means that if you have a data-dependent problem, you will probably have to do a lot of testing to find it. You may find a file that either cannot be sent across your network or that takes much longer to transfer than other similar length files. You can then examine this file for repeated patterns that you can test by using the `-p` option of `ping`.

TTL Details

The transistor-transistor logic (TTL) value of an IP packet represents the maximum number of IP routers that the packet can go through before being thrown away. Typically, each router in the Internet decrements the TTL field by exactly one. The TCP/IP specification states that the TTL field for TCP packets should be set to 60, but many systems use smaller values (4.3BSD uses 30, 4.2BSD used 15). The maximum possible value of this field is 255, and most UNIX systems set the TTL field of ICMP `ECHO_REQUEST` packets to 255. This is why you will find you can `ping` some hosts, but not reach them with `telnet`(1B) or `ftp`(1B).

In normal operation, `ping` prints the TTL value from the packet it receives. When a remote system receives a `ping` packet, it can do one of the following three things with the TTL field in its response:

- Not change it; this is what Berkeley UNIX systems did before the 4.3BSD-Tahoe release. In this case, the TTL value in the received packet will be 255 minus the number of routers in the round-trip path.
- Set it to 255; this is what current Berkeley UNIX systems do. In this case, the TTL value in the received packet will be 255 minus the number of routers in the path from the remote system to the host using `ping`.
- Set it to some other value. Some machines use the same value for ICMP packets that they use for TCP/IP packets (for example, either 30 or 60). Others may use completely random values.

BUGS

Many hosts and gateways ignore the `RECORD_ROUTE` option.

The maximum IP header length is too small for options like `RECORD_ROUTE` to be completely useful.

Using the `-f` option is not recommended in general, and using it on the broadcast address should be done only under very controlled conditions.

SEE ALSO

`ifconfig`(8)

`ftp`(1B), `netstat`(1B), `telnet`(1B) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

NAME

portmap – Converts Remote Procedure Call (RPC) program numbers into DARPA protocol port numbers

SYNOPSIS

/etc/portmap [-i] [-r] [-f *filename*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The portmap command is a server that converts Remote Procedure Call (RPC) program numbers into Defense Advanced Research Projects Administration (DARPA) protocol port numbers. It must be running to make RPCs.

When an RPC server is started, it tells portmap to which port number it is listening, and the RPC program numbers it is prepared to serve. When a client wants to make an RPC to a given program number, it first contacts portmap on the server machine to determine the port number to which RPC packets should be sent.

By default, portmap does not forward any non–null requests to the MOUNTPROG program; this would bypass security checks performed by /etc/mountd. This limitation can present a problem with some programs, especially old versions of automount. To get around this problem, portmap accepts the following option:

-i Allows nonsecure forwarding of requests to MOUNTPROG.

Use of the -i option is not supported on systems that are configured to support nonzero security labels.

The -r and -f options allow RPC servers to reregister themselves with portmap. This capability is useful if, for some reason, it is necessary to restart portmap while the RPC servers continue. If portmap were to die and then be restarted without either of these options, any active RPC process would lose its RPC registration. The options are as follows:

-r Specifies restart for standard RPC servers. This option causes portmap to send a SIGHUP signal to the following processes:

```
cnfsd
inetd
keyserv
mountd
nfsd
pcnfsd
ypbind
yppasswdd
ypserv
```

Each of these processes catches the SIGHUP signal and reregisters with portmap upon receiving the signal. In addition, any root process invoked with one of these process names is sent a SIGHUP signal if you invoke portmap with the -r option.

-f *filename*

Directs portmap to send a SIGHUP signal to each of the process/uid pairs found in the *filename* file in addition to the standard RPC servers that receive the signal when the -r option is used. (Thus, the -f option implies the -r option.)

The format of the information in *filename* is as follows:

program_1 [*uid1*]
program_2 [*uid2*]
.
.
.
program_n [*uidn*]

The *program* field is the name of the program (for example, rpc_server). The *uid* field is optional and specifies that SIGHUP be sent only to processes of the specified name running under the specified *uid*. If the *uid* field is blank, SIGHUP is sent to all processes of the specified name, regardless of their user ID. For example, assume that portmap has been invoked with the -f *rpcfile* option, and assume that *rpcfile* contains the following lines:

```
rpc_server      0
test_server     123
test_server     456
new_server
```

The following table indicates which processes will and will not be sent a SIGHUP signal:

Process name	UID	Sent SIGHUP?
rpc_server	0	Yes
rpc_server	123	No
rpc_server	456	No

Process name	UID	Sent SIGHUP?
rpc_server	789	No
test_server	0	No
test_server	123	Yes
test_server	456	Yes
test_server	789	No
new_server	0	Yes
new_server	123	Yes
new_server	456	Yes
new_server	789	Yes
other_proc	0	No
other_proc	123	No
other_proc	456	No
other_proc	789	No

The -f option is useful when site-specific RPC servers are running. However, to make this option useful, you must code the servers to catch the SIGHUP signal and to reregister with portmap when they receive it.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

SEE ALSO

rpcinfo(8)

privtext(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

prctmp – Prints login session file

SYNOPSIS

/usr/lib/acct/prctmp *files*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The prctmp command prints the login session file, usually the /usr/adm/acct/nite/ctmp file created by the acctcon1 command (see acctcon(8)). The login session file is generated as part of the daily run of standard UNIX System V accounting by using the runacct(8) command. The output has the following format:

```
MAJ/MIN                 CONNECT SECONDS START TIME      SESSION START
DEVICE  UID     LOGIN   PRIME   NPRIME  (NUMERIC)       DATE    TIME
```

FILES

/usr/adm/nite/ctmp Login session file

SEE ALSO

acct(8), acctcon(8), acctsh(8), runacct(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

prdaily – Prints daily accounting report

SYNOPSIS

/usr/lib/acct/prdaily [-c] [-l] [*mmdd*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The prdaily command prints the standard UNIX System V accounting daily report to standard output. The runacct(8) command invokes prdaily to generate a report at the end of the daily accounting run. This report is found in /usr/adm/acct/sum/rprt*mmdd*, where *mmdd* is the month and day of invocation.

The prdaily command accepts the following options and operand:

-c Specifies that a report of exceptional usage, sorted by command, be generated. This option is available only for the current day's accounting data.

-l Specifies that a report of exceptional usage, identified by login ID, be generated.

mmdd Specifies that a report be generated for each combination of month and day listed. By default, prdaily generated only the current day's report.

NOTES

After you invoke the monacct(8) command, reports for dates prior to the invocation are inaccessible, because monacct creates a file that contains a summary of the previous daily reports and deletes the original reports.

EXAMPLES

The following example generates the daily accounting report for August 1 and 2:

```
/usr/lib/acct/prdaily  0801 0802 >> rprt.0102
```

SEE ALSO

acct(8), acctsh(8), monacct(8), runacct(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

privcmd – Sets file privileges, security label, permissions mode, owner, owning group, and security flags

SYNOPSIS

```
/etc/privcmd [-l label] [-h label] [-i] [-I directory] [-b alt_db] [-S grammar_objects] [files]
/etc/privcmd -R [-I directory] [-b alt_db] [files]
/etc/privcmd [-a plst] [-f plst] [-s plst] [-m label] [-d dac] [-p pentry] files
/etc/privcmd -t grammar_objects files
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The privcmd command sets the security label, permissions mode, owner, owning group, and security flags of files and directories. This command also sets file privileges. privcmd reads the names of the files and directories, and the security attributes to be assigned, from the /etc/privdb/main.db database file.

If you choose to modify the default attribute values, you should specify your changes in the /etc/config/localpriv.db database file, which is included automatically in /etc/privdb/main.db. Do not modify any of the database files other than the /etc/config/localpriv.db file.

Your site may not want to install every file that is specified in /etc/privdb/main.db. privcmd checks the system configuration options in /etc/config/config.mh to determine which files are appropriate for your system's configuration.

You can specify file and directory names on the command line. privcmd assigns the attributes that are defined in /etc/privdb/main.db for the specified files and directories.

If no file or directory names are specified on the command line, attributes are assigned to all files and directories as defined in /etc/privdb/main.db.

You can also specify both names and attributes on the command line. In this case, the names and attributes defined in /etc/privdb/main.db are ignored.

The privcmd command accepts the following options:

-l Assigns the specified security label to the files and directories that would otherwise be assigned the syslow security label. If this option is used with the -S option, MAC must be one of the members of the argument list passed to the -S option.

-h Assigns the specified security label to the files and directories that would otherwise be assigned the syshigh security label. If this option is used with the -S option, MAC must be one of the members of the argument list passed to the -S option.

`-i` Compares the attributes that are actually assigned to files and directories against the attributes that are specified in the database entries for those files and directories. Output is produced using a `+-` format. A `+` means that the file or directory is assigned an attribute that is not specified in its database entry. A `-` means that the object is missing an attribute that is specified in its database entry.

`-I` Changes the algorithm for searching for `#include` files whose names do not begin with `/` to look in *directory* before looking in the directories on the standard list. `#include` files whose names are enclosed in double quotation marks in the `include` directive are searched for first in the directory of the input file. All `#include` files (those whose names are enclosed in `< >` or `" "`) are then searched for in directories named in `-I` options, and, finally, in the standard directories `/etc/config` and `/etc/privdb`.

Each additional directory is specified by a separate `-I` option. If multiple `-I` options are specified, the directories are searched in the order specified on the command line.

`-b` Forces `privcmd` to use the database specified by *alt_db* instead of `/etc/privdb/main.db`. If the `-b` option is not specified, the `/etc/privdb/main.db` file is used as the default.

`-S` Specifies the selection criteria for the database grammar objects to be used when setting or doing an integrity check. If this option is not specified, the default value is `ALL`. The list of valid grammar objects is defined by *grammar_objects*.

`-R` Clears all privileges from *files*. When the `-R` option is specified, no mandatory access control, discretionary access control, or flag attributes from the database file are applied to any files.

`-a` Sets the file's allowed privileges to the specified value. If this option is not specified, the old allowed privilege set associated with *files* does not change. To delete the set of allowed privileges, assign it `PRIV_NULL`.

`-f` Sets the file's forced privileges to the specified value. If this option is not specified, the old forced privilege set associated with *files* does not change. To delete the set of forced privileges, assign it `PRIV_NULL`.

`-s` Sets the file's set_effective privileges to the specified value. If this option is not specified, the old set_effective privilege set associated with *files* does not change. To delete the set of set_effective privileges, assign it `PRIV_NULL`.

`-m` Sets the file's security label to the specified value.

`-d` Sets the file permissions mode, owner, and owning group to the specified value.

`-p` Updates the file privilege assignment list (PAL) with the specified PAL category record. Multiple PAL category records can be set by specifying the `-p` option multiple times.

`-t` Displays the output in the format of a `privcmd` database entry. You can use this output to generate `privcmd` database entries for specific files.

The `privcmd` command accepts the following operands:

files The name(s) of the file(s) to which attributes are assigned. Multiple file names must be separated by white space.

alt_db The name of the database file to be used instead of `/etc/privdb/main.db`.

directory The name of the directory in which to search for the database files prior to searching the `/etc/config` and `/etc/privdb` directories.

label A character string that represents the security label to be assigned. It has the following format:

level_name[,*compartment_name*[,*compartment_name*[...]]]

level_name is a character sequence that represents the name of a security level (for example `level2).` *compartment_name* is a character sequence that represents the name of a compartment (for example, `comp39`). If no compartments are specified, or the text string `none` or 0 is used, then the compartment bit mask is set to 0. If one or more compartment names are specified, components of the specified security label must be separated by commas with no intervening white space.

plst A character string that represents the privileges to be assigned. It has the following format:

privilege_name[,*privilege_name*[,*privilege_name*[...]]]

privilege_name is a character sequence that represents the name of a privilege (for example `PRIV_MAC_READ`). If one or more privilege names are specified, they must be separated by commas with no intervening white space. The `PRIV_ALL` character string represents the list of all privileges. Privileges can be cleared by specifying `PRIV_NULL`.

dac A character string that represents the permissions mode, owner, and owning group to be assigned. It has the following format:

permissions_mode,*owner_name*,*group_name*

permissions_mode must be specified in an octal value. *owner_name* is a character string that represents the name of the file owner. *group_name* is a character sequence that represents the name of the file group.

pentry A character string that represents the PAL category record to be assigned. This entry has the following format:

category_name[:[*plst*][:[*ptext*]]]

category_name is a character string that represents the name of category (for example, `secadm`). *plst* is a character sequence that represents one or more privileges to be associated with this PAL category record. See the previous description of *plst* for more information. *ptext* is a character sequence that represents the privilege text that is to be associated with this PAL category record. A maximum of 8 alpha-numeric characters can be specified for this field.

To delete a PAL category record, specify the following:

category_name::

The following syntax clears the privileges and privilege text of a PAL category record, but it does not delete the record from the PAL:

category_name:PRIV_NULL:TEXT_NULL

grammar_objects

A character string that represents the desired grammar objects to be used when generating, setting, or verifying a database record. If one or more *grammar_objects* are specified, they must be separated by commas, with no intervening white space. The following are acceptable values for *grammar_objects*:

Value	Description
MAC	Processes mandatory access control (MAC) security information.
DAC	Processes discretionary access control (DAC) security information.
FLAGS	Processes security flags information.
PAL	Processes PAL-related information.
ALL	Processes all of the MAC, DAC, FLAGS, and PAL.

NOTES

When using the `privcmd` command, the `syshigh` and `syslow` security labels are not applied to the specified objects unless the `SECURE_MAC` configuration option is enabled.

When `privcmd` is executed, file attributes for any given file are set in the following specific order:

1. Flag attributes
2. DAC attributes
3. MAC attributes
4. Privileges and PALs

If a failure occurs when trying to set MAC attributes, DAC attributes, privileges, or PALs, `privcmd` returns an exit status of 1, and all file attributes that have already been set remain set. If a failure occurs when trying to set flag attributes, an error message is issued and privcmd continues.

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

You can specify file record fields in any order. Also, you can specify multiple categories within a PAL field. For example, if you want to assign the `secadm` and `system` categories the exact same privileges and privilege text, use a comma-separated list of categories as follows:

```
PAL=[
     secadm,system:PRIV_DAC_OVERRIDE,...,PRIV_RESOURCE:exec;

    ];
```

The previous example is equivalent to specifying two separate entries with the same exact privileges and privilege text as shown in the following example:

```
PAL=[
     secadm:PRIV_DAC_OVERRIDE,...,PRIV_RESOURCE:exec;
     system:PRIV_DAC_OVERRIDE,...,PRIV_RESOURCE:exec;
    ];
```

EXIT STATUS

The `privcmd` command exits with one of the following values:

0 `privcmd` was successful.

1 Failed to set file security information.

2 Integrity checking found discrepancy.

EXAMPLES

Example 1: The following example applies the `syslow` label on `file1` and `file2`:

```
$ /etc/privcmd -m syslow file1 file2
```

Example 2: The following example applies a security label that consists of level 0 and the `comp24` and `comp39` compartments on all files in the current directory:

```
$ /etc/privcmd -m level0,comp24,comp39 *
```

Example 3: The following example removes the PALs associated with /bin/cat and /bin/chmod, and sets the security label for both to level 0:

```
$ /etc/privcmd -R -l level0 /bin/cat /bin/chmod
```

Example 4: The following example removes all PALs found in /etc/privdb/main.db. Relabels every file entry in the database that currently has a syslow label with level0, and relabels every file entry that currently has a syshigh label with level3:

```
$ /etc/privcmd -R -l level0 -h level3
```

Example 5: For the following example, assume that mydbfile contains the following record:

```
file= { name = /bin/xyz;
                    allowed = [PRIV_NULL];
                    forced = [PRIV_DAC_OVERRIDE,
                                  PRIV_MAC_READ,PRIV_MAC_WRITE];
                    set_effective = [PRIV_DAC_OVERRIDE,
                                  PRIV_MAC_READ,PRIV_MAC_WRITE];
                    MAC = [syslow,none];
                    DAC = [0755 bin bin];
                    FLAGS = [priv_root,exec];
                    PAL = [
                           other :PRIV_NULL:TEXT_NULL;
                           secadm:PRIV_DAC_OVERRIDE,PRIV_FOWNER,
                                  PRIV_MAC_READ,PRIV_MAC_WRITE:
                                  TEXT_NULL;
                           sysadm:PRIV_DAC_OVERRIDE:showall;
                           system : PRIV_DAC_OVERRIDE,
                                    PRIV_MAC_READ,
                                    PRIV_MAC_WRITE:TEXT_NULL;
                       ];
            };
```

Executing the following command labels the /bin/xyz file with the allowed, forced, and set_effective privileges, and creates a PAL with a secadm, sysadm, system, and other entries. The MAC, DAC, and FLAGS values are also applied; the MAC values are applied only if the SECURE_MAC parameter is enabled:

```
$ /etc/privcmd -b mydbfile
```

The two following methods can be used for changing record entries from within the database:

- Create a duplicate record entry and change the attributes.
- Use a special record called an update record to redefine categories, create new categories, and/or assign privilege text.

The second method is the recommended way for changing record entries. The default privilege definition and assignment involved extensive code analysis by Cray Research. For this reason, it is not recommended that you change individual privileged for a category. Rather, if you want to create and define new categories, they should be defined in terms of an already-defined category. This ensures that the new category is assigned the right privileges.

Example 6: The following example shows the recommended method outlined previously.

Assume that the following special update record has been appended to `mydbfile`:

```
file.update = { name = /bin/xyz;
                    PAL = [
                            secadm:secadm:sysadm;
                            sysops:system:TEXT_NULL;
                            sysadm::;
                    ];
              };
```

Adding this record keeps the `secadm` privileges intact and changes the privilege text for `secadm` from `TEXT_NULL` to `showall`. It also defines a new entry for the `sysops` category, which has the same privileges as the `system` entry, but with `TEXT_NULL` for the privilege text. The last entry deletes the `sysadm` category. The sequence in which these operations are performed is important, because the definition of categories is modified and any further reference to that category name receives the updated or modified definition.

Example 7: The following example shows how to change individual privileges and categories "on the fly." Making changes in this way does not change the databases. In this example, the `PRIV_FOWNER` privilege is deleted and the `showall` privilege text is assigned for the `secadm` category:

```
$ /etc/privcmd -b mydbfile
$ /etc/getpal /bin/xyz
other:PRIV_NULL:TEXT_NULL
secadm:PRIV_FOWNER,PRIV_DAC_OVERRIDE,PRIV_MAC_READ,PRIV_MAC_WRITE:TEXT_NULL
sysadm:PRIV_DAC_OVERRIDE:showall
system:PRIV_DAC_OVERRIDE,PRIV_MAC_READ,PRIV_MAC_WRITE:TEXT_NULL
$
$ /etc/privcmd -p secadm:PRIV_DAC_OVERRIDE,PRIV_MAC_READ,\
> PRIV_MAC_WRITE:showall /bin/xyz
```

Note that there are no intervening spaces between continuation marks and the new line.

The following example shows the resulting PAL entry from the previous example:

```
$ /etc/getpal /bin/xyz
other:PRIV_NULL:TEXT_NULL
secadm:PRIV_DAC_OVERRIDE,PRIV_MAC_READ,PRIV_MAC_WRITE:showall
sysadm:PRIV_DAC_OVERRIDE:showall
system:PRIV_DAC_OVERRIDE,PRIV_MAC_READ,PRIV_MAC_WRITE:TEXT_NULL
```

Another way to compare PAL differences is to use the `privcmd -i` option, as shown in the following example. Use of this option produces verification and validation results between the `/bin/xyz` database entry and the actual object's PAL entry:

```
$ /etc/privcmd -i -b mydbfile
/bin/xyz:secadm:- PRIV_FOWNER
/bin/xyz:secadm: showall/TEXT_NULL
```

The `-` sign means the `secadm` PAL entry has `PRIV_FOWNER` while the `/bin/xyz` file does not have it. Also, `/bin/xyz` has a `showall` privilege text while the database entry shows `TEXT_NULL` as privilege text for that category.

Example 8: The following example shows how to generate a database entry with the modified `secadm` category from the previous example. Use the `-t` option to generate a formatted database record:

```
$ /etc/privcmd -t PAL,MAC /bin/xyz >> /etc/config/localpriv.db
```

The generated record is appended to your `localpriv.db` database. This new entry supersedes any previously existing entries with the same file name.

Example 9: The following example shows a PAL entry with the same privileges specified for two categories and an additional PAL entry for a different category:

```
file = { name = /bin/dog;
                 forced = [PRIV_DAC_OVERRIDE,
                               PRIV_MAC_READ,PRIV_MAC_WRITE];
                 set_effective = [PRIV_DAC_OVERRIDE,
                               PRIV_MAC_READ,PRIV_MAC_WRITE];
                 allowed = [PRIV_NULL];
                 DAC = [ 755 bin bin ];
                 PAL = [
                        other:PRIV_NULL:TEXT_NULL;
                        secadm,system:PRIV_DAC_OVERRIDE,
                               PRIV_MAC_READ,PRIV_MAC_WRITE:TEXT_NULL;
                        ];
                 PAL = [
                        sysadm:PRIV_MAC_READ:mytext;
                        ];
                 MAC = [syslow,none];
```

The following shows the output from the executing the `getprivs` command after this PAL has been set:

```
$ getprivs /bin/dog
a:PRIV_NULL
f:PRIV_DAC_OVERRIDE,PRIV_MAC_READ,PRIV_MAC_WRITE
s:PRIV_DAC_OVERRIDE,PRIV_MAC_READ,PRIV_MAC_WRITE
```

The following shows the output the `getpal` command after this PAL has been set:

```
$ getpal bin/dog
other:PRIV_NULL:TEXT_NULL
secadm:PRIV_DAC_OVERRIDE,PRIV_MAC_READ,PRIV_MAC_WRITE:TEXT_NULL
system:PRIV_DAC_OVERRIDE,PRIV_MAC_READ,PRIV_MAC_WRITE:TEXT_NULL
sysadm:PRIV_MAC_READ:mytext
```

FILES

`/etc/privdb/main.db`	Database that contains the default file and directory names and the security attributes to be assigned by `privcmd`.
`/etc/config/localpriv.db`	File that contains the site-modified security attributes to be assigned by `privcmd`.

SEE ALSO

`getpal`(8)

General UNICOS System Administration, Cray Research publication SG–2301

NAME

prtacct – Prints total accounting file

SYNOPSIS

/usr/lib/acct/prtacct *file* ["*heading*"]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The prtacct shell script formats and prints any total accounting (tacct.h format) file generated by the standard UNIX System V accounting package by invoking the acctmerg(8) command. prtacct requires that you specify the name of the tacct file.

The prtacct script accepts the following operands:

file Name of the tacct file. This operand is required.

"*heading*" A string enclosed in double quotation marks that defines the page header.

EXAMPLES

The following example prints the total accounting file daytacct with a defined header:

```
/usr/lib/acct/prtacct daytacct "DAILY USAGE REPORT FOR TODAY'S DATE"
```

SEE ALSO

acct(8), acctmerg(8), acctsh(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

pwck, grpck – Checks password and group files for consistency

SYNOPSIS

/etc/pwck [*file*]
/etc/grpck [*file*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The pwck command scans the password file and notes any inconsistencies. The checks include validation of the number of fields, login name, user ID, group ID, and whether the login directory and optional program name exist. The default password file is /etc/passwd.

The grpck command verifies all entries in the group file. This verification includes a check of the number of fields, group name, group ID, and whether all login names appear in the password file. The default group file is /etc/group.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Allowed to specify any file.
sysadm	Allowed to specify any file, subject to security label restrictions. Shell redirected output is subject to security label restrictions.

If the PRIV_SU configuration option is enabled, the super user is allowed to specify any file.

Although these commands exist on a UNICOS system, /etc/udbgen (see udbgen(8)) assures the synchronization of these files.

MESSAGES

Group entries in /etc/group without login names are flagged.

FILES

/etc/group

/etc/passwd

SEE ALSO

group(5), passwd(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

quadmin – Administers file quotas

SYNOPSIS

```
/etc/quadmin -c request [-D] [-p fstab_path] [-s file_systems]
/etc/quadmin -c request [-D] [-d directive] [-p fstab_path]
/etc/quadmin -c request [-D] [-p fstab_path] [srcfile]
/etc/quadmin -m [-D] [-d directive] [-F] [-p fstab_path] [-s file_systems] [-V version]
/etc/quadmin -m [-D] [-F] [-p fstab_path] [-s file_systems] [-V version] [srcfile]
/etc/quadmin -Q [-D]
/etc/quadmin -v [-a request] [-D] [-p fstab_path] [-s file_systems]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The quadmin command provides an interface to the file quota system that allows you to control the quota system (using the -c option), to modify the information used by the quota system (using the -m option) and to view status information (using the -v option). Any use of the command, except when the -Q or -v options are selected, requires that the file system names be specified with the -d or -s option or from a source file (*srcfile*) or stdin.

Either the -c, -m, -Q, or -v option is required. The quadmin command accepts the following options and operands:

-a *request* Displays quota association information. This option may appear only with the -v or -D options. It is used to see how the current quota file systems are configured.

-c *request* Performs *request* on the specified file systems. This option cannot be used with the -m, -Q, or -v option. Valid *requests* are count, default, inform, and enforce. The requests match the quotactl(2) requests to manage the enforcement level of the quota system. If a *srcfile* has been specified, only filesystem and set directives are meaningful, and other directives are silently ignored. This is done to allow the same source file to be used with all quadmin modes of operation needing a source file. If the -s option is selected, only those file systems named will have a control action; otherwise, all file systems named in the source are affected.

-d *directive* Supplies directive input on the command line. Valid directives are described later in the entry. This option presents the command with a *directive* as though it had been read from a file. The normal source rules are followed, allowing multiple directives separated by semicolons and escaped new lines. The only way to change a quota is through an explicit directive; omitting an explicit directive will **not** cause a quota to be reset to the default. This option must not be specified with the -Q or -v options or *srcfile*.

`-D` Sets debug mode. This option is useful only for testing.

`-F` Forces quota file access. This option bypasses any use of the `quotactl`(2) interface and directly accesses the quota control files. It is intended for use only when the quota system is inactive; otherwise, information in the files could be altered by the kernel at any time. An example of when the `-F` option is used is when `quadmin` is used to create a quota control file from `qudu`(8) output. This option cannot be used with the `-c`, `-Q`, or `-v` options.

`-m` Changes quota information if it exists; otherwise, creates new quota control files. This option must not be specified with the `-c`, `-Q`, or `-v` options.

`-p` *fstab_path* Reads */fstab_path*/*fstab* and/or the kernel mount table to find the configuration information.

`-Q` Displays the default name of the quota file on `stdout`. Scripts that manipulate the quota files have need of this feature. Only the `-D` option can be used with this option.

`-s` *file_systems* Defines the specified file systems. The *file_systems* argument is a list of one or more file system names separated by commas. When this option is specified, `quadmin` bypasses reading the kernel mount table; the specified file systems are the only ones known to the command. The root directory names rather than device names should be used with this option. If this option is used with the `-c` option, only those file systems listed have a control action, and no source file is read. If this option is used with the `-v` option, the quota file name associated with the named file system or systems is written to the `stdout` file. If other file systems share the same quota file, the names of those are appended to the same line of output. The file system that appears in `/etc/fstab` with the quota option defining the control file is always first, following the quota control file name. To get a display of all quota file systems, use the option `-s all`.

`-v` Views status of quota control information. This option must not appear with the `-c`, `-m`, or `-Q` options. The output is written to the `stdout` file.

`-V` *version* Controls conversion from previous source language forms. Only the values 5 and 6 are allowed for *version*. The value 5 signifies conversion from the first release (UNICOS 5.1) format; the value 6 signifies that the source is in UNICOS 6.0 format (this is the default). This option takes precedence over a version directive in the source file. The `-V` option is allowed only with the `-m` option.

Directives

Directives that can be specified in the input file for `quadmin` or with the `-d` option have the following functions:

- They establish the relationship between file systems and quota control files.
- They contain information to place in the quota files.

The following conventions are used in the descriptions of directives:

- Optional fields are indicated by square brackets ([ ]).
- *Italics* indicates data that you supply.
- The vertical bar (|) character indicates that you should enter one of the items separated by the symbol or symbols and enclosed in curly braces ({ }).
- An unescaped end of line, semicolon, or null ends a directive.

The directives are processed sequentially in the order in which they appear within the file. Words in typewriter font can be shortened to their minimum unique length as shown in the Abbreviations subsection.

The following directives are valid:

# Indicates a comment. All text from this character to the end of the current line is ignored.

`version` *ver*

Specifies which version of the software was used to generate directive files. The `quota`(1) and `qudu`(8) commands insert this directive when generating `quadmin`-style output. This directive must appear before anything version-specific is encountered in the file; it is effective for all subsequent lines of directive input. (Quota commands prior to UNICOS 6.0 did not generate or recognize this directive, so if files generated on release 5.1 are to be used, this directive should be placed at the beginning of each file with *ver* set to 5.) *ver* may take the value 5 or 6. If *ver* is 5, conversion of warning windows to warning values will be done in order to compensate for the warning mechanism change between UNICOS 5.1 and 6.0. The value 6 means that the directives are in 6.0 form, so no conversion needs to be done (this is the default).

`set` *vname string*

Associates the variable name *vname* with the string *string*. Only alphanumeric characters are allowed in *vname*. As with shell variables, the string may be substituted into other directives by using the notation $*vname* or ${*vname*}. The string begins at the first nonwhite-space character following *vname* and ends at the end of the line with trailing white space removed. (A comment (#) or semicolon is treated as an end of line for the purposes of editing trailing white space.) A `set` directive naming an already specified *vname* replaces the old definition with the new. The variable name `QFILE` is predefined to be the default name of the quota file. Shell variable names are also allowed but they will be accessed only if the name is not predefined and has not been defined through `set`.

`filesystem` *fsname* [*qfname*]
: Specifies a file system, *fsname*, and associates it with a quota control file, *qfname*. *qfname* should be omitted if the quota configuration is defined in `/etc/fstab` as recommended. If this is not the case, *qfname* may be `.` or omitted if the quota file is to have the default name and is to reside in the root directory of *fsname*; otherwise, the full name is required. Once established, this relationship is permanent within a directive file. This directive declares the file system and its association with a quota control file, but `open` directives must be used to make each file system eligible for use within the directive file. Root directory names rather than device names are recommended for *fsname*. A new *fsname* automatically becomes a member of the current list of open file systems.

`open` *fsname* [*fsname* ...]
: Makes the listed file system or systems, *fsname*, eligible for subsequent `account`, `acid`, `default`, `gid`, `group`, `uid`, and `user` directives. The special *fsname* `*` may be used to refer to all file systems known through previous `filesystem` directives. Each `open` redefines the list of eligible file systems.

The `default` directives below are used to set *generic* file system parameters (those not involved with specific ID classes).

`default algorithm` {`exponential` | `linear` | `none` | `site1` | `site2`}
: Three soft-quota control algorithms are provided with the released software and two additional algorithms can be supplied by the installation. Algorithm selection is specified in each quota control file and is `none` by default. The algorithm named `none` is compatible with UNICOS 5.1.

`default {ef1 | ef2}` [+ | -]*number*
: These are algorithm-specific fields declared as `long` in the header structure. Signed values are acceptable. The algorithm-specific fields depend on the algorithm selected by the `default algorithm` directive.

`default level` {`count` | `enforce` | `inform` }
: Specifies the quota enforcement level that will be imposed by `mount`(8) or `quadmin` with the `-c default` option. The default enforcement level is `count`.

`default time` *number*[`s | m | h | d`]
: This feature is not currently implemented, although the field is present. The time field is used to specify a minimum migration preference threshold. The default unit is seconds, but a suffix selected from the list `s` (seconds), `m` (minutes), `h` (hours) or `d` (days) may be used. The suffix must immediately follow the final digit of the number. A value of 0, the default, disables this feature.

`default style [online | aggregate]`

Specifies whether quota counts are maintained only for files that are physically on disk (online), or whether files migrated offline by the Data Migration Facility (DMF) are included in the count (aggregate).

The default is online quotas. In this case, when a file is migrated offline by DMF, the disk blocks released are subtracted from the user, group, and account totals. This allows a user, group, or account to have more "virtual" disk blocks then allowed for by the quota limit. The quota limit only applies to files actually on the disk.

If aggregate quotas are selected, the space for both online and migrated (offline) files are counted as part of the user, group, or account usage. In this case, the quota limits apply to the sum of the online disk blocks and the offline disk blocks. Aggregate quotas would be selected in order to limit the amount of space used in the offline storage facility, e.g., a tape silo.

The remaining `default` directives are ID-class specific.

`default` {`account` | `group` | `user`} `flags` *flags*

Binds the enable *flags* in the quota file header associated with the specified `account`, `group`, or `user` to all currently eligible *fsname*s. If a flag is off, quotas are not controlled for that class and type. The following values can be used for *flags*:

`f` File quotas only.

`i` Inode quotas only.

`fi` Both file and inode quotas. This value is the default.

`off` Quota checking disabled by default for the named ID class.

The binding is permanent within a directive file.

`default` {`account` | `group` | `user`} `file quota` *integer*

Binds the default `account`, `group`, or `user` file quota to the number of Cray file blocks (4096 bytes) specified by *integer* to all currently eligible *fsname*s. The value specified as the default file quota for the respective account, group, or user in the `sys/quota.h` file is used if this directive is not present. The binding is permanent within a directive file.

`default` {`account` | `group` | `user`} `inode quota` *integer*

Binds the default `account`, `group`, or `user` inode quota to the number of inodes specified by *integer* to all currently eligible *fsname*s. The value specified as the default inode quota for the respective account, group, or user in the `sys/quota.h` file is used if this directive is not present. The binding is permanent within a directive file.

`default` {`account` | `group` | `user`} `file warning` *float*
: Binds the default `account`, `group`, or `user` file quota warning to the value specified by *float* to all currently eligible *fsname*s. If the value is in the range 0.0 < *float* < 1.0, it is the fraction of the file quota at which a warning occurs. If the value is > 1, it is an absolute warning value. If the value is 0 or 1, the warning is disabled, and no warning is issued. The value specified as the default file warning for the respective account, group, or user in the `sys/quota.h` file is used if this directive is not present. The binding is permanent within a directive file.

`default` {`account` | `group` | `user`} `inode warning` *float*
: Binds the default `account`, `group`, or `user` inode quota warning to the value specified by *float* to all currently eligible *fsname*s. If the value is in the range 0.0 < *float* < 1.0, it is the fraction of the inode quota at which a warning occurs. If the value is > 1, it is an absolute warning value. If the value is 0 or 1, the warning is disabled, and no warning is issued. The value specified as the default inode warning for the respective account, group, or user in the `sys/quota.h` file is used if this directive is not present. The binding is permanent within a directive file.

The directives listed previously set the defaults for the specified file system or quota file. You may specify the defaults individually for each file system or apply the same defaults to as many file systems as were eligible when the default directives were encountered.

`remove` {`acid` | `account` | `gid` | `group` | `uid` | `user` | `all`} `usage`
: This directive will set file and inode usage values for all quota entries in the named category on all open file systems to 0. This directive is acted upon when it is encountered in the input file. You should use this whenever an entirely new set of usage values are to be set and no memory of previous usage is wanted. `qudu`(8) uses this directive when it writes a file of current usage information.

The remaining directives establish quota values for individual or groups of IDs. All file systems already opened with an `open` directive are affected. The special name `*` means all users, account IDs, user IDs, or group IDs.

`enable` {`uid` | `acid` | `gid`} [*range*]
: Evaluates the specified ID with respect to the inclusive *range* and, if true, enables the directive (this is used only with `*`). Valid expressions are as follows: `n-nn`, `n-`, `-nn`, and `n`. If two values are separated by a dash (`-`), the left side must be less than or equal to the right side. All values are unsigned integers or named variables that resolve into unsigned integers or recognized range constructs. To select values 50 and above, you would use the expression 50-; for values 1023 or less, you would use -1023; values from 100 through 2049 would be selected with the expression 100-2049; a single value of 35 would be selected with 35 or 35-35. You can specify multiple ranges by separating one range from another with a comma. To enable ranges 10 through 50 and 1000 through 1099, you would enter 10-50, 1000-1099. When multiple ranges are specified, the first satisfied range reading from left to right makes the condition true. White space is ignored in the range description. Each `enable` directive redefines the range for the specified ID class and removes the previous range description. The default range is 0-. To restore the default range, leave the range field empty.

The directives used to set values in individual or groups of quota file entries are made up of the components in the following list. The EXAMPLES section shows how the parts may be combined.

When creating directives, you may shorten them by using *implicit value typing*, which is built into the input process. As the line is scanned from left to right, previous classifications are remembered and used if needed. Therefore, instead of typing `default account file quota 123 file warning .8 inode quota 12 inode warning .95`, you could type `default account file quota 123 warning .8 inode quota 12 warning .95`. This applies to each line individually and is equally valid with the following directives.

The classification part of the directive must be first in the statement and consist of one of the following:

`account` *aname*
: Sets the specified values in account *aname*. If *aname* is `*`, all accounts in the user database (UDB) whose `acid`s satisfy the conditions established through the `enable acid` directive are set.

`acid` *id*
: Sets the specified values in account *id*. If *id* is `*`, all accounts in the UDB whose `acid`s satisfy the conditions established through the `enable acid` directive are set.

`group` *gname*
: Sets the specified values in group *gname*. If *gname* is `*`, all accounts in the UDB whose `gid`s satisfy the conditions established through the `enable gid` directive are set.

`gid` *id*
: Sets the specified values in group *id*. If *id* is `*`, all accounts in the UDB whose `gid`s satisfy the conditions established through the `enable gid` directive are set.

`user` *uname*
: Sets the specified values for user *uname*. If *uname* is `*`, all accounts in the UDB whose `uid`s satisfy the conditions established through the `enable uid` directive are set.

`uid` *id*
: Sets the specified values for user *id*. If *id* is `*`, all accounts in the UDB whose `uid`s satisfy the conditions established through the `enable uid` directive are set.

The next part of the statement must consist of one or more of the following value specification parts (these may be combined as necessary to establish all of the information; implicit value typing is allowed).

`file quota` {*integer* | `default` | `unlimited` | `infinite`}
: Sets the file quota to a specific *integer* value, `default`, or `unlimited`. `infinite` is a synonym for `unlimited`. The *integer* value specifies the number of 4096-byte blocks, by default. The value 0 means that file allocation is prevented on the file system.

`file usage` *integer*
: Sets the file usage to a specific value, *integer*. Typically, this would be used with only one file system open.

`file warning` {*float* | `default`}
: Sets the file warning to the value specified by *float* or `default`. If the value is in the range 0.0 < *float* < 1.0, it is the fraction of the file quota at which a warning occurs. If the value is > 1, it is an absolute warning value. If the value is 0 or 1, the warning is disabled, and no warning is issued. If the value of *float* is between 0 and 1, the warning value is computed by multiplying the inode quota by *float*.

`inode quota` {*integer* | `default` | `unlimited` | `infinite`}
: Sets the inode quota to a specific *integer* value, `default`, or `unlimited`. `infinite` is a synonym for `unlimited`. The value 0 means that new inode allocation is prevented on the file system.

`inode usage` *integer*
: Sets the inode usage to a specific *integer* value. Typically, this would be used with only one file system open.

`inode warning` {*float* | `default`}
: Sets the inode warning to the value specified by the floating value, *float*, or `default`. If the value is in the range 0.0 < *float* < 1.0, it is the fraction of the inode quota at which a warning occurs. If the value is > 1, it is an absolute warning value. If the value is 0 or 1, the warning is disabled, and no warning is issued. If the value of *float* is between 0 and 1, the warning value is computed by multiplying the inode quota by *float*.

The fields described below apply to the ID class and are not related to file or inode quotas. These fields can be placed between any of the previous specifications but will be more easily read if they are placed before file and inode specifications are written.

`time` [`+ | -`]*number*[`s | m | h | d`]
: This sets the time when the warning threshold was exceeded. Typically, this field need not be specified, but it is included for testing or other special needs. Time may be specified in either absolute or relative (to the time the directive is processed) form. Absolute form expects time in internal notation (seconds since January 1, 1970), so it is not very easy to use. The relative form is a signed value, while the absolute form is unsigned. The default unit for both forms is seconds, but a suffix selected from the list `s` (seconds), `m` (minutes), `h` (hours), or `d` (days) may be used. If the relative form is used, minus (-) means the past and plus (+) means the future. If present, the sign must immediately precede and the suffix must immediately follow the number. Set the current time using the relative notation +0. The value of this field is important with respect to the algorithm in use. See the algorithm descriptions for more information.

{`ef1 | ef2 | ef3 | ef4`} *number*
: These four fields are 32-bit unsigned integers whose meaning is algorithm-dependent; the meaning depends on the algorithm selected by the `default algorithm` directive.

`ef5` [`+ | -`]*number*
: This is a signed `long` field whose meaning is algorithm-dependent; the meaning depends on the algorithm selected by the `default algorithm` directive.

`runquota` *number*
If this field is nonzero, it is assumed to be less than `file quota` and becomes the currently enforced file quota. The soft-quota algorithms use this field to adjust the file quota enforcement value without altering the actual file quota.

`delete` Deletes the quota entry by setting the usage value to 0 and the limit and warning values to their default values.

Abbreviations

The following minimum abbreviations are allowed in directives. Square brackets surround the optional part of each word. If an abbreviation is too short to be unique, an `ambiguous word` fatal error occurs.

Statement introductory words:
`acc`[ount], `aci`[d], `d`[efault], `e`[nable], `f`[ilesystem], `gi`[d], `gr`[oup], `o`[pen], `r`[emove], `s`[et], `ui`[d], `us`[er] `v`[ersion].

Value typing and other modifier words:
`acc`[ount], `aci`[d], `ag`[gregate], `alg`[orithm], `all`, `c`[ount], `def`[ault], `del`[ete], `ef1`, `ef2`, `ef3`, `ef4`, `ef5`, `en`[force], `ex`[ponential], `fi`[le], `fl`[ags], `gi`[d], `gr`[oup], `infi`[nite], `info`[rm], `ino`[de], `le`[vel], `li`[near], `n`[one], `on`[line], `q`[uota], `r`[unquota], `site1`, `site2`, `st`[yle], `t`[ime], `ui`[d], `un`[limited], `usa`[ge], `use`[r], `w`[arning].

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to specify any file or device name.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to specify any file or device name.

MESSAGES

Possible messages from `quadmin` fall into the following categories:

- Usage error messages caused by improper option selection, and warning or fatal messages from the source.
- Messages caused by errors in the source; these include the line number on which the error was detected.
- Source lines that consist of multiple input lines with escaped new-line terminators; these inherit the line number of the first line of the group.

EXAMPLES

Example 1: Assume that you have file systems `/u`, `/v`, and `/w` defined as user home file systems, and their quota control files reside on the respective file system. In your environment, account IDs greater than 10, group IDs between 10 and 300, and user IDs greater than 49 are to have quotas. Also, file system `/tmp` is to have only user file quotas of 10,000 blocks. To define quotas for these file systems and give all users, accounts, and groups default quotas, you need the following source file:

```
    version 6
    filesystem /u ; filesystem /v ; filesystem /w ; filesystem /tmp
    open /tmp

      default account flags off
      default group flags off
      default user flags fi
      default user file quota 10000 warning .9 inode quota 200 warning .9
      user * file quota unlimited inode quota unlimited

open /u /v /w
  default account flags fi
  default account group fi
  default account user fi
  default account file quota 35000 file warning 0.85
  default account inode quota 400 warning 0.85
  default group file quota 35000 warning 0.85
  default group inode quota 400 inode warning 0.85
  default user file quota 15000 warning 0.85 inode quota 200 warning 0.85

#       Set special ranges of IDs to unlimited quota
  enable acid -10       # all acids 0 through 10
  enable gid -10, 300-  # all gids 0 through 10 and 300 and up
  enable uid -49        # all uids 0 through 49
  account * file quota unlimited inode quota unlimited
  group * file quota unlimited inode quota unlimited
  user * file quota unlimited inode quota unlimited

  #  Give user 'def' a small file quota on file system /v
open /v
  user def file quota 5000 warning default
```

Example 2: To set the disk quota for multiple groups and users, on file system `/core`, and quota control file `q`, first create a source file (*srcfile*) such as the following:

```
version 6
filesystem /core q
open /core
  group 16 file quota 6000 inode quota 2000
  group 21 file quota 6000 inode quota 2000
  user 100 file quota 6000 inode quota 2000
  user 101 file quota 6000 inode quota 2000
  user 102 file quota 6000 inode quota 2000
```

The first three lines of this file are required. You can add any number of groups or users. Then execute `quadmin` using the following command-line option:

`/etc/quadmin -m` *srcfile*

The file warning and inode warning values remain unchanged.

Example 3: For a single user (`uid 100`), on file system `/core`, and quota control file `q`, to change the file usage to `800`:

```
/etc/quadmin -Fmd 'filesys /core q; uid 100 file usage 800'
```

This changes only the file usage field; other fields in the quota control file remain unchanged.

FILES

`*/.Quota60` Quota control file

SEE ALSO

`qudu`(8)

`quota`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

qudu – Reports file system quota usage information

SYNOPSIS

/etc/qudu [-a] [-A] [-D] [-f] [-G] [-p *fstab_path*] [-q *quota_file*] [-U] *device_names*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The qudu command traverses all of the inodes on the specified *device_names* and formats usage information for input to the quadmin(8) command. Output is sent to the stdout file.

The manner in which information is recorded by this command matches the manner in which the kernel records file system space usage when quota enforcement is active; it may not exactly match the usage information derived from other usage analysis tools.

The qudu command accepts the following options and operand:

-a Uses aggregate quota counting method. Files migrated offline by DMF are included in the block counter.

-A Selects account IDs. If -A, -G, or -U is not present, the default is -AGU.

-D Sets debug mode. qudu sends information helpful in determining its internal workings to the stderr file. The function of the command is not affected.

-f Reads /etc/fstab to find the quota configuration information for *device_names*. This option is not allowed with the -p or -q option.

-G Selects group IDs. If -A, -G, or -U is not present, the default is -AGU.

-p *fstab_path* Reads */fstab_path/fstab* to find the configuration information for *device_names*. This option is not allowed with the -f or -q option.

-q *quota_file* Places the specified quota file name on the filesystem directive to be written to the stdout file. This option is intended for use when the output from qudu will be used as input to quadmin(8). For compatibility with the previous release of qudu, if -f or -p has not been specified, *quota_file* is placed on the filesystem directive as the name of the quota file (null if -q is not specified). If either -f or -p has been specified, the quota file name is derived from fstab, and -q is not allowed.

-U Selects user IDs. If -A, -G, or -U is not present, the default is -AGU.

device_names The names of the device nodes from which usage should be extracted. Multiple device names may be included only if the `-f` or `-p` option is not present. In this case, usage over all specified devices is summed in the output. If `-f` or `-p` is present, one of the devices in a quota group (the only device if the group is singular) must be named; `qudu` sums usage over all devices in the group, as configured in `fstab`.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to specify any file or device name.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to specify any file or device name.

EXAMPLES

The following example collects disk usage for the `/dev/dsk/slash_a` file system mounted on the `/a` directory. Assume that `/etc/fstab` is set up to configure the file systems `/dev/dsk/slash_a` mounted on `/a`, `/dev/dsk/slash_b` mounted on `/b`, and `/dev/dsk/slash_c` mounted on `/c` as a quota group, with the quota file defined to be `/a/$QFILE`. Sample output follows the command line.

```
/etc/qudu  -f  /dev/dsk/slash_a

# Usage report by qudu: (SN-1203) on Mon Feb 12 14:11:01 CST 1990
version 6
filesystem dsk/slash_a /a/$QFILE
filesystem dsk/slash_b /a/$QFILE
filesystem dsk/slash_c /a/$QFILE
open dsk/slash_a dsk/slash_b dsk/slash_c
remove all usage
uid 0 inode usage 2 file usage 610
uid 559 inode usage 13 file usage 91
gid 0 inode usage 2 file usage 610
gid 117 inode usage 13 file usage 91
acid 0 inode usage 19 file usage 713
acid 559 inode usage 13 file usage 91
```

FILES

`sys/quota.h`	Quota control definition file
`/etc/fstab`	File describing file system and swapping partitions used by UNICOS

SEE ALSO

`quadmin`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

rdump – Performs a file system dump across a TCP/IP network

SYNOPSIS

/etc/rdump -f *machine*:*device* [-a] [-b *nbs*,*pbs*] [-d *density*] [-D *device*] [-e] [-m *capacity*] [-n] [-t *dump_level*] [-T *t_fmt*] [-u] [-v *vsn_list*] *filesystem*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The rdump command is a shell script that establishes a data path from *filesystem* to *machine*:*device* for backup purposes. The path goes from *filesystem* through an /etc/dump process across a pipe into an /etc/lrmt process across a socket connection into a process running /etc/rmt on a remote machine and from there into *machine*:*device*.

You can use any pre-existing *device* on the remote *machine* to contain the output; *device* does not have to be a tape.

The parameters that are accepted are a combination of the parameters for /etc/dump and for /etc/lrmt.

The rdump command accepts the following options:

-f *machine*:*device*
: A required parameter. This is passed to lrmt as the network destination.

-a
: A flag that is passed to dump.

-b *nbs*, *pbs*
: This option is passed without change to lrmt.

-d *density*
: An option that is passed to dump.

-D *device*
: An option that is passed to dump.

-e
: A flag that is passed to dump.

-m *capacity*
: This option provides for volume switching control on Cray Research systems. The value *capacity* is taken to be the number of megabytes in each output volume. This is passed to dump as -m *capacity* and to lrmt as -c *capacity*. If dump or lrmt report a short write error, reduce the *capacity* value. This condition can occur when many small files are being dumped or when the block size of tape records is small on the destination machine.

-n
: A flag that is passed to dump.

-t *dump_level*
: An option that is passed to dump.

-T *t_fmt*
: An option that is passed to dump.

-u
: A flag that is passed to dump.

`-v` *vsn_list* An option that is passed to `dump`.

filesystem The `/dev/dsk` name of a mountable file system to be passed to the `dump` command.

The following `dump` options are not supported through the `rdump` interface:

`-f` `rdump` sets `-f -` when executing `dump`.

`-g` Parameter for local `rsv` and `tpmnt`.

`-l` Parameter for local `tpmnt`.

`-c` Equivalent to `-g CART`.

`-s` Input to capacity computation for round tapes.

`-M` A format for `tpmnt`.

`-R` A format for local `rsv`.

`-P` A format for prompting operator before `tpmnt`.

`-w` Use dump for this listing.

`-W` Use dump for this listing.

The following `lrmt` options are not supported through the `rdump` interface

`-p` Use a named pipe rather than `stdin` and `stdout`.

`-d` This is set to `-d n`.

FILES

`/etc/dumpdates`	Dump date record
`/etc/fstab`	File systems description
`/etc/group`	Group-information file (to find group *operator*)

SEE ALSO

`dump`(8), `lrmt`(8), `restore`(8), `rmt`(8), `rrestore`(8)

NAME

reduce – Extracts, formats, and outputs UNICOS security event files

SYNOPSIS

/etc/reduce [[-i] | [-I *optfile*]] [-s *date*] [-e *date*] [-t *typelist*] [-l *uidlist*] [-u *uidlist*] [-c *functionlist*] [-g *gidlist*] [[-r] | [-R *rawfile*]] [-f *logfile*] [-m *num*] [-v] [-j *jidlist*] [-b *pidlist*] [-n *filename*] [-p] [-x] [-o *object_level_list*] [-O [*level*[,*compartment*[,*compartment*[...]]]]] [-S]

/etc/reduce [[-i] | [-I *optfile*]] [-s *date*] [-e *date*] [-t *typelist*] [-l *uidlist*] [-u *uidlist*] [-c *functionlist*] [-g *gidlist*] [[-r] | [-R *rawfile*]] [-f *logfile*] [-m *num*] [-v] [-j *jidlist*] [-b *pidlist*] [-n *filename*] [-p] [-x] [-o *object_level_list*] [-O [*level*[,*compartment*[,*compartment*[...]]]]] [-L]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The reduce command extracts and formats entries collected by the security log daemon, slogdemon(8), and generates a report of that data. The default is to report all entries currently in the active security log file. The default path for the log file is /usr/adm/sl/slogfile; however, this path can be site-defined.

If the SECURE_MAC configuration option is enabled, the security log file is labeled at syshigh.

Alternatively, you can process one of the retired security log files that has been date-named in the security log directory (/usr/adm/sl) or any file that is in the correct security log file format. You can use one or more of the options to limit the report to specific entries that are of immediate interest. Using the -r or -R option to create a pre-reduced security log file can be followed by issuing reduce with the -f option to access that file.

The -i and -I options should not be used on the same command line; only one is needed for correct results. They have identical functions, except that -i uses a default file, and you must specify an input file for -I.

The -r and -R options also should not be used on the same command line. They have identical functions, except that -r uses a default file, and you must specify an output file for -R.

The reduce command accepts the following options:

-i Reads program options from the opt.reduce file in the current directory. Any options occurring in the command line override those found in the option file.

-I *optfile* Reads program options from the file *optfile*, which may be a relative or full path name. Any options specified in the command line override those found in the option file.

`-s` *date* Ignores all entries posted earlier than *date*; *date* has the format *mmddhhmm*[*yy*[*ss*]] and is consistent in meaning with the `date`(1) command. This option also causes `reduce` to select files for processing (using parameters in the include file `/uts/cf.SN/config.h` that define retired security log file names) to ensure that the requested start time is reached. The `reduce` command searches the security log directory for the first (earliest) retired or active log containing the start date. This log and all later logs residing in the directory are processed until an end-of-processing condition is reached. If `now` is specified for *date*, `reduce` begins reading new entries; this should be used with the `-x` option.

`-e` *date* Ignores all entries posted later than *date*; *date* has the format *mmddhhmm*[*yy*[*ss*]] and is consistent in meaning with the `date`(1) command. When an entry with a date later than *date* is encountered, processing ends.

`-t` *typelist* Selects only entries matching the types in the *typelist* (a list of type identifiers separated by spaces or commas). Valid types and their meanings are as follows:

`audit`	Security auditing criteria selection changes
`cchg`	System configuration change
`chdir`	Change directory
`crl`	Cray/REELlibrarian activity
`dac`	Discretionary access control changes
`disc`	Discretionary access event
`disk`	Disk I/O error
`eoj`	End-of-job
`go`	System startup
`logn`	Login validation process
`mand`	Mandatory access event
`nami`	File manipulation system calls
`netcf`	Network configuration changes
`netip`	IP layer security violations
`netw`	Network access
`nfs`	Cray NFS activity
`nqs`	NQS activity
`nqscf`	NQS configuration changes
`oper`	Operational access event
`priv`	Use of privilege
`secsys`	Enhanced security system calls
`setuid`	`setuid` system calls
`ssd`	SSD I/O error
`stop`	System shutdown
`suid`	`setuid` file manipulation system calls
`sulog`	`su`(1) attempts
`tape`	Tape I/O error
`tchg`	System time change
`trust`	Trusted process activity

`-l` *uidlist* Selects only entries whose login user ID (which cannot change) matches one in *uidlist* (a list of numeric user IDs or ASCII user name specifiers separated by spaces or commas). This option is the surest way of getting true accountability. All actions performed after a user logs in contain the login user ID in the security log entry for that action.

`-u` *uidlist* Selects only entries whose subject user ID (real or effective) matches one in *uidlist* (a list of numeric user IDs or ASCII user name specifiers separated by spaces or commas).

`-c` *functionlist* Selects only the function-related security log entries (for example, mandatory/discretionary access, enhanced security system calls, and so on) whose function fields match one of those in *functionlist* (a list of valid functions separated by spaces or commas).

Valid functions are as follows:

```
acctid
alloc
alloct
chdir
chmod
chown
cpselect
creat
exec
exit
fork
fsecstat
getacl
getfacl
getsysv
ialloc
kill
limit
link
mkdir
mknod
mount
nicem
open
read
reada
resume
rmacl
rmfacl
rmdir
scrub
secstat
```

```
select
setacls
setdevs
setfcmp
setfflg
setflvl
setregid
setsid
setsockopt
setsysl
setsysv
setucat
setucmp
setulvl
setusrv
shutdown
suspend
tabinfo
tabread
trunc
umount
unlink
write
writea
```

`-g` *gidlist* Selects only entries whose subject group ID (real or effective) matches one in *gidlist* (a list of numeric group IDs or ASCII group name specifiers separated by spaces).

`-r` Writes a raw dump of the selected entries to the file `raw.reduce` in the current directory. An existing `raw.reduce` file is overwritten.

`-R` *rawfile* Writes a raw dump of the selected entries to the file *rawfile*; *rawfile* can be a full or relative path name. Any existing *rawfile* is overwritten.

`-f` *logfile* Uses the file *logfile* instead of the default file defined in the kernal via `config.h`. This option results in only one file being processed.

`-m` *num* Halts processing after a specified number of entries, *num*, have been selected for output.

`-v` Turns on verbose mode. Default is off. When on, `reduce` echoes the options.

`-j` *jidlist* Selects only entries whose job ID (session) matches those in *jidlist*, a numeric list of job IDs separated by spaces or commas.

`-b` *pidlist* Selects only entries whose process ID matches those in *pidlist*, a numeric list of process IDs separated by spaces or commas.

`-n` *filename* Selects only the entries for which the object file name component of the relative path name is the same as *filename*. This is applicable only to entry types that contain a relative path name.

`-p` Reconstructs a path name. For each entry that contains a relative path name, `reduce` attempts to reconstruct the full path name being referenced from the previous path history. The `SLG_PATH_TRACK` parameter (in `/uts/cf.SN/config.h`) must be enabled to use this option.

`-x` Continues to read the log file past EOF in the manner of the `tail`(1) command, reporting entries according to the selection criteria as they are received. This option functions correctly only if the file being processed is the same as the file currently being written to by the security logging daemon, `slogdemon`. The `-x` option can be used in conjunction with the `-s now` option to select only events occurring after `reduce` is initiated.

`-o` *object_level_list*
Selects only the entries whose object level matches one of the levels in *object_level_list*, a list of decimal level values separated by commas or spaces. Cannot be used with the `-O` option.

`-O` Selects only entries whose object label matches the specified value. The object label is specified in the form *level*[,*compartment*[,*compartment*[...]]]. *level* or *compartment* is the name or numeric value that represents the appropriate *level* or *compartment*, respectively. If no *compartments* are specified, or the text string `none` or `0` is used, then the compartment bitmask is set to `0`. Cannot be used with the `-o` option.

`-S` Prints the complete security label of both the subject and the object as part of the record header. The form of the subject's security label is `S_Label`: *level,comp1,comp2,...,compn*. The form of the object's security label is `O_Label`: *level,comp1,comp2,...,compn*. When the subject's and object's security labels are printed in the header, no subject or object label information is printed in the record body. This option will become the default header format in a future UNICOS release. See the `-L` option.

`-L` Prints the subject's and object's security level only as part of the record header. This record header format is the default format for the UNICOS 8.0 and later systems; it is needed to maintain compatibility with previous UNICOS releases. It will not be supported in future releases of the UNICOS operating system. See the `-S` option.

See *General UNICOS System Administration*, Cray Research publication SG–2301, for information on the `reduce` header fields.

NOTES

If this command is installed with a privilege assignment list (PAL), a user who is assigned the following privilege text upon execution of this command is allowed to perform the action shown:

Privilege Text	Action
`exec`	Allowed to use this command.

If this command is installed with a PAL, a user with one of the following active categories is allowed to perform the action shown:

Active Category	Action
`system`, `secadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

Using the `-p` option may cause `reduce` to operate more slowly and consume more memory than usual. Because of this effect, if the `-s` option is specified, path name reconstruction does not begin until the start time is reached. This may cause incomplete path name reconstruction if the process started before the start time selected.

FILES

`/usr/include/sys/slrec.h`	Defines security log record format
`/usr/include/sys/slog.h`	Defines security log file internal (kernel) structure
`/usr/src/uts/cf.SN/config.h`	Defines security log file parameters
`/usr/adm/sl/slogfile`	Repository for security event records (defined in the kernal through `config.h`)

SEE ALSO

`slogdemon`(8)

`privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`slrec`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

regpar – Generates register parity errors

SYNOPSIS

/etc/regpar -r *register* -p *parity*

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The regpar command allows you to generate register parity errors in Revision 4 and subsequent Cray PVP systems CPUs by using the /dev/secded ioctl(2) interface to set bad parity into particular registers. The CPU must be in maintenance mode (you must have the maintenance mode switches on the maintenance panel on for all CPUs in which register parity errors are to be artificially generated).

The regpar command must be specified with the following options:

-r *register* Specifies the register in which the bad parity bit is to be stored. The *register* argument can be v, indicating the V register.

-p *parity* Sets the parity to even (0) or odd (1).

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	In a privileged administrator shell environment, allowed to write shell redirected output to any file.
sysadm	Shell redirected output is subject to security label restrictions.

If the PRIV_SU configuration option is enabled, the super user can write shell redirected output to any file.

SEE ALSO

ioctl(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

secded(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

remove – Removes temporary accounting files

SYNOPSIS

/usr/lib/acct/remove

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The remove command removes the temporary files /usr/adm/acct/sum/wtmp*, /usr/adm/acct/sum/pacct*, and /usr/adm/acct/nite/lock*. It is invoked by startup(8), which is the accounting start-up script, to clean up any files that may have been left over from a previous run.

FILES

/usr/adm/acct/nite/lock*	Controls simultaneous invocations of csarun(8) or runacct(8).
/usr/adm/acct/sum/wtmp.*MMDD*	A copy of wtmp file for *MMDD*; generated by standard UNIX System V accounting.
/usr/adm/acct/sum/pacct.*MMDD*	A concatenated version of all pacct files for *MMDD*; generated by standard UNIX System V accounting.

SEE ALSO

acct(8), acctsh(8), csa(8), startup(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

rep – Allows the operator to respond to action messages

SYNOPSIS

/usr/lib/msg/rep *msg_number* [*reply_string*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The rep command allows the operator to respond to action messages, such as tape mount messages. The *msg_number* operand indicates the number of the action message to which you want to reply. The *reply_string* operand indicates an optional string to be returned to the sender of the action message. If no string is specified, a null string is returned to the sender.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
repall	Replies are not subject to security label restrictions.

If this command is installed with a PAL, a user with one of the following active categories is allowed to perform the action shown:

Active Category	Action
system, secadm, sysadm, sysops	Allowed to use this command to reply to any messages.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command to reply to any message.

Enclose any special characters or strings containing special characters in quotation marks, so they are not misinterpreted.

SEE ALSO

infd(8), msgd(8), msgdaemon(8), msgdstop(8)

msgi(1), msgr(1), privtext(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

NAME

restore – Performs an incremental file system restore

SYNOPSIS

/etc/restore -i [-c] [-d] [-D *device_name*] [-f *file*] [-F *file_id*] [-g *device_group*] [-h] [-l *lt*] [-m] [-q] [-S *symfile*] [-T] [-v] [-V *vsn*] [-y]

/etc/restore -r [-c] [-d] [-D *device_name*] [-f *file*] [-F *file_id*] [-g *device_group*] [-h] [-l *lt*] [-m] [-q] [-S *symfile*] [-T] [-v] [-V *vsn*] [-y]

/etc/restore -M [-c] [-d] [-D *device_name*] [-f *file*] [-F *file_id*] [-g *device_group*] [-h] [-l *lt*] [-m] [-q] [-S *symfile*] [-T] [-v] [-V *vsn*] [-y]

/etc/restore -t [-c] [-d] [-D *device_name*] [-f *file*] [-F *file_id*] [-g *device_group*] [-h] [-l *lt*] [-m] [-q] [-S *symfile*] [-T] [-v] [-V *vsn*] [-y] *files*

/etc/restore -x [-c] [-d] [-D *device_name*] [-f *file*] [-F *file_id*] [-g *device_group*] [-h] [-l *lt*] [-m] [-q] [-S *symfile*] [-T] [-v] [-V *vsn*] [-y] *files*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The restore command reads data written by the dump(8) command. Its actions are controlled by the options selected. Other arguments to the command are file or directory names that specify the files to be restored.

The restore command restores the original file attributes when invoked by an authorized user (see NOTES for a description of an authorized user). Otherwise, the command restores the files using the attributes of the invoking user.

You must choose exactly one of the following options: -i, -M, -r, -t, or -x. Unless you specify the -h option, the appearance of a directory name refers to the files and (recursively) subdirectories of that directory.

The restore command accepts the following options:

-i This mode allows interactive restoration of files from a dump tape. After reading in the directory information from the tape, restore provides a shell-like interface that lets you move around the directory tree, selecting files to be extracted. The available commands follow; for those commands that require an argument, the default is the current directory:

Only the first character of a command is significant. An incorrect command of exit, for example, would yield the same results as extract.

ls [*directory*]
Lists the current or specified directory. Entries that are directories are appended with a slash (/). Entries that are marked for extraction are prefixed with an * (as in **file*). If you set the verbose option, the inode number of each entry also is listed.

cd [*directory*]
Changes the current working directory to the specified *directory*. Use commands of the form `cd` *name* rather than the form `cd ./`*name*.

pwd Prints the full path name of the current working directory.

add [*directory*] | [*filename*]
Adds the current directory or the specified *directory* or *filename* to the list of files to be extracted. If a directory is specified, it and all its descendents are added to the extraction list (unless you specify the `-h` option on the command line). Use `add` with the current directory as default, rather than commands of the form "`add .`". Files that are on the extraction list are prepended with a * symbol when they are listed by `ls`.

The `restore` command ensures that all files and directories that are added to the list of files to be extracted have the requisite directory structure in place on the file system being restored. If any directories are missing, they are immediately re-created.

delete [*directory*] | [*filename*]
Deletes the current directory or specified *directory* or *filename* from the list of files to be extracted. If a directory is specified, it and all its descendents are deleted from the extraction list (unless the `-h` option is specified on the command line). To extract most of the files from a directory expediently, add the directory to the extraction list and then delete those files that are not needed. Use `delete` with the current directory as default, rather than commands of the form "`delete .`".

The `restore` command does not remove anything from the actual file system being restored, but only from the list of files to be extracted. Any directories created by a previous `add` command are not removed from disk. Thus, the following commands ensure that directories `w`, `x`, and `y` exist after the commands are executed:

```
restore > add w/x/y/z
restore > delete w/x/y/z
```

extract Extracts all files that are on the extraction list from the dump tape.

verbose Enables verbose mode. When set, the `verbose` command causes the `ls` command to list the inode numbers of all entries. It also causes `restore` to print out information about each file as it is extracted.

help Summarizes the available commands.

quit The `restore` command immediately exits, even if the extraction list is not empty.

`-r` Rebuilds an entire file system by using a level 0 backup and a collection of one or more incrementals. The first `restore -r` invocation processes the level 0 backup. Each subsequent `restore -r` invocation processes the next incremental backup, as ordered by increasing incremental levels (`-t` option to `dump`). Data is passed between invocations of `restore -r` in a file named `restoresymtabl`. You should remove this file after the last invocation.

`-M` Sets ownership, times, modes (permissions), and security labels on directories. The `restore` command does not usually require this option because these labels are usually set at the end of a `restore` procedure. If a `restore` procedure was interrupted before completion, however, you can run `restore` again with the `-M` option to set them.

`-t` Lists the names of the specified files if they occur on the tape. If you omit the *file* argument, the root directory is listed, which results in the entire contents of the tape being listed, unless the `-h` option has been specified.

`-x` Extracts the specified *files* from the tape. If the specified file matches a directory whose contents had been written onto the tape, and the `-h` option is omitted, the directory is recursively extracted. The owner, modification time, and mode are restored (if possible). If the *file* argument is omitted, the root directory is extracted, which results in the entire contents of the tape being extracted, unless the `-h` option was specified.

`-c` Causes `restore` to use cartridge tapes rather than round tapes.

`-d` Causes `restore` to enter another type of verbose mode (debug).

`-D` *device_name*
Specifies *device_name* as a specific tape device to be used for the restore. This lets you know on which tape drive the tapes have been requested to mount. If the specified device is unavailable, another drive of the same device type is substituted. You cannot use the `-D` option with the `-f` option. Specific device names are installation-specified parameters.

`-f` *file*
Performs the restore from *file*, rather than tape. If the name of the file is `-`, `restore` reads from standard input. For example, you can use `dump`(8) and `restore` in a pipeline to move a file system by executing the following command:

```
cd /mnt
dump -t  0 -f - /dev/dsk/usr | restore -x -f -
```

`-F` *file_id*
Specifies the tape file ID for a labeled tape. By default, `restore` uses the volume serial number (VSN) of the first tape of the dump file.

`-g` *device_group*
Specifies the name of the group to which the requested device belongs. The default is `-g TAPE`. The `-c` flag is equivalent to `-g CART`. Other types of devices may be accessed (for example, `-g SILO`).

`-h` Causes `restore` to extract the actual directory, rather than the files that it references, which prevents hierarchical restoration of complete subtrees from the tape.

`-l` *lt* Specifies *lt* as the type of label to be read on the tape. Values for *label_type* can be one of the following:

`nl`	Nonlabeled tapes
`sl`	IBM standard labels
`al`	ANSI labels (default)

`-m` Causes `restore` to extract a file in your current directory. This is useful only if a few files are being extracted and you want to avoid regenerating the complete path name to the file.

You also must use the `-x` or `-i` option in addition to `-m`, and you must specify a file name. The `restore` command extracts the file and places it in your current working directory. The restored file has the name of its corresponding inode.

For example, the following command restores only *directory/filename*, rather than any subtree below *directory/filename*:

```
restore -x -m directory/filename
```

`-q` Suppresses all error messages for inode modification failures when `restore` executes with user authority. This "quiet" flag does not suppress messages for data failures.

`-S` *symfile*
Creates the restore symbol table file in *symfile*. If you omit *file*, `restore` leaves the symbol table in `restoresymtabl`.

`-T` Generates a list of all files that were modified while `dump`(8) was executing.

`-v` Causes `restore` to type the name of each file it treats, preceded by its file type. Usually, `restore` does its work silently.

`-V` *vsn*
Causes `restore` to use the specified *vsn* list as a list of volume serial numbers to be used for the restore. Each VSN is a set of 1 to 6 alphanumeric characters, separated by colons (:). You cannot use this option with the `-f` option (restore from a file). If you do not use the `-V` option, `restore` solicits you to type in a VSN list.

The following sample command line illustrates the usage:

```
restore -i -V root1:root2:root3
```

`-y` Prevents `restore` from asking whether it should abort the restore if it gets a tape error. It tries to skip over the bad tape block(s) and to continue as best it can.

To use other specific tape mount options that `restore` does not cover, you may reserve a tape drive by using `rsv`(1) and mount the tape by using `tpmnt`(1), as follows:

```
rsv
tpmnt -l al -v a1:a2 -M -p /tmp/tapedev
restore -x -f /tmp/tapedev filename
rls -a
```

NOTES

The `restore` command is slowed by synchronous writes which are used on the kernel for file system robustness. To make `restore` run faster, assign ldcache to the file system.

You must do a level 0 dump after a full restore. Because `restore` runs in user code, it cannot control inode allocation; thus, you must do a full dump to get the new inode numbering.

The security of the files on the dump tape depends on the ability to read the tape.

The interface between `restore` and the data in the inode contains several fields. The owner, group, mode, and timestamps fields also include account ID. The inode data includes compartments, flags, levels, privilege assignment lists (PALs), access control lists (ACLs), and categories.

Unauthorized users are not allowed to restore file attributes from the dump file. The first 100 such failures generate warning messages; however, the files are restored with the attributes of the invoking user. To suppress the warning messages, use the `-q` option.

If this command is installed with a PAL, a user who has one of the following active categories is allowed to perform the action shown:

Active Category	**Action**
`system`, `secadm`	Restores all attributes associated with the files in the dump.

If the `PRIV_SU` configuration option is enabled, the super user can restore all attributes associated with the files in the dump.

CAUTIONS

Usually, the `set owner/mode for '.'` question that appears in the `-i` or `-x` option should be answered `NO`. The answer to this question determines whether the `restore` command should try to set the owner/mode for the root directory of the file system being restored. Unless you are restoring the root directory of the file system, you do not have to do this. If you are restoring a few files to somewhere other than the file system that was dumped (for example, into `/tmp`) answering `YES` to this question could cause difficulties because it will set the owner and mode of the root directory of the file system you are restoring (for example, `/tmp`), to the owner and mode of the root directory of the file system that was dumped.

The `dump`(8) and `restore` commands share an internal inode number that is used to pass information between levels of an incremental restore. If an inode region is added to your file system, the internal number of a given inode may change, making an incremental restore impossible. If the super-block time stamps indicate that this may have happened, the `dump` command prints a warning. If the maximum inode number has changed, an incremental restore fails. You can use `restore -x` or `restore -i`.

When you do a full restore of a file system, disk quotas should not be running. This prevents overwriting an active quota control file.

MESSAGES

The `restore` command complains about bad option characters. The `restore` command complains if it gets a read error. If `y` was specified, or a user responds with a `y`, `restore` tries to continue the restore.

The `restore` command can list numerous consistency checks. Most checks are self-explanatory or can never happen. Common errors are as follows:

"*filename*`: not found on tape`"
: The specified file name was listed in the tape directory, but it was not found on the tape. This is caused by tape read errors while looking for the file, and from using a dump tape created on an active file system.

"`expected next file` *inumber*`, got` *inumber*"
: A file that was not listed in the directory showed up. This can occur when using a dump tape created on an active file system, or can be caused by trying to restore sockets and unlinked temporary files.

"`Tape read error while restoring` *filename*"
"`Tape read error while skipping over inode` *inumber*"
"`Tape read error while trying to resynchronize`"
: A tape read error has occurred. If a file name is specified, its contents are probably partially wrong. If an inode is being skipped, or the tape is trying to resynchronize, no extracted files were corrupted, although files may not be found on the tape.

"`resync restore, skipped` *num* `blocks`"
: After a tape read error, `restore` may have to resynchronize itself. This message lists the number of blocks that were skipped over.

EXAMPLES

Example 1: The following example does an interactive `restore` from IBM standard labeled cartridge tapes, using a specified volume serial number list:

```
restore -i -l sl -V a1:a2 -c
```

Example 2: The following example does a complete restore from ANSI labeled tapes, requesting specific device `tape00`:

```
restore -r -l al -D tape00
```

FILES

`/tmp/rstdir*`	File that contains directories on the tape
`/tmp/rstmode*`	Owner, mode, and time stamps for directories
`./restorsymtabl`	Information passed between incremental restores

SEE ALSO

`dump`(8), `mkfs`(8), `mount`(8), `rdump`(8), `rrestore`(8)

`rls`(1), `rsv`(1), `tpmnt`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

NAME

rexecd – Invokes the remote execution server

SYNOPSIS

/etc/rexecd [-S *tos*] [-U *umask*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The rexecd command is the server for the rexec(3C) routine. The server provides remote execution facilities with authentication that is based on user names and encrypted passwords.

The rexecd command invokes the centralized identification and authorization library routines to validate the user ID and password.

The rexecd command accepts the following options:

-S *tos* Directs rexecd to set the IP Type-of-Service (TOS) option on the connection to the value *tos*. This value can be a numeric TOS value or a symbolic TOS name found in the /etc/iptos file.

-U *umask* Directs rexecd to set the umask value of the process to the octal value *umask*.

Usually, rexecd is called by inetd(8) to wait for service requests at the port that is indicated in the exec service specification (see services(5)). When rexecd receives a service request, it initiates the following actions:

1. The server reads characters from the socket or transport endpoint up to a null byte (\0). The resultant string is interpreted as a base 10 ASCII number.
2. If the number received in step 1 is nonzero, rexecd interprets it as the port number of a secondary stream to be used for standard error. A second connection to the specified port on the client's machine is then created.
3. A null-terminated user name of 16 characters maximum is retrieved from the initial socket.
4. A null-terminated, encrypted password of 16 characters maximum is retrieved on the initial socket.
5. A null-terminated string that is to be passed to a shell as a command is retrieved on the initial socket. The length of the command is limited by the upper bound on the size of the system's argument list.
6. The user is validated at login time; if the authentication is successful, rexecd changes to the user's home directory and establishes the user and the user's group protections. If any of these steps fail, the connection is aborted and a diagnostic message is returned.

7. A null byte is returned on the connection associated with standard error, and the command line is passed to the normal user's login shell. The shell inherits the network connections that are established by `rexecd`.
8. The `rexecd` command performs a `setusrv`(2) system call with the user's default security parameters that are found in `/etc/udb`. Users cannot change any security attribute during the `rexecd` session.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

MESSAGES

All diagnostic messages are returned on the connection that is associated with standard error. Then any network connections are closed. An error is indicated by a leading byte with a value of 1 (0 is returned in step 7 after all steps prior to the command execution complete successfully).

`/bin/sh: ...`	The user's login shell could not be started.
`command too long`	The command line that is passed exceeds the size of the argument list (as configured into the system).
`Login incorrect`	Your name and the password may not match. This is a generic message for any of several login failure causes. No more information is given. The user name may not be valid or the password may be wrong. You may have been denied access by the WAL check. Your login may be locked, disabled, or invalidated for security violations.
`No remote directory`	The `chdir` command to the home directory failed.
`password too long`	The password consists of more than 16 characters.
`Try again`	A `fork` process by the server failed.
`username too long`	The name consists of more than 16 characters.

BUGS

No facility is available that allows all data exchanges to be encrypted.

SEE ALSO

`ia_failure`(3C), `ia_mlsuser`(3C), `ia_success`(3C), `ia_user`(3C), `rexec`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`services`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

ripquery – Queries routing information protocol (RIP) gateways

SYNOPSIS

/etc/ripquery [-1] [-2] [-a *authkey*] [-n] [-p] [-r] [-v] [-w *time*] *gateways*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The ripquery command requests all routes known by another routing information protocol (RIP) gateway by sending a RIP request or POLL command. The routing information in any routing packets returned is displayed numerically and symbolically. The ripquery command is intended to be used as a tool for debugging gateways, not for network management. Simple network management protocol (SNMP) is the preferred protocol for network management.

The ripquery command, by default, uses the RIP POLL command, which is an undocumented extension to the RIP specification supported by gated(8). The RIP POLL command is preferred over the RIP REQUEST command because it is not subject to split horizon and/or poisoned reverse effects. See RFC 1058 for more information.

The ripquery command accepts the following options:

-1 Sends the query as a version 1 packet. This is the default for UNICOS systems.

-2 Sends the query as a version 2 packet.

-a *authkey* Specifies the authentication password to use for queries. If specified, an authentication type of SIMPLE is used, otherwise the default is an authentication type of NONE. Authentication fields in incoming packets are displayed, but not validated. This option is valid only with version 2 packets.

-n Prevents the address of the responding host from being looked up to determine the symbolic name.

-p Uses the RIP POLL command to request information from the routing table. This is the default. If no response occurs to the RIP POLL command, the RIP REQUEST command is tried. The gated(8) daemon responds to a POLL command with all the routes learned through RIP.

`-r` Uses the RIP `REQUEST` command to request information from the gateway's routing table. Unlike the RIP `POLL` command, all gateways should support the RIP `REQUEST`. If no response occurs to the RIP `REQUEST` command, the RIP `POLL` command is tried. The `gated`(8) command responds to a `REQUEST` command with all the routes being announced through the specified interface. The `REQUEST` command provides information about the interface used to route packets back to the sender. This can be avoided by running a `ripquery` on the host being queried.

`-v` Displays version information about `ripquery` before querying the gateways.

`-w` *time* Specifies the time (in seconds) to wait for the initial response from a gateway. The default value is 5 seconds.

gateways Specifies the gateways to be queried.

SEE ALSO

`gated`(8), `gdc`(8)

`gated-config`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

RFC 1058 - *Routing Information Protocol*

COPYRIGHT INFORMATION

NAME

rlogind – Invokes the remote login server

SYNOPSIS

/etc/rlogind [-d] [-I *initid*] [-r *highpty-lowpty*] [-S *tos*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The rlogind command is the server for the rlogin(1B) command. The server provides a remote login facility with authentication that is based on privileged port numbers.

The rlogind command accepts the following options:

-d Specifies that each socket that is created by rlogind has debugging turned on (SO_DEBUG). With debugging enabled, the system traces all TCP packets sent and received on a socket. You can then use the trpt(8) command to interpret the packet traces.

-I *initid* Specifies the ID used by /etc/init. For more information, see the inittab(5) file format.

-r *highpty-lowpty*
Invokes the login program with *highpty-lowpty* default range of 0 through 100.

-S *tos* Directs rlogind to set the IP Type-of-Service (TOS) option for the connection to the value *tos*. This value can be a numeric TOS value or a symbolic TOS name found in the /etc/iptos file.

The inetd(8) daemon starts rlogind to receive service requests at the port indicated by the login service specification; see services(5). When a service request is received, the following protocol is initiated:

1. The server checks the client's source port. When the port is not in the range of 0 through 1023, the server aborts the connection.
2. The server checks the client's source address. When the address is associated with a host for which no corresponding entry exists in the host name database (see hosts(5)), the server aborts the connection.

After the source port and address are checked, rlogind allocates a pseudo terminal (see pty(4)) and manipulates file descriptors so that the slave half of the pseudo terminal becomes standard input, standard output, and standard error for a login process. The login process is an instance of the login(1) program invoked with the -r option. The login process then proceeds with the authentication process as described in rshd(8), but if automatic authentication fails, it reprompts the user to log in as indicated on a standard terminal line. The default range (*highpty-lowpty*) of pseudo terminals to use with rlogind is 0 through 100.

The parent of the login process manipulates the master side of the pseudo terminal, operating as an intermediary between the login process and the client instance of the `rlogin`(1B) command. In normal operation, the packet protocol that is described in `pty`(4) is invoked to provide `CONTROL-s` and `CONTROL-q` facilities and propagate interrupt signals to the remote programs. The login process propagates the client's terminal types, as found in the `$TERM` shell variable and terminal band.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

MESSAGES

All diagnostic messages are returned on the connection that is associated with `stderr`; then any network connections are closed. An error is indicated by a leading byte that has a value of 1.

`Host name for your address unknown`
: No entry in the host name database existed for the client's machine.

`Try again`
: A `fork` process by the server failed.

`/bin/sh: ...`
: The user's login shell could not be started.

BUGS

The authentication procedure assumes the integrity of each client machine and the connecting medium. The scheme relies on the concept of privileged ports (those numbered less than 1024, which is a nonstandard notion that applies only to 4.2 BSD, 4.3 BSD, and their descendants). This is not a good security practice, but it is useful in an open environment.

No facility is available that allows all data exchanges to be encrypted.

SEE ALSO

`ftpd`(8), `rshd`(8), `trpt`(8)

`privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`pty`(4), `hosts`(5), `services`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

rmt – Invokes the remote magnetic tape protocol module

SYNOPSIS

/etc/rmt

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The rmt command is a program that the remote dump and restore programs use in manipulating a magnetic tape drive through an interprocess communication connection (see rdump(8) and rrestore(8)). Usually, rmt is started with an rcmd(3C) or rexec(3C) call.

The rmt program accepts requests to manipulate the magnetic tapes, performs the commands, and then responds with a status indication. All responses are in ASCII, and they are in one of the following two forms (you must press <RETURN> after each response):

A*number* Response to a successful command. The *number* argument is an ASCII representation of a decimal number.

E*error-number* error-message
Response to unsuccessful commands. The *error-number* argument is one of the possible error numbers described in intro(2); *error-message* is the corresponding error string as printed from a call to perror(3C).

Commands

The rmt command accepts the following commands (a space is present between each token):

O *device flags* Opens the specified *device* by using the indicated *flags*. The *device* argument is a full path name, and *flags* is an ASCII representation of a decimal number suitable for passing to open(2). Because the rmt protocol was originally defined on Berkeley UNIX systems, you must specify the flag bits by using the Berkeley definitions; rmt converts them to the appropriate UNICOS equivalents. The flag bits and their equivalents are as follows:

Flag Bits	Equivalents
00001	O_WRONLY
00002	O_RDWR
00004	O_NDELAY
00010	O_APPEND
01000	O_CREAT

```
02000     O_TRUNC
04000     O_EXCL
```

If a device was already opened, it is closed before a new `open` is performed.

`C` *device* — Closes the currently open device and ignores the *device* specified.

`L` *whence offset* — Performs an `lseek`(2) operation by using the specified parameters. The response value is that returned from the `lseek`(2) operation.

`W` *count* — Writes data onto the open device. `rmt` reads *count* bytes from the connection, aborting if a premature end-of-file is encountered. The response value is that returned from the `write`(2) system call.

`R` *count* — Reads *count* bytes of data from the open device. If *count* exceeds the size of the data buffer (10 Kbytes), it is truncated to the data buffer size. If the read was successful, `rmt` then performs the requested `read`(2) and responds with A*count_read*; otherwise, an error in the standard format is returned. If the read is successful, the data read is sent.

`I` *operation count* — Tries to perform a `MTIOCOP ioctl`(2) call on the device by using the specified parameters. Because UNICOS tape drivers do not support this `ioctl` request, the only operation allowed is `MTFSF` (skip file forward), which is simulated in `rmt` by a series of `read` calls (see `read`(2)).

`S` — Returns the status of the open device, obtained with a `MTIOCGET ioctl` call. Because UNICOS tape drivers do not support this operation, an `ack` is sent, followed by a status buffer filled with binary 0's.

Any other command causes `rmt` to exit.

BUGS

Do not use the `rmt` command for remote file access.

SEE ALSO

`rdump`(8), `rrestore`(8)

`read`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

`rcmd`(3C), `rexec`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

NAME

`route` – Manipulates network routing tables manually

SYNOPSIS

`/etc/route` [`-d`] [`-n`] [`-q`] [`-t`] [`-v`] *commands* [`-net | -host`] *destination gateway* [*modifiers* [*arg*]]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `route` command manually manipulates the network routing tables.

A given route is uniquely identified by its destination address, netmask, and Type of Service (TOS) attributes. When deleting or changing a route, you must specify these values so the kernel can locate the route.

The `route` command accepts the following options:

`-d` Debug mode. Operates similarly to the `-t` option, but it also displays the `route` message that would have been sent to the kernel. Useful for debugging the `route` command.

`-n` Prevents attempts to print host and network names symbolically when reporting actions. This is useful, for example, when all name servers are down on your local network and you need a route before you can contact the name server.

`-q` Suppresses the output of the `flush` directive.

`-t` Test directives. The `route` command processes the directives, but it makes no changes to the kernel routing table. Useful for testing configuration changes.

`-v` Prints additional routing details.

commands

The `route` command accepts several directives. You can use one of the following:

`add` Adds a route.

`change` Changes the gateway or metrics of a route.

`delete` Deletes a route. If the `-s` *tos* or `-netmask` *mask* modifiers are specified when adding a route, they must also be specified when deleting a route.

`flush -inet`
Flushes the routing tables of all gateway entries or, by specifying an optional keyword modifier that describes the address family, flushes only those routes whose destinations are of the specified address family. Specify `-inet` for the Internet address family.

`get` Looks up and displays the route for a destination.

`monitor` Reports changes to the routing information base, routing lookup misses, or suspected network partitioning.

`-net | -host`

Forces *destination* to be interpreted as a network or a host address, respectively. If you omit these optional keywords and *destination* has a local address portion of `INADDR_ANY`, or *destination* is the symbolic name of a network, the route is assumed to be to a network; otherwise, it is assumed to be to a host. For example, `128.32` is interpreted as `-host 128.0.0.32`; `128.32.130` is interpreted as `-host 128.32.0.130`; `-net 128.32` is interpreted as `128.32.0.0`; and `-net 128.32.130` is interpreted as `128.32.130.0`. To eliminate ambiguity, use the `-net` or `-host` option.

destination gateway

destination is the destination host or network. *gateway* is the next-hop gateway to which packets should be addressed. Routes to a particular host are distinguished from those to a network by the interpretation of *destination*.

If you are adding an Internet route, *destination* is interpreted as a dot-notation host address. If this fails, the algorithm tries to interpret *destination* as a dot-notation network address. A dot-notation representation of a subnetted network address appears to be a host address and is interpreted as such. This interpretation affects the implicit network mask generation in that no network mask is generated for a host route (see the `-netmask` modifier). To ensure that a subnetted network address in dot-notation format is interpreted correctly, include the `-net` option on the command line.

If *destination* cannot be interpreted as a dot-notation address, the algorithm tries to resolve it by using `getnetbyname`(3C). If this fails, it uses `gethostbyname`(3C).

modifiers [*arg*]

The `route` command accepts several modifiers, some with arguments (*arg*). You can use any of the following:

`-admmtu` *mtu* Specifies the largest size packet that will be sent to the next hop gateway when this route is used; *mtu* is a numeric argument.

`-blackhole` Silently discards packets during updates

`-cloning` Generates a new link level route on use

`-gid` *gidlist* Limits the use of a route to particular groups of users. *gidlist* is of the form +|-*group1,group2,group3, ...*. The + indicates an inclusive list; that is, only the specified groups may use this routing entry. The - indicates an exclusive list that excludes the specified groups from using the routing entry. The groups may be either numeric or ASCII names as shown in the `/etc/group` file.

`-inet` Specifies that all subsequent addresses are in the Internet family.

`-interface` Specifies that the destination is directly reachable through an interface rather than through an intermediary system as a gateway. The gateway given is the address of this host on the common network, indicating the interface to be used for transmission.

`-link` Specifies that all subsequent addresses are specified as link-level addresses, and the names must be numeric specifications rather than symbolic names.

`-llinfo` Validly translates proto address to link address

`-netmask` *mask* The optional `-netmask` qualifier is intended to be used when manually adding subnet routes with different netmasks from that of the implied network interface. The required *mask* is an address to be interpreted as a network mask. You can use this modifier to override the implicit network mask generated in the `AF_INET` case by making sure this option follows the destination parameter.

If you specify this modifier when adding a route, you must also specify it when either deleting or changing the route.

`-noforward` Disables forwarding for this route.

`-nomtudisc` Disables path mtu discovery for this route.

`-nostatic` Pretend route added by kernel or daemon

`-proto1` Sets protocol specific routing flag #1

`-proto2` Sets protocol specific routing flag #2

`-reject` Emits an ICMP `unreachable` when matched

`-S` *arg* | `-service` *arg*
Requires an argument (symbolic name for option bits), such as `delay` or `throughput`, to mark the specified route with an IP Type-of-Service. See `iptos`(5) for other possible arguments. `-S` and `-service` are functionally identical.

If you specify this modifier when adding a route, you must also specify it when either deleting or changing the route.

`-static` Specifies a manually added route

`-tosmatch` Affects whether the specified routine can satisfy certain routing lookups. If you specify `-tosmatch` with a route, this route can be used to satisfy only those lookups that specify all of the type of service bits that are set in the route (using `-S` or `-service` modifiers). Lookups with more bits set can use this route if it is the best match available.

`-xresolve` Emits message on use (for external lookup)

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	May manipulate routing tables

If the `PRIV_SU` configuration option is enabled, the super user may manipulate routing tables.

Only an appropriately authorized user can use the `route` command to modify the routing tables.

The kernel installs point-to-point interface routes with a network mask of 255.255.255.255. To delete a point-to-point interface route, specify this network mask.

The `change` directive cannot be used on destinations with more than one route.

MESSAGES

`add [ host | network ] %s: gateway %s flags %x`
: The specified route that is being added to the tables. The values printed are from the routing table entry entered into the kernel routing table. If the gateway address used was not the primary address of the gateway (the first one returned by `gethostbyname`), the gateway address is printed numerically and symbolically.

`delete [ host | network ] %s: gateway %s flags %x`
: The specified route that is being deleted from the tables. The values printed are from the routing table entry entered into the kernel routing table. If the gateway address used was not the primary address of the gateway (the first one returned by `gethostbyname`), the gateway address is printed numerically and symbolically.

`%s %s done`
: When the `flush` directive is used, a message of this form indicates the destination and gateway address of each routing table entry that is deleted. The `-q` option disables this function.

`network is unreachable`
: An attempt to add a route failed because the gateway listed was not on a directly connected network. You must specify the next-hop gateway.

`not in table`
: A delete operation was tried for an entry that was not present in the tables.

`routing table overflow`
: An add operation was tried, but the system was low on resources and could not allocate memory to create the new entry.

`too many routes for destination`
: The `change` operation cannot be used for destinations that have multiple routes.

SEE ALSO

`gated`(8)

`privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`hosts`(5), `intro`(4), `networks`(5), `route`(4P) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

rpcbind – Maps universal addresses to an RPC program number

SYNOPSIS

rpcbind [-d] [-f *filename*] [-i] [-r] [-w]

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The rpcbind server converts Remote Prcedure Call (RPC) program numbers into universal addresses. It must be running on the host to be able to make RPC calls on a server on that machine.

When an RPC service is started, it tells rpcbind the address at which it is listening, and the RPC program numbers it is prepared to serve. When a client wants to make an RPC call to a program number, it first contacts rpcbind on the server machine to determine the address where the RPC requests should be sent.

The rpcbind server should be started before any other RPC service. Normally, port monitors start standard RPC servers, so rpcbind must be started before port monitors are invoked.

When rpcbind is started, it checks that certain name-to-address translation calls function correctly. If they fail, the network configuration databases may be corrupt. Since RPC services cannot function correctly in this situation, rpcbind reports the condition and terminates.

The -r and -f options allow RPC servers to reregister themselves with rpcbind. This capability is useful if, for some reason, it is necessary to restart rpcbind while the RPC servers continue. If rpcbind stopped executing and then was restarted without either of these options, any active RPC process would lose its RPC registration.

The rpcbind command accepts the following options:

-d Runs in debug mode. In this mode, rpcbind does not fork when it starts, prints additional information during operation, and aborts on certain errors. With this option, the name-to-address translation consistency checks are shown in detail.

-f *filename*
Directs rpcbind to send a SIGHUP signal to each of the process/user ID pairs found in the specified *filename* file, in addition to the standard RPC servers that receive the signal when the -r option is used. (Thus, the -f option implies the -r option.) The format of the information in *filename* is as follows:

```
program_1  [uid1]
program_2  [uid2]
     .
     .
     .
program_n  [uidn]
```

The `program` field is the name of the program (for example, `rpc_server`). The `uid` field is optional and specifies that `SIGHUP` be sent only to processes of the specified name running under the specified `uid`. If the `uid` field is blank, `SIGHUP` is sent to all processes of the specified name, regardless of their user ID. For example, assume that `rpcbind` has been invoked using the `-f rpcfile` option, and assume that `rpcfile` contains the following lines:

```
rpc_server       0
test_server      123
test_server      456
new_server
```

The following table indicates which processes will and will not be sent a `SIGHUP` signal:

Process name	UID	Sent `SIGHUP`?
`rpc_server`	`0`	Yes
`rpc_server`	`123`	No
`rpc_server`	`456`	No
`rpc_server`	`789`	No
`test_server`	`0`	No
`test_server`	`123`	Yes
`test_server`	`456`	Yes
`test_server`	`789`	No
`new_server`	`0`	Yes
`new_server`	`123`	Yes
`new_server`	`456`	Yes
`new_server`	`789`	Yes
`other_proc`	`0`	No
`other_proc`	`123`	No
`other_proc`	`456`	No
`other_proc`	`789`	No

The `-f` option is useful when site-specific RPC servers are running. However, to make this option useful, you must code the servers to catch the `SIGHUP` signal and to reregister with `rpcbind` when they receive it.

`-i` Allows nonsecure forwarding of requests to the `MOUNTPROG` program.

`-r` Specifies the command `restart` for standard RPC servers. This option causes `rpcbind` to send a `SIGHUP` signal to the following processes:

```
cnfsd
inetd
keyserv
lockd
mountd
nfsd
pcnfsd
statd
ypbind
yppasswdd
ypserv
```

Each of these processes catches the `SIGHUP` signal and reregisters with `rpcbind` upon receiving the signal. Any root process invoked with one of these process names is sent a `SIGHUP` signal if you invoke `rpcbind` using the `-r` option.

`-w` Performs a warm start. If `rpcbind` aborts or terminates on receipt of a `SIGINT` or `SIGTERM` signal, it writes the current list of registered services to the `/tmp/portmap.file` and `/tmp/rpcbind.file` files. When the `-w` option is specified, `rpcbind` searches for these files and starts operation with the registrations found in them. This allows `rpcbind` to resume operation without requiring all RPC services to be restarted.

NOTES

Terminating `rpcbind` using `SIGKILL` prevents the warm-start files from being written.

All RPC servers must be restarted if the following occurs: `rpcbind` crashes (or is killed using `SIGKILL`) and is unable to to write the warm-start files; `rpcbind` is started without the `-w` option after a graceful termination; or, `rpcbind` does not find the warm-start files.

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the action shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

SEE ALSO

`portmap`(8), `rpcinfo`(8)

`privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

rpcbstart – Starts rpcbind(8) or portmap(8)

SYNOPSIS

/etc/rpcbstart

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The rpcbstart command starts either an rpcbind(8) or portmap(8) server. The command uses the following criteria for determining which server to start:

1. If rpcbind(8) is on your system and your site has an Open Network Computing plus (ONC+™) license, rpcbind(8) is started.
2. Otherwise, portmap(8) is started.

SEE ALSO

portmap(8), rpcbind(8)

NAME

rpcinfo – Reports RPC information

SYNOPSIS

```
/etc/rpcinfo -p [host]
/etc/rpcinfo -u host program-number [version-number]
/etc/rpcinfo -t host program-number [version-number]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The rpcinfo command makes a Remote Procedure Call (RPC) to an RPC server and reports information according to the option you select.

The following options are available:

-p *host* Probes the portmapper on the specified *host* computer and prints a list of all registered RPC programs. If you do not specify the *host*, it defaults to the value returned by hostname(1).

-u *host program-number version-number*
Makes an RPC to procedure 0 of *program_number* by using the user datagram protocol (UDP) and reports whether a response was received.

-t *host program-number version-number*
Makes an RPC to procedure 0 of *program_number* by using Transmission Control Protocol (TCP) and reports whether a response was received.

SEE ALSO

portmap(8)

rpc(3C) in the *Remote Procedure Call (RPC) Reference Manual*, Cray Research publication SR–2089

NAME

rrestore – Restores a file system dump across a TCP/IP network

SYNOPSIS

/etc/rrestore -f *machine*:*device* [-i] [-d] [-h] [-m] [-q] [-v] [-y] [-S *symfile*] [-b *nbs*,*pbs*]

/etc/rrestore -f *machine*:*device* [-M] [-d] [-h] [-m] [-q] [-v] [-y] [-S *symfile*] [-b *nbs*,*pbs*]

/etc/rrestore -f *machine*:*device* [-r] [-d] [-h] [-m] [-q] [-v] [-y] [-S *symfile*] [-b *nbs*,*pbs*]

/etc/rrestore -f *machine*:*device* [-t] [-d] [-h] [-m] [-q] [-v] [-y] [-S *symfile*] [-b *nbs*,*pbs*] [*file1 file2* ...]

/etc/rrestore -f *machine*:*device* [-x] [-d] [-h] [-m] [-q] [-v] [-y] [-S *symfile*] [-b *nbs*,*pbs*] [*file1 file2* ...]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The rrestore command is a shell script that establishes a data path from file *device* at *machine* to a process running /etc/restore. The path comes from *machine*:*device* through a remote process running /etc/rmt across a socket connection into a local process running /etc/lrmt across a pipe and finally into a process running /etc/restore. *device* may be any file on the remote *machine*; it does not need to be a tape.

The options that are accepted are a combination of the parameters for /etc/restore and /etc/lrmt.

-f *machine*:*device*	A required parameter. This is passed to lrmt as the network destination.
-i, -r, -M, -t, -x	Passed without change to restore.
-d, -h, -m, -q, -v, -y	Passed without change to restore.
-S *symfile*	Passed without change to restore.
-b *nbs*,*pbs*	Passed without change to lrmt.
file1 file2 ...	Passed to restore.

The following restore options are not supported:

-c	Equivalent to -g CART.
-D *device_name*	Parameter for tpmnt.
-f *file*	Performs the restore from *file*, rather than round tape. If the name of the file is -, restore reads from lrmt through stdin.
-F *file_id*	Parameter for tpmnt.

`-g` *device_group*	Parameter for `tpmnt` and `rsv`
`-l` *lt*	Parameter for `tpmnt`.
`-V` *vsn*	Parameter for `tpmnt`.

The following `lrmt` options are not supported:

`-d` *dest*	This is set to `-d p`.
`-p` *pname*	A named pipe to be used for passing data on the local machine (rather than `stdin` or `stdout`).
`-c` *capacity*	The capacity, in megabytes, of a remote output tape device.

SEE ALSO

`dump`(8), `lrmt`(8), `orestore`(8), `rdump`(8), `restore`(8), `rmt`(8)

NAME

rshd – Invokes the remote shell server

SYNOPSIS

/etc/rshd [-S *tos*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The rshd command is the server for the rcmd(3C) routine and the remsh(1B) and rsh(1) commands. The server provides remote execution facilities with authentication based on privileged port numbers. inetd(8) invokes rshd.

The rshd command invokes the centralized identification and authorization library routines to validate the user ID and password.

The rshd command accepts the following option:

-S *tos* Directs rshd to set the IP Type-of-Service (TOS) option on the connection to the value *tos*. This value can be a numeric TOS value or a symbolic TOS name found in the /etc/iptos file.

When a service request is received, the following protocol is initiated:

1. The server checks the client's source port. If the port is not in the range of 0 through 1023, the server aborts the connection.
2. The server reads characters from the socket up to a null ( ) byte. The resultant string is interpreted as a base 10 ASCII number.
3. If the number received in step 1 is nonzero, it is interpreted as the port number of a secondary stream to be used for stderr. A second connection to the specified port on the client's machine is then created. The source port of the second connection also must be in the range of 0 through 1023.
4. The server checks the client's source address. If the address is associated with a host for which no corresponding entry exists in the host name database /etc/host (see hosts(5)), the server aborts the connection.
5. A null-terminated user name that consists of 16 characters maximum is retrieved on the initial socket. This user name is interpreted as a user identity to use on the server's machine.
6. A null-terminated user name that consists of 16 characters maximum is retrieved on the initial socket. This user name is interpreted as the user identity on the client's machine.
7. A null-terminated command to be passed to a shell is retrieved on the initial socket. The upper limit of the size of the system's argument list determines the length of the command.

8. The `rshd` command then validates the user according to the following steps. The remote user name is searched for in the password file and a `chdir`(2) call is made to move to the user's home directory.
   - If either the lookup or `chdir`(2) call fails, the connection is terminated.
   - If the user is not the super user (user ID 0), the `/etc/hosts.equiv` file is consulted for a list of hosts that are considered equivalent; see `hosts.equiv`(5).
   - If the client's host name is in this file, the authentication is considered successful. Some systems may be configured to require extra authentication. See the NOTES section.
   - If the lookup fails, or the user is the super user, the `.rhosts` file (see `rhosts`(5)) in the home directory of the remote user is checked for the machine name and identity of the user on the client's machine.
   - If this lookup fails, the connection is terminated. Also, the `.rhosts` file must have read/write permissions only for the user.
9. A null byte is returned on the connection that is associated with `stderr` and the command line is passed to the normal login shell of the user. The shell inherits the network connections established by `rshd`.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

The system configuration may require the `/etc/hosts.equiv` and `.rhosts` files each to contain a match for the remote host, and also require the remote user and local user names to match.

MESSAGES

All of the following diagnostic messages are returned on the connection associated with `stderr`; then any network connections are closed. An error is indicated by a leading byte with a value of 1 (0 is returned in step 9 after successful completion of all steps preceding the command execution).

`locuser too long` The name of the user on the client's machine is longer than 16 characters.

`remuser too long` The name of the user on the remote machine is longer than 16 characters.

`command too long` The command line passed exceeds the size of the argument list (as configured into the system).

`Hostname for your address unknown`
The host name database contains no entry for the client's machine.

`Login incorrect` The password file entry does not exist for the user name.

`No remote directory`	The `chdir`(2) call to the home directory failed.
`Permission denied`	The authentication procedure that is previously described failed.
`Can't make pipe`	The pipe that the `stderr` needs is not created.
`Try again`	The server's `fork` process server failed.
`/bin/sh: . . .`	The user's login shell could not be started.

BUGS

The authentication procedure used in `rshd` assumes the integrity of each client machine and the connecting medium. Also, the scheme relies on the concept of privileged ports (those numbered less than 1024, which is a nonstandard notion that applies only to 4.2 BSD, 4.3 BSD, and their descendants). This is not a good security practice, but it is useful in an open environment.

No facility that allows all data exchanges to be encrypted is available.

SEE ALSO

`inetd`(8)

`remsh`(1B) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`chdir`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

`ia_failure`(3C), `ia_mlsuser`(3C), `ia_success`(3C), `ia_user`(3C), `rcmd`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`hosts`(5), `hosts.equiv`(5), `rhosts`(5), `services`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

TCP/IP Network User's Guide, Cray Research publication SG–2009

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

`rstatd` – Performs kernel statistics server function

SYNOPSIS

`/etc/rstatd`

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `rstatd` command is a Remote Procedure Call (RPC) server that returns performance statistics obtained from the kernel to the client process that issued the RPC request. It is always registered as RPC program number 100,001. `inetd`(8) usually invokes the `rstatd` daemon.

FILES

`/etc/inetd.conf`	Contains configuration information and listens for incoming service requests
`/etc/rpc`	File that makes remote procedure calls

SEE ALSO

`inetd`(8)

`havedisk`(3R), `rstat`(3R) in the *Remote Procedure Call (RPC) Reference Manual*, Cray Research publication SR–2089

NAME

rsvportbm – Creates a bit map of well-known reserved ports

SYNOPSIS

/etc/rsvportbm [-c]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The rsvportbm command updates the port bit map in the kernel memory, based on the well-known reserved port numbers defined in the /etc/services file. Each time the command runs, it creates a new bit map, based on the port numbers in the /etc/services file, compares it with the one in the kernel memory, and replaces the one in the kernel memory if the two bit maps are different.

The rsvportbm command accepts the following option:

-c Specifies that a check will be made only if the bit map of well-known reserved ports from the /etc/services file differs from the one set in the kernel memory. If the bit maps are different, the following message is displayed:

```
The kernel memory port bitmap is not up-to-date.
```

Only an appropriately authorized user can use this command.

NOTES

This command should be run only at multiuser startup and each time the /etc/services file is modified.

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

FILES

/etc/services	File that contains a list of port numbers

SEE ALSO

services(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

runacct – Runs daily accounting

SYNOPSIS

/usr/lib/acct/runacct [*mmdd* [*state*]]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The runacct command is the main daily accounting shell procedure used by the standard UNIX accounting system. It is usually initiated by using cron(8). The runacct command processes connect, fee, disk, and process accounting files. It also prepares summary files for prdaily(8) or billing purposes.

The runacct command ensures that active accounting files or summary files are not damaged if errors occur. It records its progress by writing descriptive diagnostic messages into the active file. When an error is detected, a message is written to /dev/console, mail (see mail(1)) is sent to root and adm, and runacct terminates. To protect against reinvocation, runacct uses a series of lock files. The lock and lock1 files prevent simultaneous invocation, and lastdate prevents more than one invocation per day.

The runacct command accepts the following options:

mmdd specifies the month and day for which runacct will rerun the accounting.

state Designates where processing should begin. Overrides the contents of statefile.

runacct breaks its processing into separate, restartable *states* by using statefile to remember the last *state* completed. It accomplishes this by writing the *state* name into statefile; it then looks in statefile to see what it has done and to determine what to process next. The *states* are executed in the following order:

SETUP	Moves active accounting files into working files
WTMPFIX	Verifies integrity of /etc/wtmp file (see utmp(5)), correcting date changes if necessary
CONNECT1	Produces connect session records in ctmp.h format
CONNECT2	Converts ctmp.h format records into tacct.h format
PROCESS	Converts process accounting records into tacct.h format
MERGE	Merges the connect and process accounting records
FEES	Converts output of chargefee into tacct.h format and merges with connect and process accounting records
DISK	Merges disk accounting records with connect, process, and fee accounting records

MERGETACCT	Merges the daily total accounting records in daytacct with the summary total accounting records in /usr/adm/acct/sum/tacct
CMS	Produces command summaries
USEREXIT	User exit that executes site-dependent accounting program or script
CLEANUP	Cleans up temporary files and exits

To restart runacct after a failure, first check the active file for diagnostics; then fix up any corrupted data files such as pacct or wtmp. You must remove the lock files and lastdate file before runacct can be restarted. If runacct is being restarted, you must specify the *mmdd* argument. The entry point for processing is based on the contents of statefile; to override this, include the desired *state* on the command line.

NOTES

The mail recipients (root and adm) can be changed by modifying the MAIL_LIST parameter in /etc/config/acct_config.

BUGS

You should not restart runacct in the SETUP *state*, because doing so could overwrite valid data. Run SETUP manually and restart by using the following command line:

```
runacct mmdd WTMPFIX
```

If runacct failed in the PROCESS *state*, remove the last ptacct file because it is not complete.

If runacct terminates abnormally and leaves the lock files in place, the next execution of runacct will remove these locks, but it will also terminate abnormally.

EXAMPLES

Example 1: To start runacct, use the following command line:

```
nohup runacct 2> /usr/adm/acct/nite/fd2log &
```

Example 2: To restart runacct, use the following command line:

```
nohup runacct 0601 2>> /usr/adm/acct/nite/fd2log &
```

Example 3: To restart runacct at a specific *state*, use the following command line:

```
nohup runacct 0601 MERGE 2>> /usr/adm/acct/nite/fd2log &
```

FILES

`/etc/config/acct_config`	Accounting configuration file
`/etc/wtmp`	Connect time accounting data
`/usr/adm/acct/day/pacct*`	Process-accounting files switched using `turnacct`(8)
`/usr/adm/acct/nite/active`	Used by `runacct` to record progress and to issue error messages
`/usr/adm/acct/nite/daytacct`	Total accounting records for one day in `tacct.h` format
`/usr/adm/acct/nite/lock`	Used to control serial use of `runacct`
`/usr/adm/acct/nite/lock1`	Used to control serial use of `runacct`
`/usr/adm/acct/nite/lastdate`	Last day `runacct` executed in *date* `+%m%d` format
`/usr/adm/acct/nite/statefile`	A record of the current state during execution of `runacct`
`/usr/adm/acct/nite/ptacct*.`*mmdd*	Process-accounting information in `tacct.h` format
`/usr/lib/acct/run.user`	Executable called from the `runacct` script

SEE ALSO

`acct`(8), `acctcms`(8), `acctcon`(8), `acctmerg`(8), `acctprc`(8), `acctsh`(8), `cron`(8), `fwtmp`(8)

`acctcom`(1), `mail`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`acct`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

`acct`(5), `utmp`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

runsequence – Performs automatic test sequencing

SYNOPSIS

/etc/diag/scripts/runsequence *seqfile* [*cpu*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The runsequence program is used with the crontab(1) command to perform automatic test sequencing (scheduling and testing without operator intervention). Error messages are returned to specified users through email and can also be sent to the system operator. The CPU in which the error occurred is downed if the error is repeatable and the down-CPU feature is enabled.

When an error occurs and a CPU is downed, error messages alert service personnel, who then examine the error log to determine where the error occurred. The goal is to detect and isolate failures before a system or application failure occurs.

To initiate automatic test sequencing, do the following:

1. Set the shell variables in the runsequence shell script.
2. Customize (or create) the sequence files.
3. Create the input file for the crontab command.
4. Execute the crontab command.

Cray Research recommends that a site check with its customer service representative before running diagnostics on CRAY J90 and CRAY EL systems on a regular basis (for example, using runsequence).

crontab input file

The crontab input file is read by the /etc/cron utility to determine the tasks to be done and the times. /etc/cron is the mechanism for scheduling runsequence execution. The crontab input file contains the following information:

- Times at which the sequences are to be run.
- Calls to runsequence (Each call to runsequence must contain an appropriate sequence file name and, optionally, a CPU designator.)

The runsequence command accepts the following options:

seqfile Name of the file containing the sequence of diagnostic tests to be run, the test command options, and comments. Comments are the same as shell script comments; they start with a pound sign (#) and continue to the end of the line.

cpu CPU in which the tests are to be run. *cpu* can be a, b, c, and so on, up to the maximum number of CPUs, or a digit can be used to designate a CPU. If this option is specified, the diagnostic tests in the sequence file must be CPU tests. All of the log and core files are placed in subdirectories of the DIAGLOG directory. The name of the subdirectory corresponds to the CPU designator. The subdirectory is created if it does not already exist. For example, the log and core files from a CPU diagnostic test executing in CPU b will be placed in subdirectory b. If the *cpu* option is not specified, the log and core files will be placed in the other subdirectory in the DIAGLOG directory.

If the *cpu* option is specified and the DOWNCPU variable is set to ON, the CPU will be downed if *cpu* consists of a single character or digits and the error is repeatable.

The following is the suggested crontab input file for a system with four CPUs:

```
#
# cronfile: Crontab input file for running diagnostic testing
# sequences.  See CRONTAB(1) for a complete definition of the
# first 6 fields in the cronfile.
#
# The example below runs cpuseq once an hour in each cpu,
# every day of the year.
#
# findseq removes extraneous files once a day at midnight.
#
# dtseq or cfdtseq need to be set up at the site.
#
# This file should be installed as follows:
#     host$ crontab <filename>
#
# Note: To run olcfdt, set FSPATH and DT to appropriate values.
#1  * * * 0 FSPATH=/??? DT=???? $HOME/scripts/runsequence cfdtseq
# Note: To run oldt, set DEVNAME to the appropriate value.
#1  * * * 0 DEVNAME=devname $HOME/scripts/runsequence oldt
1  * * * * $HOME/scripts/runsequence cpuseq a
15 * * * * $HOME/scripts/runsequence cpuseq b
30 * * * * $HOME/scripts/runsequence cpuseq c
45 * * * * $HOME/scripts/runsequence cpuseq d
1  0 * * * $HOME/scripts/runsequence dailyseq a
15 0 * * * $HOME/scripts/runsequence dailyseq b
30 0 * * * $HOME/scripts/runsequence dailyseq c
45 0 * * * $HOME/scripts/runsequence dailyseq d
1  0 * * * FINDPATH=$HOME/log $HOME/scripts/findseq
```

The minute field is set to 1 to offset the diagnostic program execution to 1 minute after the hour. This allows scheduled system activities to be performed at the start of each hour.

Note that FSPATH, DT, and DEVNAME must be set to the appropriate values for your site when the crontab input file is first installed. For example, these values might be set as follows:

```
1  * * * 0 FSPATH=/tmp DT=DD49 $HOME/scripts/runsequence cfdtseq
```

The following is an optional addition to the suggested crontab input file for a system with four CPUs. Note that olsbt(8) will have large wall-clock completion times on heavily loaded systems. olsbt should only be run if problems with semaphore-related hardware are suspected. Regularly scheduled use of olsbt is not recommended.

```
# Optionally runs olsbt to test semaphore-related hardware
1  0 * * * $HOME/scripts/runsequence sbtseq a,b,c,d
15 0 * * * $HOME/scripts/runsequence sbtseq b,c,d,a
30 0 * * * $HOME/scripts/runsequence sbtseq c,d,a,b
45 0 * * * $HOME/scripts/runsequence sbtseq d,a,b,c
```

Sequence files

The sequence files contain a list of the diagnostic tests to be executed and their related command options. A recommended sequence file is supplied with the diagnostics. A site may determine that more or less testing is desirable, or circumstances may require that testing be varied. You can either customize the recommended sequence file for your site or create new sequence files. Place the sequence files in the directory specified by the DIAGSCRIPT shell variable.

The following are the recommended sequence files, which contain the maximum suggested diagnostic execution times:

cpuseq:

```
#
#  cpuseq: This sequence is a list of diagnostics to be
#  run by runsequence.  The cpu is selected in the crontab
#  file (cronfileX).
#
olcrit cputime 3 +getseed       Reads seed from olcrit.seed if available
olcsvc cputime 3 +getseed       Reads seed from olcsvc.seed if available
```

dailyseq:

```
#
#  dailyseq: This sequence is a list of diagnostics to be
#  run by runsequence once a day.  The cpu is selected in the crontab
#  file (cronfileX).
#
olcrit cputime 30 +getseed      Reads seed from olcrit.seed if available
olcsvc cputime 30 +getseed      Reads seed from olcsvc.seed if available
olibuf cputime 30 +getseed      Reads seed from olibuf.seed if available
olcfpt cputime 30 +getseed      Reads seed from olcfpt.seed if available
olcm   cputime 30 +getseed      Reads seed from olcm.seed if available
```

The following is an optional addition to the recommended sequence input file. It should be run only if semaphore-related problems are suspected.

`sbtseq`:

```
#
#  sbtseq: This sequence tests olsbt in all cpu's available.
#  The cpu is selected in the crontab file (cronfileX).
#
olsbt cputime 2 +getseed
```

The following is a sample `cfdtseq` sequence file:

`cfdtseq`:

```
#
#  cfdtseq: This sequence tests a mass storage device.
#  FSPATH and DT are defined in the cronfile.  If more than
#  one copy of olcfdt is run, it must have a unique value
#  specified for fn.
#  cfdtseq should be run from the installed cronfile as follows:
#
olcfdt maxp 20 fn $FSPATH/workfil. sz 250 dt $DT
rm -f $FSPATH/workfil
```

The following is a sample `dtseq` sequence file:

`dtseq`:

```
#
#  dtseq: This sequence tests a mass storage device.
#  DEVAME is defined in the crontab.
#
oldt -nF2 -m1 -d $DEVNAME
```

The following is a sample `findseq` sequence file:

`findseq`:

```
#
# findseq: This sequence finds (starting at FINDPATH) and removes
# small log files, stderr files and log files older than TOO OLD.
# FINDPATH is defined in cronfile (default:  FINDPATH=$HOME/log).
#

TOO_OLD=180     # Number of days to save log files

#
# Remove small log files or stderr files that runsequence created.
#
find $FINDPATH   -name '*.[0-9]*[0-9]' -size -300c ) -o -name

#
#Remove any log file that has not been touched recently
#
find $FINDPATH -name '*.[0-9]*[0-9]' -type f -atime +$TOO_OLD -exec rm -f {} ;
```

`runsequence` shell script

The `runsequence` shell script runs under the POSIX shell and executes a series of diagnostic tests by reading a file containing a list of the tests to be run. The tests should be run with the verbose option disabled (`-verbose`; the default), because the size of each diagnostic output file is used to determine whether the test has failed.

The shell script maintains the diagnostic output and sends messages to a specified list of users when an error is detected. You can set the following variables in the `runsequence` shell script:

`CONSOLE=ON|OFF`
: Enables (`ON`) or disables (`OFF`) the option that sends error messages to the system operator. The default is `ON`.

`DIAGBIN=`*path*
: Indicates the full path name of the directory in which the executable binary files of the diagnostic tests reside. If they reside in more than one directory, enter colons between directories. The following entry defines a single directory:

```
DIAGBIN=/ce/bin
```

The following entry defines several directories:

```
DIAGBIN=/ce/bin:$HOME/bin
```

The default is `/etc/diag:/etc:/ce/bin`.

DIAGLOG=*path*
: Indicates the full path name of the directory in which the log files are saved when a diagnostic test detects an error. The default is `/usr/spool/diag`.

DIAGSCRIPT=*path*
: Indicates the full path name of the directory in which the sequence files reside. You can specify only one full path name. The default is `/etc/diag/scripts`.

DOWNCPU=ON|OFF
: Enables (`ON`) or disables (`OFF`) the option that downs a failing CPU. For a CPU to be downed, `runsequence` must be given a single character or digits in the *cpu* option, and the error must be repeatable. The default is `OFF`.

MAILLIST="*user ...user*"
: Provides a list of users to be notified when a diagnostic test detects an error. Enter a space between user names and enclose the list in double quotation marks. It is recommended that the list contain more than one user name. The default is `LOGNAME`.

NICE=*n*
: Indicates the amount by which the diagnostic test's priority in the execution queue is to be lowered. *n* can be any integer in the range 1 through 19. If a value greater than 19 is entered, it is processed as if it were 19. If a value of 0 or less is entered, it has no effect. The default is 4.

RUNLOG=*logfile*
: Indicates the name of the log file containing information on the sequence being run and any errors detected. The log file resides in the `DIAGLOG` directory.

SAVECORE=ON|OFF
: Enables (`ON`) or disables (`OFF`) the option that renames and saves each core file generated. If `SAVECORE` is set to `OFF`, any new core file overwrites an existing one. The default is `OFF`.

The default values for the variables in the `runsequence` shell script are as follows:

```
CONSOLE=ON      # Set to ON to send messages to the operator
DIAGBIN=/etc/diag:/etc:/ce/bin      # Location of the executable diagnostic tests
DIAGLOG=/usr/spool/diag      # Location of the diagnostic log files
DIAGSCRIPT=/etc/diag/scripts      # Location of the diagnostic sequence lists
DOWNCPU=OFF     # Down failing CPU if repeatable error
RUNLOG=$DIAGLOG/runlog  # Program log
NICE=4  # Amount by which to lower the diagnostic
          test's priority
SAVECORE=OFF    # Existing core file will be overwritten
MAILIST="$LOGNAME"      # List of people to receive error messages
```

EXAMPLES

The following example shows a `runsequence` error message, which is sent to the system operator and to users specified by the `MAILLIST` variable.

```
Mon May 20 11:44:37 CST 1996: diagnostic found an ERROR: olcrit cputime 1
 +getseed cpu a
Dump in /usr/spool/diag/a/olcrit.111931
```

SEE ALSO

`crontab`(1) for information on the `crontab` file

Online Maintenance Tools Guide for Cray PVP Systems, Cray Research publication SD–1012. (This document contains information private to Cray Research, Inc. It can be distributed to non-CRI personnel only with approval of the appropriate Cray manager.)

NAME

rusersd – Performs RPC-based network user name server function

SYNOPSIS

/etc/rusersd

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The rusersd command is a Remote Procedure Call (RPC) based server that returns a list of users on a host to a requesting RPC-based client. It is always registered as RPC program number 100002. Typically, inetd(8) invokes the rusersd daemon.

FILES

/etc/inetd.conf	Contains configuration information and listens for incoming service requests
/etc/rpc	File that makes remote procedure calls

SEE ALSO

inetd(8)

rusers(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

rnusers(3R) in the *Remote Procedure Call (RPC) Reference Manual*, Cray Research publication SR–2089

NAME

rwall – Writes to all users on a network

SYNOPSIS

/etc/rwall *host...*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The rwall command reads a message from standard input until the end-of-file (EOF). It then sends this message, preceded by the following line, to all users who are logged in on the specified host machines:

```
Broadcast Message at [Time] ...
```

A machine can receive a message only if it is running rwalld(8), which is started typically by the daemon inetd(8).

The rwall command takes the following argument:

host... Specifies the host machines on which users must be logged in to receive the message that rwall sends.

BUGS

The time-out is fairly short so that messages can be sent to a large group of machines (some of which might be down) in a reasonable amount of time. Thus, the message might not get through to a heavily loaded machine.

FILES

/etc/inetd.conf Contains configuration information and listens for incoming service requests

SEE ALSO

inetd(8), rwalld(8), wall(8)

NAME

rwalld – Performs RPC-based network rwall(8) server function

SYNOPSIS

/etc/rwalld

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The rwalld command is a Remote Procedure Call (RPC) based server that processes rwall(8) requests. It is always registered as RPC program 100008. Usually, inetd(8) invokes the rwalld daemon.

FILES

/etc/inetd.conf	Contains configuration information and listens for incoming service requests
/etc/rpc	File that makes remote procedure calls

SEE ALSO

inetd(8), rwall(8), wall(8)

NAME

sam – Displays data about system activity

SYNOPSIS

/usr/bin/sam [*csam_options*]
/usr/bin/sam [*xsam_options*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The sam command is a generic interface to the csam(8) and xsam(8) utilities. Both utilities display data received from samdaemon(8). xsam is a graphic user interface based on the X Window system, while csam is a menu driven display based on curses, which is useful on dumb terminals. sam invokes either program using the following rules:

- If the -h *hostname* option has been specified, csam is started.
- If the DISPLAY variable is part of the user's shell environment, xsam is started.
- If the -display *value* option is specified, xsam is started.
- In any other case, csam is started.

sam accepts the following options:

csam_options Options accepted by the csam utility; refer to the csam(8) man page for more information.

xsam_options Options accepted by the xsam utility; refer to the xsam(8) man page for more information.

SEE ALSO

csam(8), samdaemon(8), xsam(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

samdaemon – System activity data daemon

SYNOPSIS

/usr/lib/sam/samdaemon [-?] [-d] [-f] [-l *logfile*] [-D]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The system activity monitor daemon, samdaemon, reads kernel data structures in memory as requested by the display clients, xsam(8) and csam(8). samdaemon can run on a Cray Research mainframe or on a connected OWS operator workstation. The daemon and clients communicate through Remote Procedure Call (RPC).

samdaemon contains a list of kernel data structures. When a client requests a structure, the daemon reads that structure and passes the information back to the client. At fixed intervals, the daemon re-reads the requested information, but passes it to a client only upon a subsequent client request.

The samdaemon command accepts the following options:

-? Prints a short synopsis of the command-line arguments.

-d Turns on debugging mode.

-f Forces a start up of the daemon, even when the RPC program number is still defined. samdaemon uses the RPC program number as a *lock* to ensure that only one daemon is running at any given time. If abnormal termination occurs, this *lock* is not cleared.

-l *logfile* Specifies an alternative log file; the default is /usr/lib/sam/samdaemon.log.

-D Prints a list of all built-in default values.

samdaemon provides access control to UNICOS data with the use of a validation file. If samdaemon finds a validation file (by default, /usr/lib/sam/samdaemon.val), it will return data only for validated users. The validation file consists of entries that take the following format:

```
host_name user_name
host_name *
host_name -user_name
```

The first format allows access for the specified user from the specified host. The second format allows access for all users from the specified host. The third format denies access for the specified user from the specified host.

On startup, samdaemon attempts to read the two configuration files, ./samdaemon.rc and /etc/config/samdaemon.rc. If the first file is found, the second is not read. If neither file is found, samdaemon uses default values.

The rules governing the format of the samdaemon.rc are as follows:

- Lines starting with # are treated as comments.
- All other lines have the format *token value*. The following *tokens* are valid:

AUTOGEN	Name of the configuration binary; the default is /usr/lib/sam/sama.
MAP_FILE	Name of the map file generated by AUTOGEN; the default is /usr/lib/sam/samdaemon.map.
VALID_FILE	Name of the validation file; the default is /usr/lib/sam/samdaemon.val.
NLIST_FILE	Name of the *name list* file; the default is /unicos. This token is valid only for daemons running on Cray Research systems.
HEARTBEAT	Time in seconds between structure updates; the default is 5 seconds.
MASTER	Host name of the Cray Research system to monitor; the default is cray. This token is valid only for daemons running on an operator workstation. Another samdaemon must be running on the target Cray Research system.
MASTER_MODE	If set, samdaemon does not respond to data requests. You can use this token to restrict the use of samdaemon on a Cray Research system and force users to use a samdaemon resident on an operator workstation. The default value is 0. This token is valid only for daemons running on the Cray Research system.
TABLE_HBEAT	Number of structure updates from the kernel without any client request; the default is 10.
PROC_HBEAT	Heartbeat of the *proc table* updates from the kernel. Because the *proc table* is big, updates can be configured to occur at a slower pace. The unit of PROC_HBEAT is HEARTBEAT. The default value is 2.
PROC_READS	Number of *proc table* entries samdaemon is to read. To limit the amount of memory needed to hold a copy of the kernel *proc table*, samdaemon reads only PROC_READS entries at one time. If PROC_READS is set to a negative number, all entries of the kernel *proc table* are read simultaneously. You should use caution when changing the value of PROC_READS; too small a value results in higher system overhead, while too great a value significantly increases the size of samdaemon. The default value is 50.
MEM_DEV	Name of the device for memory access. For the UNICOS operating system, the default value is /dev/mem. For Sun Workstations, the default value is /dev/fy0/xc0i0.
RPC_VERSION	RPC version number; the default is 1. This token may be used for debugging.

`MODEL_E` Used to configure nonstandard configurations between operator workstations and the IOS. When running on a Sun Workstation, the daemon assumes a connected IOS model E. This value can be used to override this default. This is only valid for daemons running on an operator workstation.

FILES

`/etc/config/samdaemon.rc`	System activity monitoring (`sam`) configuration file
`/usr/lib/sam/samdaemon.log`	System activity monitoring (`sam`) configuration file
`/usr/lib/sam/samdaemon.val`	System activity monitoring (`sam`) validation file

SEE ALSO

`csam`(8), `sam`(8), `xsam`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

sadc, sa1, sa2 – Generates system activity data on a routine basis

SYNOPSIS

/usr/lib/sa/sadc [*t n*] [*ofile*]

/usr/lib/sa/sa1 [*t n*]

/usr/lib/sa/sa2 [-a] [-b] [-c] [-d] [-e *time*] [-g] [-h] [-i *sec*] [-j] [-k] [-l] [-p] [-q] [-s *time*] [-t] [-u] [-v] [-w] [-x] [-y] [-z] [-A] [-B] [-H] [-M] [-T] [-W] [-X] [-Z]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

System activity data can be accessed upon special request of a user (see sar(1)); it can also be accessed automatically, on a routine basis, by invoking sa1, sa2, and sadc. The UNICOS operating system contains a number of counters that are incremented as various system actions occur. These counters include CPU usage, buffer usage, disk and tape I/O activity, tty device activity, switching and system call, file access, queue activity, and interprocess communication.

The sadc, sa1, and sa2 commands sample, save, and process this data.

The shell collector, sadc, samples system data *n* times every *t* seconds and writes in binary format to *ofile*, if specified, or to standard output. If you do not specify *t* and *n*, sadc writes a special record. Use this facility at system boot time to mark the time at which the counters restart from 0. The following /etc/rc entry writes the special record to the daily data file to mark the system restart:

```
/usr/lib/sa/sadc /usr/adm/sa/sa‘date +%d‘
```

The sa1 command, a variant of sadc, collects and stores data in the binary file /usr/adm/sa/sa*dd; dd* is the current day. The *t* and *n* operands specify that records are to be written *n* times every *t* seconds; the default is 1. The following entries in an appropriately authorized user's crontab file (see cron(8)) produce records every 20 minutes during working hours and hourly otherwise:

```
0 * * * 0,6 /usr/lib/sa/sa1
0 8-17 * * 1-5 /usr/lib/sa/sa1 1200 3
0 18-7 * * 1-5 /usr/lib/sa/sa1
```

The shell script sa2, a variant of sar(1), writes a daily report in the /usr/adm/sa/sar*dd* file. The options are explained in the sar(1) man page. The following crontab entry reports important activities hourly during the working day:

```
5 18 * * 1-5 /usr/lib/sa/sa2 -A
```

NOTES

Any `ncyls` values reported are valid only on Cray Research systems with an I/O subsystem model E (IOS-E).

`sa1` must be executed by an appropriately authorized user to get all possible data collected. `sadc` collects `ldcache` statistics from `/dev/dsk`, which only appropriately authorized users can access.

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to collect all possible data.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to collect all possible data.

FILES

`/usr/adm/sa/sa`*dd*	Daily data file
`/usr/adm/sa/sar`*dd*	Daily report file
`/usr/adm/sa.adrfl`	Address file

SEE ALSO

`cron`(8)

`privtext`(1), `sag`(1), `sar`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

sdaemon – Starts or kills daemon processes

SYNOPSIS

```
/etc/sdaemon -c [-f file] [-g daemongroup] daemon
/etc/sdaemon [-s] [-[qnv]] [-f file] [-h header] daemon [ ... ]
/etc/sdaemon [-s] [-[qnv]] [-f file] [-h header] -g daemongroup
/etc/sdaemon -k [-[qnv]] [-f file] [-h header] daemon [ ... ]
/etc/sdaemon -k [-[qnv]] [-f file] [-h header] -g daemongroup
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The sdaemon script starts or kills daemons (and other commands) according to information that is contained in a tabular configuration file. It is invoked at system startup by many start-up scripts (including /etc/rc, /etc/netstart, /etc/tcpstart (invoked by /etc/netstart), and /etc/nfsstart) to start the daemons necessary for normal system operation.

Each daemon that is to be started or killed can be specified on the command line according to a tag name that is given for the daemon in the configuration file. Alternatively, you can start or kill all of the daemons listed in the configuration file as members of the group name *daemongroup* by specifying the -g option. As part of its configuration, the sdaemon script with the -c option can return a simple status, indicating whether the daemon is in the configuration file and the path given points to an executable file.

When starting or killing daemons, sdaemon prints a header line, which is not terminated by a newline character, and it prints a tag name after the header for each daemon that starts successfully or is killed. The list terminates with a period and a newline character. The default header strings are Starting [*group*] daemon(s): and Stopping [*group*] daemon(s):; these strings are used for starting and killing daemons, respectively. Use the -q option to suppress this information.

The sdaemon script accepts the following options:

-c Determines whether the specified daemon is listed in the daemon's file and if its path name is executable. The sdaemon script returns a true value (that is, exits with a status of 0) if the specified daemon is listed as a tag in the configuration file and if the path name that is specified for the tag exists and is executable; otherwise, sdaemon returns a false value (that is, exits with a nonzero exit status). When used together with the -g option, sdaemon returns a true value only when the specified daemon is listed as a tag in the configuration file with a group membership of *group*, a *groupstart* value of YES for the tag, and a path name for the tag that exists and is executable.

-f *file* Uses *file* as the configuration file from which to fetch information about daemons to be started or killed.

`-g` *daemongroup*
: Starts or kills all daemons in the specified *daemongroup*. When starting a group of daemons, start each daemon as directed by its *groupstart* action in the configuration file (see the File Format subsection).

daemon
: Specifies the name of the daemon to be started or killed.

`-s`
: Starts the specified daemon, or all of the daemons that are listed in the configuration file as part of the *daemongroup* specified with the `-g` option. This is the default action.

`-q`
: Specifies quiet mode. Does not print either a header or the tag names of daemons that are started or killed.

`-n`
: Specifies no execution commands. The `sdaemon` script processes arguments and prints headers and tags, as usual, but does not actually execute the commands to start or kill the specified daemons. (This is very useful, in conjunction with the `-v` option, to double-check the actions that `sdaemon` will perform before actually performing them.)

`-v`
: Specifies verbose mode. The `sdaemon` script prints the command that it would actually execute to start or kill the specified daemons. When the commands that kill a daemon are printed, each command is followed by the actual process ID of the daemon, which is determined by the `sdaemon` and enclosed in brackets.

`-h` *header*
: Uses the string *header* as the header that is printed before the list of daemons that are started or killed. White space in a header string must be escaped from the shell.

`-k`
: Kills the specified daemon that is currently running, or all daemons that are listed in the configuration file as part of the *daemongroup* specified with the `-g` option, as directed by the kill information supplied in the configuration file for the appropriate daemon.

File Format

A configuration file for `sdaemon` consists of a series of lines with the following format:

daemongroup tag groupstart kill pathname arguments ...

The fields on each line have the following meanings:

daemongroup
: The name of a group to which this daemon belongs.

tag
: The tag name by which the individual daemon can be named as a command-line argument, and which `sdaemon` prints to indicate successful starting or killing of the daemon. Adjacent lines that have the same *tag* have only a single *tag* printed after the last line is executed.

groupstart The group start-up action for this daemon, which may be YES, NO, or ASK, or a run-level specific designation of the three. If the *groupstart* action is YES, the daemon is started when this daemon's group is specified with the -g flag. If the *groupstart* action is NO, the daemon is not started when this daemon's group is specified with the -g flag (however, you can start the daemon by not using the -g flag but explicitly specifying the tag on the command line). If the *groupstart* action is ASK, the operator of the command is prompted about whether to start this daemon when this daemon's group is specified with the -g flag. The prompt line is as follows:

```
Do you want to start tag (y/n) ?
```

To start the daemon, press <RETURN>, or any response that begins with the characters Y or y.

To tailor a group start-up action for a daemon to specific system run levels, use a run-level *groupstart* designation that has the following format:

runlevels=*action*

In the preceding example, *runlevels* is a concatenation of the characters for which *action* is the appropriate group start-up action. Alternatively, you can use the asterisk character (*) as a synonym for all run levels; omitting *runlevels*= from the beginning of the string can be used as a synonym for all run levels.

To specify different group start-up actions for different run levels for the same daemon specification, use colon characters to separate multiple run-level *groupstart* designations. The sdaemon script interprets these multiple designations from left to right and uses the first action specified for the current run level.

As examples, sdaemon interprets the following *groupstart* actions as follows:

Action	Interpretation
NO	Do not start this daemon as part of the group.
YES	Always start this daemon as part of the group.
ASK	Prompt for whether to start this daemon as part of the group.
2=YES	Start this daemon as part of its group if the system is at run level 2. (Implicitly, the user is prompted as to whether this daemon will be started, if the system is at any other run level.)
2346=YES	Start this daemon as part of its group when the system is at run level 2, 3, 4, or 6.
2=ASK:*=YES	Prompt for whether or not to start this daemon as part of its group when the system is at run level 2, and always start this daemon when the system is at any other run level.

`2=YES:1=NO:ASK`
: Always start this daemon as part of its group when the system is at run level 2; do not start this daemon when the system is at run level 1, and prompt for whether to start this daemon when the system is at any other run level. (The string `ASK` implicitly functions as a synonym for `*=ASK`.)

(See `init`(8) for further description of system run levels.)

kill
: The following information describes how to kill a currently running copy of the daemon:

- If *kill* is one hyphen (`-`), `sdaemon` must not attempt to find or kill a currently running copy of this daemon.
- If *kill* is the name of an executable file, `sdaemon` executes the file, expecting the program or script that is executed to exit with a status of 0 after a successful kill of the running daemon.
- If *kill* is the name of a nonexecutable file, `sdaemon` assumes that it is a text file that contains the process identification number (PID) of a currently running copy of the daemon, to which `sdaemon` sends a `SIGTERM`.
- If *kill* is one asterisk (`*`), `sdaemon` sends a `SIGTERM` to all running processes (as determined by `ps`(1)) that have a name equal to the file name component (the base name) of *pathname*.
- Otherwise, `sdaemon` sends a `SIGTERM` to all running processes (as determined by `ps`(1)) that have a name of *kill*.

pathname
: The path name of the file to execute to start the daemon.

arguments
: The command-line arguments to pass to the executable file when the daemon is started.

An initial # character at the beginning of a line indicates that the line is a comment.

MESSAGES

If the daemon with the tag name *tag* is not listed in the configuration file *file*, the following message is issued:

```
sdaemon:  daemon 'tag' not found in 'file'
```

BUGS

The `sdaemon` script has no flow-control ability in its configuration file; that is, the successful execution of an earlier daemon or command must not be a condition of executing any given daemon or command.

Because `sdaemon` relies on output from `ps`(1) and the base name of commands to determine the process IDs of many running daemons, the possibility exists that it can kill unintended programs if they share the same base name as the daemon being killed.

When using the `-g` option to start more than one daemon, if a daemon reads from standard input when it is started, it clears the pipe used internally in the `sdaemon` script. The result is that no daemons are started after that, and `sdaemon` exits normally.

The daemons started by `sdaemon` may inherit some characteristics of the user evironment that is running `sdaemon`. This may influence how the daemon executes. For example, if the user's operator has restricted CPU or memory limits when starting a daemon, that daemon may fail when those limits are reached.

FILES

`/etc/config/daemons`	Default configuration file

SEE ALSO

`init`(8)

`kill`(1), `ps`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`signal`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

NAME

sdc, sdcx – Collects system data

SYNOPSIS

/usr/bin/sdc [-c *path*] [-d *debug_file*] [-h *host*[:*port*]] [-i *interval*] [-n *sample_count*] [-B | -S] [-R *request_file*] [*output_file*]

sdcx [-i *interval*] [-n *sample_count*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The sdc command generates, compiles, and executes a C program named sdcx, which samples system activity data from the kernel. The sdcx source code cannot be accessed, but the executable can be retained by using the -c option.

When the -c, -B, or -S option is specified, sdc writes only a header record to *output_file*. When these options are not used, data records are also written. If *output_file* is not specified, output is written to a file named dcf in the current directory.

Each time sdc is invoked, it generates and compiles code for sdcx, and writes a header record. sdc may then start a new session and use the fork(2) system call to create a child process, which executes sdcx. The parent sdc process exits, and the child process collects *sample_count* data samples at a rate of one sample each *interval*.

The sdcx command samples the system activity counters and writes data records. sdcx does not generate header records. To use sdcx, first save the sdcx binary, and write a header record to the output file by invoking sdc with the -c option. Next, sample the system data by executing sdcx. You must append the sdcx output, which is written to stdout, to the sdc output file. Failure to do so can cause the tsar(8) command to abort or can generate erroneous reports.

UNICOS Resource Administration, Cray Research publication SG–2302, contains the list of data items that sdc and sdcx collect. The sdc -R option allows you to sample a subset of the data items.

The tsar(8) command formats the output file into an ASCII report.

The sdc command accepts the following options:

-c *path* Specifies the path name of the directory where sdcx, the data collector executable, is placed. By default, the data collector is placed in $TMPDIR and is not accessible after the parent sdc process exits. This option writes a header record to the output file and does not gather data samples.

-d *debug_file* Specifies the name of the file where debug information is written.

`-h` *host*[`:`*port*] Specifies the name of the remote *host* where the data sampling is to occur and the *port* which is used. The default *port* is 0. By default, data sampling is done on the host on which `sdc` is invoked.

`-i` *interval* Specifies the data sampling interval. *interval* is of the form *xx*`h`*yy*`m`*zz*`s` (*xx* hours, *yy* minutes, *zz* seconds) or *zz* seconds. By default, data is sampled every 5 minutes.

`-n` *sample_count* Specifies the number of samples to be taken. By default, `sdc` collects 5 data samples.

`-B` Specifies that a system boot header record be written to the output file. No data is collected with this option. The `-B` and `-S` options are mutually exclusive.

`-S` Specifies that a system shutdown header record be written to the output file. No data is collected with this option. The `-B` and `-S` options are mutually exclusive.

`-R` *request_file* Specifies the name of the file that contains a list of data items to be sampled. By default, all of the data items sampled. A complete list of data items that `sdc` samples is found in *UNICOS Resource Administration*, Cray Research publication SG–2302.

The `sdcx` command accepts the following options:

`-i` *interval* Specifies the data sampling interval. `interval` may be in the form *xx*`h`*yy*`m`*zz*`s` (*xx* hours, *yy* minutes, *zz* seconds) or *zz* seconds. By default, `sdcx` uses the interval specified to `sdc`.

`-n` *sample_count* Specifies the number of intervals for which samples are to be taken. By default, `sdcx` uses the number of samples specified to `sdc`.

NOTES

The `sdc` and `sdcx` commands can be executed only by a user who has permission to read the `/unicos` and `/dev/kmem` files.

If you will not reconfigure the system during the sampling interval, use `sdcx` instead of `sdc` to collect the data samples. `sdcx` takes less time to execute than `sdc`, because it does not generate and compile `C` code. `sdc` always generates and compiles `sdcx`; thus, it may take significant time to execute on a busy system.

The `sdc` command creates a new session and forks a child process to perform the actual data collection and does not wait for the child process to terminate. Thus, the child process can continue to write data to the data file even though the parent process has exited. The name of the child process is generated by the `tmpnam`(3C) routine.

EXAMPLES

Example 1: The following example gathers data only for the items listed in the file `disk_data`. Twelve samples are collected at a rate of one sample every 5 minutes. The data is written to the `dcf` file.

```
$ sdc -R disk_data -n 12 -i 5m dcf
```

Example 2: In the following example, `sdc` writes a header record to the `dcf` file and saves the `sdcx` binary in the `/usr/adm/tsar` directory. `sdcx` then is invoked to gather 12 data samples. The samples are collected at a rate of one sample every 5 minutes (300 seconds). Last, the data is appended to the `dcf` file.

```
$ sdc -c /usr/adm/tsar dcf
$ /usr/adm/tsar/sdcx -i 300s -n 12 >> dcf &
```

SEE ALSO

`tsar`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

sdconf – Controls the state of a disk drive

SYNOPSIS

/etc/sdconf [*path control* ...]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The sdconf command controls the state of a disk drive. It is based on the pddconf(8) command, but allows all physical disk devices (xdd, qdd, hdd, and pdd) to be controlled. The pddconf(8) only works with pdd devices.

If no arguments are given on the command line, a summary display of the current state of all drives is shown.

The sdconf command accepts the following command line arguments:

path The I/O path of the device to be controlled. For pdd, qdd, and xdd devices, *path* takes the form *iopath.unit*. For network devices, *path* takes the form *iopath.unit.ifield*. pdd and qdd devices are controlled through per-device control nodes that reside in /dev/ddd/*path*. xdd devices are controlled through the single control node /dev/ddd/xdiag (IOS model E systems) or /dev/xdd/xdiag (GigaRing-based systems). hdd devices are controlled through the single control node /dev/ddd/hdd (IOS model E systems). Path may also be the token all; see the qsort control.

control The control operation to perform on *path*, followed by per-control arguments. The following controls are supported:

- rw — Sets the physical device mode to read/write.
- ronly — Sets the physical device mode to read-only. Read requests are permitted, writes are returned with an *errno* of EIO. Any mounted file system slice(s) residing on the device are marked as not available for allocation.
- noall — Sets the physical device mode to nonallocatable. Any mounted file system slices residing on the device are marked as not available for allocation.
- up — Sets the physical device state to up. Any previous setting of mode is still in affect.
- down — Sets the physical device state to down and terminates all queued I/O requests with errors. Any mounted file system slices residing on the device are marked as not available for allocation.

`spinup` Spinup a disk drive. Valid only for DD-60 disk drives, spinup is also done during device open.

`spindown` Spindown a disk drive. Valid only for DD-60 disk drives.

`pripath` Change disk device to primary path.

`altpath` Change disk device to alternate path if available.

`reset` Reset disk statistics.

`disable` *spindle*
Disable a given spindle of a disk array. A spindle number 0-4 must be given.

`qsort` Turn on the disk queue sorting algorithm in the disk driver for the specified device; the device flag is on by default. Specifying `all` as the path turns on a global flag that enables queue sorting for all disks in the system for which the disk queue sorting algorithm is enabled; this global flag is off by default.

`noqsort` Turn off the disk queue sorting algorithm for the indicated device; the device flag is on by default. Specifying all as the device turns off a global flag, disabling queue sorting for all disks in a system; this global flag is off by default.

`autoswitch on|off`
Enable/disable autopath switching during error recovery. The system will not switch to the alternate path if the switch is disabled. The device must be configured with an alternatepath path to work.

`racerron` Routes Recovered Disk Error messages to be printed on the console. This is the default.

`racerroff` Stops routing Recovered Disk Error messages to the console.

Not all controls are applicable for all device types. The following table lists valid device type controls:

Control	xdd	pdd	hdd	qdd
rw	Yes	Yes	Yes	Yes
ronly	Yes	Yes	Yes	Yes
noall	Yes	Yes	Yes	Yes
up	Yes	Yes	Yes	Yes
down	Yes	Yes	Yes	Yes
spinup	No	Yes	No	Yes
spindown	No	Yes	No	Yes
pripath	Yes	Yes	No	Yes
altpath	Yes	Yes	No	Yes
reset	No	Yes	No	Yes
disable	No	Yes	No	Yes

Control	xdd	pdd	hdd	qdd
qsort	No	Yes	No	Yes
noqsort	No	Yes	No	Yes
autoswitch	No	Yes	No	Yes
racerron	No	Yes	No	Yes
racerroff	No	Yes	No	Yes

If no path or control arguments are given, the following display is generated:

```
iopth.un[.ifld] type  drvr state mode  altp  flags  wstrm   rstrm  qon  q'ed
--------------- ----- ---- ----- ---- ------ ----- ------- ------- --- -------
  0230.0        DD61  pdd  up    rw    ----  01100 ------- ------- yes       0
  0230.1        DD61  pdd  up    rw    ----  01100 ------- ------- yes     169
  0232.0        DD60  pdd  up    rw    ----  01000 ------- ------- yes       0
  0232.1        DD62  pdd  up    rw    ----  01000 ------- ------- yes       0
  0234.2        DD61  pdd  up    rw    ----  01100 ------- ------- yes       0
  0234.3        DD62  pdd  up    rw    ----  01000 ------- ------- yes       0
  0236.3        DD61  pdd  up    rw    ----  01000 ------- ------- yes       0
  0236.4        DD61  pdd  up    rw    ----  01000 ------- ------- yes       0
  0236.5        DD61  pdd  up    rw    ----  01100 ------- ------- yes       0
  0236.6        DD61  pdd  up    rw    ----  01000 ------- ------- yes       0
```

The following are definitions of the headings in the preceding table:

`iopth.un[.ifld]`
: Device I/O path having the format *iopath.unit* for `pdd`, `qdd`, and `xdd` devices, *iopath.unit.ifield* for network devices.

`type`
: Device type.

`drvr`
: Driver type.

`state`
: Device state - driver dependent.

`mode`
: Device mode (read-write (`rw`), read-only (`ro`), or no allocate (`na`)).

`altp`
: Device alternate path if available.

`flags`
: Device flags - driver dependent.

`wstrm`
: Device write stream control flags (for disk arrays).

`rstrm`
: Device read stream control flags (for disk arrays).

`qon`
: Indicates that sort is enabled for a particuliar device.

`q'ed`
: Total number of requests that have been sorted since the system was booted, or since device statistics were reset.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

SEE ALSO

`pddconf`(8), `sdstat`(8)

`ioctl`(2) in *UNICOS/mk System Calls Reference Manual*, Cray Research publication SR–2612

NAME

sdstat – Displays information about disk device I/O

SYNOPSIS

/etc/sdstat

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The sdstat command displays information about disk device activity. It is based on the pddstat(8) command, but it displays information about all disk types (pdd, hdd, qdd, and xdd).

Currently, the sdstat output is limited to a summary display having the following format:

```
                                                    Sectors Moved    Errors
iopth.un[.ifld] type  drvr state mode req  reads    writes   rec  unr
--------------- ----- ---- ----- ---- --- -------- -------- ---- ----
  0130.20.13    HD32  hdd  up    rw     0        0        0    0    0
  0130.40.25    HD32  hdd  up    rw     0      512        0    0    0
  0230.0        DD60  pdd  up    rw     0       26        2    0    0
  0230.3        DD62  pdd  up    rw     0     4468     4377    0    0
  0230.4        DD60  pdd  up    rw     0       23        2    0    0
  0230.5        DD60  pdd  up    rw     0        4        0    0    0
  0232.0        DD60  pdd  up    rw     0    18587     2732    0    0
  0232.1        DD60  pdd  up    rw     0      868        4    0    0
  0232.2        DD60  pdd  up    rw     0     1054        2    0    0
  0232.3        DD62  pdd  up    rw     0      122        2    0    0
  0234.0        DD60  pdd  up    rw     0      320       38    0    0
  0234.1        DD60  pdd  up    rw     0     1157     1099    0    0
  0234.2        DD60  pdd  up    rw     0     2052      115    0    0
  0234.3        DD62  pdd  up    rw     0       92        2    0    0
  0236.2        DD62  pdd  up    rw     0     2115        1    0    0
  0236.3        DD61  pdd  up    rw     0     4343        0    0    0
  0236.4        DD60  pdd  up    rw     0     4737      282    0    0
  0236.5        DD60  pdd  up    rw     0     6806     1936    0    0
```

The following are definitions of the headings in the preceding table:

iopth.un[.ifld]
: Device I/O path having the format *iopath.unit* for pdd, qdd, and xdd devices, *iopath.unit.ifield* for network devices. The path will be suffixed by a if the device has an alternate path, and the alternate path is active.

type
: Device type.

drvr	Driver type.
state	Device state - driver dependent.
mode	Device mode (read-write (rw), read-only (ro), or no allocate (na)).
req	Number of currently outstanding requests.
reads	Number of sectors read (sector size is device type dependent).
writes	Number of sectors written.
rec	Number of recovered read and/or write errors.
unr	Number of unrecovered read and/or write errors.

SEE ALSO

sdconf(8), pddstat(8)

ioctl(2) in *UNICOS/mk System Calls Reference Manual*, Cray Research publication SR–2612

NAME

secded – Memory error correction interface SECDED hardware

SYNOPSIS

```
/etc/secded [-u uncmax] [-n cormax] [-w mint] [-i disint]
/etc/secded [-l umelife] [-m umemax] [-t umedown]
/etc/secded -a address -d data
/etc/secded -a address -c check
/etc/secded -a address -c check -d data
```

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The secded command allows any memory location to be written with any pattern of bits by using the SECDED maintenance hardware. The command also provides access to the memory error processing control parameters and control of CPU downing due to memory errors.

The secded command accepts the following options:

-a *address* Specifies absolute octal address, *address*, of the word of memory to be altered. If you specify this option, you must also specify the -d or -c option.

-d *data* Specifies data bits to be toggled in the word of memory (interpreted as an octal mask). For example, to toggle bit 2^4, use the following command line:

```
/etc/secded -a address -d 20
```

Either the -d option or the -c option must be specified if the -a option is specified.

-c *check* Specifies check bits to be toggled in the word of memory (interpreted as an octal mask). For example, to toggle bit 2^4, use the following command line:

```
/etc/secded -a address -c 20
```

The mask is limited to the 8 check bits that exist. Either -d or -c must be specified if the -a option is specified.

-u *uncmax* Specifies the number of uncorrectable memory errors that the system will allow before forcing a panic. The default is 64.

-n *cormax* Specifies the number of single-bit errors that can occur in *mint* seconds with no intervals of longer than *mint*/*cormax* seconds without an error before single-bit errors are turned off, for all user processes, for *disint* seconds. The default for *cormax* is 16. If -1 is specified, single-bit interrupts are always disabled.

`-w` *mint* Specifies the number of seconds that *cormax* single-bit errors can occur in before disabling single-bit errors. The default is 5.

`-i` *disint* Specifies the number of seconds to keep memory errors disabled. The default is 300. If `-1` is specified, single-bit errors are permanently disabled.

`-l` *umelife* Specifies the lifetime (in seconds) of uncorrectable memory errors. The default is 86400 (24 hours). This is related only to the downing of CPUs due to uncorrectable memory errors and not to the `-u` option.

`-m` *umemax* Specifies the maximum number of uncorrectable memory errors before a CPU is downed. The default is 0. A value of zero disables downing of CPUs due to uncorrectable memory errors.

`-t` *umedown* Specifies the time (in seconds) that a CPU will remain down after uncorrectable memory errors. The default is 0. A value of zero means that the CPU will remain down until restarted by the /etc/cpu command.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	**Action**
`system`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

BUGS

On Cray PVP systems, error maintenance switches must be on for the CPUs used to execute the `secded` command. Software cannot determine whether or not error maintenance is enabled.

Because of its ability to write memory errors anywhere in physical memory, the `secded` command should be used with caution. For instance, if a double-bit or multibit memory error is written in memory and the UNICOS kernel reads that word next (as opposed to a user process), the kernel panics. Generally, use of the SECDED maintenance feature should be done with the mainframe in single-user mode.

FILES

`/dev/secded` SECDED maintenance special device

SEE ALSO

`cpu`(8), `errdemon`(8), `errpt`(8)

`secded`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

sendmail, newaliases, mailq – Sends mail over the Internet

SYNOPSIS

/usr/lib/sendmail [-B*type*] [-ba] [-bd] [-bi] [-bm] [-bp] [-bs] [-bt] [-bv] [-C*file*] [-d*X*] [-F*fullname*] [-f*name*] [-h*N*] [-n] [-o*x value*] [-p*protocol*] [-q[*time*]] [-qI*substr*] [-qR*substr*] [-qS*substr*] [-r*name*] [-t] [-v] [-X*logfile*] *address* ...

/usr/bin/newaliases

/usr/bin/mailq [-v]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The sendmail command sends a message to one or more recipients, routing the message over whatever networks are necessary. It does internetwork forwarding, as necessary, to deliver the message to the correct place.

The sendmail command is not intended as a user interface routine. Other programs provide user-friendly front ends; sendmail is used only to deliver preformatted messages.

With no options specified, sendmail reads its standard input up to an end-of-file or a CONTROL-d or a line with a single dot (.) and sends a copy of the message found there to all of the addresses listed. It determines the network(s) to use based upon the syntax and contents of the addresses.

The sendmail command looks up local addresses in a file and determines aliases. To prevent aliases from being determined, precede the address with a backslash. Usually, the sender is not included in any alias expansions (for example, if john sends to group, and group includes john in the expansion, the letter will not be delivered to john).

The sendmail command accepts the following options:

-B*type* Sets the body type to *type*. Current legal values are 7BIT or 8BITMIME.

-ba Goes into ARPAnet mode. All input lines must end with a carriage return-line feed (CR-LF), and all messages are generated with a CR-LF at the end. Also, the From: and Sender: fields are examined for the name of the sender.

-bd Runs as a daemon. sendmail will fork and run in background, listening on socket 25 for incoming SMTP connections.

-bi Initializes the alias database.

-bm Delivers mail in the usual way (default).

-bp Prints a listing of the queue.

`-bs`	Uses the SMTP protocol as described in RFC 821 on standard input and output. This option implies that all operations are compatible with SMTP.
`-bt`	Runs in address test mode, which reads addresses and shows the steps in parsing. It is used for debugging configuration tables. This mode is used for debugging configuration tables.
`-bv`	Verifies names only; does not try to collect or to deliver a message. This option is usually used for validating user or mailing lists.
`-C`*file*	Uses an alternate configuration file. `sendmail` refuses to run as root if an alternate configuration file is specified.
`-d`*X*	Sets debugging value to *X*.
`-F`*fullname*	Sets the full name of the sender.
`-f`*name*	Sets the name of the sender of the mail. Only special users (`root`, `daemon`, and `network`) can use the `-f` option, unless the sender you are trying to set has the same status as yours.
`-h`*N*	Sets the hop count to *N*. The hop count is incremented each time the mail is processed. When it reaches the limit (*N*), the mail is returned with an error message, the victim of an aliasing loop. If not specified, `Received:` lines in the message are counted.
`-n`	Prevents alias determination.
`-o`*x value*	Sets option *x* to the specified *value*. Options are described below.
`-p`*protocol*	Sets the protocol used to receive the message. This can be a simple protocol, such as `UUCP`, or a protocol and hostname, such as `UUCP:ucbvax`.
`-q`[*time*]	Processes saved messages in the queue at given intervals. If you omit *time*, the queue is processed once. *time* is specified in seconds (`s`), minutes (`m`), hours (`h`), days (`d`), and weeks (`w`) (for example, `-q1h30m` or `-q90m` both set the *timeout* to 1 hour, 30 minutes). If *time* is specified, `sendmail` will run in background. This option can be used safely with `-bd`.
`-qI`*substr*	Limits processed jobs to those containing *substr* as a substring of the queue id.
`-qR`*substr*	Limits processed jobs to those containing *substr* as a substring of the recipients.
`-qS`*substr*	Limits processed jobs to those containing *substr* as a substring of the sender.
`-r`*name*	Indicates an alternate and obsolete form of the `-f` option.
`-t`	Reads message for recipients, and scans `To:`, `Cc:`, and `Bcc:` lines for recipient addresses. The `Bcc:` line is deleted before transmission. Any addresses in the argument list are suppressed.
`-v`	Goes into verbose mode. Alias expansions are announced, and so on.
`-X` *logfile*	Logs all traffic in and out of mailers in the indicated log file. This option should be used only as a last resort for debugging mailer bugs. This option will log large amounts of data very quickly.
address ...	Specifies the address to which the mail is sent.

Options

You can set several processing options. Usually, these are used only by a system administrator. Options can be set either on the command line by using the `-o` option or in the configuration file. The processing options are as follows:

`A` *file* Uses an alternate alias file.

`b` *nblocks* Indicates the minimum number of free blocks needed on the spool filesystem.

`c` Indicates that for mailers that are considered expensive to which to connect, `sendmail` does not initiate immediate connection. This requires queueing.

`C` *N* Checkpoints the queue file after every *N* successful deliveries (the default value is 10). This options avoids excessive duplicate deliveries when sending to long mailing lists and the delivery is interrupted by a system crash.

`d` *x* Sets the delivery mode to *x*. Delivery modes are `i` for interactive (synchronous) delivery, `b` for background (asynchronous) delivery, and `q` for queue only; that is, actual delivery is done the next time the queue is run.

`D` Tries to rebuild the alias database automatically if necessary.

`e` *x* Sets error processing to mode *x*. Valid modes are `m` to mail back the error message, `w` to write back the error message (or mail it back if the sender is not logged in), `p` to print the errors on the terminal (default), and `q` to throw away error messages (only exit status is returned). If the text of the message is not mailed back by modes `m` or `w` and if the sender is local to this machine, a copy of the message is appended to the `dead.letter` file in the sender's home directory.

`f` Saves UNIX-style `From` lines at the front of messages.

`G` Matches local mail names against the `GECOS` portion of the password file.

`g` *N* Specifies the default group ID to use when calling mailers.

`H` *file* Specifies the SMTP help file.

`h` *N* Specifies the maximum number of times a message is allowed to hop before it is considered to be in a loop.

`i` Does not take dots on a line by themselves as a message terminator.

`j` Sends error messages in MIME format.

`K`*timeout* Sets the connection cache timeout.

`k`*N* Sets the connection cache size.

`L` *n* Specifies the log level.

`l` Pays attention to the `Errors-To:` header.

`m` Sends to ''me'' (the sender) also if the sender is in an alias expansion.

n	Validates the right hand side of aliases during a newaliases(1) command.
o	If set, specifies the message may have old style headers. If not set, the message is guaranteed to have new style headers (that is, commas instead of spaces between addresses). If set, an adaptive algorithm that correctly determines the header format is used.
Q *queuedir*	Selects the directory in which to queue messages.
S *file*	Saves statistics in the specified file.
s	Instantiates the queue file, even under circumstances where it is not strictly necessary. This provides safety against system crashes during delivery.
T *time*	Sets the *timeout* on messages in the queue to the specified time. After delivery has failed for this amount of time, they are returned to the sender; the default is 3 days.
t *stz, dtz*	Sets the name of the time zone.
u *N*	Sets the default user ID for mailers.
Y	Forks each job during queue runs. This option can be convenient on memory-poor machines.
7	Strips incoming messages to seven bits.

If the first character of the user name is a vertical bar (|), the remainder of the user name is used as the name of the program to which to send the mail. It might be necessary to enclose the name of the user in quotation marks to prevent sendmail from suppressing the blanks from between arguments. For example, a common alias is:

```
msgs: "|/usr/bin/msgs -s"
```

Aliases can also have the syntax :include:filename, asking sendmail to read the named file for a list of recipients. For example, the alias:

```
poets: ":include:/usr/local/lib/poets.list"
```

would read /usr/local/lib/poets.list for the list of addresses making up the group.

The sendmail command returns an exit status that describes what it did. The codes are defined in the /usr/include/sysexits.h include file.

Code	Description
EX_OK	Successful completion at all addresses
EX_NOUSER	User name not recognized
EX_UNAVAILABLE	Necessary resources are not available
EX_SYNTAX	Syntax error in address
EX_SOFTWARE	Internal software error, including bad arguments
EX_OSERR	Temporary operating system error, such as cannot perform fork process
EX_NOHOST	Host name not recognized

EX_TEMPFAIL	Message could not be sent immediately, but it was queued

If invoked as `newaliases`, `sendmail` rebuilds the alias database. If invoked as `mailq`, `sendmail` prints the contents of the mail queue.

NOTES

If this command is installed with a privilege assignment list (PAL), a user who is assigned the following privilege text upon execution of this command is allowed to perform the actions shown:

Privilege Text	Action
`daemon`	Allowed to start the `sendmail` daemon.
`dmstart`	Allowed to start the `sendmail` daemon.

If this command is installed with a PAL, a user with one of the following active categories is allowed to perform the action shown:

Active Category	Action
`system`, `secadm`	Allowed to start the `sendmail` daemon.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to start the `sendmail` daemon.

For the root user's `.forward` file to be processed, the file must have the other-readable mode bit set.

All undefined command-line switches are silently ignored. No error message is printed to notify the user of the invalid command-line switch.

All unknown options are silently accepted, but ignored. No error message is printed to notify the user of the invalid option.

The name of the local host should be fully qualified in the `/etc/hosts` file. Otherwise, a delay will occur when initiating `sendmail`.

For information about `/usr/lib/sendmail.cf`, see the *UNICOS Networking Facilities Administrator's Guide*, Cray Research publication SG–2304. The documentation for `sendmail` is also available in the UNIX4.3 BSD System Manager's Manual, University of California at Berkeley.

BUGS

The `sendmail` command converts blanks in addresses to dots. This is incorrect according to the removed ARPAnet mail protocol RFC 733 (NIC 41952), but it is consistent with the new protocols (RFC 822).

FILES

The following path names are site-specific, except for `/usr/lib/sendmail.cf`. You must specify the path names in `/usr/lib/sendmail.cf`.

`/usr/lib/aliases`	Raw data for alias names
`/usr/lib/aliases.dir`	Alias database

`/usr/lib/aliases.pag`	Alias database
`/usr/lib/sendmail.cf`	Configuration file
`/usr/lib/sendmail.hf`	Help file
`/usr/spool/mqueue/sendmail.st`	Collected statistics
`/usr/spool/mqueue/*`	Temporary files
`/etc/sendmail.pid`	Process ID of the daemon
`$HOME/dead.letter`	Copy of message

SEE ALSO

`mail`(1), `privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`syslog`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`aliases`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

`ia_mlsuser`(3C) for information about users' mandatory access control (MAC) attributes in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

DARPA Internet Request For Comments RFC 819, RFC 821, and RFC 822

Allman, E. and M. Amos, ''SENDMAIL Revisited,'' *Proceedings of the Summer 1985 USENIX Conference.* USENIX Association

NAME

`setacid` – Sets default accounting ID

SYNOPSIS

`/usr/lib/acct/setacid` [`-v`] [`-z`] *directory*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `setacid` command assigns a default accounting ID to each file in a file system, or directory tree.

The `setacid` command accepts the following options and operand:

`-v` Sets verbose mode on.

`-z` Sets a file's account ID to the file owner's default account ID. This is done only on files that have an account ID of 0.

directory Starting directory in which `setacid` starts when making changes.

NOTES

When running `setacid` on a directory that has account IDs set, always use the `-z` option to assure that the owner's default account ID is set for that directory.

FILES

`/etc/acid` Account ID information file

`/etc/udb` User database control information file

SEE ALSO

`diskusg`(8)

`chacid`(1), `newacct`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`chacid`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

NAME

setdate – Sets the system date from the IOS

SYNOPSIS

/etc/setdate

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The setdate command uses a date packet passed into central memory by the I/O subsystem (IOS) at boot time to determine the time of day.

The setdate command also sets the system time-zone information into the kernel extended time-zone structure as described in the sys/time.h file. This is usually done at boot time, but setdate may be run at any time to set or update this information. The time-zone information is obtained from the TZ environment variable. When setdate detects that the time-zone information has already been set, it updates the time-zone information as required, but does not modify the system time.

The init(8) command should execute setdate at boot time. To execute the procedure, use the following inittab command:

```
tz::sysinit:TZ=CST6CDT
sd::sysinit:/etc/setdate 1 >/dev/console 2>&1
```

For the time to be correct, you must change the time-zone string (TZ=CST6CDT) in the inittab file (see inittab(5)). The first line is usually set prior to the setdate in /etc/inittab.

Only an appropriately authorized user can set the system date.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Allowed to set the system date.
sysadm	Allowed to set the system date. Shell-redirected I/O is subject to security label restrictions.

If the PRIV_SU configuration option is enabled, the super user is allowed to set the system date.

SEE ALSO

`init`(8)

`settimeofday`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

`ctime`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`inittab`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

seterr – Set maximum user error counts

SYNOPSIS

/etc/seterr [-e *err*] [-o *ore*] [-p *pre*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The seterr command allows values to be set in the kernel maxerrint table. These values specify the maximum number of Error Exits, Operand Range Errors or Program Range Errors that a process may accumulate before being aborted. The error counts are maintained on a per-CPU basis and are cleared each time a process is connected to a CPU. If any of the error counts exceeds the value in the maxerrint table during a connection interval, the connected process is terminated with a SIGKILL signal. The seterr command will print the values in the maxerrint table after changes specified by the parameters, if any, have been made.

The seterr command accepts the following options:

-e *err* Specifies the maximum allowable count of Error Exits, *err*.

-o *ore* Specifies the maximum allowable count of Operand Range Errors, *ore*.

-p *pre* Specifies the maximum allowable count of Program Range Errors, *pre*.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Allowed to alter system values.
sysadm	Allowed to alter system values. Shell redirected I/O and access to specified processes are subject to security label restrictions.

If the PRIV_SU configuration option is enabled, the super user is allowed to alter system values.

MESSAGES

seterr sends messages about parameter values that cannot be converted correctly, or about problems reading or setting the values in the system maxerrint table.

FILES

`/dev/cpu/any` CPU control device

SEE ALSO

`cpu`(8)

`cpu`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

setfs – Changes dynamic information in file system super block

SYNOPSIS

/etc/setfs [-B *bf*] [-A *bu*] [-L *sl*] [-U *sl*] [-a *al*] [-b *flaw_list*] [-c] [-i] [-s *arbiter:semaphore_count*] [-z] [-*flag*] *special*

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The setfs command makes changes to the file system super block without requiring you to make an entire new file system. You must unmount the file system before using this command to make alterations.

The setfs command accepts the following options and argument:

-B *bf* Big file size (in bytes) for the file system.

-A *bu* Big file allocation units (in 512-word blocks) used when a file exceeds the "Big file" size.

-L *sl* Lower-level security for this file system.

-U *sl* Upper-level security for this file system.

-a *al* Allocation strategy for this file system. The values for *al* may be one of the following:

- rrd1 Handles the first level directories in a round-robin manner.
- rrda Handles all directories in a round-robin manner.
- rrf Handles all files in a round-robin manner. Directories and inodes are allocated in the first partition of the multiple partition file system whenever possible.

-b *flaw_list* Reads a list of decimal tuples (pairs of numbers) from the file *flaw_list* that specifies the starting block number and the number of blocks in the bad sections of the file system. If the name of the file is –, setfs will read standard input.

The setfs command avoids using the specified areas, when possible: If information for which the location is critical (such as the super block) falls within the specified area, that area of the disk is used; if the specified area is not a critical area, the area specified is reserved when mkfs is executed and is not allocated as new file space. Reserving bad blocks with the -b option avoids using the bad block areas on disk that are remapped to spare cylinders. Using the spare cylinders may case an I/O performance degradation when reading or writing the bad block area; the tradeoff can be disk fragmentation caused by bad block avoidance, if there are a large number of flaws on a particular disk.

Blokcs that are already reserved when setfs declares them as suspect could still be in use by files. This does not affect prior allocation of these blocks.

`-c` Clears any existing flaw list from the file system. Specifying both the `-b` and the `-c` options replaces an old flaw list with the flaw list specified by the file *flaw_list*.

`-i` Toggles inode allocation preference placement.

`-s` *arbiter:semaphore_count*
Used to change `SFS` (Shared File System) file system parameters.

The argument to the `-s` option provides the *arbiter* name or number, and the number of semaphores from that arbiter to be assigned to this file system at mount time.

The *arbiter* name or number must match one of the valid configuration entries in the `/etc/config/sfs` configuration file.

With the `-s` option, an unmounted shared file system may be changed to an `NC1FS` file system by specifying a *semaphore_count* of zero. Similarly, an unmounted `NC1FS` file system may be changed to a shared file system with the `-s` option.

`-z` Toggles the setting of the file system panic flag. When enabled, the system panics on file system errors encountered. When disabled, the error is logged to `/dev/fslog` and handled by the `fslogd`(8) daemon.

-flag Sets a test flag for the file system. *flag* can be 1, 2, or 3.

special Specifies the name of the block special device on which the file system exists. If no options other than *special* are specified, the `setfs` command displays the current values in effect for the file system.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to specify any file system.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to specify any file system.

When using either the `-b` or `-c` option, `fsck`(8) must be run on the file system.

SEE ALSO

General UNICOS System Administration, Cray Research publication SG–2301

NAME

`setpal` – Sets privilege assignment list (PAL) category entries of a file

SYNOPSIS

`setpal` [`-f`] [`-p` *privlist*] [`-t` *privtext*] *catlist files...*
`setpal -d` *catlist files...*
`setpal -c` *files...*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `setpal` command adds or deletes category entries from the privilege assignment lists (PALs) of the specified regular file(s).

Only one PAL entry may exist for a specified category.

The `setpal` command accepts the following options and operands:

`-c` Clears the PAL such that only the `other:PRIV_NULL:TEXT_NULL` entry remains.

`-d` Deletes the PAL entries of categories specified in *catlist*. Specified categories that do not exist in the PAL are ignored. The `other` entry cannot be deleted. Attempting to delete the `other` category entry clears the privilege and privilege text values.

`-f` Forces the overwrite of any PAL entries for privileges specified in *catlist*. If this option is not specified, and a PAL entry already exists for a category specified in *catlist*, the `setpal` command returns an unsuccessful error status.

`-p` Specifies the set of privileges to associate with PAL entries of the specified categories. Privileges are specified in *privlist*. If this option is not specified, the privilege set associated with each PAL entry is `PRIV_NULL`.

`-t` Specifies the privilege text to associate with PAL entries of the specified categories. Privilege text is specified by *privtext*. If this option is not specified, the privilege text associated with each PAL entry is `TEXT_NULL`.

catlist A character string that represents one or more category names (for example, `secadm`). Multiple category names must be separated by commas, with no intervening white space.

privlist A character string that represents one or more privilege names (for example, `PRIV_MAC_READ`). Multiple privilege names must be separated by commas, with no intervening white space. The `PRIV_ALL` character string represents the list of all privileges.

privtext A sequence of one to eight alphanumeric characters, or the word `TEXT_NULL`, that represents privilege text.

files Represents the name(s) of the file(s) whose PALs are to be changed.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

EXIT STATUS

The `setpal` command exits with one of the following values:

Value	Description
0	The specified file's PAL was successfully updated.
1	A badly formed option or an option that was not valid was supplied.
2	When multiple files are supplied, both failure and success occurred.
4	The PAL(s) for the specified file(s) could not be displayed.

EXAMPLES

Example 1: The following example clears the contents of the PAL associated with `testfile`, such that it contains only the PAL entry `other:PRIV_NULL:TEXT_NULL`:

```
$ setpal -c testfile
```

Example 2: The following example creates a `sysadm:PRIV_FOWNER,PRIV_KILL:adm` entry in PAL associated with `testfile`, overwriting a `sysadm` category entry if it exists:

```
$ setpal -f -p PRIV_FOWNER,PRIV_KILL -t adm sysadm testfile
```

Example 3: The following example creates a `secadm:PRIV_NULL:sec` entry in the PAL associated with `testfile`, overwriting a `sysadm` category if it exists:

```
$ setpal -f -t sec secadm testfile
```

Example 4: The following example creates a `secadm:PRIV_NULL:TEXT_NULL` entry in the PAL associated with `testfile`, overwriting a `secadm` category entry if it exists:

```
$ setpal -f -t TEXT_NULL secadm testfile
```

An alternative way of creating a `secadm:PRIV_NULL:TEXT_NULL` entry in the PAL associated with `testfile`, overwriting a `secadm` category entry if it exists, is as follows:

```
$ setpal -f secadm testfile
```

Example 5: The following example creates a `secadm:PRIV_ALL:TEXT_NULL` entry in the PAL associated with `testfile`, overwriting the `secadm` and `system` category entries if they exist:

```
$ setpal -f -p PRIV_ALL secadm,system testfile
```

Example 6: The following example deletes the `secadm` category entry from the PAL associated with `testfile`:

```
$ setpal -d secadm testfile
```

SEE ALSO

`getpal`(8)

General UNICOS System Administration, Cray Research publication SG–2301

NAME

setprivs – Modifies the file privilege sets of a file

SYNOPSIS

setprivs [-a] [-f] [-s] *privlist files...*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The setprivs command updates the allowed, forced, and set-effective privilege sets of the specified regular files.

Only an appropriately authorized user can use this command.

The setprivs command accepts the following options and operands:

-a Updates the allowed privilege set to the value specified in *privlist*.

-f Updates the forced privilege set to the value specified in *privlist*.

-s Updates the set-effective privilege set to the value specified in *privlist*.

privlist A character string that represents one or more privilege names (for example: PRIV_MAC_READ). Multiple privilege names must be separated by commas with no intervening white space. To clear privileges, the value of *privlist* should be the PRIV_NULL character string. The PRIV_ALL character string represents the list of all privileges.

files Represents the name(s) of the file(s) whose privilege set(s) are to be updated.

If no options are specified, then all privilege sets are updated.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Allowed to use this command.

If PRIV_SU is enabled, the super user is allowed to use this command.

EXIT STATUS

The `setprivs` command exits with one of the following values:

Value	Description
0	The specified privilege state was successfully updated.
1	A badly formed option or an option that was not valid was supplied.
2	When multiple files are supplied, failure and success both occurred.
4	The privilege state for the specified file(s) could not be updated.

EXAMPLES

Example 1: The following example sets the forced and set-effective privilege sets of `testfile` to `PRIV_MAC_READ,PRIV_MAC_WRITE`:

```
$ setprivs -fs PRIV_MAC_READ,PRIV_MAC_WRITE testfile
```

Example 2: The following example sets the forced and set-effective privilege sets of `testfile` to contain all of the defined privileges:

```
$ setprivs -fs PRIV_ALL testfile
```

Example 3: The following example sets the allowed, forced, and set-effective privilege sets of `testfile` to `PRIV_MAC_READ`:

```
$ setprivs -fsa PRIV_MAC_READ testfile
```

An alternative way to set the allowed, forced, and set-effective privilege sets of `testfile` to `PRIV_MAC_READ` is shown in the following example:

```
$ setprivs PRIV_MAC_READ testfile
```

Example 4: The following example clears the allowed privilege set of `testfile`:

```
$ setprivs -a PRIV_NULL testfile
```

Example 5: The following example clears all privilege sets of `testfile`:

```
$ setprivs -a -f -s PRIV_NULL testfile
```

An alternative way to clear all privilege sets of `testfile` is shown in the following example:

```
$ setprivs PRIV_NULL testfile
```

SEE ALSO

`getprivs`(8)

NAME

sfsd – Initializes and monitors the External Semaphore Device

SYNOPSIS

/etc/sfsd [-v] [-F]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The sfsd command is used to initialize and monitor the External Semaphore Devices.

The sfsd command replaces the esdaemon command.

The sfsd command uses the special devices /dev/sfsd..., /dev/smp... and /dev/smnt... to interface to the respective device drivers. The actual names of the special devices is provided in the /etc/config/sfs Shared File System (SFS) configuration file. Usually, sfsd tries to determine the necessity of clearing and initializing the External Semaphore Device by scanning the state of all the semaphores. The presence of any errors usually indicates a *power-up* condition, and initialization will be done. Use the -F option to force unconditional initialization.

Executing the sfsd command requires super-user permissions.

-v (Verbose) Causes additional information to be displayed.

-F (Force) Causes the unconditional clearing and initialization of the External Semaphore Device.

FILES

/etc/config/sfs The Shared File System configuration file.

SEE ALSO

sfs(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

shradmin – Changes and displays costs for fair-share scheduler usage calculations

SYNOPSIS

/etc/shradmin [-b *bio*] [-c *clicks*] [-D *h1,h2*] [-E *half_life*] [-F *flags*] [-G *maxgroups*] [-K *half-life*] [-m *click*] [-n *interval*] [-P *maxpri*] [-p *procs*] [-Q *maxupri*] [-R *delta*] [-S *maxusers*] [-s *syscall*] [-t *tick*] [-U *maxusage*] [-v] [-X *maxushare*] [-Y *mingshare*] [-y *tio*] [-Z *sharemin*] [*percent*]

/etc/shradmin -r

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The shradmin command changes the costs associated with the usage calculations for the fair-share scheduler.

The -r option must be used by itself, because the order of option processing by shradmin is controlled by the getopt(3C) library routine).

The shradmin command accepts the following options and operand. You must specify at least one of these options; otherwise, shradmin does not execute.

-b *bio* Sets the charge for a logical I/O request.

-c *clicks* Specifies the maximum number of clicks of memory that can be used by the processes attached to an lnode. (This amount is divided among all the processes.)

-D *h1,h2* Sets the decay rate for process priorities to normal nice, so that they decay to half their initial value in *h1* seconds, and sets the decay rate for process priorities to maximum nice, so that they decay to half their initial value in *h2* seconds.

-E *half_life* Sets the decay rate for users' process rates so that they decay to half their initial value in *half_life* seconds.

-F *flags* Sets various global scheduling flags. The *flags* value must be in octal. The following flags are available:

Flag	Description
NOSHARE	(001) Turns off the fair-share scheduler. Leaves accumulated charges in the user database (UDB) unless they are cleared by the system administrator.
ADJGROUPS	(002) Specifies adjustments by group IDs (group share allocation).
LIMSHARE	(004) Specifies limits on maximum share.

NOSCHED (020) Gathers fair-share charges and usage information, but does not use these values for CPU scheduling.

SHAREBYACCT (010) Specifies Share by Account mode; share priorities are calculated based on active account IDs (shareholders) instead of active user IDs. The correct value for the SHAREHOLDER flag in the UDB is now required; see the NOTES section for more information.

USRLEVLFSS (0100) Specifies *user-level fair-share mode*; in this mode, the fair-share daemon (shrdaemon(8)) performs the share calculations and updates the lnodes. This will be the default mode in future releases of the UNICOS operating system.

See the share(5) man page for more information on the global scheduling flags.

-G *maxgroups* Sets the maximum depth for the fair-share hierarchy. *maxgroups* is 4 by default.

-K *half-life* Sets the decay rate for users' usages so that they decay to half their initial value in *half-life* hours. If *half-life* has the suffix s, the decay rate is set to *half-life* seconds rather than hours.

-m *click* Sets the charge for a memory tick.

-n *interval* Specifies the interval, in seconds, to be used for copying lnode information to the UDB.

-P *maxpri* Specifies the absolute upper bound for the priority of a process. (The value must be less than the largest nonnegative integer.)

-p *procs* Specifies the maximum number of processes that can be attached to an lnode.

-Q *maxupri* Specifies the upper bound for normal processes' priorities. Idle processes run with priorities in the range *maxupri* < *pri* < *maxpri*.

-R *delta* Sets the run-rate for the fair-share scheduler in seconds.

-r Causes shradmin to reset to the default values. No other options may be used with this option.

-S *maxusers* Sets the maximum number of users and groups that can be active. This value cannot exceed the size of the lnode table configured in the kernel (set by the NUSERS parameter).

-s *syscall* Sets the charge for a system call.

-t *tick* Sets the charge for a CPU tick.

-U *maxusage* Sets the MAXUSAGE parameter (the upper bound for reasonable usages). Users with usages larger than this are grouped together and given a normalized usage that prevents them from interfering with users that do not have large usages.

-v Displays the scheduling feed-back parameters. Do not attempt to set the changed parameters.

`-X` *maxushare* If the `LIMSHARE` scheduling flag is on, then this option limits the maximum effective share an individual user can have to *maxushare* times his or her allocated share. The default is 2.0, or 2%.

`-Y` *mingshare* If the `ADJGROUPS` scheduling flag is on, the priority is adjusted for any group that is getting less than *mingshare* times its allocated share.

`-y` *tio* Sets the charge for a stream I/O operation. This is dependent on the number of kernel buffer operations, so a "`write(1)`" operation costs the same as a "`write(64)`" operation to an ordinary stream, or a "`write(1024)`" operation to a pipe.

`-Z` *sharemin* Specifies the minimum machine share allocated to a user. This option, combined with the `-X` (*maxushare*) option, provides a floor on individual priorities to provide all users with reasonable interactive response, regardless of share allocations or past usage. The default is 0.

percent The percentage to apply to all the charges.

NOTES

If this command is installed with a privilege assignment list (PAL), a user who is assigned the following privilege text upon execution of this command is allowed to perform the action shown:

Privilege Text	Action
`chgany`	Allowed to modify charges.

If this command is installed with a PAL, a user with one of the following active categories is allowed to perform the action shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to modify charges.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to modify charges.

As of UNICOS release 8.3, all entries in the UDB must be correct. The correct use of flags in the UDB is enforced. For example, in Share by Account mode, account ID entries (shareholders) must now have the the `SHAREHOLDER` flag set to 01000000. Lnodes are no longer generated for incorrect entries; users whose UDB entries reference the incorrect entries are unable to log in and their Network Queuing Environment (NQE) jobs are rejected. The `shrtree`(8) command can be used to analyze entries in the UDB; this command reports problems that can cause unexpected system behavior or prevent login for affected users. See the `shrtree`(8) man page for information on running this command.

BUGS

The percent value also affects any new constants, so bias them accordingly.

The defaults are rarely relevant.

EXAMPLES

Example 1: The following example invokes `shradmin` with the `-v` (view) option to show the current charges:

```
$ shradmin -v

sn1703a ice 9.0.0ab wdp.0 CRAY Y-MP
Scheduling flags     = ADJGROUPS,LIMSHARE
Charging percentage = 100%,
Usage decay rate    = 0.99807644 (half-life of 3600.0 seconds),
active users = 16, active groups = 10.
max. users = 200, max group nesting = 4.

Charge: syscall      0%, bio      0%, tio      0%, tick    100%, click      0%.
Costs:  syscall       0, bio       0, tio       0, tick     100, click       0.
Counts: syscall 3916030, bio  627083, tio   38762, tick 3858349, click       0.

Process priority decay rate biased by "nice":-
    high priority (nice -20) 0.4044 (half-life of   0.8 seconds),
    avg  priority (nice   0) 0.7039 (half-life of   2.0 seconds),
    low  priority (nice  19) 0.9885 (half-life of  60.0 seconds).
Run rate decay rate          0.8409 (half-life of   4.0 seconds).

Max. value for normal usage        = 1.000000e+12,
Max. value for normal p_sharepri   = 1.000000e+28,
Max. value for idle   p_sharepri   = 1.000000e+38.

High value of current normal usage = 9.902495e+08,
high value of current p_sharepri   = 1.041637e+01.
```

Example 2: The following command line changes the costs to 10% of the cost values:

```
shradmin 10
```

Example 3: The following command line alters the default parameters at system boot time in `/etc/config/daemons`; this is unnecessary if the default values set in the kernel are correct.

```
shradmin -F04 -K1 -R10 -Y0.90 -Z0.001 -b1666 -m3 -s523 -t600 -y4700
```

Example 4: The following example sets share hierarchy levels at 7:

```
shradmin -G 7
```

Example 5: The following example sets up Share by User mode by setting the LIMSHARE and ADJGROUPS flags:

```
shradmin -F 06
```

Example 6: The following example sets up Share by Account mode by setting the ADJGROUPS and SHAREBYACCT flags:

```
shradmin -F 012
```

Example 7: The following command line turns off the fair-share scheduler. (The fair-share daemon, shrdaemon(8), continues to accrue charges in the UDB until it is stopped by the system administrator.)

```
shradmin -F 01
```

FILES

/usr/include/sys/share.h	Definition of shconsts structure

SEE ALSO

shrdaemon(8), shrtree(8)

privtext(1), shrview(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

limits(2), policy(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

getopt(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

share(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Resource Administration, Cray Research publication SG–2302

Henry, G.J., "The Fair Share Scheduler," *Bell Labs Technical Journal*, LVIII-8b, 10-84, pp. 1845-1858.

NAME

shrdaemon – Performs system functions for the fair-share scheduler

SYNOPSIS

/etc/shrdaemon [-t]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The shrdaemon command cleans up kernel limits structures (lnodes) after users have logged off, and writes the usage and charge information to the user database (UDB). shrdaemon is invoked at system startup in the /etc/config/daemons configuration file and runs as a system daemon (with UID 0) in the background.

The shrdaemon command uses the limits(2) system call to initialize the default idle lnode, configured at system boot time, for use of the idle processes. shrdaemon goes into a loop scanning for dead user lnodes, updating them in the UDB as they appear, and checkpointing active lnodes every few minutes.

The shrdaemon command also maintains a checkpoint file of active users, /etc/lnodes.chkpt, so that after a system crash, charges accumulated by users active at the time of the crash are not lost. The /etc/lnodes.chkpt file, if present, is read when shrdaemon is first invoked, and active lnodes found in it are updated in the user database (UDB).

shrdaemon catches the system termination signal 27 (SIGSHUTDOWN), which causes it to remove /etc/lnodes.chkpt, update the UDB for any remaining active lnodes, and terminate.

If the user-level fair-share mode is enabled (that is, the USRLEVLFSS flag is set), shrdaemon performs the functions that the kernel would perform in kernel-level fair-share. shrdaemon applies CPU scheduling policy calculations to data in the lnodes, then updates the lnode table in the kernel. In addition, a user exit allows sites to substitute site-specific scheduling policy algorithms for the standard policies Share by User and Share by Account (SHAREBYACCT), and the ADJGROUPS, NOSHARE, LIMSHARE, and NOSCHED kernel calculations.

The shrdaemon command accepts the following option:

-t Creates a log file for debugging purposes; log information is written to standard error (stderr). This option is intended for internal debugging only; it is not recommended for use at a site because the log mechanism consumes significant CPU time and the log file can become quite large.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to start the daemon.

If the PRIV_SU configuration option is enabled, the super user is allowed to start the daemon.

The checkpoint file used by shrdaemon is not in any way related to the checkpointing accomplished by the chkpnt(1) and restart(1) commands, and the chkpnt(2) and restart(2) system calls.

BUGS

If the shrdaemon process should die for any reason, then new users logging into the system gradually consume all available kernel lnodes. You can use the automated incident reporting (AIR) daemon, aird(8), to restart shrdaemon.

FILES

/etc/udb	User validation file containing user control limits
/etc/lnodes.chkpt	Checkpoint file for active lnodes

SEE ALSO

aird(8), shradmin(8)

limits(2), policy(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

lnode(5), share(5), udb(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

shrdist – Redistributes shares among a resource group for the fair-share scheduler

SYNOPSIS

shrdist [-b] [-g *group*] [-p *directory*] [-s *shares*] [-u *user*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The shrdist command reassigns shares within a resource group. The point-of-contact (POC) for each resource group can control the allocation of resource shares in his or her group. This control is authorized by an authorization file, which shrdist checks before executing. If the POC administers more than one resource group, the particular group must be specified (by using the -g *group* option and argument) when the shrdist command is invoked.

The shrdist command accepts the following options:

-b	Enables batch mode.
-g *group*	Specifies a resource group.
-p *directory*	Specifies a path to a directory other than /etc (this option allows you to test the execution of shrdist).
-s *shares*	Specifies the new shares value (not valid in interactive mode).
-u *user*	Specifies the user name to be updated (not valid in interactive mode).

The shrdist command has two modes of execution: interactive and batch.

To use interactive mode, you must be able to communicate in full-screen mode (for example, vi).

When you execute shrdist in interactive mode, the command displays the current share distribution for each account under the resource group. You control the cursor position with the following keys:

Key	Cursor Movement
<h>	Left
<j>	Down
<k>	Up
<l>	Right
<+>	Next page
<->	Previous page
</>	Prompts for a search pattern

You can change (reallocate) the share values by positioning the cursor (which should always be positioned over the rightmost digit of the share allocation field) at the desired account and entering the new values. All other accounts are then automatically recalculated. Digits shift to the left as you enter them. The current value for the <ERASE> key can be used to erase characters.

After you have reallocated all new share values, enter `u` to update the share allocation database, and enter `q` to quit (exit) the `shrdist` command. If the share distribution has been updated and you exit (`q`) without having updated (`u`), `shrdist` prompts for verification.

To use `shrdist` in batch mode, use the `-b` option on the command line. In this mode, only one account may be updated per execution of `shrdist`. Specify the user name to be updated with the `-u` option, and the new shares value with the `-s` option. (The `-u` and `-s` options are valid only with the `-b` option.)

The `shrdist` command supports test mode execution (similar to `udbgen` and `udbsee`). You can specify the `-p` option to specify a path to a directory other than `/etc`, in which test versions of the `udb`, `group`, `acid`, and `shrdist.auth` files can be found.

The `shrdist` command uses `/etc/shrdist.auth` as the authentication file; therefore, all of the resource points of contact should have read access to `/etc/shrdist.auth`. The `shrdist.auth` file is a simple ASCII text file, consisting of two fields per line, separated by white space. The first field is the user identity of the resource point of contact, and the second field is the name of the resource group. The resource group name may be terminated by end-of-line or any white space. If white space follows the resource group name, anything beyond that white space is treated as a comment.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

Example `shrdist.auth` file:

```
user1 resource1
user1 resource-group2
user2 resource-group2          Comments
```

In this example, `user1` is allowed to function as point of contact for more than one resource group. If a resource group is not specified with the `-g` option, the first match (in this case, `resource1`) is used.

FILES

`/etc/shrdist.auth` Share authentication file

SEE ALSO

`shradmin`(8), `shrsync`(8)

`shrview`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

shrlimit – Sets another user's limits for the fair-share scheduler

SYNOPSIS

/etc/shrlimit [-u] *user_name* [*command*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The shrlimit command creates a new process (with limits per the values for the specified *user_name* in the user database (UDB)), changes to the user and group IDs *user_name*, prefixes *command* with the login shell for *user_name* (or bin/sh -c by default), and executes using execv (see exec(2)) with the resulting string. This command can be executed only by an appropriately authorized user.

The shrlimit command accepts the following option and operands:

-u	Does not change user and group IDs.
user_name	Specifies the user name; this operand is not optional.
command	Passes any optional extra arguments as one argument to the invoked shell's -c option.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	**Action**
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

FILES

/etc/udb	User validation file containing user control limits

SEE ALSO

limits(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

share(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

shrmon – Monitors detailed system fair-share scheduling information

SYNOPSIS

/etc/shrmon [-c] [-i *n*] [-l *n*] [-r *n*] [-u] [-v]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The shrmon command displays a columnar table of information in order by user and group.

The number of columns displayed depends upon the -v (verbose) option. Each usage of the -v option increases the level of verboseness by one. For example, the following command line specifies verbose level 2:

```
shrmon -vv
```

See the Column Descriptions subsection to determine the level of verboseness at which that particular column will appear.

If no options are specified, shrmon prints the minimum report and exits.

The shrmon command accepts the following options:

-c Specifies continuous operation, using curses(3) routines for full-screen update.

-i *n* Specifies the interval between updates as *n* seconds. The default is 4.

-l *n* Limits the display to the first *n* levels of the share hierarchy. For example, the following command line reports on only the first level groups:

```
shrmon -l 1
```

-r *n* Specifies repeat mode of operation. shrmon repeats the display *n* times, each separated by a form-feed, and then exits.

-u Prints the user and group names on each line, separated by a /. For example, user work in group busy would be displayed as follows:

```
System/busy/work
```

-v Increases the amount of information displayed. Each usage of the -v option increases the verboseness level by one.

Column Descriptions

The following columns are always displayed:

Column	Description
CPUs	The number of CPU seconds of time accumulated by this user during the sample period.
charge	A relative number indicating the amount charged to this user during the sample period. This number is a derivation of the cost factors described on the shradmin(8) man page.
%Rate	The percentage of all charges ascribed to this user during the sample period.
%share	The percentage of the effective share of the machine to which this user or group is entitled (as described by the allocation of shares to individual users and groups defined by the system administrator).
%CPU	The percentage of all available CPU resources that were allocated to this user or group during the sample period.
Name	The name of the user or group, as defined in the user database (UDB), with indentation to indicate hierarchy level.

The following additional columns are displayed at verbose level 1 (the -v option specified once):

Column	Description
rate	The kl_rate field, which is a floating-point number used as an indication of the rate of system usage by this user or group.
nrun	Estimate of the maximum amount of the rshare value (machine shares) that can be used by all runnable processes under the lnode.
%rshare	The percentage representing the dynamic share of the system that this user or group has been allocated in order that they achieve their effective share (see the %share column).

The following additional columns are displayed at verbose level 2 (the -v option specified twice):

Column	Description
kids	The number of group members currently active on the system.
ref	The number of processes that this user, or all of the users in this group, have currently active on the system.

The following additional columns are displayed at verbose level 3 (the -v option specified three times):

Column	Description
usage	The floating-point number that represents the kl_usage field, and relates to the system use by this user or group.
temp	The floating-point number that represents the kl_temp field, and that is an intermediate variable used in the calculation of kl_usage.

NOTES

The shmon command will be removed in a future UNICOS release. The shrview(1) command provides an improved method to monitor fair-share scheduling information.

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Shell redirected I/O is not subject to security label restrictions.

If the PRIV_SU configuration option is enabled, for the super user, shell redirected I/O is not subject to security label restrictions.

EXAMPLES

The following command line displays the minimum report in continuous-update mode:

```
$ shrmon -c

Mon Oct 11 18:10:24 1993                    CPUs charge  %Rate %share   %CPU
    0.0    1.8  100.0  100.0    0.0 Root
    0.0    0.0    0.0    0.0    0.0  Idle
    0.0    0.0    0.0   10.0    0.0  System
    0.0    0.0    0.0   10.0    0.0   operator
    0.0    1.8  100.0   90.0    0.0  Users
    0.0    1.8   96.6    0.8    0.0   SoftDev
    0.0    0.0    0.3    1.4    0.0   Mktg
    0.0    0.1    3.2    0.0    0.0   CCN
    0.0    0.0    0.0    5.0    0.0   HardDev
```

SEE ALSO

shradmin(8), shrtree(8)

shrview(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

share(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

shrsync – Synchronizes the UDB and fair-share information for active users

SYNOPSIS

shrsync [-n] [-p *directory*] [-q] [-s] [-u]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The shrsync command synchronizes the fair-share information contained in the user database (UDB) and the active system. By default, shrsync examines the system to determine what active users and resource groups are present in the system. Then it updates the active users and resource groups on the system with the current value of allocated shares as read from the user database. This process allows you to make routine changes to share allocation, such as daytime/nighttime, or shift-to-shift, without users having to log out and log in again.

The shrsync command accepts the following options:

-n (Negative) Checks for negative values in the usage and charge fields of the lnode (the l_usage and l_charge fields, respectively), and replaces them with valid information from the UDB.

-p *directory*
(Path) Specifies a path name other than /etc so that test versions of udb, group, and acid files can be used.

-q (Quotas) Updates the CPU-quota-used information with the current values from the UDB. See the EXAMPLES section for more information.

-s (Silent) Suppresses the normal messages concerning the updating of a user's UDB entry.

-u (Update) Updates the usage, charge, and quota information from the active system in the UDB so that it is an accurate reflection of all users' usage and charges when shrsync was executed.

NOTES

If this command is installed with a privilege assignment list (PAL), a user who is assigned the following privilege text upon execution of this command is allowed to perform the action shown:

Privilege Text	Action
exec	Allowed to use this command.

If this command is installed with a PAL, a user with one of the following active categories is allowed to perform the action shown:

Active Category	Action

`system`, `secadm`, `sysadm` Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

The CPU-quota-used information in the UDB is updated only when a user logs off the system. For this reason, you should ensure that the UDB contains current information before updating the active system from the UDB. This is accomplished by first updating the CPU-quota-used information in the UDB with information from the active system.

The following procedure shows the preferred sequence of events when clearing the CPU-quota-used information for a group of users.

1. Use the `shrsync -u` command to update the UDB with the information from the active system, as follows:

```
$ shrsync -u
```

2. Clear the desired CPU-quota-used fields in the UDB with the `udbgen` command. (See the `udbgen`(8) man page for more information.)
3. Use the following command to update the CPU-quota-used information from the UDB:

```
$ shrsync -q
```

FILES

`/etc/udb` User database (UDB)

SEE ALSO

`shradmin`(8), `shrdist`(8), `udbgen`(8)

`privtext`(1), `shrview`(1), `udbsee`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

shrtree – Displays the fair-share hierarchy defined in the user database (UDB)

SYNOPSIS

/etc/shrtree [-e] [-f] [-F *mode*] [-G *maxgroups*] [-L *level*] [-l] [-m] [-p *udb_path*] [-s] [*id*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The shrtree command displays the share tree, or fair-share hierarchy, that is defined in the user database (UDB), highlighting problems that could prevent a user from logging in. This command can be executed by all users. The *share tree* is the set of all resource group chains starting from the base node (also called the *root entry*, labeled _ROOT_) and ending with the user entries. Any problems with the configuration of the fair-share scheduler in the UDB are marked in the shrtree output. In addition, each entry in the hierarchy is marked with a status indicating whether it is a resource group, account ID (shareholder), or user entry.

When invoked with no options, shrtree displays the short version of output (equivalent to the -s option).

The shrtree command accepts the following options and arguments:

-e (Errors) Displays errors only. UDB entries that are set up incorrectly for use with the fair-share scheduler are displayed, along with a description of the problem. If a single UDB entry has more than one problem, the shrtree output displays an error for each problem. Entries marked with WARN are warnings; entries marked with ERROR are problems that will prevent logging in.

-f (Full display) Displays the full share tree, including user entries. By default, only resource group entries are displayed.

-F *mode* (Force fair-share mode) Overrides the current fair-share mode. Use ACCT to specify Share by Account mode; use UID to specify Share by User mode. By default, the mode specified in the kernel shconsts structure is used. This option is useful if you are analyzing a UDB that is in a different mode from the current system mode.

-G *maxgroups* (Group depth) Sets the maximum depth of the hierarchy to the specified value. By default, the value of MAXGROUPS (as defined in the shconsts structure) is used to specify the maximum level of the hierarchy.

-L *level* (Maximum level) Specifies the maximum level of the fair-share hierarchy to be displayed. By default, all levels of the hierarchy are displayed.

-l (Long display) Produces a wider display than the default display, with additional columns for the number of shares and the usage value for each entry.

-m (Medium display) Displays share usages values as decimal fractions rather than percentages.

`-p` *udb_path* (UDB path) Specifies an alternate UDB as found in the directory specified by *udb_path*.

`-s` (Short display) Displays the short version of the share tree. The `Share` and `Usage` fields are omitted; fair-share information is displayed as percentages; and only resource groups are listed (unless the `-f` option is specified). This display is the default.

id Displays information on all possible resource group chains that contain an entry that matches *id*. The value *id* can be specified as either the entry name or the ID number.

Resource Group Hierarchy Display

The `shrtree` command displays the resource group hierarchy by default (with all options except `-e`). The column headings for this display and their meanings are as follows:

Heading	Description
`Lv`	Level of the entry in the resource group chain. The root entry (`_ROOT_`) is always at level 0.
`Name`	Name of the entry in the UDB.
`ID`	ID number of the entry in the UDB.
`Shares`	(Long listing only) Number of shares from the UDB.
`System Share`	System-wide percentage of shares for the entry. For example, if an entry had a resource group share of 20% and the resource group had a 30% share of the system, the system share value would be 6% (`0.20 * 0.30 = 0.06`). If the `-m` option is specified, this value is displayed as a decimal fraction.
`Group Share`	Relative percentage of shares for the entry within the resource group. For example, if an entry was assigned 10 shares and the total shares in its resource group was 100, the group share value would be 10% (`10 / 100 = 0.10`). If the `-m` option is specified, this value is displayed as a decimal fraction.
`Usage`	(Long listing only) Usage value from the UDB (`shusage` field). The share usage for `_ROOT_` is calculated by adding the share usages of the first-level resource groups.
`System Usage`	Accumulated system share usage for the entry based on the usage value in the `shusage` field of the UDB. The value is first decayed and then divided by the total system usage obtained from `_ROOT_`. If the `-m` option is specified, this value is displayed as a decimal fraction.
`Status`	Flags to indicate the entry type (user, resource group, or account ID) and to indicate problems with the configuration of the fair-share hierarchy in the UDB. Note that problems of type `ERROR` can prevent the affected entries from logging in. An entry can have several flags, separated by the `|` character. The following flags are available:

Flag	Description
`A`	Account; marks an account ID (shareholder) entry.
`G`	Resource Group; marks a resource group entry.

U	User; marks a user entry.
Ba	Bad Account. (Severity: ERROR) An entry is being used as an account ID without the SHAREHOLDER flag set. To enable this flag, set the shflags field in the UDB to 01000000; this marks the entry as an account ID. (See the udbgen(8) man page for more information).
Bf	Bad flag. (Severity: ERROR) For a resource group or account ID entry, this error indicates that the entry's share flags are incorrect for its position in the resource group chain (for example, if the entry appears to be used as a resource group but does not have the NOTSHARED flag set). For a user entry, this error indicates that either that the entry references a resource group when Share by Account mode is enabled, or that it references an account ID when Share by User mode is enabled. The referenced entry will be marked with the Bg flag.
Bu	Bad UID. (Severity: ERROR) The UID assigned to this UDB entry is greater than the maximum UID allowed by the system.
Mg	Maximum groups exceeded. (Severity: ERROR) The level of this resource group exceeds the maximum level allowed in the share tree hierarchy (as defined by MAXGROUPS in the shconsts structure, or as set by the -G option).
Bg	Bad group. (Severity: ERROR) This error indicates that the entry is being used as a resource group but does not have the NOTSHARED flag set. The user entries referencing this resource group will be marked with the Bf status. To correctly set the NOTSHARED flag for a resource group entry, set the shflags field in the UDB to 040000. (See the udbgen(8) man page for more information).
Nc	No children. (Severity: WARN) This resource group does not have any user entries (no references to the resource group).
Ng	No group. (Severity: ERROR) If this error appears for a resource group entry, it indicates that a user entry references this group as its resource group (or, if Share by Account mode is enabled, in its acid list) but the entry does not exist in the UDB. If this error appears for a user entry, it indicates that the user entry references an undefined resource group.
Nr	No root. (Severity: ERROR) Indicates that this resource group chain is circular and does not start at _ROOT_.
Rl	Root link. (Severity: WARN) A user or account ID entry is referencing _ROOT_ directly. There should be at least one level of resource group or shareholder entries between _ROOT_ and the user or account ID entries.
Zs	Zero shares. (Severity: WARN) A resource group or user entry does not have any shares. This problem can prevent login of the affected entry.

Error Display

The `shrtree` error display is produced by the `-e` option. The column headings for the error display and their meanings are as follows:

Heading	Description
`Type`	Severity of the problem; either `ERROR` or `WARN`. Problems of type `ERROR` can prevent login.
`Name`	Name of the entry in the UDB.
`ID`	ID number of the entry in the UDB.
`Status`	Numeric flag to indicate the type of error.
`Description`	Flag to indicate the type of error, with a brief description of the problem. See the "Resource Group Hierarchy Display" subsection for a list of the available flags and their meanings.

EXAMPLES

Example 1: The following example shows sample output for `shrtree` with no options (short form report). The system is running Share by User mode, with a maximum of four levels in the fair-share hierarchy. Note that in this example, the `Serv` entry has warning flag `Nc`; `Unknown` has warning flag `Zs`; and `Country`, `Region`, and `TechOps` have flags `Nc` and `Zs`.

```
$ shrtree

DISPLAY OF SHARE TREE
_____________________
UDB path:      DEFAULT
Analyzed:      By UID
Format:        Groups only
Maxgroups:     4
Node:          ALL
Group Count:   15
Account Count: 0
User Count:    1370
Warnings:      9
Errors:        0

Warning Count: 4     (Nc) Group has no references
Warning Count: 1     (Zs) User has zero shares
Warning Count: 4     (Zs) Group has zero shares

                                    System Group  System
Lv        Name                 ID   Share  Share  Usage  Status   Flags
__ ____________________________ _____ ______ ______ ______ ________ ________
 0 _ROOT_                        0 100.0% 100.0% 100.0% G           40000
 1   Demos                    8367 100.0% 100.0%   0.0% G           40000
 2     Serv                   8001 100.0% 100.0%   0.0% G|Nc        40000
 1   System                   8389   0.0%   0.0%   0.0% G           40000
 2     Admin                  8306   0.0%   2.9%   0.0% G           40000
 1   Unknown                  8393   0.0%   0.0%   0.0% G|Zs        40000
 1   Users                    8395   0.0%   0.0%  28.8% G           40000
 2     CCN                    8354   0.0%   4.8%   0.2% G           40000
 2     Country                8359   0.0%   0.0%   0.0% G|Nc|Zs     40000
 2     HardDev                8372   0.0%   4.8%   0.0% G           40000
 2     Intl                   8374   0.0%  14.3%   0.0% G           40000
 2     Mktg                   8381   0.0%  28.6%   0.1% G           40000
 2     Region                 8385   0.0%   0.0%   0.0% G|Nc|Zs     40000
 2     SoftDev                8386   0.0%  47.6%  28.5% G           40000
 2     TechOps                8390   0.0%   0.0%   0.0% G|Nc|Zs     40000
```

Example 2: The following example shows the error display for the same system (`shrtree` with the `-e` option). Only the entries with errors are displayed.

```
$ shrtree -e

DISPLAY OF SHARE TREE
_____________________
UDB path:      DEFAULT
Analyzed:      By UID
Format:        Groups only
Maxgroups:     4
Node:          ALL
Group Count:   15
Account Count: 0
User Count:    1370
Warnings:      9
Errors:        0

Warning Count: 4     (Nc) Group has no references
Warning Count: 1     (Zs) User has zero shares
Warning Count: 4     (Zs) Group has zero shares

  Type      Name    ID    Status  Description
_________ ________ _____ ________ ___________________________________
**WARN*** Serv      8001       10 Nc: Group has no references
**WARN*** Unknown   8393     1000 Zs: Group has zero shares
**WARN*** unknown     12     1001 Zs: User has zero shares
**WARN*** Country   8359     1010 Nc: Group has no references
**WARN*** Country   8359     1010 Zs: Group has zero shares
**WARN*** Region    8385     1010 Nc: Group has no references
**WARN*** Region    8385     1010 Zs: Group has zero shares
**WARN*** TechOps   8390     1010 Nc: Group has no references
**WARN*** TechOps   8390     1010 Zs: Group has zero shares
```

FILES

`/usr/include/sys/share.h`	Definition of `shconsts` structure
`/usr/include/udb.h`	Definition of fields in the UDB

SEE ALSO

shradmin(8)

shrview(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

lnode(5), share(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

shutacct – Shuts down accounting

SYNOPSIS

/usr/lib/acct/shutacct ["*reason*"]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The shutacct shell script turns off process and daemon accounting and is usually called during a system shutdown in /etc/shutdown. shutacct invokes the turnacct(8) and turndacct(8) commands with the off operand.

The shutacct shell script accepts the following operand:

"*reason*" Specifies that *reason* enclosed in double quotation marks should be appended to /etc/wtmp, the login and logoff summary file. A maximum of 11 characters is allowed. By default, the reason is specified by the ACCTOFF parameter in the accounting configuration file /etc/config/acct_config.

FILES

/etc/config/acct_config	Accounting configuration file
/etc/wtmp	Login/logoff summary
/usr/adm/acct/day	Directory that contains current accounting files

SEE ALSO

acct(8), acctsh(8), csa(8), shutdown(8), turnacct(8), turndacct(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

shutdown – Terminates all processing

SYNOPSIS

/etc/shutdown [-y] [*grace*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The shutdown utility is part of the operation procedures on systems based on the UNIX system. As implemented in the UNICOS system, the shutdown utility uses the killall(8) command to terminate all currently running processes belonging to sessions other than that of the person who invoked shutdown. The shutdown utility is designed to bring the system from multiuser mode to single-user mode in an orderly and cautious manner.

The utility is designed to interact with the person who invoked shutdown, unless the -y option is specified.

The shutdown command accepts the following options:

-y Quiet mode (no interactive session); executes shutdown without waiting for responses from the person who invoked shutdown.

grace Specifies the grace period, in seconds, before the shutdown will begin. Default is 60 seconds. Unless the -y option is specified, the shutdown procedure may instruct you to perform some specific tasks or to supply certain responses before execution can resume.

The shutdown procedure goes through the following steps:

1. Checks for and reports on active guest systems. If there is an active guest system, asks whether you want to continue shutdown.

   A return to single-user mode in the host should not affect a multiuser guest, but a subsequent host dump or reboot will destroy the guest and probably will adversely affect its mounted filesystems. If possible, it is best to stop and release guests prior to host system shutdown.

2. If it exists, executes the /etc/shutdown.pre user exit.
3. All users logged on the system are notified by a broadcasted message to log off the system. If you did not specify the -y option, you may display your own message at this time. Otherwise, the standard file save message is displayed. The shutdown script then sleeps for *grace* seconds, to allow users to log out.
4. Shuts down daemons in the SYS1 and SYS2 groups (defined in the /etc/config/daemons file), using the sdaemon(8) command.
5. Terminates remaining processes, using the killall(8) command.

6. Releases logical device cache, using the `ldcache`(8) command. This ensures that all logical device buffers are flushed.
7. Shuts down all configured network interfaces (defined in the `/etc/config/interfaces` file), using the `ifconfig`(8) command.
8. If it exists, executes the `/etc/shutdown.pst` user exit.
9. Unmounts all file systems.
10. Enters single-user mode.

After the `shutdown` procedure completes, the system is in single-user mode and is essentially equivalent to the state of the system after a reboot procedure.

User Exits

To allow each site to modify the shutdown procedure, `shutdown` provides two user exits: `/etc/shutdown.pre` and `/etc/shutdown.pst`. The user exits are described as follows:

`/etc/shutdown.pre` This is the first user exit in `shutdown`. If an executable named `/etc/shutdown.pre` exists, it will be executed by `shutdown`.

At this point, nothing has been done in shutting down the system. All daemons are still running, all file systems are still mounted, and all users are still active and unaware that the shutdown processing has begun. This exit could be used to verify the user's permission to run the shutdown script or to run some system cleanup routines.

Shutdown will check the return status from the `shutdown.pre` program. If the return status is nonzero, and the `-y` option was not specified, the user will be queried as to whether or not to continue the shutdown processing. At this point, the shutdown can be stopped without any effect on the system.

`/etc/shutdown.pst` This is the second (and last) user exit in `shutdown`. If an executable named `/etc/shutdown.pst` exists, it will be executed by `shutdown`.

At this point, all processes (users and daemons) have been terminated, but all of the file systems are still mounted. This is virtually single-user mode, except that the file systems are still mounted.

NOTES

If this command is installed with a privilege assignment list (PAL), a user who is assigned the following privilege text upon execution of this command is allowed to perform the action shown:

Privilege Text	Action
`valid`	Allowed to use this command.

If this command is installed with a PAL, a user with one of the following active categories is allowed to perform the action shown:

Active Category	Action
`system`, `secadm`, `sysadm`, `sysops`, `diagadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

Caution should be used in choosing what is executed in `/etc/shutdown.pst`. At this point, the various log daemons have been terminated. Since the log daemons are not available to free up space, the file systems containing the various system log files could fill up.

MESSAGES

The most common error message is *device busy*. This message is issued when a particular file system could not be unmounted.

SEE ALSO

`brc`(8), `ifconfig`(8), `init`(8), `killall`(8), `ldcache`(8), `mount`(8), `sdaemon`(8), `slogdemon`(8)

`privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

General UNICOS System Administration, Cray Research publication SG–2301

NAME

slogdemon – Security event logging daemon

SYNOPSIS

/etc/slogdemon

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The security auditable event logging daemon, slogdaemon, collects security-relevant audit records from the operating system by reading the character special pseudo device /dev/slog (defined by SLG_DEV in sys/slog.h) and places them in the security audit log file. At daemon startup time, the security audit log file is created if it does not already exist. Security audit records are appended to this file.

The directory in which the security audit log file exists is defined by the SLG_DIR configuration option. The name of the security audit log file is defined by the SLG_FILE configuration option. The default path name of the security audit log file is /usr/adm/sl/slogfile. The file's maximum size is defined by the SLG_MAXSIZE configuration option. The default value for the maximum size is 8192000 bytes. When the security audit log file reaches its maximum size, the daemon renames the file according to the following naming convention:

SLG_DIR/s.*yymmddhhmmss*

The s. prefix of the renamed file is defined by the SLG_FPREFIX configuration option. The suffix of the renamed file identifies the date and time at which the file exceeded its maximum size. After the file has been renamed, a new security audit log file is created by using the path name of the original security audit log file.

Analysis of security audit records can be performed using the reduce(8) utility. The daemon is stopped during system shutdown after all user processes have been terminated.

NOTES

Processes are placed into a sleep state when the operating system audit record buffer becomes greater than 50% full. If the buffer becomes full, the system panics to ensure that no security audit records are lost.

The daemon should be started for all valid init levels by an entry in /etc/inittab (see inittab(5)), or for multiuser mode by an entry in /etc/rc (see brc(8)).

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the action shown:

Active Category	Action
system, secadm	Allowed to start the daemon.

If PRIV_SU is enabled, the super user is allowed to start the daemon.

FILES

/dev/slog	Source of security event records.
/slogdemon_err	Contains a copy of the error messages sent out by the current slogdemon process.
/usr/adm/sl/slogfile	Repository for security event records.
/etc/slogd.pid	Contains the process ID of the currently-running version of the security log daemon. Used by shutdown(8) to exclude the security log daemon from receiving the SIGTERM signal and to kill the user processes.

SEE ALSO

brc(8), killall(8), reduce(8), shutdown(8)

kill(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

inittab(5), slog(4), slrec(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

smpmon – Interactively monitors the physical-layer External Semaphore Device

SYNOPSIS

/etc/smpmon [-s *SFS-Arbiter*]

IMPLEMENTATION

Cray PVP systems (except CRAY J90 series and CRAY EL series)

DESCRIPTION

The smpmon command is used to interactively monitor the physical-layer External Semaphore Device. esdmon uses the special device /dev/smp to interface to the device driver.

Executing the smpmon command requires super-user permissions.

-s *SFS-Arbiter* Specifies the name of the Shared File System (SFS) arbitration service to be monitored.

The *SFS-Arbiter* name must match one of the valid entries in the /etc/config/sfs configuration file.

smpmon always operates in full-screen mode.

At the end of each repeat interval, which is initially set to 2 seconds, keyboard input is sampled for one of the following commands:

Command	Description
^l	Clear and refresh the display.
+	Increase the repeat interval by 1 second.
-	Decrease the repeat interval by 1 second.
h,?	Display the *help* menu.
q	Quit, exit the program.

FILES

/dev/smp Character Device Special used to interface to the channel-level device driver for the External Semaphore Device

NAME

smtd – FDDI station management daemon

SYNOPSIS

/etc/smtd [-z *device*] [-s *sec*] [-n *sec*] [-v *sec*] [-c *file*] [-u] [-p *port*] [-d] [-i *infc*]

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

DESCRIPTION

The smtd is the daemon process that handles all frame-based station management functions for the Cray FDDI network interface. Cray FDDI supports Station Management (SMT) version 6.2, which is comprised of two major components: frame-based SMT and Connection Management (CMT). Frame-based SMT performs three major functions: neighborhood notification (NN), duplicate address detection (DAD), and response to SMT request frames from the FDDI network such as neighbor information request frames (NIF), and station information request frames (SIF) requests. (For an explanation of Connection Management, see fddi(4).)

To provide a mechanism for other processes such as utility programs (see fddimap(8)) and network management applications (see snmpd(8)), smtd provides a UDP-based socket utility over which requests and replies can be sent between processes to obtain information about the FDDI network. The smtd acts as a type of proxy to these other processes. The socket utility is discussed in detail in the Socket Utility section of this man page.

The command-line options are as follows:

-z *device* Specifies the UNICOS character special device name to open for communications on the FDDI network. Default is /dev/fddi0/smt.

-s *sec* Specifies the status report threshold (in seconds) for sending SRF announcements. If 0 seconds is specified, the status reporting protocol is disabled. The default is 2 seconds, the minimum is 2 seconds, and the maximum is 32 seconds.

-n *sec* Specifies the interval (in seconds) for sending NIF requests during the neighbor notification protocol (NNP). The default is 5 seconds and the minimum is 2 seconds.

-v *sec* Specifies the number of seconds that is allowed between NIF requests from upstream and downstream neighbors before considering the UNA and DNA not valid. The default is 228 seconds.

-c *file* Specifies the name of the configuration file to use.

-u Enables communications across the socket utility. The default is disabled.

-p *port* Sets the UDP port number of the socket utility. The default is 3000. See also fddimap(8).

-d Enables debug mode; verbose output.

`-i` *infc* Specifies the interface ordinal of the FDDI network that will be managed by this copy of `smtd`. If the Cray Research system has more than one FDDI interface, and each interface is attached to a different ring, copy of `smtd` must exist for each ring.

Configuration File

Any parameters that you can set on the command line can also be set in a more permanent fashion by placing them in a configuration file; the exception is the `-d` debug flag, which can be entered only on the command line. The following shows the format of the configuration file. Only parameters that will change must appear in the file; others are set to their default values.

```
#
#    These three lines are comment lines.
#
device=path
interface=ordinal
udpport=portnumber
tnotify=seconds
tnnout=seconds
srthreshold=seconds
```

You can place comment lines anywhere in the file if the comment starts in column 1 of a line. Comments are designated by a # character in column 1.

Neighbor Notification Protocol

The `smtd` performs neighborhood notification (NN) by sending NIF request frames to the broadcast address by using next station addressing (SMT NSA) every `T_NOTIFY` seconds. The NSA mode of addressing lets only a single station on the FDDI ring copy the frame, even though it is addressed to the broadcast address. In this way, it should be the station's downstream neighbor (DNA) that copies the frame. The DNA sets its UNA to our address and sends a response to the NIF request. The DNA is set to the MAC address of the station that responds to our NIF request. All stations on the FDDI ring perform this same protocol; therefore, at some point in time, the same type of NIF request is received from our upstream neighbor (UNA). When received, set your UNA to the MAC address of the sender of that NIF request before replying to the frame. When the UNA gets your response, it sets its DNA to your address. This process continues forever and the stations on the FDDI ring eventually learn their UNAs and DNAs.

Another part of the Neighbor Notification Protocol involves the timing out of the receipt of NIF requests from our UNA and NIF responses from our DNA. If we do not hear from our neighbors for a period longer than `T_NN_Out`, we set the appropriate neighbor's address to the FDDI unknown address to indicate that the contact with that neighbor was lost.

Duplicate Address Detection Protocol

The `smtd` also performs duplicate address detection (DAD) by monitoring the NIF request frames that arrive at this station. Because our hardware can copy an NIF request that we send and one that is sent from our UNA, we can determine whether that may be a duplicate MAC address on the ring. By receiving our own NIF request frames back, we can check the Transaction ID of the frame to ensure that it is the same Transaction ID that we sent; if it is not, a station on the ring must have the same MAC address as ours.

Another way you can check for duplicates is by ensuring that when we receive a NIF response frame to one of our NIF request frames, the Address Recognized (A bit) is not set. If it is, this would also indicate a duplicate address, because the A bit being set indicates some other station on the ring has recognized the destination address of the response frame as their own.

If the `smtd` detects a duplicate address condition on the FDDI ring, it notifies the FDDI driver (`fd.c`) by using an `ioctl` request indicating that the duplicate address test results have changed and that the new state is `failed`. When the driver receives this notification, it will first notify the CMT process on the channel adapter of the duplicate address condition. The CMT process causes the FDDI interface to disconnect from the ring. The driver also marks the logical link control (LLC) services unavailable, which prevents any further traffic from leaving the Cray Research system. When the ring becomes operational again, the CMT process notifies the FDDI driver and LLC services are reenabled. During the time the ring is not operational, the `smtd` continues to try to send the NIF request frames. These attempts will fail because the ring is down, but will not fail after the ring comes back up.

Status Reporting Protocol

The `smtd` also supports the Status Reporing Protocol of the SMT standard. When one of several predefined conditions exists or events occur, the `smtd` sends a status report frame (SRF) to the ANSI-owned IEEE SRF multicast address (`80:01:43:00:80:00`). Each SRF frame contains all of the currently active conditions and any new events that have occurred since the last SRF was sent. The protocol limits the frequency of sending the SRF announcements to a maximum of one frame every 2 seconds. The protocol also defines a report threshold, which can vary from a minimum of 2 seconds to a maximum of 32 seconds. This threshold is the amount of time the FDDI station can wait before sending an SRF frame when conditions exist or events occur that require it. Setting this time to a higher value is useful if conditions persist or events occur at an unusually high rate because a higher value minimizes the number and frequency of SRF frames added to network traffic load.

Socket Utility

To provide a mechanism for other processes that want to obtain information about FDDI stations and FDDI station management information, `smtd` supports a UDP-based socket over which it can receive requests and send replies. You can use this socket for such things as sending SMT request frames, receiving SMT response frames, obtaining the MAC addresses of upstream and downstream FDDI stations, and obtaining a copy of all FDDI MIB variables for the station.

After a process has opened a utility socket connection to the `smtd`, it can then send requests to the daemon over the socket and receive the results of the requests over the socket. The format of the structures and the contents of the structures that are used across the UDP socket is specified in the `smtd` header files `defines.h` and `typedefs.h`. The following shows the format of the socket structures (for a more accurate definition of the structures, see the header files themselves):

```
/*
 *      Request-Response packet format
 */
typedef struct {
        QueueElement                    qe;
        int                             len;
        int                             fromlen;
        struct sockaddr_in              from;
        union {
                int     request;
                int     response;
        } rr;
        union {
                util_xmtframeparms      xmt;
                util_rcvframe           rcv;
                util_macs               macs;
                util_mibvar             mib;
                util_cancel             cancel;
        } arg;
} util_packet;

/*
 *      Transmit Frame packet format
 */
typedef struct {
        uchar   fc;
        uint    dest;
        uchar   class;
        uchar   type;
        uint    tid;
} util_xmtframeparms;

/*
 *      Receive Frame packet format
 */
typedef struct {
        QueueElement    qe;
        int             age;
        int             hold;
        int             tid;
        int             len;
        smt_frame       frame;
} util_rcvframe;
```

```
/*
 *      Get MAC addresses packet format
 */
typedef struct {
        uint    ME;
        uint    UNA;
        uint    DNA;
} util_macs;

/*
 *      Get FDDI MIB variables packet format
 */
typedef struct {
        uint    dummy;          /* To be determined */
} util_mibvar;
```

The `qe`, `len`, `fromlen`, and `from` fields of the `util_packet` are all used internally by `smtd` and do not have to be manipulated by the user.

The `rr` field of the `util_packet` is used for request and response codes.

The `arg` field holds the actual data associated with the request or response.

When requests are compiled, they can then be sent to `smtd` by using a `sendto`(2) system call. You can obtain the response to the request by using a `recvfrom`(2) system call.

To test the response for success, compare the `rr.response` field to `UTIL_RSP_OK`.

The following section describes the way in which the socket utility requests are formed and the contents of the responses for all of the different types of operations across the socket.

To Transmit an SMT Request Frame

- Set the `rr.request` field to `UTIL_REQ_XMT_FRAME`.
- Set the `arg.xmt.fc` field to the desired FDDI frame control.
- Set the `arg.xmt.dest` field to the destination station's MAC address (in MSB form).
- Set the `arg.xmt.class` field to the frame class.
- Set the `arg.xmt.type` field to the frame type.
- Set the `arg.xmt.tid` to the Transaction ID of the user's request. This ID should be a number that is unique to each request that is sent by the process and unique to the process itself. Because many programs can communicate with the `smtd` over the socket simultaneously, this is the only mechanism that is available to maintain order to all of the various requests and responses that will be seen by the `smtd`. The `smtd` uses the Transaction ID to differentiate response frames that it receives due to request frames that it sends, from those response frames that are received because of request frames sent by users communicating over the socket utility.

To Receive an SMT Response Frame

- Set the `rr.request` field to `UTIL_REQ_RCV_FRAME`.
- Set the `arg.rcv.tid` to the Transaction ID of the request that generated this SMT response frame. (This must be the same Transaction ID that was in the transmit request.)
- Set the `arg.rcv.hold` to the number of seconds the user wants to wait for a response frame to arrive. If no response frame arrives with the desired Transaction ID within this much time, the request will fail with a `UTIL_RSP_RCV_TIMEOUT` error. When a response frame with the designated Transaction ID is received, it will be copied into the `arg.rcv.frame` field of the request, the length of the frame will be placed into the `arg.rcv.len` field, and the response delivered to the user. If no receive request is posted, frames that arrive are queued. These frames are held until either a receive request with the proper Transaction ID is received or a time-out occurs. If the time-out occurs before the receive request is received, the frame(s) will be discarded.

To Cancel a Receive Request

- Set the `rr.request` field to `UTIL_REQ_CANCEL_RCV`.
- Set the `arg.cancel.tid` to the Transaction ID of the request to cancel.

To Obtain the MAC Addresses of This Station and Its Neighbors

Set the `rr.request` field to `UTIL_REQ_GET_MACS`. The addresses that are returned are in MSB form.

Get FDDI MIB Variables

This request is not yet defined.

SEE ALSO

`fddimap`(8), `xfddimap`(8)

`fddi`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

The ANSI documents for FDDI:

FDDI MAC (Media Access Protocol) Specification (FDDI-MAC), document number X3.139–1987, November 5, 1986

FDDI PHY (Physical Layer Protocol) Specification (FDDI-PHY), document number X3.148–1988, June 30, 1988

FDDI PMD (Physical Medium Dependent) Specification (FDDI-PMD), document number X3.166–1990, September 28, 1989

FDDI SMT (Station Management) Specification (FDDI-SMT), document number X3T9.5/84-49 Rev 6.2, May 18, 1990

Other documents related to FDDI:

RFC 1188 Proposed standard for the transmission of IP datagrams over FDDI networks. October 1990: D. Katz

Logical Link Control Specification (802.2 LLC), document number 802.2–1985, July 16, 1984

NAME

snmpd – Simple network management protocol agent/server for Cray Research systems

SYNOPSIS

/etc/snmpd [-b *size*] [-d] [-D] [-P *portnumber*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The snmpd command is a server that acts as a management agent, implementing the Simple Network Management Protocol (SNMP). After receiving a message, it authenticates the request, attempts the operation, and then returns a response.

The managed objects that snmpd manipulates are defined in the snmpd.defs file, which is kept in the system administrator's area. These objects conform to the Internet standard management information base (MIB), which is defined in RFC 1213 as MIB-II. The rules used for naming and describing objects are taken from the Internet standard structure of management information (SMI), which is defined in RFC 1155 and enhanced by RFC 1212.

The snmpd command reads the /dev/kmem file to obtain most objects. Some objects can be specified in the /etc/snmpd.conf configuration file instead, which can be read only when the daemon starts. The /etc/snmpd.conf configuration file provides a mechanism to list all management stations that can access snmpd.

The snmpd command accepts the following options and operands:

-b *size* Specifies the "too big" packet size. Using this option determines the largest packet size that snmpd will accept.

-d Turns on debugging. This option specifies that the daemon will not be run in the background; it enables debug messages to print to the screen.

-D Turns on debugging and packet dumps. Using this option enables the -d options and also prints out each packet that is received and sent.

-P *portnumber*
Listens on port *portnumber*. Using this option overides the default port 161 to which snmpd listens for requests.

Configuration

The /etc/snmpd.conf file contains the following customization directives:

`community` *name* [*address*] [*access*] [*view*]

Defines an SNMP community called *name* and indicates the level of *access*.

The *address* token is either a host name, an IP address, or a network address (using dot notation). If *address* is present and you use a value other than 0.0.0.0, incoming messages that claim to belong to the specified community must come from this address.

If the *access* token is present, its value is `readOnly`, `readWrite`, or `none`; it defaults to `readOnly`.

If the *view* token is present, it is an object identifier that names the corresponding view of MIB objects that this community can access; otherwise, the view contains all variables known to the agent.

`logging` *attribute=value*

Sets the logging parameters according to the values specified for attributes. Attributes are as follows:

`file`	Specifies the file name for the log, whose value is interpreted relative to the ISODE logging area unless the value starts with a slash.
`size`	Specifies the maximum file size (an integer value in Kbytes) that the log should be allowed to grow.
`slevel`	Specifies which one of the following events should be logged: `none`, `fatal`, `exceptions`, `notice`, `trace`, `pdus`, `debug`, or `all`.
`dlevel`	Specifies the events (listed under `slevel`) that should not be logged.
`sflag`	Specifies which one of the following logging options should be enabled: `close` (to close the log after each entry), `create` (to create the log if it does not already exist), `zero` (to reset the log if the size is exceeded), and `tty` (to log events to the user's terminal in addition to the file).
`dflags`	Specifies the logging options (listed under `sflag`) that should be disabled.

`trap` *name address* [*view*]

Defines a trap sink for the SNMP community called *name*, on the indicated address, for the indicated view. *address* can be a host name, an IP address, or a network address (using using dot notation). If you omit *view*, a view is not named for the trap sink. Note: UDP must be able to reach the trap sink.

`variable` *name value*

Sets the *name* variable to the indicated *value*. Following are the variables that can be set:

`sysDescr`	Can be set to a string value that describes the management agent.
`sysObjectID`	Can be set to an object identifier value that identifies this system.
`sysContact`	Can be set to a string value that identifies the person who is responsible for the node.

`sysName` Can be set to a string value that gives an administratively assigned name for the node.

`sysLocation` Can be set to a string value that describes the location of the node.

`sysServices` Can be set to an integer that describes the services that are offered by the node.

`snmpEnableAuthTraps`
Can be set to either `enabled` or `disabled`, which enables or disables the generation of `authenticationFailure` traps, respectively.

See RFC 1213 for a more thorough explanation of these objects.

`variable interface` *name attribute=value...*
Sets attributes for the specified interface, *name*. The *name* token is the same interface name that results from the `netstat -i` command. You can set the following attributes for each interface:

`ifType` Can be set to an integer value that identifies the type of interface.

`ifSpeed` Can be set to an integer value that describes the speed of the interface.

See RFC 1213 for a more thorough explanation of these objects.

`view` *name* [*subtree* ...]
Determines the part of the MIB that can be accessed. *name* specifies for which view the MIB is being accessed. *subtree...* specifies the MIB variables that can be accessed. If no subtrees are listed, the view contains all variables known to the agent.

The following parameters are supported for compatability with previous versions of the configuration file.

`readall`;
Allows any management station to access the agent.

`community` *community_name address/hostname type*;
Specifies additional options for a community. The *community_name* argument specifies the name of a community (in ASCII); *address/hostname* is either the Internet address (in dot notation format) or the name of the host; *type* can be one of the following values:

`read-only` Allows read-only access

`traps` Specifies the hosts to which `trap` messages will be sent

A value of `0.0.0.0` in the *address/hostname* field allows any host in a community to access `snmpd`.

`traps enabled`;
Tells `snmpd` to send traps. You must define a community to specify the management station to receive traps.

`authentication_traps enabled`;
Sends authentication traps when an unknown community attempts to query `snmpd`. If you specify `readall`, no authentication traps are generated. If an unknown host tries to use a valid community, an authentication trap is sent.

Debug Operation

To enter debug mode, specify the `-d` or `-D` option. In debug mode, all logging activity is displayed on the user's terminal in verbose mode. In addition, when you specify the `-D` option, each SNMP packet that the agent receives and sends is dumped in hexadecimal format.

You can use the `-b` option to specify the maximum message size that is supported by the daemon. (This is useful for testing how management stations recover from `tooBig` errors.)

You can use the `-P` option to specify a port other than 161. You can use the same option with the SNMP commands to allow `snmpd` to be debugged without affecting the system `snmpd`.

NOTES

The names of the objects in the `snmpd.defs` file are case sensitive. This was necessary to improve the efficiency of the hashing algorithm used for object lookup.

FILES

`/etc/services`	Network service name database
`/etc/snmpd.conf`	Configuration file
`/etc/snmpd.defs`	MIB definitions
`/etc/snmpd.pid`	Daemon PID file
`/usr/spool/osi/snmpd.log`	Log file

SEE ALSO

RFCs 1155, 1212, 1213, and 1157

NAME

snmproute – Performs route tracing with the Simple Network Management Protocol

SYNOPSIS

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

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The snmproute command 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 (see 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

spaudit – Changes security auditing criteria

SYNOPSIS

/etc/spaudit [-c] [-e *enableopts*] [-d *disableopts*] [-l] [-s]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The spaudit command allows an appropriately authorized user to make a change in the security audit log edition. On a UNICOS system, it also allows an appropriately authorized user to change the selection of auditable events when the UNICOS operating system is running.

To change the security audit log edition automatically at specified time intervals, initiate a cron job that executes the spaudit -c command. If a security audit log exceeds its threshold before an edition change is requested, the security audit log daemon automatically creates a new edition of the security audit log. This prevents the possibility of an edition of the security audit log overflowing its disk space and causing the UNICOS operating system to panic.

On a UNICOS system, the spaudit command can be used in either single-user or multiuser mode to enable or disable logging options. This command cannot be used to both enable and disable security logging options at the same time. If both the -e and -d options are specified, an error message is issued and no action is taken. Any changes in logging options are recorded in the security audit log.

The spaudit command accepts the following options:

-c Forces a change in the security audit log edition.

-e *enableopts*
Enables the specified logging option(s).

-d *disableopts*
Disables the specified logging option(s).

-l Lists and describes the valid logging options.

-s Lists the state (on or off) of valid logging options.

The following list contains valid logging options:

Option	Description
all_nami	All mkdir, rmdir, link, and rm calls
all_rm	All remove requests
all_valid	All access requests
audit	All security auditing criteria changes

`chdir`	All change directory requests
`config`	All UNICOS configuration changes
`crl`	Cray/REELlibrarian activity
`dac`	Discretionary access control changes
`discv`	Discretionary access violations
`filexfr`	All file transfer requests
`ipnet`	All IP layer activities
`jend`	End-of-job
`jstart`	Start-of-job
`linkv`	All link (`ln`) violations
`mandv`	Mandatory access violations
`mkdirv`	All make directory (`mkdir`) violations
`netcf`	Network configuration changes
`netwv`	Network violations
`nfs`	All NFS activity
`nqs`	NQS activity
`nqscf`	NQS configuration changes
`object_path`	Object's full path name on accesses
`operator`	Operator actions
`physio_err`	Currently not used
`priv`	Use of privilege
`removev`	All remove violations
`rmdirv`	All remove directory (`rmdir`) violations
`secsys`	All security system call requests
`setuid`	All setuid requests
`shutdown`	System shutdown requests
`startup`	System start-up requests
`state`	Defines if security audit log is on (`1`) or off (`0`)
`sulog`	All `su` attempts
`tapes`	Tape activities
`time_change`	System time change

`trust`	Trusted process activity
`user`	User name for failed login attempts

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the action shown:

Active Category	**Action**
`system`, `secadm`	Allowed to use this command.

If `PRIV_SU` is enabled, the super user is allowed to use this command.

EXAMPLES

Example 1: The following is an example of a `cron` entry that allows `spaudit` to execute at midnight each night to force a security audit log edition change:

```
crontab
0 0 * * * spaudit -c
```

To interactively force a security audit log edition change, execute the `spaudit -c` command.

Example 2: The following example shows the resultant display when using the `-l` (which lists the valid logging options and `-s` (which lists the state of the options) options:

```
$ spaudit -l -s
state           ON      security auditing state
all_nami        OFF     all nami requests
all_valid       OFF     all valid accesses
all_rm          OFF     all rm requests
audit           ON      audit criteria changes
chdir           ON      chdir requests
config          ON      config changes (UNICOS)
crl             OFF     Cray/REELlibrarian
dac             ON      discretionary access control changes
discv           ON      discretionary violations
filexfr         ON      file transfer requests
io_err          OFF     I/O errors
ipnet           ON      IPnet layer activities
jend            ON      job end
jstart          ON      job initiation
linkv           ON      link violations
mandv           ON      mandatory violations
mkdirv          ON      mkdir violations
netcf           ON      network config changes
netwv           ON      network violations
nfs             OFF     NFS activity
nqs             ON      NQS activity
nqscf           ON      NQS config changes
operator        ON      operator actions
object_path     ON      object path tracking
priv            OFF     privilege use
removev         ON      remove violations
rmdirv          ON      rmdir violations
secsys          ON      security syscalls
shutdown        ON      system shutdown
startup         ON      system startup
time_change     ON      system time change
setuid          ON      setuid requests
sulog           ON      su requests
tapes           OFF     tape activity
trust           ON      trusted process activity
user            OFF     user password on login fail
```

Example 3: The following example shows the use of the `-d` option, which disables the `nqs` and `nqscf` options. The `-s` option is then executed to show the state of the options:

```
$ spaudit -d nqs,nqscf
$ spaudit -s
state           ON
all_nami        OFF
all_valid       OFF
all_rm          OFF
audit           ON
chdir           ON
config          ON
crl             OFF
dac             ON
discv           ON
filexfr         ON
io_err          OFF
ipnet           ON
jend            ON
jstart          ON
linkv           ON
mandv           ON
mkdirv          ON
netcf           ON
netwv           ON
nfs             OFF
nqs             OFF
nqscf           OFF
operator        ON
object_path     ON
priv            OFF
removev         ON
rmdirv          ON
secsys          ON
shutdown        ON
startup         ON
time_change     ON
setuid          ON
sulog           ON
tapes           OFF
trust           ON
user            OFF
```

Example 4: The following example shows the use of the `-e` option, which enables the `tapes`, `nqs` and `nqscf` options. The `-l` and `-s` options are then executed to show the description and the state of the options:

```
$ spaudit -e tapes,nqs,nqscf
$ spaudit -ls
state           ON      security auditing state
all_nami        OFF     all nami requests
all_valid       OFF     all valid accesses
all_rm          OFF     all rm requests
audit           ON      audit criteria changes
chdir           ON      chdir requests
config          ON      config changes (UNICOS)
crl             OFF     Cray/REELlibrarian
dac             ON      discretionary access control changes
discv           ON      discretionary violations
filexfr         ON      file transfer requests
io_err          OFF     I/O errors
ipnet           ON      IPnet layer activities
jend            ON      job end
jstart          ON      job initiation
linkv           ON      link violations
mandv           ON      mandatory violations
mkdirv          ON      mkdir violations
netcf           ON      network config changes
netwv           ON      network violations
nfs             OFF     NFS activity
nqs             ON      NQS activity
nqscf           ON      NQS config changes
operator        ON      operator actions
object_path     ON      object path tracking
priv            OFF     privilege use
removev         ON      remove violations
rmdirv          ON      rmdir violations
secsys          ON      security syscalls
shutdown        ON      system shutdown
startup         ON      system startup
time_change     ON      system time change
setuid          ON      setuid requests
sulog           ON      su requests
tapes           ON      tape activity
trust           ON      trusted process activity
user            OFF     user password on login fail
```

FILES

`/usr/adm/sl/slogfile`	Security log
`/usr/include/sys/slrec.h`	Format of security audit log record

SEE ALSO

`reduce`(8), `slogdemon`(8)

`slrec`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

spcheck – Performs security checks on a UNICOS system

SYNOPSIS

/etc/spcheck [-a] [-b] [-B] [-c] [-f *filesystem*] [-g] [-G] [-l] [-p] [-q] [-Q *file*] [-r] [-s] [-w]
/etc/spcheck [-a] [-b] [-B] [-c] [-g] [-G] [-l] [-p] [-q] [-Q *file*] [-r] [-s] [-w] [-u *user*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The spcheck command performs a variety of security-relevant checks on a UNICOS system. This command can be used only by an appropriately authorized user. spcheck makes administrative and logical file system checks according to the selected options. If you run spcheck with no options, the -a, -b, -g, -l, -p, -q, -s, and -w options are executed by default.

The -f and -u options cannot be used together.

The spcheck command accepts the following options:

-a Lists users who have administrative categories.

-b Lists all setuid and setgid files in the /bin, /usr/bin, /usr/lib, and /etc directories.

-B Performs the same function as the -b option, but uses the directory list in /etc/bincheck. The /etc/bincheck file is a one-record file and has the following format:

directory1 directory2 directory3 . . . directoryn

-c Runs the spfilck(8) program in check mode (with the -c option), using /etc/permlist as input to spfilck.

-f *filesystem*
Performs a security check on the logical *filesystem* (for example, /usr). This option should be used in conjunction with the -r, -s, and -w options, and it cannot be used with the -u option.

-g Reports users in groups root, adm, bin, and sys. This option also runs the grpck (see pwck(8)) command.

-G Performs the same function as the -g option but uses the group list in /etc/grpcheck. The /etc/grpcheck file is a one-record file and has the following format:

group1 group2 group3 . . . groupn

-l Reports infrequently used login IDs (users who have not logged in for at least 14 days and users who have not logged in for 180 days) and lists .profile and .cshrc files that are writable by anyone.

`-p` Reports users with duplicate user IDs, users who cannot change their passwords, and users whose passwords do not expire. This option also runs the `pwck`(8) command.

`-q` Reports all users who have accumulated 10 or more `su`(1) command failures in any one day, as logged in the `/usr/adm/sulog` file.

`-Q` *file* Performs the same function as the `-q` option, but it uses *file* instead of `/usr/adm/sulog` as the search log.

`-r` Reports files in the system that have the world read permission set (ignores files in `/tmp`, `/usr/tmp`, and `/usr/spool`). This option is to be used in conjunction with the `-f` and `-u` options.

`-s` Reports list of programs with either the setuid or setgid bits set. If you are root and you specify this option, `spcheck` also reports block/character special files not in `/dev`. This option is to be used in conjunction with the `-f` and `-u` options.

`-w` Reports files in the system that have the world write permission set (ignores files in `/tmp`, `/usr/tmp`, and `/usr/spool`). This option is to be used in conjunction with the `-f` or `-u` options.

`-u` *user* Performs security check on the specified *user*'s home directory. See the `-r`, `-s`, and `-w` options for security check options. This option cannot be used with the `-f` option.

NOTES

If this command is installed with a privilege assignment list (PAL), a user who is assigned the following privilege text upon execution of this command is allowed to perform the action shown:

Privilege Text	Action
`exec`	Allowed to use this command.

If this command is installed with a PAL, a user with one of the following active categories is allowed to perform the action shown:

Active Category	Action
`system`, `secadm`	Allowed to use this command.

If `PRIV_SU` is enabled, the super user is allowed to use this command.

FILES

```
etc/bincheck
etc/grpcheck
usr/adm/sulog
```

SEE ALSO

`grpck`(8), `pwck`(8), `spfilck`(8)

`chmod`(1), `privtext`(1), `su`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

General UNICOS System Administration, Cray Research publication SG–2301

NAME

spdev – Sets, clears, and displays objects

SYNOPSIS

```
/etc/spdev [-l minlvl] [-c mincmp] [-L actlvl] [-K actcmp] [-u maxlvl] [-v maxcmp] [-m] [-s]
[-p] devf ...
/etc/spdev -C [-p] devf ...
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The spdev command sets, clears, and displays the security labels on objects. spdev can be executed by anyone; kernel-level restrictions limit the actions available to users who are not appropriately authorized.

The spdev command accepts the following options:

-l *minlvl*	Sets the minimum level for the requested devices to *minlvl*. The default minimum level is 0. It is not possible to set the minimum level on files, so this option is ignored for nondevice objects.
-c *mincmp*	This option is provided for future use in setting minimum compartments on devices. Because the device labeling mechanism currently does not support a nonzero set of minimum compartments on devices, this option is recognized but not implemented.
-L *actlvl*	Sets the active level for single-level objects to *actlvl* and sets the active level for multilevel files. The active level on multilevel devices is always set to the maximum level by the kernel, so this option has no effect on multilevel device labels. The default active level is 0.
-K *actcmp*	Sets the active compartments for single-level objects to *actcmp* and sets the active compartments for multilevel files. The active compartments for multilevel devices are always set to the authorized compartments by the kernel, so this option has no effect on multilevel device labels. The default active compartment set is the empty set.
-u *maxlvl*	Sets the maximum level for the requested devices to *maxlvl*. For multilevel devices, the kernel also sets the active level to *maxlvl*. The default maximum level is 0. It is not possible to set the maximum level for files, so this option is ignored for nondevice objects.
-v *maxcmp*	Sets the authorized compartments for the requested devices to *maxcmp*. For multilevel devices, the kernel also sets the active compartments to *maxcmp*. The default authorized compartment set is the empty set. It is not possible to set the authorized compartments for files, so this option is ignored for nondevice objects.
-m	Sets the requested objects to multilevel mode. The default mode is single-level.

`-s` Sets the requested devices to the `ON` state. Setting the device state to `ON` enables nonprivileged access to single-level devices, and signifies to privileged software using multilevel devices that the device is now available for use. The default state is `OFF`. Once the state is set to `ON` (when using this option), the `-C` option must be used to clear the device and set its state to `OFF` again.

`-C` Sets the device labels on the requested objects to the default settings described for each of the previously described options.

`-p` Prints the device labels of the requested objects.

NOTES

When the path name supplied to the `spdev` command specifies a multilevel symbolic link (the name of a multilevel directory), the attributes are changed only on the root of the multilevel directory tree. In the case of access control lists (ACLs), this affects all subsequently created labeled subdirectories. In any case, the attribute change does not affect existing labeled subdirectories. To set attributes on the existing labeled subdirectories, you must specify the path names of the existing labeled subdirectory found in the root of the multilevel directory to the `spdev` command.

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`	Allowed to set security attributes of any object.
`sysadm`	Allowed to set security attributes of any object, subject to security label restrictions on the object's path. Shell redirected I/O is subject to security label restrictions.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to set security attributes for any object.

MESSAGES

`spdev` writes the following error messages to standard error:

`spdev: Stat failed on` *objects*
The device *devf* does not exist or user does not have access.

`spdev: Cannot secstat` *objects*
The user does not have the security privileges necessary to access the device.

`spdev: Invalid security level` or `Invalid compartments`
If the `MLS_OBJ_RANGES` configuration option is enabled, and the requested device security levels or compartment settings are not within UNICOS system security settings.

`spdev: Unable to set label on` *objects*
The user is not authorized to change the label on the object, or the requested label is not valid.

SEE ALSO

`spget`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`secstat`(2), `setdevs`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

General UNICOS System Administration, Cray Research publication SG–2301

NAME

spfilck – Checks and sets file ownership, access, and security parameters

SYNOPSIS

```
/etc/spfilck -c [-f file] [-q]
/etc/spfilck -s [-f file] [-q]
/etc/spfilck -m [-f file] [-q]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The spfilck command monitors and controls ownership, mode, and security parameters of files. It reads or creates a control file, /etc/permlist (see the -f option description), which defines the actual or desired file attribute (owner, mode, security) definition. spfilck can set from, check against, or make /etc/permlist.

The spfilck command operates in one of three modes:

- Checks the owner, group owner, access modes, security levels, security compartments, integrity class, categories, and integrity flags against a master list of files, and flags any discrepancies (with the -c option)
- Sets the owner, group owner, access modes, security levels, security compartments, integrity class, categories, and integrity flags for a list of files (with the -s option)
- Creates a master list of files (with the -m option)

The spfilck command accepts the following options. Only one of the -c, -s, and -m options may be used on a command line.

-c Checks file parameters against input from the /etc/permlist file. An input line from /etc/permlist is in the following format:

owner group mode level compartments class categories flags files

The components of the input line are the following:

owner	Owner name.
group	Group name.
mode	Octal representation of the user, group, and other permissions.
level	Decimal representation of the file's security level.
compartments	Either a list of valid compartments separated by commas or the word none.
class	Decimal representation of the file's integrity class. This value is no longer used.

categories Either a list of valid categories separated by commas or the word `none`. This value is no longer used.

flags Either a list of valid integrity flags separated by commas or the word `none`.

files A nonempty list of path names that are compared against the attributes previously described. The names can include the following substitution characters, whose meaning is analogous to their use in `ed`(1): `*`, `.`, `[`, and `]`.

`-s` Sets file attributes according to `/etc/permlist`. The input from `/etc/permlist` is in the same format as that for the `-c` option. `spfilck` attempts to set the file attributes for the *files*, the format of which is identical to the `-c` option.

`-m` Reads a list of files from standard input or from those specified with the `-f` option and writes the current file attributes to standard output in the `/etc/permlist` format to be used by `spfilck` with the `-c` and `-s` options. The input line used by `spfilck` is in the following format:

file [*file*]

This is identical to the corresponding `-c` file name definition.

`-f` *file* Causes the `-s` and `-c` options to use *file* instead of `/etc/permlist` for input against which to check or set file parameters, and `-m` to use *file* instead of standard input for reading file names.

`-q` Causes `spfilck` not to perform the character substitution described above in the explanation of the *file* field of the `-c` and `-s` options. This is useful when used with the `-c` and `-s` options, which use the output of an `-m` option run for input. The `-m` option output does not insert any special characters in the *file* field of its output, and this precludes the necessity of checking for them if `-c` or `-s` are used with `-m` output as their input. `-m` might contain '.', '*', '[', or ']' in its output. If it does, this means that the literal character is used in the actual path name. Therefore, you do not want character substitution (for example, 'file *' means that the file name is really 'file*.')

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	**Action**
`system`, `secadm`	Allowed to access all files.

On a UNICOS MLS system with `PRIV_SU` enabled, the super user is allowed to access all files.

`*` does not have the same meaning in `ed`(1) as in `sh`(1). For example, `/bin/*` matches `/bin/a` but not `/bin/aa`, and `/bin/.*` matches both `/bin/a` and `/bin/aa`. Standard output receives a list of any files that fail the checks performed.

FILES

`/etc/permlist`

SEE ALSO

`ed`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`udb`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

spmap – Reads or generates and writes a physical disk spare map

SYNOPSIS

spmap [-r] *special*
spmap [-w] *special*

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

DESCRIPTION

With no options selected, the spmap command reads physical device flaw information from standard input and writes to standard output a formatted spare sector map for the specified special device. The flaw information read is of the same format written to the standard output by the ift(8) command. The options can be used to read or write the spare sector map from or to a specified slice.

The spmap command accepts the following options:

-r Reads the spare sector map from the slice described by the character special file *special*. In this mode, no data is read from standard input.

-w Writes the spare sector map to the slice described by the character special file *special*. The spare sector map is generated from flaw information read from standard input. The -w option initializes only the spare sector map on the disk. The incore spare map is initialized only at device open time, normally by means of a file system mount.

The -r and -w options are mutually exclusive.

By convention, the special files describing the spare disk slices are described by the character special files in the /dev/spare directory. Typically, they are named after the physical device on which they reside.

EXAMPLES

Example 1: The following two commands write a spare sector map on device 0130.

```
ift /dev/ift/0130 | spmap -w /dev/spare/0130

cat /etc/aft/0130 | spmap -w /dev/spare/0130
```

Example 2: The following command reads the spare sector map from device 0130:

```
spmap -r /dev/spare/0130
```

SEE ALSO

`ift`(8), `mkspice`(8)

`pdd`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

spnet – Manages NAL, WAL, and IP Security Options (IPSO) maps stored in network security tables

SYNOPSIS

/etc/spnet [-f *filename*] [-d] [-n] [-q] [-v] *command* -*table* [*table_args*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The spnet command is a combined tool for managing security network configuration information. The spnet command currently maintains the network access list (NAL), the workstation access list (WAL), and the Internet Protocol Security Options (IPSO) mapping tables. It converts between external human-readable text and kernel-internal binary forms of the information. It adds, deletes, and lists the binary information in the kernel.

The typical use of the spnet command is to load all of these tables from the default input file when the UNICOS system is started, but it can be used with a small input file that contains a few entries to make a run-time change to the configuration. The spnet command reads a text file (/etc/config/spnet.conf by default) that contains descriptions of any or all of these tables and performs the operations specified on the command line.

The spnet command recognizes shortened option names. The following options are accepted:

-d[efaults] When used with the list command for the NAL table, causes spnet to list all the NAL field values. Normally, fields using default values are not listed.

-f[ile] *filename*
Specifies text files other than /etc/config/spnet.conf.

-n[umeric] Prints IP addresses in dotted-decimal form.

-q[uiet] Suppresses display of entries that spnet is adding to or deleting from the kernel tables.

-v[erbose] When used with the parse command, displays the input file information in a canonical format. When used with the spnet list command, it provides a text display of the radix tree (kernel-internal form of the NAL and WAL tables).

command Specifies an operation to perform on the NAL, WAL, or IPSO maps. You can specify any of the following commands:

- add — Adds an entry or entries from /etc/config/spnet.conf or the specified text file to the kernel table. Specify the entries in the *table_args* argument.
- delete — Removes an entry or entries from the kernel table. Specify the entries in the *table_args* argument.

`export` Reads the `*.cfg` files (listed under the `import` command) and writes to the `stdout` file the `spnet` input language version of the information. This command is not intended for use outside of the installation and configuration menu system because compatibility might not be maintained in the next UNICOS release.

`help` Displays usage information.

`import` Reads the input file and produces the following files for the installation and configuration menu system: `ssnalset.cfg`, `sswalset.cfg` `sscipsomap.cfg`, and `ssspnet.cfg`. This command is not intended for use outside of the installation and configuration menu system because compatibility might not be maintained in the next UNICOS release.

`list` Lists the kernel table.

`parse` Parses the input file, but it performs no actions except to report errors.

`resolve` Lists individual NAL or WAL entries from the kernel table. Specify the entries in the *table_args* argument.

-table Specifies the type of table on which to operate. Valid types are as follows:

`-nal` Indicates NAL operations for Internet Protocol (IP) addresses

`-wal` Indicates WAL operations for IP addresses

`-map` Indicates IPSO map operations

table_args Specifies table arguments for the `add`, `delete`, `convert`, and `resolve` `spnet` commands. The format of *table_args* is as follows:

[*address_list*] [*ifile*] [*ofile*]

For the `add`, `delete`, and `resolve` commands, *table_args* is an address list based on the type of table specified with the *table* argument. If you specify more than one *table* argument, you must specify a separate *table_args* entry for each one. You must specify at least one of the following *address_list* items with the `add`, `delete`, and `resolve` commands:

address_list

Specifies one of the following:

- A list of addresses from the NAL or WAL definitions in the `/etc/config/spnet.conf` file or the kernel.
- A list of addresses from the alternate input file specified by the `-f` *filename* option.
- A list of DOI numbers for the IPSO maps defined in the input file.
- One or more of the following address indicators:

`-net` Limits the selection of NAL or WAL addresses to network entries

`-host` Limits the selection of NAL or WAL addresses to host entries

`-ip` Limits the selection of NAL or WAL addresses to IP protocol family entries

`-station` or
`-cipso` Limits the selection of IPSO maps to CIPSO entries
`-basic` Limits the selection of IPSO maps to basic entries
`-all` Specifies all entries within the constraints set by the modifiers in the preceding list. Use of this indicator in an `add` command saves kernel space because several addresses can be specified to share a single NAL or WAL entry.

For the `convert` command, *table_args* is the following argument:

ifile Specifies the name of the UNICOS 7.0 binary-format file to be converted to `spnet` input text format.

Following are examples of the `spnet` command:

```
spnet add -map -all
spnet add -nal -ip -all
spnet add -wal -net -all
spnet -file special.nal add -nal pecan poplar14 pecan0-net
spnet delete -map -basic
spnet list -map

spnet convert -wal /etc/hosts.wal >> /etc/config/spnet.conf
```

Input File

The input file for the `spnet` command (`/etc/config/spnet.conf` or a file specified by the `-f` *filename* option) contains a readable text language in which NAL, WAL, and IPSO map specifications are given. A simple grammar specification for the language is provided at the end of this man page. Comments can be placed on any line. They begin with the `!` character and continue until the end of the line. Numeric values are scanned by using the `strtol`(3C) function, which means that numbers beginning with 0 are octal, numbers beginning with 0x are hexadecimal, and numbers beginning with a nonzero digit are decimal. Capitalized words in the following subsections describe rules in the grammar. The remaining subsections describe the following elements of the input language:

- Addresses
- Labels
- NAL entries
- WAL entries
- IPSO map entries

Addresses

One or more addresses are associated with each WAL or NAL entry. Each address applies to the IP family. In the `spnet.conf` file, a single NAL or WAL definition can have several addresses associated with it. In addition to enhancing readability and organization of the file, this can result in space savings in the kernel. Use of the `-all` indicator in an `add` operation causes `spnet` to load entries into the kernel in a way that retains the multi-address association so that only one record is stored per definition.

An IP address can apply to either a single host or to a network (default is host). An IP network address can have a subnet mask, which the NETMASK specifies in dotted-decimal form. The IP_ADDRESS can be either a name or numeric address in dotted-decimal form. Names are looked up in the system host database and then in the system network database. The special network name `default` applies to any host that is not explicitly addressed. Separate default entries exist for `ip net` and for `station`.

The following are examples of address specifications:

```
troll.cray.com
128.162.28.3
ip net cray-net
ip net 128.162.28.80 ( 255.255.255.240 )
ip host 127.0.0.1
localhost
station "NSC,V3,128,0"
```

Labels

Labels contain a level component and a compartments component. The level can be either a name (see `secnames`(3C)), including `syslow` and `syshigh`, or an integer from 0 through 16. The compartments can be names (see `secnames`(3C)), an integer preceded by the # character that represents a bit position as a power of 2, or an integer, ortal, or hex bit mask.

The following are examples of levels:

```
level1
16
syshigh
syslow
```

The following are examples of compartments:

```
secadm
sysadm
human_resources
director
#0, #1, #2, #3, #4
037
0x1f
```

NAL Entries

The network access list (NAL) defines the hosts that are granted access to a UNICOS system. The NAL is an essential mechanism for maintaining the UNICOS security policy in a network environment. Only those hosts that are listed in the NAL, either as a single node or as a member of a network, can communicate over the network with the UNICOS system. A packet from any other host is simply dropped. The NAL holds a collection of attributes for each host, which allows the UNICOS administrators to adjust for the various security implementations and levels of trust of each of the nodes on the network.

The following are the available NAL attributes:

- `name`
- `authority-in` or `auth-in`
- `authority-out` or `auth-out`
- `authority`
- `class`
- `domain` or `doi`
- `ipso`
- `label`
- `min label`
- `max label`
- `modes` or `mode`

The `name` attribute attaches an optional name to a NAL entry. It serves no function as input to the `spnet` command, but it can be useful for labeling the various security domains in a large network. The installation and configuration menu system requires this name, but a default name is generated automatically if it is missing when the `spnet` input file is imported. The name is used to associate NAL entries with network addresses because the creation of NAL entries is a separate step in the installation and configuration menu system.

The name is stored in kernel NAL entries and printed in the `list` output, but the kernel does not interpret the name. Only the first eight characters are saved.

The `authority`, `authority-in`, `authority-out`, `auth-in`, and `auth-out` attributes are used when the `ipso` attribute value is `basic`. These attributes specify the predefined protection-authority flags that are required to be included on input and output IP packets. The flags are specified as a comma-separated list of one or more of the following numeric values:

- 0x80 (`GENSER`)
- 0x40 (`SIOP-ESI`)
- 0x20 (`SCI`)
- 0x10 (`NSA`)
- 0x8 (`DOE`)

The `class` attribute specifies the level of trust of a remote host. The class value can be either uppercase or lowercase, and is one of the following values:

```
class = D | C1 | C2 | B1 | B2 | B3 | A1;
```

The default value is `B1`.

The ipso attribute specifies the type of packet labeling in use between the remote host and the UNICOS system. It can be set to one of the following values:

Packet Type	Description
none	No packet labeling is specified. This is the default.
basic	Indicates that the IP Basic Security Option (IP BSO), defined in RFC 1108, will be used. Protection authority information is described in the preceding section for the auth-in and auth-out attributes. To select optional security level mapping, define and activate an IPSO map table with an identifier of 0 or the name basic (see the IPSO Map Entries subsection).
cipso	Indicates that the Common IP Security Option (CIPSO) defined by revision 2.2 of the Trusted Systems Interoperability Group CIPSO specification will be used. In this case, a Domain of Interpretation (DOI) is required in the domain attribute. An IPSO mapping table definition is also required (see "IPSO Map Entries" in a succeeding subsection).

The label, min label, and max label attributes specify the single label, or the label range at which a remote host is allowed access to the UNICOS system. When the label form is used, both the min label and max label are set to that value. The default min and max labels are level 0 and no compartments.

The mode or modes attribute specifies flags that have the following effects:

Mode	Description
receive	Allows Transmission Control Protocol (TCP) connections from the local system to the remote system to complete.
send	Allows TCP connections from the remote system to the local system to complete.
trap	Implementation of this flag is deferred to a future release.
none	No mode flags will be set.

If omitted, the mode attribute defaults to send and receive.

WAL Entries

The workstation access list (WAL) defines the individuals (user IDs) and groups (group IDs) granted access to the local UNICOS system from a specific remote node. The UNICOS security administrator can define the users and groups per node who are allowed to log in to or to access network services from each remote node in the network. The user IDs and group IDs are those of users and groups on the local UNICOS system. For example, whenever a user logs in to UNICOS, the login process references the WAL to validate the user's right to access the UNICOS system from the workstation. The user ID and group ID are formatted with a period separating the uid and gid, as follows: uid.gid. A comma-separated list of allowed services follows the uid.gid pair; for example, uid.gid = ftp, rexec, remsh..

The name attribute attaches an optional name to a WAL entry. It serves no function as input to the spnet command, but it can be useful for labeling the various security domains in a large network. The installation and configuration menu system requires this name, but a default name is generated automatically if it is missing when the spnet input file is imported. The name is used to associate WAL entries with network addresses because the creation of WAL entries is a separate step in the installation and configuration menu system.

The name is stored in kernel WAL entries and printed in the `list` output, but the kernel does not interpret the name. Only the first 8 characters are saved.

By default, all services are permitted to all users from a particular node if no entry exists in the WAL for that node. But when at least one entry exists for a node, a user from that node is granted remote access to UNICOS services only if there is a WAL entry for the user.

In the search for a WAL entry for a user, the applicable WAL entry applies if its address matches the user's remote node and it meets a stronger rule from the following list than any other entry (the rules are in order from strongest to weakest):

1. Its user ID and group ID match the user's user ID and group ID.
2. Its user ID matches the user's user ID and its group ID is the wildcard (`*`).
3. Its group ID matches the user's group ID and its user ID is `*`.
4. Its user ID and group ID are both `*` (this is the default entry).

The values for services controlled by WAL entries are as follows:

Service	Description
`all`	Allows all services.
`none`	Allows no service.
`ftp`	Allows `ftp`(1B) connection from the node to the Cray Research system.
`login`	Allows any kind of interactive access to UNICOS, including `remsh`(1B) interactive login access.
`lpd`	Allows `lpr`(1B) connection from the node.
`mail`	Allows mail connection from the node (deferred).
`nfs`	Allows NFS services to be accessed from the node (deferred).
`nqs`	Allows NQS transfers from the node.
`rexec`	Allows `rexec`(3C) connection from the node.
`rlogin`	Synonymous with `login`.
`rsh`	Allows `remsh`(1B) command access from the node, but not interactive access.
`telnet`	Synonymous with `login`.

IPSO Map Entries

To conserve space in the IP protocol, security levels and compartments are represented by numbers rather than by their ASCII equivalent. Internet Protocol Security Options (IPSO) maps are translation tables that allow the security administrator to control the numeric network representations and to adjust for differences in the implementation of labels among hosts on the network. Every incoming and outgoing packet from a UNICOS system to a particular host on the network has its label information translated through a designated IPSO map.

IPSO maps are numbered and optionally named. The map number, known as the `doi` number, is a 32-bit quantity that is included in every IP packet. A Domain of Interpretation (DOI) is a collection of hosts that share a common definition of label values and meanings. The UNICOS system validates the DOI number against the DOI value in the NAL entry for the remote host. When the `ipso` attribute in the NAL is set to `basic`, the DOI number is 0 implicitly. You can use the map domain name as a convenience within an `spnet` input file for associating NAL entries with maps. The installation and configuration menu system requires this name, but a default name is generated automatically when the `spnet` input file is imported if it is missing.

When you are creating an IPSO map, you must consider the following issues:

- The host-internal numeric level values range as follows: `syslow`, 0 through 16, `syshigh`.
- The host-internal numeric compartment values are each a power of 2 ranging from 2**0 to 2**62.
- The numeric network representations for levels range from 0 through 255, and for compartments, from 0 through 65534.
- Each local level or compartment value must correspond to only one network value.
- Each network level or compartment value must correspond to only one local value.
- Network administrators must ensure that network-wide mappings are nonpermuting; that is, a translation from host A to host B to host C to host A would result in the original label.
- DOI 0 supports four levels and no compartments.

Sample Input File

Following is a sample input file:

```
nal {
      ip net default {
            name = public;
            class = d;
            ipso = none;
            label = 0;
            mode = receive;
      }

      ip net 128.162.72 {
            name = pecannet;
            label = level4;
            ipso = none;
            class = c1;
      }

      station "NSC,V3,128,0" {
            name = station1;
            label = level4;
            authority = 0x80,0x10;
            ipso = none;
            class = c1;
      }

      cool {
            ipso = cipso;
            max label = level16, secadm, sysadm, sysops,
                  netadm, unicos, crayri, nqsdev, nqstst,
                  comp24, comp39, comp63;
            doi = yellow;
      }
}
```

```
wal {
      ip net default {
            *.* = none;
      }

      pecan-0net {
            mgc.* = ftp, login, rsh;
            *.group1 = all;
      }
}
map {
      cipso yellow = 42 {
            levels {
                  Unclassified = 0;
                  Confidential = 1;
                  Secret = 2;
                  TopSecret = 3;
            }
            compartments {
                  usa = 1;
                  france = 5;
                  germany = 11;
            }
      }
}
```

Grammar

This subsection defines the grammar for the `spnet` command. The following conventions are used in this grammar:

`courier` Indicates literal text, to be typed verbatim. Courier UPPERCASE denotes a grammar rule.

: Indicates that a definition follows. Words for this symbol are "is defined as."

| Indicates a succession of items from which to choose one item. The word for this symbol is "or."

ellipses (. . .) Indicates 0 or more repetitions of the preceding item.

[ ] Indicates that the enclosed item is optional.

italics Indicates variable text or numbers, to be replaced with a value.

number..number
Indicates a range (for example, 0..16 indicates any number in the range 0 through 16).

! Indicates a comment line. A comment can be placed on any line. Comments begin with a ! character and continue until the end of the line.

Name specification
The full specification of names is as follows:

```
[%a-zA-Z*][a-zA-Z0-9_.*-]*
```

This specification indicates one of the characters in the first set enclosed in brackets, plus any number of the characters in the second set.

Variable number specification

Variable numbers can be expressed in octal, decimal, or hexadecimal notation.

The following are reserved keywords in the `spnet` grammar:

A1	B2	cipso	ip	nal	{}
a1	b2	class	ipso	name	;
auth-in	B3	compartments	label	net	,
auth-out	b3	D	levels	none	
authority	basic	d	map	receive	
authority-in	C1	debug	max	send	
authority-out	c1	doi	min	station	
B1	C2	domain	mode	trap	
b1	c2	host	modes	wal	

Following are the simplified `spnet` grammar rules:

Grammar Rule		**Definition**
FILE	:	SECTION ...
SECTION	:	NAL | WAL | MAP
NAL	:	nal { NAL_ENTRY ... }
WAL	:	wal { WAL_ENTRY ... }
MAP	:	map { MAP_ENTRY ... }
NAL_ENTRY	:	ADDRESS_LIST { NAL_ATTRIBUTES ... }
WAL_ENTRY	:	ADDRESS_LIST { WAL_ATTRIBUTES ... }
MAP_ENTRY	:	basic { [MAP_LEVELS] }
	|	cipso [[domain] [*domain_name* =]] *number* { [MAP_LEVELS] [MAP_COMPARTMENTS] }
ADDRESS_LIST	:	ADDRESS_VALUE [, ADDRESS_VALUE] ...
ADDRESS_VALUE	:	[ip [host]] IP_ADDRESS
	|	[ip [net]] IP_ADDRESS [(NETMASK)]
IP_ADDRESS	:	*host_or_network_name_or_dotted_IP_address*
NETMASK	:	*dotted_IP_address_mask*
NAL_ATTRIBUTES	:	name = *name* ;
	|	auth-in = AUTHORITY_LIST ;

```
                        |     authority-in = AUTHORITY_LIST ;
                        |     auth-out = AUTHORITY_LIST ;
                        |     authority-out = AUTHORITY_LIST ;
                        |     authority = AUTHORITY_LIST ;
                        |     class = d; | c1; | c2; | b1; | b2; | b3; | a1 ;
                        |     class = D; | C1; | C2; | B1; | B2; | B3; | A1 ;
                        |     domain = NAME; | number ;
                        |     doi = NAME; | number ;
                        |     ipso = none; | basic; | cipso ;
                        |     label = LABEL ;
                        |     min label = LABEL ;
                        |     max label = LABEL ;
                        |     mode = MODE [ [,] MODE ] ... ;
                        |     modes = MODE [ [,] MODE ] ... ;


AUTHORITY_LIST          :     AUTHORITY [ [,] AUTHORITY ] ...
AUTHORITY               :     authority_number
                        |     # authority_bit

LABEL                   :     LEVEL [ : COMPARTMENTS ]
                        |     LEVEL [ , COMPARTMENTS ]

LEVEL                   :     level_name
                        |     0..16

COMPARTMENTS            :     COMPARTMENT [ [,] COMPARTMENT ] ...

COMPARTMENT             :     compartment_name
                        |     compartment_mask
                        |     # compartment_bit

MODE                    :     receive
                        |     send
                        |     trap

WAL_ATTRIBUTES          :     name = name ;
                        |     PERMENTRY [ PERMENTRY ] ...

PERMENTRY               :     USER_LIST = SERVICES ;

USER_LIST               :     USER.GROUP [ [,] USER.GROUP ] ...

USER                    :     user_name | uid | *
```

```
GROUP                   :     group_name | gid | *

SERVICES                :     SERVICE [ [,] SERVICE ] ...

SERVICE                 :     all
                        |     none
                        |     ftp
                        |     login
                        |     lpd
                        |     mail
                        |     nfs
                        |     nqs
                        |     rexec
                        |     rlogin
                        |     rsh
                        |     telnet

MAP_LEVELS              :     levels { MAP_LEVEL_ENTRY;
                                    [ MAP_LEVEL_ENTRY; ] ... }

MAP_LEVEL_ENTRY         :     local_level_name = remote_number
                        |     local_level_number = remote_number

MAP_COMPARTMENTS        :     compartments { MAP_COMPARTMENT_ENTRY;
                                    [ MAP_COMPARTMENT_ENTRY; ] ... }

MAP_COMPARTMENT_ENTRY   :     compartment_name = number
                        |     local_compartment_mask = remote_number
                        |     # local_compartment_bit = remote_number
```

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

SEE ALSO

`ftpd`(8), `nfsd`(8) `rexecd`(8), `rlogind`(8), `rshd`(8), `telnetd`(8)

`login`(1), `privtext`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`secnames`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`ip`(4P), `tcp`(4P), `udp`(4P) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

spray – Spray RPC packets

SYNOPSIS

/etc/spray [-c *count*] [-d *delay*] [-i *delay*] [-l *length*] *host*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The spray command sends a one-way stream of packets to *host* using Remote Procedure Call (RPC) requests and reports how many were received and the transfer rate. The *host* argument can be either a name or an Internet dot address.

The spray command accepts the following options:

-c *count* Specifies the number of packets to send. The default value of *count* is the number of packets required to make the total stream size 100000 bytes.

-d *delay* Specifies the number of microseconds to pause between sending each packet. The default is 0.

-i *delay* Specifies that Internet control message protocol (ICMP) echo packets must be used, rather than RPC requests. Because ICMP automatically echos, this creates a two-way stream.

-l *length* Specifies the number of bytes in the Ethernet packet that holds the RPC call message. Since the data is encoded using external Data Representation (XDR), which only deals with 32-bit quantities, not all values of *length* are possible, and spray rounds up to the nearest possible value. The default value of *length* is 86 bytes (the size of the RPC and user datagram protocol (UDP) headers).

host Specifies name or Internet dot address to which packets are sent.

SEE ALSO

ping(8), sprayd(8)

icmp(4P) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

sprayd – Performs RPC-based spray server function

SYNOPSIS

/etc/sprayd

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The sprayd command is a Remote Procedure Call (RPC)-based server that records the packets sent by spray(8). The sprayd daemon is always registered as RPC program 100012. Usually, inetd(8) invokes the sprayd daemon.

FILES

/etc/inetd.conf	Contains configuration information and listens for incoming service requests
/etc/rpc	File that makes remote procedure calls

SEE ALSO

inetd(8), spray(8)

NAME

spwcard – Sets wildcard levels on system directories in a trusted environment

SYNOPSIS

/etc/spwcard

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The spwcard sets the wildcard security level on the appropriate system directories required for the UNICOS system to operate in multilevel mode. The directory names are defined within the spwcard program, and no arguments are required or allowed when executing the command.

/dev/tty* devices are reset to level 0 and null compartments. Usually, spwcard is executed by the /etc/rc (see brc(8)) script when the system is entering multiuser mode, but can be executed at any time by root to label system directories with the wildcard level.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

FILES

/etc/spwcard

SEE ALSO

General UNICOS System Administration, Cray Research publication SG–2301

NAME

ssd – Display and change SSD configuration information

SYNOPSIS

/etc/ssd [-a] [-c] [-e] [-p] [-q] [-R] [-u *channel*] [-d *channel*] [-t *blocks*] [-T *blocks*]

IMPLEMENTATION

Cray PVP systems (except CRAY J90 series and CRAY EL series)

DESCRIPTION

The ssd command is used to display and modify SSD configuration information. The ssd command can be used to *up* or *down* SSD VHISP channels.

You must be an appropriately authorized user to execute the ssd command.

-a Display all VHISP and SSD information. The output of this option is not meant to be very readable, it is used mainly for debugging.

-c Display the VHISP and SSD configuration. See the EXAMPLES section.

-e Display the VHISP and SSD error counter information. See the EXAMPLES section.

-p Display the VHISP and SSD performance information. See the EXAMPLES section.

-q Display the VHISP and SSD queuing information. See the EXAMPLES section.

-R Reset all VHISP and SSD performance and queuing information to an initial state of all zeroes.

-u *channel* *Up* the specified *channel*. This option makes the *channel* available for use by the system.

-d *channel* *Down* the specified *channel*. This option makes the *channel* unavailable for use by the system.

-t *blocks* Change the synchronous ssd threshold. The *blocks* argument is expressed in terms of 64-word blocks. SSD I/O transfers whose size is equal or less than this size will be done synchronously (*spin wait for channel interrupt*) within the low-level channel handler.

-T *blocks* Change the synchronous sds threshold. The *blocks* argument is expressed in terms of 64-word blocks. SDS I/O transfers whose size is equal or less than this size will be done synchronously (*spin wait for channel interrupt*) within the low-level channel handler.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

Example 1: The following examples shows the -c option of the ssd command:

```
# /etc/ssd -c
Vhisps: 3 configured, 3 up, avail mask 07
  # indx chan(oct) avail
  0    0    1( 01)  up
  1    1    9(011)  up
  2    2   13(015)  up

SSDs: 2 configured
  # indx unit iopath hspmsk .size. ..bias.. ldthresh sdsthrsh maxtrans
  0    0    0     01     03    2Gw        0       200    10000   262136
  1    1    0     06     03    2Gw        0       200    10000   524272
```

Example 2: The following example shows the -e option of the ssd command:

```
# /etc/ssd -e
Vhisps: 3 configured, 3 up, avail mask 07
  # indx avail chan  errors recvrd unrcvd
  0    0     1    1       0      0      0
  1    1     1    9       0      0      0
  2    2     1   13       0      0      0

SSDs: 2 configured
  # indx unit iopath hspmsk errors recvrd unrcvd.
  0    0    0     01     03      0      0      0
  1    1    0     06     03      0      0      0
```

Example 3: The following example shows the -p option of the ssd command:

```
# /etc/ssd -p
Vhisps: 3 configured, 3 up, avail mask 07     (1 block = 64 words)
  # chan #transfers ..#.blocks.. blks/xfer .chan.time. ..time/xfer  .Xfer..Rate
  0    9          0            0       0.0         0ns          0ns 0.000000E+00
  1    1          4       524288  131072.0    269078us  67269635ns 9.976101E+08
  2    5          4       524288  131072.0    269061us  67265345ns 9.976737E+08

SSDs: 2 configured
  # unit iopath #transfers ..#.blocks.. blks/xfer #q'd maxq'd avg.q'd
  0    0     01          0            0       0.0    0      0   0.000
  1    0     06          2      1048576  524288.0    0      1   1.000
```

Example 4: The following example shows the `-q` option of the `ssd` command:

```
# /etc/ssd -q
SSDs: 2 configured
  # unit vhmask #q'd maxq'd  Qdonly Qdfirst  Qdlast Qdinbtw #cumq'd avg.q'd
  0    0     01    0      0       0       0       0       0       0   0.000
                                 0.0%    0.0%    0.0%    0.0%

  1    0     06    0     17    6235       0    2768       0   31453   3.494
                             69.3%    0.0%   30.7%    0.0%
```

SEE ALSO

General UNICOS System Administration, Cray Research publication SG–2301

NAME

ssddconf – Controls the state of a model E SSD drive

SYNOPSIS

ssddconf [-d *device*] [*state*]

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

DESCRIPTION

The ssddconf command controls the state of a model E SSD drive. If no device is specified, all open devices are displayed. ioctls are issued to both the logical and physical SSD drivers to mark the file systems that are affected by the SSD state change and mark the SSD driver tables that are necessary.

The ssddconf command accepts the following options:

-d *device* Device I/O path (for example, 0132 or 0230.3). If no device is specified, all open devices are shown.

state One of the following:

- rw — Sets the device to be operationally read/write.
- ronly — Sets the physical device in read-only mode. Read requests are permitted, writes are returned with errno set to EIO, indicating an I/O error. If any mounted file systems are residing on that device, it marks that file system slice(s) not available for allocation.
- noall — Sets a device to be operationally nonallocatable.
- up — Sets the physical SSD device in the up state, and if any mounted file systems are residing on the SSD, marks that file system that its slice(s) are available for allocation.
- down — Sets the physical device in the down state and terminates all queued I/O requests with errors. If there are any mounted file systems residing on that device, it marks that file system slice(s) not available for allocation.

The display has the following format:

```
command line: ssddconf

      SSD  devices
  iopath  type unit # open  state mode
 -------- ---- ---- ------ ------ ----
      01  SSD    0       2    up  rw
```

Where:

`iopath`	Device I/O path
`type`	Device type
`unit`	Device unit number
`state`	Device state (for example: up or down)
`mode`	Device mode: read/write (`rw`), read-only (`ro`), or nonallocatable (`na`)

SEE ALSO

`ioctl`(2)

NAME

start_air – Starts the automated incident reporting (AIR) system

SYNOPSIS

/usr/air/bin/start_air

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The start_air shell script initiates the automated incident reporting (AIR) daemon (aird(8)) after checking for the existence of key AIR system components. By default, the /etc/config/daemons file initiates AIR by invoking this script.

The start_air shell script checks for the existence of the AIR home and log file directories and uses the airexist(8) command to check for a currently running aird(8) process. If one does not exist, start_air initiates an aird(8) process by using the default configuration and standard output files (/usr/air/config_file and /usr/spool/air/logs/ttylog, respectively).

SEE ALSO

aird(8), airexist(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

startup – Puts system in multiuser mode or turns on system accounting

SYNOPSIS

/usr/lib/acct/startup

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

This man page describes two startup shell scripts: /etc/startup puts the system in multiuser mode, and /usr/lib/acct/startup turns on system accounting.

The /usr/lib/acct/startup shell script turns on process accounting and, if directed, also turns on daemon accounting. startup checks the values of NQS_START, TAPE_START and SOCKET_START in the accounting configuration file /etc/config/acct_config to determine whether to turn on any daemon accounting. The ACCTON string, as defined in /etc/config/acct_config, is written to /etc/wtmp when startup is invoked.

Typically, startup is called from /etc/rc at each system boot.

NOTES

If Network Queuing System (NQS) accounting is enabled by startup, it also must be enabled by NQS by using the qmgr set accounting on command (see qmgr(8)).

If tape accounting is enabled by startup, you must use the -c option when starting the tape daemon, tpdaemon(8).

FILES

/etc/config/acct_config	Accounting configuration file
/etc/wtmp	Login/logoff summary
/usr/adm/acct/day	Directory that contains current accounting files

SEE ALSO

For the `/etc/startup` shell script:
`brc`(8), `init`(8), `killall`(8), `mount`(8), `slogdemon`(8)

For the `/usr/lib/acct/startup` shell script:
`acct`(8), `acctsh`(8), `brc`(8), `csa`(8), `rc`(8), `turnacct`(8), `turndacct`(8)

`sync`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Resource Administration, Cray Research publication SG–2302

NQE Administration, Cray Research publication SG–2150, for information about NQS accounting

NAME

`statd` – Provides the crash and recovery functions for NFS locking services

SYNOPSIS

`/etc/statd`

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `statd` daemon is an intermediate version of the status monitor. It interacts with `lockd`(8) to provide the crash and recovery functions for the locking services on the network file system (NFS).

NOTES

The crash of a site is detected only on its recovery.

FILES

`/usr/spool/statmon/sm`	Contains client names having current lock requests.
`/usr/spool/statmon/sm.bak`	Used in recovery of lock requests.
`/usr/spool/statmon/state`	Contains current state of `statd`.

SEE ALSO

`lockd`(8)

NAME

staticrts – Installs static routing information

SYNOPSIS

/etc/staticrts [-n] [-f *conffile*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The staticrts command uses the route(8) command to install static routes defined in the /etc/gated.conf configuration file.

The staticrts command accepts the following options:

-n Specifies that the route commands will be echoed, but not executed.

-f *conffile* Specifies an alternative configuration file.

The routes listed in the configuration file are expected to be in the format of the dynamic routing daemon gated(8) (see the gated-config(5) man page for details). This is true regardless of whether the site actually chooses to run gated, to implement a single routing standard. staticrts searches the configuration file for any specifications of default routes, static network routes, or static host routes, and it issues the appropriate route(8) commands to install those routes in the kernel.

The staticrts command is intended primarily for those sites that choose not to run gated(8), and as an emergency backup in case the gated(8) program becomes unexecutable.

FILES

/etc/gated.conf Default configuration file for routing information

SEE ALSO

gated(8), netstart(8), route(8)

gated-config(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

stdhosts – Converts/prints Internet addresses from hosts(5) or networks(5) files

SYNOPSIS

/etc/yp/stdhosts [*file*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The stdhosts command reads a file, converts the byte-ordered Internet addresses in the file to a standard format, and writes the converted file to standard output.

The stdhosts command accepts the following argument:

file Specifies the file to be read by stdhosts. If you do not specify a file, stdhosts reads from standard input. stdhosts expects the file to be in hosts(5) or networks(5) file format.

SEE ALSO

inet(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

hosts(5), networks(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

stor – Defines physical storage

SYNOPSIS

/etc/stor [-d *number*] *special_files*

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

CRAY J90 series

CRAY EL series

DESCRIPTION

The stor command sorts the special files by physical device numbers and writes, to standard output, information about starting and ending disk addresses and sizes.

The stor command accepts the following option and operand:

-d *number* Prints those special files listed that match the specified physical device number.

special_files Specifies one or more special files to process.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to specify any file.

If the PRIV_SU configuration option is enabled, the super user is allowed to specify any file.

EXAMPLES

The following command line displays information for disk device 0–101 only:

```
stor -d 0101 /dev/pdd/*
```

SEE ALSO

ddstat(8), mknod(8)

file(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

dsk(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

suspend, resume – Suspends or resumes a process or group of processes

SYNOPSIS

```
/etc/suspend [-p] pid ...
/etc/suspend [-g] pgrp ...
/etc/suspend [-j] jid ...

/etc/resume [-p] pid ...
/etc/resume [-g] pgrp ...
/etc/resume [-j] jid ...
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The suspend procedure makes a process (or group of processes) ineligible to execute. The resume procedure restores a process (or group of processes) so it is eligible to execute.

The suspend and resume commands accept the following options and arguments:

-p *pid*	When invoked without arguments or with the -p option, affects one or more process IDs (*pid*).
-g *pgrp*	Affects process group IDs (*pgrp*).
-j *jid*	Affects job IDs (*jid*).

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm	Allowed to suspend or resume any process.
sysadm	Allowed to suspend or resume any process, subject to security label restrictions. Shell redirected I/O is subject to security label restrictions.

If the PRIV_SU configuration option is enabled, the super user is allowed to suspend or resume any process.

The csh(1) command has a built-in suspend command with slightly different characteristics. See csh(1) for more information.

In the standard shell, suspend is not a shell built-in, but an alias that the shell creates automatically. That is, suspend is actually an alias for 'kill -STOP $$'.

SEE ALSO

csh(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

suspend(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

UNICOS File Formats and Special Files Reference Manual, Cray Research publication SR–2014

NAME

swapper – Displays current system swapping activity

SYNOPSIS

/etc/swapper [-f]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The swapper command displays current system swapping activity. It displays processes waiting to swap in and the priorities of in-memory processes that are candidates to swap out.

The swapper command prints the entries in the swap-in queue and then prints the processes that are available for swap-out if the swap-in queue is nonempty **and** one of the swapped processes is:

- Not suspended and has been out longer than the Max swap-out time

  or its UID is 0

  or its SUID is 0

- **and** its size is not greater than the total hog memory value

For any process in the swap-in queue that is greater than the total hog memory value, an asterisk is placed after the size field of that process.

The swapper command accepts the following option:

-f Force the printing of the swap-out candidates even if the swap-in queue is empty.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Retrieving process information is not subject to security label restrictions.

If the PRIV_SU configuration option is enabled, for the super user, retrieving process information is not subject to security label restrictions. Shell redirected I/O is not subject to security label restrictions.

EXAMPLES

```
Total swap space:  1748828      Available swap space:      1652136

Total swap-ins:        688      Total blocks swapped in:    297764
Total swap-outs:       782      Total blocks swapped out:   388976

Total hog memory:   224000      Available hog memory:       147116

Max swap-out time:       0      Max in-memory time:              0

Swap-in queue:

time    size    pri  shpri  nic    swpri     hog    pid   name
-----  -------  ---   ---   ---  ----------  ---   -----  --------
 7460      44    39    56    29   7459.6071  no     4252  quotamon
 7460      44    39    56    29   7459.6071  no    15598  quotamon
 7460      64    39    56    29   7459.4286  no     4243  chkpntd
 7460      64    39    56    29   7459.4286  no    15595  chkpntd
 7459      44    39    56    35   7458.6071  no     4172  quotamon
 7459      64    39    56    35   7458.4286  no     4169  chkpntd
 7460     228    56    56    29   7457.9643  yes    4267  a.out
 7460   13260    56    56    29   7341.6071  yes   15610  kiva
 7459   63360    39    56    35   6893.2857  yes    4195  fluent

Swap-out candidates:

time    size    pri  shpri  nic    swpri     hog    pid   name
-----  -------  ---   ---   ---  ----------  ---   -----  --------
low priority sleepers:
 1681     352    26    40    20     -3.1429  no     1944  ftp
 1681     152    26    46    20     -0.3214  no     2999  oper
  886      44    39    56    25      0.0000  no    10784  quotamon
  886      64    39    56    25      0.0000  no    10781  chkpntd
  701     160    39    40    20      0.0000  no    11268  csh
  186      60    26    40    20      0.0000  no    14876  telnetd
  400     292    26    56    24      0.0000  no    13013  sjc_11b
    1      64    30    52    20      0.0000  no    16330  sh
high priority sleepers:
  884   13836    20   487    25      0.0000  yes   10794  a.out
    1     436    20    71    20      0.0000  no    16333  cpp
 1205   11152    20   429    29      0.0000  yes    9353  Mc21.cray
   53    2012    20    60    24      0.0000  no    15879  EXE
```

```
runnable:
27590   11364   995   995    29  -25806.5357  yes    4761  a.out
12326   11364   994   994    29  -10543.5357  yes   47428  a.out
 8644    5820   987   987    27  -6911.0357  yes   60289  mndo91
 3380    5684   989   989    27  -1648.2500  yes   93721  new.out
 3104   13836   996   996    27  -1299.4643  yes   96757  tgie1
 2022    3828   987   987    27   -306.8214  yes    4981  kiva
 1681     140    57    57    20     -1.2500  no    85102  fta
    0     460    40    40    20      0.0000  no    16340  swapper
    1     164    68    68    20      0.0000  no    16337  quota
```

NAME

sysctl – Gets or sets kernel state

SYNOPSIS

```
sysctl [-n] name ...
sysctl [-n] -w name=value ...
sysctl [-n] -a
sysctl [-n] -A
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The sysctl utility retrieves kernel state and allows processes with appropriate privilege to set kernel state. The state to be retrieved or set is described using a Management Information Base (MIB) style name, described as a dotted set of components. The -a flag can be used to list all the currently available string or integer values. The -A flag will list all the known MIB names including tables. Those with string or integer values will be printed as with the -a flag; for the table values, the name of the utility to retrieve them is given.

The -n flag specifies that the printing of the field name should be suppressed and that only its value should be output. This flag is useful for setting shell variables. For example, to save the maximum size of the *ipintrq* in variable *qlen*, use:

```
set qlen=‘sysctl -n net.inet.ip.qmaxlen‘
```

If just a MIB style name is given, the corresponding value is retrieved. If a value is to be set, the -w flag must be specified and the MIB name followed by an equal sign and the new value to be used.

The information available from sysctl consists of integers, strings, and tables. The tabular information can only be retrieved by special purpose programs such as netstat, arp, and spnet. The string and integer information is summaried below. For a detailed description of these variables see sysctl(3C). The changeable column indicates whether a process with appropriate privilege can change the value.

Name	Type	Changeable
net.unix.stream.sendspace	integer	yes
net.unix.stream.recvspace	integer	yes
net.unix.dgram.sendspace	integer	yes
net.unix.dgram.recvspace	integer	yes
net.inet.ip.forwarding	integer	yes
net.inet.ip.redirect	integer	yes

Name	Type	Changeable
net.inet.ip.qmaxlen	integer	yes
net.inet.ip.subnetsarelocal	integer	yes
net.inet.ip.mrtproto	integer	no
net.inet.ip.mrtdebug	integer	yes
net.inet.ip.dynamic_mtu	integer	yes
net.inet.ip.admin_override_mtu	integer	yes
net.inet.ip.ipmaxpkts	integer	yes
net.inet.ip.iploadleveling	integer	yes
net.inet.icmp.maskrepl	integer	yes
net.inet.tcp.printfs	integer	yes
net.inet.tcp.rexmtthresh	integer	yes
net.inet.tcp.defttl	integer	yes
net.inet.tcp.sendspace	integer	yes
net.inet.tcp.recvspace	integer	yes
net.inet.tcp.keepidle	integer	yes
net.inet.tcp.debx	integer	no
net.inet.tcp.ndebug	integer	no
net.inet.tcp.autowinshft	integer	yes
net.inet.udp.checksum	integer	yes
net.inet.udp.sendspace	integer	yes
net.inet.udp.recvspace	integer	yes
net.inet.udp.ttl	integer	yes
net.trace.tcpwaitq	integer	no
net.trace.udpwaitq	integer	no
net.trace.recvspace	integer	yes
net.link_layer.0.ifqmaxlen	integer	yes
net.link_layer.0.bufstatlen	integer	no
net.link_layer.0.bufstatsize	integer	no
net.sock.maxsock	integer	yes
net.sock.sbmax	integer	yes
net.sock.printdelay	integer	yes
user.cs_path	string	no
user.bc_base_max	integer	no

Name	Type	Changeable
user.bc_dim_max	integer	no
user.bc_scale_max	integer	no
user.bc_string_max	integer	no
user.coll_weights_max	integer	no
user.expr_nest_max	integer	no
user.line_max	integer	no
user.re_dup_max	integer	no
user.posix2_version	integer	no
user.posix2_c_bind	integer	no
user.posix2_c_dev	integer	no
user.posix2_char_term	integer	no
user.posix2_fort_dev	integer	no
user.posix2_fort_run	integer	no
user.posix2_localedef	integer	no
user.posix2_sw_dev	integer	no
user.posix2_upe	integer	no
user.stream_max	integer	no
user.tzname_max	integer	no
mbuf.nmbspace	integer	no
mbuf.mhbase	integer	no
mbuf.mdbase	integer	no
netsec.mls_enabled	integer	no

EXAMPLES

To retrieve the maximum size of the *ipintrq*, use the following:

```
sysctl net.inet.ip.qmaxlen
```

To set the maximum size of the *ipintrq* to 100, use the following:

```
sysctl -w net.inet.ip.qmaxlen=100
```

FILES

`<sys/sysctl.h>`	Definitions for top level identifiers, second level kernel and hardware identifiers, and user level identifiers
`<sys/socket.h>`	Definitions for second level network identifiers
`<sys/un.h>`	Definitions for fourth level UNIX domain identifiers
`<net/if.h>`	Definitions for fourth level IF identifiers
`<netinet/in.h>`	Definitions for third level Internet identifiers and fourth level IP identifiers
`<netinet/icmp_var.h>`	Definitions for fourth level ICMP identifiers
`<netinet/igmp_var.h>`	Definitions for fourth level IGMP identifiers
`<netinet/tcp_var.h>`	Definitions for fourth level TCP identifiers
`<netinet/udp_var.h>`	Definitions for fourth level UDP identifiers
`<sys/tr_pcb.h>`	Definitions of fourth level TRACE domain identifiers

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	**Action**
`system`, `secadm`, `sysadm`	Allowed to use the [`-w`] option to set the kernal state.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use the [`-w`] option to set the kernal state.

SEE ALSO

`sysctl`(3C) in the *UNICOS Networking Facilities Administrator's Guide*, Cray Research publication SG–2304

NAME

syslogd – Logs system messages

SYNOPSIS

/etc/syslogd [-f *configfile*] [-m *markinterval*] [-p *path*] [-P *port*] [-d]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The syslogd command reads and logs system messages into a set of files described by the configuration file /etc/syslog.conf. Each message is one line of text. A message can contain a priority code, marked by a number in angle braces at the beginning of the line. Priorities are defined in the include file sys/syslog.h. syslogd reads from the named pipe /dev/log, from an Internet domain socket specified in /etc/services (see services(5)), and from the special device /dev/klog to read kernel messages (see log(4)).

The shell script newsys(8) starts syslogd; however, before it starts the daemon, newsys renames the log files so that they have a length of 0. This is because if the daemon simply started the files being renamed, they would eventually grow to fill the file systems on which they reside (syslogd never truncates the log files).

newsys also saves the 10 most recent copies of the files and deletes any older copies. After this is done for all log files, newsys starts the syslogd daemon.

The syslogd command accepts the following options:

-f *configfile*	Specifies an alternate configuration file.
-m *markinterval*	Selects the number of minutes between mark messages.
-p *path*	Specifies a named pipe other than /dev/log.
-P *port*	Specifies an alternate port. This is a handy feature if more than one syslogd is to run at the same time.
-d	Turns on debugging.

The syslogd command configures when it starts and whenever it receives a hang-up signal. Lines in the configuration file have a *selector* to determine the message priorities to which the line applies and an *action*. The *action* field is separated from the selector by one or more tabs.

Selectors are lists of priority specifiers, separated by semicolons. Each priority has a *facility* describing the part of the system that generated the message, a dot, and a *level* indicating the severity of the message. You may use symbolic names. Use an asterisk to select all facilities. All messages of the specified level or higher (greater severity) are selected. You may select more than one facility, using commas to separate them. Known facilities and levels recognized by `syslogd` are those listed in `syslog`(3C) without the prefix `LOG_`. The additional facility `mark` has a message at priority `LOG_INFO` sent to it every 20 minutes (this may be changed with the `-m` option). The `mark` facility is not enabled by a facility field containing an asterisk. The level `none` may be used to disable a particular facility. The second part of each line describes where the message is to be logged if this line is selected. There are four forms, as follows:

- A file name (beginning with a leading slash); the file is opened in append mode.
- A host name preceded by an at-sign (@); selected messages are forwarded to the `syslog` daemon on the specified host.
- A comma-separated list of users; selected messages are written to those users if they are logged in.
- An asterisk; selected messages are written to all logged-in users.

Blank lines and lines beginning with # are ignored.

`syslogd` creates the `/etc/syslog.pid` file, if possible, containing a single line with its process ID. You can use this to kill or reconfigure `syslogd.`

To bring `syslogd` down, send it a terminate signal (for example, `kill` `` `cat /etc/syslog.pid` ``).

To change what is logged through `syslogd`, edit the configuration file (`/etc/syslog.conf`). The default configuration file is described in *General UNICOS System Administration*, Cray Research publication SG–2301.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

On a UNICOS system with the `FORCED_SOCKET` configuration option enabled, the named pipe interface, `/dev/log`, is not available. In this case, the `-p` option is forced to the TCP/IP default socket specified in the `/etc/services`. If the pipe interface is requested using the `-p` option, the request is ignored.

EXAMPLES

Example 1: Be careful when using wildcards when selecting facilities; the last level specified on the line is the one that is used. Therefore, the following line selects messages of the `err` level for the facility `kern`, not the `debug` level, because the wildcard applies to all facilities:

```
kern.debug;*.err
```

Example 2: The following line selects messages of the `debug` level for the facility `kern`:

```
*.err;kern.debug
```

Example 3: The following line selects all facilities at the `emerg` level and the `mail` and `daemon` facilities at the `crit` level:

```
*.emerg;mail,daemon.crit
```

Example 4: The following line sends all messages except mail messages to the selected file:

```
*.debug;mail.none
```

Example 5: The following configuration file logs all kernel messages and 20-minute marks onto the system console, all notice (or higher) level messages and all mail system messages, except debug messages, into the file `/usr/spool/adm/syslog`, and all critical messages into `/usr/adm/critical`. Kernel messages of error severity or higher are forwarded to ucbarpa. All users are informed of any emergency messages. Users `eric` and `kridle` are informed of any alert messages, and user `ralph` is informed of any alert message or any warning message (or higher) from the authorization system.

```
kern,mark.debug          /dev/console
*.notice;mail.info       /usr/spool/adm/syslog
*.crit                   /usr/adm/critical
kern.err                 @ucbarpa
*.emerg                  *
*.alert                  eric,kridle
*.alert;auth.warning     ralph
```

FILES

`/etc/syslog.conf`	Configuration file
`/etc/syslog.pid`	Process ID
`/dev/log`	Name of the named pipe
`/dev/klog`	Kernel log device

SEE ALSO

`newsys`(8)

`logger`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`syslog`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`log`(4), `services`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

`tcpstart` – Starts the TCP/IP networking software

SYNOPSIS

`/etc/tcpstart`

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `tcpstart` script is called from the `netstart`(8) command to initialize the TCP/IP networking software. `tcpstart` relies on underlying configuration files for system-specific initialization, and thus, it should never need modification.

The `tcpstart` command performs the following functions:

- Executes the local script `/etc/tcpstart.pre` to perform any local initializations configured by the system administrator.
- Sets kernel run-time networking variables by executing the `netvar`(8) command with the contents of the `/etc/config/netvar.conf` file as arguments.
- Creates the binary host file (see `mkbinhost`(8)).
- Sets the system host name, using the `hostname`(1) command to one of the following:
  - The output of the `/etc/config/makehostname` script if it exists and is executable
  - The contents of the `/etc/config/hostname.txt` file if it exists
  - The compiled-in kernel name, as reported by `uname`(1)
- Initializes the TCP/IP networking interfaces by executing the `initif`(8) script.
- Executes the local script `/etc/tcpstart.mid` to perform any local initializations, configured by the system administrator, that must be performed after the interfaces have been configured but before static routes have been installed in the kernel.
- Initializes routing by executing the `staticrts` script if the `gated`(8) daemon is not configured as part of system startup.
- Starts the TCP/IP networking daemons by calling the `sdaemon`(8) utility.
- Executes the local script `/etc/tcpstart.pst` to perform any local initializations configured by the system administrator.

FILES

`/etc/config/netvar.conf`	Start-up arguments to `netvar`
`/etc/config/makehostname`	Executable script to generate a host name
`/etc/config/hostname.txt`	File that contains system host name

SEE ALSO

`gated`(8), `initif`(8), `mkbinhost`(8), `netvar`(8), `sdaemon`(8), `staticrts`(8)

`hostname`(1), `uname`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

telnetd – Invokes the DARPA TELNET protocol server

SYNOPSIS

/etc/telnetd [-debug] [-h] [-I*initid*] [-l] [-r *lowpty*-*highpty*] [-D options] [-D report] [-D exercise] [-D netdata] [-D ptydata] [-B] [-S *tos*] [*port*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The telnetd command is a server to the DARPA standard TELNET virtual terminal protocol. The TELNET server operates at the port indicated in the telnet service description; see services(5). Specify a port number on the command line to override this port number (for debugging, diagnostics, or bftp(1B) server mode).

The telnetd command operates by allocating a pseudo-terminal device (see pty(4)) for a client, then by creating a login process that has the slave side of the pseudo-terminal device as stdin, stdout, and stderr. telnetd manipulates the master side of the pseudo terminal, implementing the TELNET protocol and passing characters between the client and login process.

The telnetd command accepts the following options:

-debug Enables debugging on each socket created by telnetd (see SO_DEBUG in socket(2)).

-h Disables the printing of host-specific information before login has been completed.

-I*initid* Specifies the ID from /etc/inittab to use when init starts login sessions. The default ID is fe.

-l Specifies line mode. Tries to force clients to use line-at-a-time mode.

-r *lowpty*-*highpty*
Specifies an inclusive range of pseudo-terminal devices to use. If the system has sysconf variable _SC_CRAY_NPTY configured, the default pty search range is 0 to _SC_CRAY_NPTY; otherwise, the default range is 0 to 128. Either *lowpty* or *highpty* may be omitted to allow changing either end of the search range. If *lowpty* is omitted, the - character is still required so that telnetd can differentiate *highpty* from *lowpty*.

[-D options]
[-D report]
[-D exercise]
[-D netdata]

[-D ptydata]

Specifies diagnostics output from the telnet server. This is useful for debugging interoperability problems with telnet clients that do not have debugging capability of their own. A telnetd used as a telnet diagnostics server should be used only on a port that is different from the normal telnet port. One argument is required with the -D option; multiple arguments require separate -D options. For example, -D netdata ptydata will not have the desired effect; use -D netdata -D ptydata.

The options argument causes telnetd to include clear text of telnet options to be included in the output stream. This is similar to the options toggle available from telnet.

The report argument produces options as well as other information about the status of telnetd. Setting the options argument when the report argument has been set has no additional effect.

The exercise argument is ignored.

The netdata argument causes telnetd to echo data read from its network connection, similar to the netdata toggle available from telnet.

The ptydata argument causes telnetd to echo the data it writes to the pty that the user's process is reading from. This can be useful when used in conjunction with the netdata option to see what processing telnetd is performing on the incoming data stream, such as end-of-line processing.

-B Specifies bftp server mode. In this mode, telnetd causes login to start a bftp(1B) session rather than the user's normal shell. bftp daemon mode does not support normal logins, and it must be used on a port other than the normal telnet port.

-S *tos* Sets the IP Type-of-Service (TOS) option on the connection to the value *tos*, which may be a numeric TOS value or a symbolic TOS name found in the /etc/iptos file.

port Specifies an alternative port.

The UNICOS TCP/IP telnet daemon tries to use telnet-line-mode (see RFC 1116) to support line-at-a-time mode. If the client does not support line mode, the telnet daemon tries to use an alternative method of supporting line-at-a-time mode by proposing the use of telnet go-ahead (SGA) options.

The telnetd command supports the following TELNET options:

- Binary mode
- Status of options
- Suppress go-ahead
- Remote echoing
- Negotiate about window size
- telnet status
- Flow control

- Terminal type
- Terminal speed
- Options on the extended options list
- Automatic authentication (by using Kerberos)
- Data stream encryption (not supported outside of the USA and Canada)

It does not support timing mark. Currently, no options are defined on the extended options list.

The implementation of the TELNET options follow the TELNET specifications. For a detailed description of the options, see the TELNET specifications RFC 854 to 861, RFC 1073, RFC 1079, RFC 1080, RFC 1091, and RFC 1116.

Many `telnet` client implementations do not support line-at-a-time `telnet` sessions correctly.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

The `telnetd` command cannot be serviced directly; it must be run through `inetd`(8).

SEE ALSO

`inetd`(8)

`bftp`(1B), `privtext`(1), `telnet`(1B) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`socket`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

`inittab`(5), `pty`(4), `services`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

tftpd – Server to the DARPA trivial file transfer protocol

SYNOPSIS

/etc/tftpd [-d] [*port*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The tftpd program is a server to the DARPA Trivial File Transfer Protocol (TFTP). The TFTP server operates at the port indicated in the TFTP service description; see services(5). The tftpd program accepts the following option and operand:

-d Enables debugging for each socket created by tftpd (see SO_DEBUG in socket(2)).

port Overrides the port number in the TFTP service description (for debugging purposes). Instead, tftp uses the port number specified on the command line.

You do not need an account or password on the remote system to use tftp. Because of the lack of authentication information, tftpd allows only publicly readable files to be accessed by the default get and put commands. This extends the concept of ‘‘public’’ to include all users on all hosts that can be reached through the network; this may not be appropriate on all systems, and its implications should be considered before enabling tftpd service. However, tftpd also supports Kerberos-authenticated requests from hosts that provide a kerberized tftp client. For Kerberos-authenticated requests, tftpd provides access to all files readable by the authenticated network principal.

Because tftpd does not require an account name or password, inbound tftpd is a security problem for non-Kerberos-authenticated requests. To reduce this problem, the /etc/tftpd.conf file should be maintained by the system administrator. This file contains a list of directories that are accessible by tftpd. Each directory has to be specified on a separate line. The administrator must provide the absolute path names for these directories and their access permissions. Access permissions are as follows:

R Read-only access by tftpd

W Write-only access

RW or WR Read/write access by tftpd

The tftpd access permissions differ from the directory’s normal access permissions, and the directory’s permissions override tftpd’s access. That is, if a directory cannot be read by others, putting that directory in this file with read permission does not automatically permit access.

BUGS

This server is known only to be self-consistent (that is, it operates with the user TFTP program, tftp(1B)). Because of the unreliability of the transport protocol (udp(4P)) and the scarcity of TFTP implementations, it is uncertain whether it really works.

The search permissions of the directories leading to the files accessed are not checked.

EXAMPLES

In the following example of the /etc/tftpd.conf file, /garbage is readable and writable, and /tmp is readable through tftp, provided that these directories have their world read/write and read permissions set, respectively.

By default, no directories are accessible by tftpd.

```
# This is the tftpd configuration file.  It contains
# the list of directories, and the type of access for
# these directories for tftpd.

# The format is as follows:
#       Mode    Directory
#       where Mode is either R (for only read access
#       via tftpd), W (only write) and RW or WR (for read
#       write access via tftpd).  Each directory has to
#       be specified on a separate line.
#
#
# NOTE:  tftpd is a big security risk and should not be used
# unless necessary.

      RW    /garbage
      R     /tmp
```

SEE ALSO

tftp(1B) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

services(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

tic – Invokes the terminfo compiler

SYNOPSIS

/usr/bin/tic [-v [n]] [-c] *file*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The tic command translates a terminfo(5) file from the source format into the compiled format. The results are placed in the directory /usr/lib/terminfo. The compiled format is necessary for use with the library routines described in curses(3).

-v [n] (Verbose) Output to standard error trace information showing the progress of tic. The optional integer n is a number from 1 to 10, inclusive, indicating the desired level of detail of information. If n is omitted, the default level is 1. If n is specified and greater than 1, the level of detail is increased.

-c Check only *file* for errors. Errors in use= links are not detected.

file Contains one or more terminfo(5) terminal descriptions in source format (see terminfo(5)). Each description in the file describes the capabilities of a particular terminal. When a use=*entry-name* field is discovered in a terminal entry currently being compiled, tic reads in the binary from /usr/lib/terminfo to complete the entry. (Entries created from *file* will be used first. If the TERMINFO environment variable is set, that directory is searched instead of /usr/lib/terminfo.) tic duplicates the capabilities in entry-name for the current entry, with the exception of those capabilities that are defined explicitly in the current entry.

If the TERMINFO environment variable is set, the compiled results are placed there instead of /usr/lib/terminfo.

NOTES

The files in /usr/lib/terminfo are created by the tic administrator utility. This utility produces a word-oriented format. Although not installed, an otic is created during the build of the libcurses library. An administrator may use otic to build old (pre-UNICOS 7.0) binary terminal information files and place them in an alternate directory, such as /usr/lib/oterminfo. These files may then be accessed successfully by executable files by setting the TERMINFO shell variable (and exporting it, if necessary) to the alternate directory name. setupterm in libcurses will then read the terminal information for the directory specified by TERMINFO.

WARNINGS

Total compiled entries cannot exceed 4096 bytes. The name field cannot exceed 128 bytes.

When the `-c` option is used, duplicate terminal names will not be diagnosed; however, when `-c` is not used, they will be.

Use of the `-v` option should be done with great care. This option will cause `tic` to send terminal control (commmonly called escape) sequences to `stderr`. These escape sequences may directly affect the operation of the user's terminal.

ENVIRONMENT VARIABLES

`TERMINFO` Identifies the path to an alternative terminal information (`TERMINFO`) directory. The `/usr/lib/terminfo` default directory file contains terminal definitions. You can create your own terminal definitions, compile them and store them in one of your own directories. For the system to recognize an alternative directory, you must set `TERMINFO` (and export it, if necessary). For example, you could set `TERMINFO=$HOME/terminfo`.

MESSAGES

Most diagnostic messages produced by `tic` during the compilation of the source file are preceded with the approximate line number and the name of the terminal currently being worked on.

`mkdir ... returned bad status`
: The named directory could not be created.

`File does not start with terminal names in column one`
: The first thing seen in the file, after comments, must be the list of terminal names.

`Token after a lseek not NAMES`
: Somehow the file being compiled changed during the compilation.

`Not enough memory for use_list element`
`Out of memory`
: Not enough free memory was available (`malloc`(3C) failed).

`Can't open ...`
: The named file could not be created.

`Error in writing ...`
: The named file could not be written to.

`Can't link ... to ...`
: A link failed.

`Error in re-reading compiled file ...`
: The compiled file could not be read back in.

`Premature EOF`
: The current entry ended prematurely.

`Backspaced off beginning of line`
: This error indicates something wrong happened within `tic`.

`Unknown Capability - "..."`
: The named invalid capability was found within the file.

`Wrong type used for capability "..."`
: For example, a string capability was given a numeric value.

`Unknown token type`
: Tokens must be followed by `@` to cancel, `,` for Booleans, # for numbers, or `=` for strings.

`"...": bad term name`
`Line ...: Illegal terminal name - "..."`
`Terminal names must start with a letter or digit`
: The given name was invalid. Names must not contain white space or slashes, and must begin with a letter or digit.

`"...": terminal name too long.`
: An extremely long terminal name was found.

`"...": terminal name too short.`
: A one-letter name was found.

`"..." filename too long, truncating to "..."`
: The given name was truncated due to UNICOS system file name length limitations.

`"..." defined in more than one entry.`
: Entry being used is "...". An entry was found more than once.

`Terminal name "..." synonym for itself`
: A name was listed twice in the list of synonyms.

`At least one synonym should begin with a letter.`
: At least one of the names of the terminal should begin with a letter.

`Illegal character - "..."`
: The given invalid character was found in the input file.

`New-line in middle of terminal name`
: The trailing comma was probably left off the list of names.

`Missing comma`
: A comma was missing.

`Missing numeric value`
: The number was missing after a numeric capability.

`NULL string value`
: The proper way to say that a string capability does not exist is to cancel it.

`Very long string found.  Missing comma?`
Self-explanatory.

`Unknown option. Usage is:`
An option that was not valid was entered.

`Too many file names.  Usage is:`
Self-explanatory.

`"..." nonexistent or permission denied`
The given directory could not be written into.

`"..." is not a directory`
Self-explanatory.

`"...": Permission denied`
Access denied.

`"...": Not a directory`
`tic` wanted to use the given name as a directory, but it already exists as a file.

`SYSTEM ERROR!! Fork failed!!!`
A `fork`(2) failed.

`Error in following up use-links.`
Either there is a loop in the links or they reference nonexistent terminals. The following is a list of the entries involved.

A `terminfo`(5) entry with a `use=`*name* capability either referenced a nonexistent terminal called *name*, or *name* somehow referred back to the given entry.

BUGS

To allow existing executables from the previous release of the UNICOS system to continue to run with the compiled `terminfo` entries created by the new `terminfo` compiler, cancelled capabilities will not be marked as cancelled within the `terminfo` binary unless the entry name has a `+` within it. (Such terminal names are only used for inclusion within other entries via a `use=` entry. Such names would not be used for real terminal names.)

Example:

```
4415+nl, kf1@, kf2@, ...

4415+base, kf1=\EOc, kf2=\EOd, ...

4415-nl|4415 terminal without keys,
   use=4415+nl, use=4415+base,
```

The above example works as expected; the definitions for the keys do not show up in the `4415-nl` entry. However, if the entry `4415+nl` did not have a plus sign within its name, the cancellations would not be marked within the compiled file and the definitions for the function keys would not be cancelled within `4415-nl`.

FILES

`/usr/lib/terminfo/?/*` Compiled terminal description database

SEE ALSO

`infocmp`(8)

`tput`(1), `tset`(1B) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`curses`(3) (available only online)

`term`(5), `terminfo`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

tpapm – Tape daemon autoloader premount program

SYNOPSIS

/etc/tpapm [-g *device-group-name*] [-l *label-type*] [-m *message-file*] [-r *ring-option*] *vsn*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The tpapm command requests that the tape daemon mount a volume (*vsn*) on any available drive that may be serviced by an autoloader. This function is useful when you must premount volumes required by some program (for example, a backup) that is performing a multivolume operation, and the required volumes reside in the autoloader storage unit (silo).

The tpapm command accepts the following options and operand:

-g *device-group-name*
: Specifies the name of the group to which the volume should be mounted. If you do not specify the device group name, it defaults to an installation-specified name. The device group name (*dgn*) field of tpstat(1) shows the allowable parameters.

-l *label-type*
: Defaults to an installation-specified label type. The label type may be one of the following:

  - al ANSI label.
  - blp Bypass label processing.
  - nl Not labeled.
  - sl IBM standard label.
  - st Single-tape mark format; single tape mark terminates reel.

  Note: Special permission is needed for blp and nl labels. See your system administrator.

-m *message-file*
: Specifies a file to which informative messages from the tape subsystem are written. The default message file is /dev/null.

-r *ring-option*
: Specifies whether the write protect ring is in or out. The allowable value for *ring-option* is either in or out. If you do not specify -r, an installation-defined default is used.

vsn
: Specifies the volume serial number (VSN) to mount on any available device.

NOTES

If this command is installed with a privilege assignment list (PAL), a user who is assigned the following privilege text upon execution of this command is allowed to perform the actions shown:

Privilege Text	Action
SMACDAC	Allowed to use this command if you have an active secadm category.
SDAC	Allowed to use this command if you have an active sysadm category.

If this command is installed with a PAL, a user with one of the following active categories is allowed to perform the action shown:

Active Category	Action
secadm, sysadm, sysops	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

The tpapm command is not functional if the Cray/REELlibrarian (CRL) is in the mandatory state. For information on displaying the state CRL is in, see tpset(8).

SEE ALSO

tpmql(8)

tpstat(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

General UNICOS System Administration, Cray Research publication SG–2301

Tape Subsystem Administration, Cray Research publication SG–2307

NAME

tpbmx – Displays operator information about tape devices

SYNOPSIS

```
/etc/tpbmx [-d]
/etc/tpbmx device_name
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The tpbmx command displays tape device information that may be useful to operators and system administrators. This information comes primarily from the bmxtab structure for each device in the kernel (the definition of bmxtab is in /usr/include/sys/bmx.h).

The tpbmx command accepts the following option and operand:

-d Displays the bmxtab structure for all devices.

device_name Specifies the device for which to display the bmxtab structure.

If you do not specify an option or operand, tpbmx displays a subset of the fields in the bmxtab structure.

The following information is provided on the status output:

device The device name for each device.

pm Path mask. The path mask is a hexadecimal digit that specifies active paths. Bit 0 is the rightmost bit; bit 3 is the leftmost bit. Bit 0 represents path 1.

st Device status.

dn Specifies down status.

up Specifies up status.

flags Flags used by the tape driver.

Symbol	Value	Description
TPD_OPEN	1	Device open for I/O
TPD_ABN	2	Device abnormal status
TPD_OIP	4	Device open in progress
TPD_NOREAD	010	No read ahead
TPD_BKSPC	020	Back up to a file mark
TPD_RDONLY	040	Device read only
TPD_USRSAVED	0100	Saved user’s I/O pointers
TPD_UERR	0200	User error flag

Symbol	Value	Description
TPD_DISCARD	0400	Discard IOP replies
TPD_PCLOSE	0100	Pending close
TPD_IORDY	02000	Device ready for I/O
TPD_HOLD	04000	Close with hold
TPD_USRIO	010000	User first I/O complete
TPD_DEMON	020000	Under daemon control
TPD_DIAG	040000	Under diagnostic control
TPD_SLEEP	0100000	Process waiting on interrupt
TPD_WIOQUIET	0200000	Wait until I/O quiet
TPD_UEOV	0400000	Process user EOV
TPD_CHKD	01000000	Ordinal checked
TPD_CLEAR	02000000	Cleared by operator
TPD_CIP	04000000	Close in progress
TPD_DEMOPEN	010000000	Opened for daemon driver
TPD_BMXOPEN	020000000	Opened for BMX driver
TPD_UNBUF	040000000	Basic unbuffered I/O
TPD_LIST	0100000000	User-supplied list I/O
TPD_MLIST	0200000000	User-supplied multilist
TPD_HLD	0400000000	FDF_HLD set
TPD_GBLK	01000000000	geteblk in progress
TPD_WAITBF	02000000000	Waiting to get a BF
TPD_WSIGDEM	04000000000	Waiting to signal daemon
TPD_ONEMORE	010000000000	Write one block at EOT
TPD_USYSBF	020000000000	Use system buffer
TPD_ACKERR	040000000000	User ack error
TPD_OPEN_DELAY	0100000000000	User delayed in open
TPD_WRITE	0200000000000	Writing to tape
TPD_RECHECK	0400000000000	Recheck response
TPD_REISSUE	01000000000000	Reissue the command
TPD_INTERRUPT	02000000000000	In interrupt routine
TPD_RESTART	04000000000000	Waiting to restart I/O
TPD_ERPA	010000000000000	Call ERPA routine
TPD_MODELE	020000000000000	This is IOS model E
TPD_LOCKP	040000000000000	Process locked in memory
TPD_INITIALIZED	0100000000000000	TPD table initialized
TPD_FLUSHBUF	0200000000000000	Flush write buffer
TPD_ROPE	0400000000000000	A reopen request
TPD_EOD	01000000000000000	At end-of-data
TPD_RECOVERY	02000000000000000	In error recovery
TPD_UEOVAPP	04000000000000000	User EOV append read

Symbol	Value	Description
TPD_PACKET	010000000000000000	Use packet interface
TPD_WAITRCV	020000000000000000	Waiting for recovery
TPD_RCVDONE	040000000000000000	Recovery routine done
TPD_DENSITY	010000000000000000	Density is set
TPD_TEST	020000000000000000	A flag for testing
TPD_LOCKSETFL	040000000000000000	Lock the BXC_SETFL ioctl
TPD_WTM_UEOV	0100000000000000000	Tape mark at UEOV time
TPD_TIMEDOUT	0200000000000000000	Request timed out
TPD_ARMED	0400000000000000000	Device is armed
TPD_RCVIO	01000000000000000000	Do recovery I/O
TPD_NOCHECK	02000000000000000000	Do not do any checking
TPD_CLOSEUNLD	04000000000000000000	Close unload in progress
TPD_PRIMED	010000000000000000000	Device primed
TPD_READOPPOSIT	020000000000000000000	Do read opposite

lreq Last request code.

For IBM compatible devices, the values for this flag are one or more of the following:

Symbol	Value	Description
echo	0001	Echo
lmcr	0202	Load adapter microcode
rmcr	0203	Read adapter microcode
strd	0004	Start driver
stpd	0005	Stop driver
abnk	0006	Add bank
dbnk	0007	Delete bank
achn	0011	Add channel
dchn	0012	Delete channel
cchn	0013	Change channel
acu	0014	Add control unit
dcu	0015	Delete control unit
ccu	0016	Change control unit
adev	0017	Add device
ddev	0020	Delete device
cdev	0021	Change device
cfcu	0022	Configure channel up
cfcd	0023	Configure channel down
chst	0024	Report channel state

Symbol	Value	Description
dvst	0025	Report device state
armd	0026	Arm device
carm	0027	Cancel arm
opdv	0030	Open device
cldv	0031	Close device
cmdl	0232	Command list
selr	0033	Selective reset
hio	0034	Halt I/O
sysr	0035	System reset
drst	0036	Report driver statistics
bkst	0037	Report bank statistics
chst	0040	Report channel statistics
cust	0041	Report control unit statistics
dvst	0042	Report device statistics
opch	0043	Open channel
clch	0044	Close channel
wchb	0245	Write channel buffer
rchb	0246	Read channel buffer
chbp	0047	Set channel buffer pointer
wrqi	0050	Enable and wait for request input
drqi	0051	Disable request in interrupts
adpf	0052	Issue adapter function
rdrt	0053	Read driver tables
rbkt	0054	Read bank tables
rcht	0055	Read channel tables
rcut	0056	Read control unit tables
rdvt	0057	Read device tables
rtrb	0260	Read trace buffer
pdsc	0061	Preset driver simulation code
schs	0062	Start channel simulator
stcs	0063	Stop channel simulator

For block multiplexer (mux) devices connected to an ESCON channel, the following requests are added to the above list:

Symbol	Value	Description
cuup	0064	Configure control unit up
cudn	0065	Configure control unit down

Symbol	Value	Description
enas	0066	Enable asynchronous responses
asre	0067	Asynchronous response

For ER90 devices, the values for this flag are one or more of the following:

Symbol	Value	Description
lmcr	0202	Load microcode
rmcr	0203	Read microcode
strd	0004	Start driver
stpd	0005	Stop driver
abnk	0006	Add bank
achn	0011	Add channel
adsl	0014	Add slave
adev	0017	Add device
cfcu	0022	Configure channel up
cfcd	0023	Configure channel down
opdv	0026	Open device
enas	0027	Enable async responses
cldv	0030	Close device
cmdl	0231	Command list
hio	0032	Halt I/O
selr	0033	Selective reset
mstr	0034	Master reset
asyn	0035	Asynchronous response

lrep Last reply code. The values for this flag are the same as those for *lreq*.

resp Response code. The values for this flag are the same as those for *lreq*.

bblk Number of blocks in buffer.

bsec Number of sectors in buffer. A sector consists of 4096 bytes.

or Number of outstanding requests.

oblk Number of outstanding blocks to transfer.

osec Number of outstanding sectors to transfer.

fnum File number. Number of tape marks encountered in the forward direction.

blks Block number. This number reflects the direction the tape is moving, and the number of blocks of the tape that have moved in that direction. The block number increases when the tape is moving forward and decreases when it is moving backward.

pid Process ID of the process that opened a file on this tape unit. If the process ID is 0, no process has opened a file on this tape.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`secadm`, `sysadm`, `sysops`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

Example 1: This example shows system output. The tape daemon uses the special device name, `tpdemreq`, to make nondevice-specific requests to the driver. `tpdemreq` always is included. Device `170` has two available paths.

```
device   pm st flags                  lreq lrep resp bblk bsec or oblk fnum blks pid
tpdemreq  0 dn 00000000000000000000 cfcu cfcu 0000 0000 0000 00 0000 0000 0000 0      0
200       1 dn 00000000000000000000 0000 0000 0000 0000 0000 00 0000 0000 0000 0      0
201       1 dn 00000000000000000000 0000 0000 0000 0000 0000 00 0000 0000 0000 0      0
202       1 dn 00000000000000000000 0000 0000 0000 0000 0000 00 0000 0000 0000 0      0
203       1 dn 00000000000000000000 0000 0000 0000 0000 0000 00 0000 0000 0000 0      0
170       3 up 00001001002002100120001 rwnd rwnd 0000 0000 0000 00 0000 0000 0000 0  71328
```

SEE ALSO

General UNICOS System Administration, Cray Research publication SG–2301

Tape Subsystem Administration, Cray Research publication SG–2307

NAME

tpclr – Clears the tape drive

SYNOPSIS

/etc/tpclr [-r] *device_name*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The tpclr command clears the last pending request on the data path to a tape drive. All tables and data associated with that device are cleared if possible.

The tpclr command accepts the following option and operand:

-r Issues a slave reset to the ER90 device. A slave reset causes the ER90 device to reset itself to the initial start-up state. All information in the slave is lost. This option is necessary only if the device is hung. A slave reset clears out the ER90 traces. If possible, you should save the device traces before issuing this type of clear. By default, a logical reset is sent to the device. A logical reset clears out only the slave commands. It neither resets the device to the start-up state nor clears the device traces. If the device is not an ER90 device, this option is ignored.

device_name Specifies name of device to be cleared.

NOTES

You should use the tpclr command only as a last resort. Use the tpfrls(8) command and other commands first.

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
secadm, sysadm, sysops	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

EXIT STATUS

If tpclr command completes successfully, 0 is returned; otherwise, a nonzero value is returned. Where possible, this exit status code is normalized to the last three digits. Exit status values are documented in the *Tape Subsystem User's Guide*, Cray Research publication SG–2051.

FILES

`/usr/include/taperr.h`	Tape daemon error codes

SEE ALSO

`tpfrls`(8)

`rls`(1), `rsv`(1), `tprst`(1), `tpstat`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

General UNICOS System Administration, Cray Research publication SG–2301

Tape Subsystem Administration, Cray Research publication SG–2307

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

`tpconf` – Verifies new format tape daemon configuration file and converts it to binary format

SYNOPSIS

`/usr/lib/tp/tpconf` [`-i` *config-file*] [`-o` *bin-config-file*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `tpconf` command verifies the new format tape daemon configuration file and converts it to binary format for the tape daemon to process when it is started.

The `tpconf` command accepts the following:

`-i` *config-file* Specifies the configuration file. If you omit this option, the standard input will be read.

`-o` *bin-config-file* Specifies the binary format configuration file. If you omit this option, the output will be sent to the standard output.

For a system with an IOS model E, `tpconf` adds diagnostic devices. The user must not specify these devices. The diagnostic devices are as follows:

- One per IOP. The name is `IOP`*nn*; *nn* is the IOP number.
- One per channel. The name is `CHAN`*ipcc*; *i* is the cluster number, *p* is the IOP number, and *cc* is the channel number (in octal).

This increases the number of tape devices in the system. The `TAPE_MAX_DEV` parameter in the system boot parameter file should be the sum of the real devices, the number of tape IOPs, and the number of tape channels.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`secadm`, `sysadm`, `sysops`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

EXIT STATUS

If no error occurs, the `tpconf` command exits with a return value of 0. If an error occurs, `tpconf` exits with a return value of nonzero.

SEE ALSO

`tpdaemon`(8)

General UNICOS System Administration, Cray Research publication SG–2301

Tape Subsystem Administration, Cray Research publication SG–2307

NAME

tpconfig – Configures tape devices up and down

SYNOPSIS

```
/etc/tpconfig -m media_loader status
/etc/tpconfig -m media_loader device_name
/etc/tpconfig -m media_loader device_name:device_name:...
/etc/tpconfig -m media_loader device_name-device_name
/etc/tpconfig -g device_group_name device_name
/etc/tpconfig -g device_group_name device_name:device_name:...
/etc/tpconfig -g device_group_name device_name-device_name
/etc/tpconfig -b bank up|down
/etc/tpconfig [-n] device_name up|down
/etc/tpconfig [-n] device_name:device_name:... up|down
/etc/tpconfig [-n] device_name-device_name up|down
```

For systems with an IOS model E:

```
/etc/tpconfig -c ipc up|down
/etc/tpconfig -u ipc/control_unit up|down
/etc/tpconfig [-n] -d ipc/control_unit/device up|down
```

For systems without an ESCON channel:

```
/etc/tpconfig -c rrnncc up|down
/etc/tpconfig -u rrnncc/control_unit up|down
/etc/tpconfig [-n] -d rrnncc/control_unit/device up|down
```

For control units connected via an ESCON channel:

```
/etc/tpconfig -u rrnncc/link_address up|down
/etc/tpconfig [-n] -d rrnncc/link_address/device up|down
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The tpconfig command configures tape devices up and down, changes the status of the associated media loaders, assigns a media loader to a device, and reassigns a device group to a device. When options are not used, the *device_name* operand is required. Control unit numbers are in hexadecimal; all other numbers used as arguments and operands are octal.

The tpconfig command accepts the following options and operands:

-m *media_loader*
: Specifies one of the valid loaders defined in the system.

status
: The *status* operand may have the following values:

`up` Configures the media loader up.

`down` Configures the media loader down.

`attd` Configures the media loader to attended mode.

`auto` Configures the Storage Technology Corporation (StorageTeK) autoloader to unattended mode or the IBM autoloader to auto mode.

device_name Specifies the device name of the device to be processed by `tpconfig`.

device_name`:`*device_name*`:...`
Specifies a list of device names separated by colons (:). Each device name identifies a device to be processed by `tpconfig`.

device_name`-`*device_name*
Specifies a range of devices to be processed by `tpconfig`. To indicate the range, specify the first device name and the last device name, separated by a dash (–). The order of the devices is determined by the order in which they appear in the output of the `tpstat`(1) command. If the first device does not appear before the last device, an error is returned.

`-g` *device_group_name*
Specifies a device group name to be reassigned to a device. The device group name must be valid in the `text_tapeconfig`(5) file. A device assigned to group `CART` could be reassigned to group `OPER`.

You can use this option to move a device or devices into a device group that is used by isolated users, such as operations. For example, you could set aside devices for system dumps. You could also put a tape drive into a special device group and reserve it for `tplabel`(8) tapes so that they do no have to compete for the device with general users.

`-b` *bank* Specifies a bank to be configured. A *bank* is a set of devices that have the same data paths. You cannot use this option with any other options.

`-n` Specifies that the following devices will not be unloaded (no unload). You can use this option to disable the automatic unloading of a volume when the user releases the tape for tapes that will be used repeatedly. When the device is configured down, this option returns an error. You can force a device to be unloaded by using the `tpu`(8) command.

If a tape is already mounted on the device, you can specify the `-n` option to keep the tape on the device. This option applies if the `status` option is either `up` or `down`.

If a tape is already mounted on the device, and you specify `status` as either `up` or `down` but omit the -n option, the tape is unloaded.

`-c` *rrnncc* Specifies a 2-digit ring number, a 2-digit node number, and a 2-digit channel number to be configured. You can use this specification only on GigaRing based systems.

`-u` *rrnncc/control unit*

Configures a control unit. Because a system may have several channels, and control unit numbers may not be unique across channels, you must specify a channel number with the control unit number. The following components compose the control unit configuration: a 2-digit ring number, a 2-digit node number, a 2-digit channel number, a slash (/), and a control unit number. You can use this specification only on GigaRing based systems.

`-d` *rrnncc/control_unit/device*

Specifies a device to be configured. The following components determine the data path to a tape device: a 2-digit ring number, a 2-digit node number, a 2-digit channel number, a slash (/), a control unit number, a slash (/), and a tape device number. You can use this specification only on GigaRing based systems.

`-d` *rrnncc/link_address/device*

Specifies device to be configured. For control units connected by an ESCON channel, the *control_unit* field is replaced by the *link_address*. You can use this specification only on GigaRing based systems.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`secadm`, `sysadm`, `sysops`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

When you use `tpconfig`, replace any reference to a control unit that is connected by an ESCON channel with a link address. The link address defines the port on which a control unit is attached to the ESCON director. If there is no ESCON director in the path being described, the link address field must be set to 0. If an ESCON director exists in the path being described, the link address must reflect the port on which the control unit is attached. Values in the range 0. to .255 are valid.

EXIT STATUS

If `tpconfig` command completes successfully, 0 is returned; otherwise, a nonzero value is returned. Where possible, this exit status code is normalized to the last three digits. Exit status values are documented in the *Tape Subsystem User's Guide*, Cray Research publication SG–2051.

The device `tpconfig` operation is an asynchronous process and as such the initial exit status is zero. If an error occurs during the configuration process, the status is displayed on `/usr/spool/tape/daemon.stderr` and the device status reflects the problem accordingly.

EXAMPLES

The following examples show possible `tpconfig` command lines and tape configurations:

Example Command	Action
`tpconfig -c 20 up`	Configures channel 20 up.
`tpconfig -c 20 -i 1 down`	Configures channel 20 on IOS 1 down.
`tpconfig -u 20/1 down`	Configures control unit 1 on channel 20 down.
`tpconfig -u 20/1 -i 0 up`	Configures control unit 1 on channel 20 on IOS 0 up.
`tpconfig -d 20/1/3 down`	Configures a device on channel 20, control unit 1, device ID 3 down.
`tpconfig -b 1 up`	Configures bank number 1 up.
`tpconfig tape03 up`	Configures device `tape03` up.
`tpconfig tape03:tape01 down`	Configures devices `tape01` and `tape03` down.
`tpconfig tape01-cart00 up`	Configures devices from `tape01` to `cart00` up.
`tpconfig -n tape200 up`	Configures device `tape200` up and sets `no unload`.
`tpconfig -m stksun up`	Configures media loader `stksun` up.
`tpconfig -m stkvm 302`	Configures device 302 to be assigned to media loader `stkvm`.
`tpconfig -c 0132 up`	Configures channel 32 in IOS cluster 0, IOP number 1 up.
`tpconfig -u 0132/A up`	Configures control unit A of channel 32 in IOS cluster 0, IOP number 1 up.
`tpconfig -d 0132/A/2 up`	Configures tape unit 2 of control unit A up. Control unit A is connected to channel 32 in IOS cluster 0, IOP number 1.
`tpconfig -g OPER 301`	Assigns device group `OPER` to device 301.

FILES

`/dev/tape/`*`device_name`*	Tape device node
`/etc/config/text_tapeconfig`	Tape subsystem configuration file
`/usr/include/taperr.h`	Tape daemon error codes

SEE ALSO

tpbmx(8), tpdaemon(8), tpdev(8), tpdstop(8), tpfrls(8), tplabel(8), tpgstat(8), tpu(8)

rls(1), rsv(1), tpmnt(1), tprst(1), tpstat(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

text_tapeconfig(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

Tape Subsystem Administration, Cray Research publication SG–2307

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

tpcore – Initiates the tape subsystem monitor

SYNOPSIS

/etc/tpcore [-a] [-A] [-b *beep_rate*] [-c *core_file*] [-d *device_name*] [-g] [-G] [-J] [-L] [-M] [-o *obj_file*] [-O] [-p *pid*] [-P] [-r *refresh_rate*] [-S *screens*] [-T *tail_count*] [-v] [-V *vsn_count*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The tpcore command initiates the tape subsystem monitor. Administrators use the monitor interactively to perform several functions on a running tape subsystem. These functions include monitoring the tape daemon and issuing warning messages when potentially dangerous conditions occur.

The tpcore command builds a buffer display of 10 screens of data. In order for the command to display this data, the COLUMNS and LINES environment variables must be set. By default, tpcore displays the online UP devices and any mount messages.

You may scroll through the display with the + (forward) and - (backward) commands. You may also control the tape daemon and send the operator wto messages from the command line.

The tpcore command accepts the following options:

-a	Disables the volume serial number (VSN) list display for active users. By default, all VSNs are listed on the assigned drive status.
-A	Displays only assigned drives.
-b *beep_rate*	Sets the interval between beeps. tpcore issues these beeps when there are outstanding mount messages. To disable beeping, set the -b option to 0.
-c *core_file*	Specifies the core file to use for monitoring. The default is /proc/*pid* where *pid* is determined from kernel information.
-d *device_name*	Displays all the information relating to the specified device.
-g	Skips the tpgstat(8) display. The default is to display a formatted tpgstat(8) display.
-G	Shows the number of users currently awaiting device allocation instead of the tpgstat(8) display.
-J	Displays the tape daemon child processes. This information is useful in determining the activity associated with a device while the user is suspended for some tape daemon service.
-L	Displays loader status by using an internal tpmls(8) command.

`-M`	Displays the internal tape daemon message queue. This queue is typically a superset of the messages that an operator can view through the `oper`(8) command.
`-o` *obj_file*	Specifies the object file for symbol table lookup. The search path default is `./tpdaemon`, then `TPDAEMON=obj_file` from the environment, followed by `/usr/lib/tp/tpdaemon`.
`-O`	Formats data dumped in octal.
`-p` *pid*	Specifies the PID to use for `/proc/pid` as the core file. The PID is determined automatically from an internal `ps`(1) command if you omit the `-p` option.
`-P`	Sets the display to page align mode. Major sections of the display are aligned on separate pages or screens of the interactive display.
`-r` *refresh_rate*	Sets the screen update rate to *refresh_rate*.
`-S` *screens*	Sets the number of screens of data to *screens*. The default number of screens is 10. Reducing this number may impact the information that is displayed.
`-T` *tail_count*	Generates a modified display of the last few lines in the `/usr/spool/tape/trace` file. The date and real time stamp are removed from the display and the data is formatted to fit the window width. The default is to tail 4 lines of `daemon.stderr`. To disable the `-T` option, set it to 0.
`-v`	Sets verbose mode.
`-V` *vsn_count*	Limits the number of VSNs displayed to *vsn_count*. This option is useful when memory is corrupted and the file information table overlaid.

The bottom line of the window is a command line, from which you may enter the following commands to control the interactive display and the tape subsystem.

Command	**Description**

`add` *command*	Adds a user-defined display to the output generated by the `tpcore` command.
	
	The output of this shell command is displayed as the last data in the last page or screen. The command must exit; otherwise, the window hangs on its completion as shown in the following example:
	
	`add tail,16 a.file`
	
	Adding this `tail` command would render the monitor useless.
	
	An `add` command that is issued while a previous command is active replaces the old command with the new command. For example, an administrator issues the following to watch device level activity through a `tpcore` display.
	
	`add tpbmx | sed "/000000000000000000000/d"`

`chopt +|- a|A|g|J|L|M|P`
: Toggles the command line options. A plus (`+`) turns the option on and a minus (`-`) turns it off.

  For example, you can turn off the display on tape daemon child processes by using the `-J` option after `tpcore` has been started with the `-J` option enabled. Invalid toggle requests are ignored.

`del`
: Deletes the command specified with the `add` command.

`ref` *rate*
: Alters the refresh rate to *rate*. See the `-r` option.

`rep nn str`
: Enables you to respond to an outstanding operator message.

`tail`[,*num_lines*]*bmx_file*
: Adds the specified number lines of the `/usr/spool/tape/trace/`*bmx_file* file to the window. See the `-T` option.

  num_lines indicates the number of lines. If *num_lines* is less than 4 or more than 40, 4 lines are added. To delete this display, issue the `tail` command without any file name.

  The trace file date is only updated when the child process exits or sleeps during tape mark processing; so the data displayed may not accurately reflect the state of the currently executing child process.

  The data is used to detect stalled child processes and child exits abnormally. For example, when a child process begins executing, the trace entry provides the following kind of information:

  ```
  ...execpgm child for 120 started: jid =...
  ```

  If the trace file fails to update, it indicates that the child is waiting for some event to occur. This event can be a normal one such as a tape mount or an abnormal one such as looping.

`tpclr` *opts*
: Executes a `tpclr`(8) command through the `system`(3C) interface. For a description of *opts*, see the options and operands on the `tpclr`(8) man page.

`tpconfig` *opts*
: Executes a `tpconfig`(8) command through the `system`(3C) interface. For a description of *opts*, see the options and operands on the `tpconfig`(8) man page.

`tpfrls` *opts*
: Executes a `tpfrls`(8) command through the `system`(3C) interface. For a description of *opts*, see the options and operands on the `tpfrls`(8) man page.

`tpset` *opts*
: Executes a `tpset`(8) command through the `system`(3C) interface. For a description of *opts*, see the options and operands on the `tpset`(8) man page.

`tpu` *opts*
: Executes a `tpu`(8) command through the `system`(3C) interface. For a description of *opts*, see the options and operands on the `tpu`(8) man page.

`wto` *message*	Executes a background `/usr/bin/msgr` with *message* as the text, but does not wait for a response from the operator.
`^L`	Causes the screen to be redrawn.
`^U`	Erases current input line.
`^H`	Backspaces current command line position.
`+`	Advances to the next screen in the buffer.
`-`	Backs up one screen.
`!`(*command*)	Executes an immediate command. May be used to exit to a C shell as shown in the following `csh`(1) command example: `!csh`

NOTES

If this command is installed on a UNICOS MLS system with the default privilege assignment list (PAL), you must have an active `secadm`, `sysadm`, or `sysops` category to use this command. To override the MAC and DAC restrictions, you must have the `SMACDAC` privilege text (if you are a security administrator) or the SDAC system administrator). On a UNICOS MLS system with `PRIV_SU` enabled, you must be `root` to use this command.

FILES

`/usr/lib/tp/tpdaemon`	Tape daemon executable file
`/usr/spool/tape/trace`	Trace files for tape devices

SEE ALSO

`oper`(8), `tpclr`(8), `tpconfig`(8), `tpfrls`(8), `tpset`(8), `tpgstat`(8), `tpmls`(8), `tpu`(8)

`csh`(1), `ps`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`system`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

General UNICOS System Administration, Cray Research publication SG–2301

Tape Subsystem Administration, Cray Research publication SG–2307

NAME

tpdaemon – Starts tape daemon

SYNOPSIS

/usr/lib/tp/tpdaemon [-b] [-c] [-C *config_file*] [-d] [-l] [-m *devs*] [-r]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The tpdaemon command starts the tape daemon. It provides the routing and control of the various components used in tape resource management, device management, volume mounts and dismounts through operator communication or autoloader requests, label processing, volume switching, accounting, security, and error recovery. By default, tape configuration is defined as part of the tape daemon start-up.

The tpdaemon command accepts the following options:

-b
: Instructs the tape daemon to bypass all configuration tasks. tpdaemon uses the existing tape configuration instead of configuring the tape subsystem based on the tape configuration file.

  If you specify this option and have not previously defined the configuration by using the tpdaemon command without the -b option, tpdaemon terminates with an error. If you omit the -b option, tpdaemon reads and processes a tape configuration file, provides the configuration information to the kernel and tape I/O processors (IOPs), creates the tape device nodes, and configures the channels and control units as specified in the tape configuration file. Options -b and -C are mutually exclusive.

-c
: Collects tape user data for accounting.

  Options -b and -C are mutually exclusive.

-C *config_file*
: Specifies the file that contains the tape configuration. If you do not specify the -C option, the tape daemon uses the /etc/config/text_tapeconfig file. If the tpinit(8) command does not find this file, it terminates and returns an error message.

-d
: Does not initiate or verify the devices, channels, control units, or banks in the tape configuration file.

-l
: Locks the tape daemon into memory so that it will not be swapped out.

-m *devs*
: Preallocates heap memory for *devs* number of devices. If the tape daemon is locked in memory and needs to grow, using the -m option reduces the probability of the tape daemon being swapped out of memory.

`-r` Removes all trace files before creating new ones. If you do not specify this option, the existing trace files are used. If possible, the old trace files are linked to files in the directory specified in `/etc/config/text_tapeconfig` by the `tape_daemon_trace_savefile_prefix`.

For more information on starting the tape daemon on a UNICOS system, see the section on enhanced security in *General UNICOS System Administration*, Cray Research publication SG–2301.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`secadm`, `sysadm`, `sysops`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

FILES

`/dev/tpddem`	Tape daemon interface device
`/dev/tape/`*device_name*	Tape device file
`/etc/config/text_tapeconfig`	Tape subsystem configuration file
`/usr/spool/tape/trace.`*bmxXXX*	Trace file for tape device. The block multiplexer (BMX) number is associated with the drive in the `tpstat`(1) display.

SEE ALSO

`tpconf`(8), `tpdstop`(8), `tpinit`(8)

`tapetrace`(5), `text_tapeconfig`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

Tape Subsystem User's Guide, Cray Research publication SG–2051

General UNICOS System Administration, Cray Research publication SG–2301

Tape Subsystem Administration, Cray Research publication SG–2307

NAME

tpdev – Displays current tape equipment configuration

SYNOPSIS

/etc/tpdev

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The tpdev command displays the status of tape devices and associated components of the tape data path. The following information is displayed in the status output (control unit numbers are hexadecimal; all other numbers are octal):

dev_name	Specifies device name
dev_grp	Specifies device group name
bk	Specifies bank number
ord	Specifies device order
did	Specifies device ID number
dvst	Specifies the device status
cu	Specifies the control unit ID number
cust	Specifies the control unit status
ipcc	(IOS-E only) Specifies a 1-digit (i) IOS cluster number, a 1-digit (p) IOP number, and a 2-digit (cc) channel number
rrnncc	(GigaRing based systems only) Specifies a 2-digit (r) ring number, a 2-digit (n) node number, and a 2-digit (cc) channel number
chst	Specifies the channel status
loader	Specifies the media loader, if any, for the device

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
secadm, sysadm, sysops	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

The following display (IOS-E only) is an example of tpdev output:

```
dev_name  dev_grp  bk ord  did  dvst  cu   cust  ipcc   chst  loader
130       CART      0   0    0  down   1   up    0321   up    Operator
131       CART      0   1    1  down   1   up    0321   up    Operator
210       TAPE      1   4    0  down   2   up    0321   up    Operator
211       TAPE      1   5    1  down   2   up    0321   up    Operator
170       3490      2  10    0  down   A   up    0326   up    Operator
171       3490      2  11    1  down   A   up    0326   up    Operator
300       SILO      3  14    0  up     6   up    0326   up    stksun
301       SILO      3  15    1  down   6   up    0326   up    stksun
B00       WOLF      8  30    0  up     A   up    0327   up    wolfy
B01       WOLF      8  31    1  up     A   up    0327   up    wolfy
```

The following display (GigaRing based systems only) is an example of tpdev output:

```
dev_name  dev_grp  bk ord  did  dvst  cu   cust  rrnncc   chst  loader
s4890s0   STK4890   1   0    0  down  01   up    000101   up    panther
s4890s1   STK4890   1   1    8  down  01   up    000101   up    panther
d4000s0   DLT4000   1   2    0  down  01   up    000101   up    panther
d4000s1   DLT4000   1   3    8  down  01   up    000101   up    panther
3490s0    IBM3490E  6   4    0  up    06   up    000106   up    ibm
3490s1    IBM3490E  6   5    1  up    06   up    000106   up    ibm
d5649JX   DAT      17   6    0  down  07   up    010107   up    Operator
s9490s0   STK9490  16   7    0  down  06   up    010106   up    wolfy
s9490s1   STK9490  16  10    8  down  06   up    010106   up    wolfy
s9490s2   STK9490  16  11    0  down  06   up    010106   up    wolfy
s9490s3   STK9490  16  12    8  down  06   up    010106   up    wolfy
3590s0    IBM3590  10  13    0  down  00   up    010100   up    ibm
3590s1    IBM3590  12  14    8  down  02   up    010102   up    ibm
ssd3_s0   STKSD3   11  15    8  down  01   up    010101   up    wolfy
ssd3_s1   STKSD3   13  16    8  down  03   up    010103   up    wolfy
```

SEE ALSO

tpconfig(8)

General UNICOS System Administration, Cray Research publication SG–2301

Tape Subsystem Administration, Cray Research publication SG–2307

NAME

`tpdstop` – Stops tape daemon

SYNOPSIS

`/etc/tpdstop`

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `tpdstop` command stops the tape daemon, which terminates in an orderly fashion if the operator uses `tpdstop` to stop it. If the tape daemon does not terminate after the `tpdstop` command, the system administrator may send the tape daemon an interrupt signal by using the `kill`(1) command (*pid* is the process ID of the tape daemon), as follows:

```
kill -2 pid
```

When the tape daemon catches the interrupt signal, it terminates in a less orderly way than if you used `tpdstop`.

If the tape daemon still refuses to terminate after being sent the interrupt signal, the super user may kill it in the following way (again, *pid* is the process ID of the tape daemon):

```
kill -9 pid
```

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`secadm`, `sysadm`, `sysops`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

SEE ALSO

`tpdaemon`(8)

`kill`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

General UNICOS System Administration, Cray Research publication SG–2301

Tape Subsystem Administration, Cray Research publication SG–2307

NAME

tpformat – Formats an ER90 tape volume

SYNOPSIS

/etc/tpformat [-C] [-D *device-name*] [-e] [-f *format-ID*] [-g *device-group-name*] [-l *length*] [-n *number-of-A-partitions* [:*number-of-B-partitions*]] [-q] [-s *A-partition-size*[:*B-partition-size*]] [-u] -v *volume-ID* [-w] [-z]

IMPLEMENTATION

All Cray Research systems except CRAY J90se systems

DESCRIPTION

The tpformat command reserves the specified resource, mounts the requested volume, and issues the volume format request to the ER90 device. After the format is completed, it releases the reserved resource. Use this command to preformat a volume or to specify that the volume be formatted during normal write operations. Preformatting formats the entire volume from beginning-of-tape (BOT) to end-of-tape (EOT), creates the specified partitions, and creates the system zones (if requested).

The tpformat command requires the following argument:

-v *volume-ID*
: Specifies the volume identifier that appears on the physical, outer label of the tape to be formatted. This identifier is used to communicate with the system operator.

The tpformat command accepts the following options:

-C
: Specifies that the character-special tape interface be used. The character-special tape interface allows access to a tape device without using the tape daemon. You are responsible for allocating a resource for the tape format and mounting a volume on a drive. You must also specify the device to which the format request is issued by using the -D option. By default, the tape daemon-assisted interface is used.

-D *device-name*
: Specifies the name of the device to use for the tape format.

  If the tape daemon-assisted interface is used, the tape daemon will issue a mount request to the specified volume. If the requested device is busy (assigned to another user), the request will be queued until the device is available. If the volume serial number (VSN) requested is premounted and idle on another drive, that drive will be used instead of the drive requested. When automatic volume recognition (AVR) is enabled, this option is not available.

-e
: Specifies that the ER90 attempt to minimize the amount of system zone discontinuities in a partition. If the ER90 device determines that a partition should be created after a system zone, the previous partition is extended to the system zone dividing the two partitions. Options -e and -w are mutually exclusive. If neither option is specified, partitions are allowed to span system zones.

`-f` *format-ID*

Specifies the identifier to be recorded on the tape during the volume format. The format identifier must not be longer than 6 alphanumeric characters. If you do not specify a format ID, the volume identifier specified with the `-v` option is recorded on the tape.

`-g` *device-group-name*

Specifies that the volume be mounted on the device belonging to the group *device-group-name*. If you omit *device-group-name*, it defaults to an installation-specified name. The device group name (*dgn*) field of the `tpstat`(1) command shows the allowable parameters.

`-l` *length*

Specifies the length, in double frames, between system zones. The length specified must be in the range 0x842 through 0xFFFFFF. You should not change this length from the device default.

`-n` *number-of-A-partitions* [`:`*number-of-B-partitions*]

Specifies the number of A partitions and the number of B partitions that should be formatted. The number of A partitions specified must be in the range 1 through 255; the size is specified by using the `-s` option. These partitions are formatted on the tape until all partitions have been created or until the end of the tape is detected. If no B partitions are specified (or if *number-of-B-partitions* is specified as 0), the ER90 device formats the tape with A partitions until the EOT is detected. If the EOT is detected after formatting all A partitions and no B partitions are specified, the system continues formatting with A partitions. The default is to create one A partition.

The number of B partitions specified must be in the range 0 through 255. B partitions are created following the last A partition. If one B partition is requested with a size of 0 (as specified with the `-s` option), the volume is formatted with one B partition spanning the remainder of the volume. If you specify more than one B partition, the volume is formatted with B partitions until all partitions are formatted or until the EOT is detected. The default is 0 B partitions.

If the end of the volume is not detected after creating the B partitions, formatting continues, beginning again with A partitions.

`-q` Specifies that the volume should be formatted during normal write operations. The default is to format the entire volume. When using this option, you cannot specify multiple formats.

`-s` *A-partition-size*[`:`*B-partition-size*]

Specifies the size (in millions of bytes) of the partitions. You must specify the number of partitions by using the `-n` option for this option to be valid. The size of the partition must be in the range 0, 0xF0 through 0x1312D00 (240 through 20,000,000 bytes).

If you specify a size of 0 for the A partition, one partition is created spanning the length of the volume. Any size specified for the B partition is then not valid.

If you specify 0 for the B partition size, one B partition is created spanning the length of the tape remaining after the A partitions.

`-u` Disables tape unload at release time. A tape usually is unloaded automatically after the format completes.

`-w` Specifies that the ER90 attempt to minimize the number of system zone discontinuities within a partition. If the ER90 device determines that a partition should be created after a system zone, the previous partition is not extended to the system zone dividing the two partitions. Instead, the area between the previous partition and the system zone is wasted. Options `-e` and `-w` are mutually exclusive. If you omit both options, the partitions may span system zones.

`-z` Specifies that the volume should be formatted without system zones.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`secadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

If the format request does not complete successfully, the integrity of the cassette cannot be guaranteed. You should not use the cassette unless it is reformatted successfully.

If the tape error message TM297 is received, the ER90 device was not able to complete the format request within the time-out period specified in the tape configuration file. This time-out period is specified with the `long_timeout` parameter for ER90 over IPI devices; it is specified with the `format_timeout` parameter for ER90 over HIPPI devices. The tape is not usable if this message is received. To correct the error condition, you can increase the time-out period in the tape configuration file, restart the tape daemon, and reissue the tape format request.

If a volume is formatted without system zones, the volume is positioned to the beginning-of-tape (BOT) or the end-of-tape (EOT) when it is unloaded. It could take up to 185 seconds to complete this unload. By using the default system zone spacing, you can reduce this time to approximately 16 seconds for a small cassette, 21 seconds for a medium cassette, and 24 seconds for a large cassette.

The actual tape format may not be equivalent to the requested format as described in the following paragraphs.

For each partition, the ER90 device creates a beginning-of-media (BOM) marker and reserves the end-of-media warning (EMW) area following the end of the partition. These areas use approximately 42 million bytes of the tape per partition in addition to the partition size requested. Of this 42 million bytes, 12 million bytes are used for the EMW area and 30 million bytes are used for the BOM marker. 8.4 million bytes of the EMW area are available to the user. (The tape daemon allows the user to write into this area by default.)

System zones use additional tape space. For example, for a small cassette in which 3 system zones are created, 809.7 million bytes are used for system zones.

EXAMPLES

Example 1: This example formats a tape with one partition spanning the length of the tape. The format ID recorded on the cassette defaults to the volume ID specified by using the `-v` option.

```
tpformat -v 000011 -g ER90
```

Example 2: This example divides a 25-Gbyte cassette into three partitions of equal length. Each partition consists of approximately 0x208d Mbytes. The format ID recorded on the tape is ER9011. The tape is left on the drive after the format completes because the `-u` option is specified.

```
tpformat -v 000011 -f ER9011 -n 3 -s 0x208d -u -g ER90
```

Example 3: This example formats a cassette into one partition consisting of 0xF0 Mbytes and one partition spanning the remainder of the tape. No system zones are created.

```
tpformat -v 000011 -n 1:1 -s 0xF0:0 -g ER90 -z
```

Example 4: This example requests that one partition consisting of 240 Mbytes be formatted on a 25-Gbyte tape. Because unformatted tape remains after formatting one 240 Mbyte partition, the ER90 device continues creating partitions that consist of 240 Mbytes until the end of tape is detected. There will be approximately 90 partitions on the tape.

```
tpformat -v 000011 -n 1 -s 240 -g ER90
```

Example 5: This example shows how to format a volume when you are using the character-special tape interface. The `tpformat` command formats the volume mounted on device `er92` with 2 partitions consisting of 3 Gbytes and 9 partitions consisting of approximately 2 to 6 bytes.

```
tpformat -f ER4810 -s 3000:2000 -n 2:9 -C -D er92
```

SEE ALSO

General UNICOS System Administration, Cray Research publication SG–2301

Tape Subsystem Administration, Cray Research publication SG–2307

NAME

tpfrls – Forcibly releases tape reservation and associated devices

SYNOPSIS

/etc/tpfrls *user jobid*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The tpfrls command lets the operator release the tape reservations made by a user. To obtain the status of all users' reservations, use the tpgstat(8) command.

You must specify the following operands with tpfrls:

user Indicates the user ID of the user whose resources will be released.

jobid Indicates the job ID of the user whose reservation will be released.

tpfrls also clears all active tape devices and kills the user process using the tape devices.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
secadm, sysadm, sysops	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

EXIT STATUS

If tpfrls completes successfully, 0 is returned; otherwise, a nonzero value is returned. Where possible, this exit status code is normalized to the last three digits. Exit status values are documented in the *Tape Subsystem User's Guide*, Cray Research publication SG–2051.

SEE ALSO

tpgstat(8)

rls(1), rsv(1), tpmnt(1), tprst(1), tpstat(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

General UNICOS System Administration, Cray Research publication SG–2301

Tape Subsystem Administration, Cray Research publication SG–2307

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

tpgstat – Displays user reservation status for all users

SYNOPSIS

/etc/tpgstat [-a]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The tpgstat command displays the reservation status for each device group in the system for each tape user. The tpgstat command accepts the following option:

-a Restricts output to each device group that has been reserved or that is being used by a tape user in the system.

The tpgstat command provides the following status:

user User ID.

jobid Job ID.

dgn Device group name.

w If the user is waiting for a device, an asterisk (*) is displayed.

rsvd Number of devices reserved.

used Number of devices used.

mins Number of minutes the user has the resources reserved.

NQSid Specifies Network Queuing System (NQS) ID if the reserved resources have been submitted through NQS. You may use this ID for qstat(1) or qdel(1).

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
secadm, sysadm, sysops	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

Any devices not reserved by other users show up as being reserved by tpdaemon.

EXIT STATUS

If `tpgstat` command completes successfully, 0 is returned; otherwise, a nonzero value is returned. Where possible, this exit status code is normalized to the last three digits. Exit status values are documented in the *Tape Subsystem User's Guide*, Cray Research publication SG–2051.

EXAMPLES

In the following example display, user `lfa` is waiting for a device that is type `TAPE`:

```
user       job id   dgn    w   rsvd   used   mins   NQSid
tpdaemon     3      TAPE       2      2     110
             3      CART       2      1     110
lfa        362      TAPE   *   1      1      37
           362      CART       0      0      37
backup     302      TAPE       0      0      19
           302      CART       1      1      19
dkl        379      TAPE       1      1      15
           379      CART       0      0      15
jwa        251      TAPE       1      0      0
           251      CART       0      0      0
cym        667      TAPE       0      0      1     4082.sn1101
           667      CART       1      1      1     4082.sn1101
```

SEE ALSO

`qdel`(1), `qstat`(1), `rls`(1), `rsv`(1), `tpmnt`(1), `tprst`(1), `tpstat`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

General UNICOS System Administration, Cray Research publication SG–2301

Tape Subsystem Administration, Cray Research publication SG–2307

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

tpinit – Initializes the tape subsystem

SYNOPSIS

/etc/tpinit [-C *config_file*] [-d]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The tpinit command initializes the tape subsystem. It reads and processes the tape configuration file, provides configuration information to the kernel and tape I/O processors (IOPs), creates the tape device nodes, and, unless instructed not to do so, configures the channels and control units or slaves as specified in the tape configuration file. The command is executed automatically at system startup if tapes are defined.

You cannot run tpinit if a tape device node is open or if the tape daemon is active.

The tpinit command accepts the following options:

-C *config_file* Specifies the file that contains the tape configuration. If you omit the -C option, the command uses the /etc/config/text_tapeconfig file as the tape configuration file.

-d Instructs the command to leave the tape hardware in its existing state rather than attempt to change the state of the hardware. If you omit the -d option, tpinit configures the channels, banks, and control units or slaves as specified in the tape configuration file.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
secadm, sysadm, sysops	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

FILES

/dev/tpddem	Tape daemon interface
/dev/tape/*device_name*	Tape device node
/etc/config/text_tapeconfig	Tape subsystem configuration file

SEE ALSO

`tpconfig`(8), `tpdaemon`(8), `tpdstop`(8)

`text_tapeconfig`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

Tape Subsystem Administration, Cray Research publication SG–2307

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

`tplabel` – Labels a magnetic tape

SYNOPSIS

`/etc/tplabel` [`-C`] [`-D` *device_name*] [`-g` *device_group_name*] [`-l` *label_type*] [`-I`] [`-o` *owner_id*] [`-s` *system_code*] [`-u`] [`-v` *int_vid*[*=ext_vid*][*=format_id*][*/partition*]]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `tplabel` command labels tapes and may perform other functions depending on the interface being used.

When you are using the character-special tape interface and want to label a tape, you must mount the desired tape on the target device prior to using a `tplabel` command. After the label is written, the tape is unloaded.

When you are using the tape daemon-assisted interface and issue a `tplabel` command, the requested resource must be configured up. The `tplabel` command reserves the specified resource, requests the tape mount, labels the tape, and finally releases the resource.

The `tplabel` command accepts the following options:

`-C`
: Specifies that the character-special tape interface be used. The character-special tape interface allows access to a tape device without using the tape daemon. You are responsible for allocating a resource for the volume labeling and mounting a volume on a drive. You must also specify the device to which the label request is issued by using the `-D` option. By default, the tape daemon-assisted interface is used.

`-D` *device_name*
: Specifies the name of the device to use for labeling the tape volume.

  If the tape daemon-assisted interface is used, the tape daemon will issue a mount request to the specified volume. If the requested device is busy (assigned to another user), the request will be queued until the device is available. If the volume serial number (VSN) requested is premounted and idle on another drive, that drive will be used instead of the drive requested. When automatic volume recognition (AVR) is enabled, this option is not available.

`-g` *device_group_name*
: Specifies that the volume be mounted on a device belonging to the group *device_group_name*. If you do not specify the device group name, the volume will be mounted on a device that belongs to the installation-defined default. `tpstat`(1) displays the valid device group names.

`-l` *label_type* Specifies the label type to create. The label type must be one of the following:

`al` ANSI label

`nl` Not labeled

`sl` IBM standard label

If you omit this option, a nonlabeled tape will be created. For nonlabeled tapes, three tape marks are written at the beginning of the tape.

`-I` ER90 volumes only. Specifies that mount verification that uses the *format_id* should be bypassed. To use this option, the user must have root user bypass label or tape manager permission.

`-o` *owner_id* Specifies the owner ID field in the VOL1 header label. The *owner_id* can consist of up to 14 characters. This value defaults to `CRI/UNICOS`.

`-s` *system_code* Specifies the system code field in the HDR1 and EOF1 label. The *system_code* can consist of up to 13 characters. This value defaults to `CRI/UNICOS`.

`-u` Disables the tape unload at release time. If you omit this option, the volume will unload after the format completes.

`-v` *int_vid*[*=ext_vid*][*=format_id*][*/partition*]
Specifies the volume ID of the tape. You can specify the *volume_id* in lowercase or uppercase letters; however, the system converts the internal and external volume IDs to uppercase letters.

int_vid Specifies the ID that will be used in the VOL1 label for `al` or `sl` label type. This ID is not used for volume verification; therefore, it may differ from the *external_volume_id*.

ext_vid Specifies the identifier that appears on the physical, outer label of the tape reel. This ID is used to communicate with the system operator. If you omit an external volume identifier, the default is the internal volume identifier.

format_id ER90 volumes only. Specifies the volume identifier that the ER90 device records on the tape during volume initialization. This ID is used for mount verification. If you omit a format identifier, the external volume identifier is used. If an external volume identifier is not specified, the internal volume identifier is used. If you omit an external volume identifier and want a format identifier, the format identifier follows two equal signs (*internal_volume_id==format_id*).

/partition ER90 volumes only. Specifies the partition number to which the volume should be positioned. The partition number must be in the range of 0 through 1023. If you omit the partition number, the default is 0.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
secadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

Example 1: This example creates an IBM standard label on a volume with the external identifier of EXT011. The VOL1 label contains the volume ID 000011 and the owner ID CRI. The system code is UNICOS.

```
tplabel -v 000011=EXT011 -l sl -o CRI -s UNICOS -g CART
```

Example 2: This example creates a nonlabeled tape on the third partition of an ER90 volume. The volume must have a format ID of FMT011. Three tape marks are created at the beginning of the volume. The -u option specifies that the volume is not unloaded.

```
tplabel -v 000011==FMT011/2 -l nl -u -g ER90
```

Example 3: This example shows how to create an ANSI labeled tape on the volume mounted on device 300 when you are using the character-special tape interface. The tplabel command expects the volume to be mounted on device 300.

```
tplabel -C -D 300 -l al -o CRI -s UNICOS -v  001940
```

Example 4: This example shows how to create IBM standard labels on partitions 2 through 5 on the volume mounted on er93 when you are using the character-special tape interface. The tplabel command expects the volume to be mounted on device er93. The default owner system codes, CRI/UNICOS, are used.

```
tplabel -C -D er93 -p 2-5 -v 004810 -l sl
```

SEE ALSO

tpdaemon(8)

General UNICOS System Administration, Cray Research publication SG–2301

Tape Subsystem Administration, Cray Research publication SG–2307

NAME

tpmls – Displays loader status

SYNOPSIS

/etc/tpmls

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The tpmls command displays the status of the tape loaders in the system. The tape loader configuration information is built from the /etc/config/text_tapeconfig file, and the tape daemon updates it during normal processing. You can use the tpconfig(8) command to configure loaders up or down or to place them in attended or unattended mode of operation.

The tpmls command display contains the following information:

loader Name by which devices and loader requests are associated.

type The type of loader. Currently supported loader types are OPERATOR, STKACS, and IBMSCR.

status UP, DOWN, MANUAL, CNFPND, SYSDWN, SYSMAN, or UNKNOWN. Any loader configured down has its drives configured down.

m Mode of operation. Possible values are as follows:

- A Attended
- U Unattended (auto mode)
- S Full system mode (IBM scratch loader)
- M Manual mode

server Name or collection of front-end IDs to which messages may be sent for service.

old Number of online drives.

m_pnd Number of mounts pending for the loader.

d_pnd Number of dismounts pending for the loader.

r_qd Requests queued waiting for a drive serviced by this loader.

comp Number of mounts completed.

avg Average time to complete mount request.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`secadm`, `sysadm`, `sysops`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

The following example shows a typical display created by executing `tpmls`:

```
  loader      type     status m  server    old  m_pnd  d_pnd  r_qd  comp  avg
========  ========  ====== = ========  ===  =====  =====  ====  ====  ===
Operator  OPERATOR  UP     A UNICOS      0      0      0     0     0    0(sec)
stkvm     STKACS    DOWN   A V3          0      0      0     0     0    0(sec)
stkmvs    STKACS    DOWN   A M4          0      0      0     0     0    0(sec)
stksun    STKACS    UP     A robot       1      0      0     0     8   41(sec)
wolfy     STKACS    UP     A stkwolf     4      0      1     0   166   31(sec)
```

FILES

`/etc/config/text_tapeconfig`	Tape subsystem configuration file

SEE ALSO

`tpconfig`(8), `tpdev`(8)

`text_tapeconfig`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

Tape Subsystem Administration, Cray Research publication SG–2307

NAME

tpmql – Displays tape daemon mount request queue list

SYNOPSIS

/etc/tpmql

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The tpmql program displays the current tape mount request list for all users who have completed initial mount processing and have a mount request pending.

If you use the -O option on the tpmnt(1) command to specify an offset to a volume identifier, the output of the tpmql command is no longer accurate. The tpmql command displays the entire list of volume identifiers. If an offset has been set, the list of identifiers from tpmql may no longer correspond to the order in which the tapes have been accessed using the tpmnt(1) command.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
secadm, sysadm, sysops	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

EXAMPLES

The following example shows the output of tpmql:

```
userid     jobid     reqid     ring     Vsn list
jas        19        14        in       UP0007
dja        208       15        out      ISCAL
bar        549       27        i/o      BARRRY1
```

In the ring field, i/o indicates that the user has omitted the -r argument on the tpmnt(1) command and the system default is to accept the tape with the ring in or out.

The fields in the display have the following meanings:

userid	The user ID of the job running tapes
jobid	The job ID of the job running tapes
reqid	The request ID (internal to the tape daemon)

`ring` The ring status associated with the `Vsn list`

`Vsn list` The volume serial number (VSN) of the next volume to be mounted

SEE ALSO

`tpapm`(8)

General UNICOS System Administration, Cray Research publication SG–2301

Tape Subsystem Administration, Cray Research publication SG–2307

NAME

tpscr – Returns volumes to loader scratch pool

SYNOPSIS

```
/etc/tpscr -v vsnlist loader_name
/etc/tpscr -V file loader_name
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The tpscr command returns volumes allocated by a user to the loader scratch pool. You must specify the following options and operands with tpscr (you cannot use the -v and -V options together):

-v *vsnlist*	Specifies a list of volume serial numbers (VSNs).
-V *file*	Specifies a file that contains the VSNs.
loader_name	The name of the loader to which the allocated volumes belong; this name must match one of the loader names in the /etc/config/text_tapeconfig file.

The VSN list and the VSN file follow the same rules as those used with the tpmnt(1) command.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

The define DEF_SCR_ACTION in the /etc/config/text_tapeconfig file controls whether VSNs allocated by the loader are returned to the scratch pool when a user issues a rls(1) command.

MESSAGES

A scratch request is issued for each VSN specified. Those scratch requests that cannot be completed are displayed with the VSN and the return code from the scratch request.

FILES

/etc/config/text_tapeconfig	Tape subsystem configuration file

SEE ALSO

`rls`(1), `tpmnt`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`text_tapeconfig`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

Tape Subsystem Administration, Cray Research publication SG–2307

NAME

tpset – Sets features for the tape daemon

SYNOPSIS

/etc/tpset [-a *state*] [-c *state*] [-f *state*] [-g *device_group*] [-M *number*] [-o *operator*] [-O *state*] [-T *state*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The tpset command sets features for the tape daemon. It changes the status of automatic volume recognition (AVR), the status of front-end servicing (FES), the status of the Cray/REELlibrarian (CRL) system, the status of overcommitted mount requests, the status of tracing for the tape daemon, or the destination of tape operator messages issued by the tape daemon. All of these features except tracing can be set in the tape configuration file. Tracing is always available for use in problem resolution unless you turn it off by specifying the -t option.

The tpset command can be used by both operators and system administrators. For information about the tape daemon, see the tpdaemon(8) man page.

If you omit the options, tpset returns the current status of all features. You may use the -a, -c, and -f options on only a quiet tape subsystem;

The tpset command accepts the following options:

-a *state* Specifies whether AVR is available. Enter one of the following for *state*:

- on — Enables AVR.
- off — Disables AVR. Overcommitted mount requests must be disabled before you can disable AVR.

-c *state* Specifies whether CRL processing is available. Enter one of the following for *state*:

- mandatory — Requires CRL processing for all tpmnt(1) requests.
- on — Enables CRL processing only when the -X option to tpmnt(1) is used.
- off — Disables CRL processing.

-f *state* Specifies whether FES is available. Enter one of the following for *state*:

- on — Enables FES.
- off — Disables FES.

-M *number* Specifies the maximum number of overcommitted mount requests that the tape subsystem can issue.

When the actual number of tape mount requests exceeds this number, the system stops processing requests until one or more of the already overcommitted mount requests are satisfied.

`-o` *operator* Changes the destination that receives operator messages from the tape daemon.

Enter one of the following for *operator*:

`UNICOS` Routes operator messages to the message daemon.

station Specifies a 32-character station ID, which signifies a servicing front end for your UNICOS system.

`-O` *state* Specifies whether the number of current mount requests can exceed the number of available tape drives.

Enter one of the following for *state*:

`on` Enables the use of overcommitted mount requests only when the AVR is enabled.

`off` Disables the use of overcommitted mount requests. You must wait to specify `off` until all overcommitted mount requests are satisfied.

An operator must respond to the regular mount requests before mounting tapes for overcommitted mount requests. When an unassigned tape drive becomes available, an operator mounts an overcommitted tape, and the system then assigns the drive to the user's job. Overcommitted mount requests work for jobs that need only one tape mount or that are requesting the last of a series of tape mounts.

If you omit the `-g` option, overcommitted mount requests are available to all device groups that have AVR enabled and that use only operator-mounted tapes. If you specify the `-g` option, overcommitted mount requests are available only to the specified device group.

`-T` *state* Specifies whether tracing is enabled for the tape daemon. Enter one of the following for *state*:

`on` Enables tracing. The default is `on`.

`off` Disables tracing.

The new state does not affect child processes executing at the time of the change.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`secadm`	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

SEE ALSO

tpdaemon(8)

tpmnt(1), tpstat(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

General UNICOS System Administration, Cray Research publication SG–2301

Tape Subsystem Administration, Cray Research publication SG–2307

NAME

tpu – Unloads tape drives

SYNOPSIS

```
/etc/tpu [-g] dvn[:dvn...]
/etc/tpu [-g] all
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The system operator uses the tpu command to unload tapes. This command has no effect on a tape that is currently in use. It is most useful for unloading tapes and freeing tape drives on systems running with automatic volume recognition (AVR).

The tpu command accepts the following option and operands:

-g Specifies that the current *no-unload* (cannot be unloaded) status will be disregarded. tpu does not affect a tape drive with no-unload status unless you specify the -g option. See the tpconfig(8) command for more information about tapes with the no-unload status.

dvn Specifies the device names of the tape drives to be unloaded.

all Specifies that all tape drives configured up and not in use will be unloaded.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
secadm, sysadm, sysops	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

SEE ALSO

tpconfig(8)

General UNICOS System Administration, Cray Research publication SG–2301

Tape Subsystem Administration, Cray Research publication SG–2307

NAME

traceroute – Prints the route that packets take to network host

SYNOPSIS

/etc/traceroute [-d] [-f] [-m *max_ttl*] [-n] [-p *port*] [-q *nqueries*] [-r] [-s *src_addr*] [-S *tos*] [-v] [-w *waittime*] *host* [*packetsize*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The traceroute command traces the route that the Internet Protocol (IP) uses when sending a packet from the local system to the given remote host. This command depends on aspects of IP that do not exist in every environment; therefore, it does not display hops through a system whose IP does not support this feature. traceroute continues until the remote host is reached. Hosts whose IP does not support the necessary feature show up as hops displayed as all * (asterisks). traceroute traces the route only from the local system to the remote host.

The Internet is a large and complex aggregation of network hardware connected by gateways. The traceroute command uses the IP protocol time-to-live (ttl) field and tries to elicit an ICMP TIME_EXCEEDED response from each gateway along the path to some host.

The only mandatory argument in the traceroute command is the destination host name or IP number. The name may be a source routed address; the format is @*hop1*@*hop2*. . .@*dst*. By default, a loose source route is used; if the route is preceded by the ! character, a strict source route is generated. The default probe datagram consists of 38 bytes; to increase this size, specify a packet size (in bytes) after the destination host name.

The traceroute command accepts the following options:

-d	Turns on the SO_DEBUG option on the socket.
-f	Turns on the DONT_FRAGMENT bit in the IP header.
-m *max_ttl*	Sets the maximum time-to-live (maximum number of hops) used in outgoing probe packets. The default is 30 hops, which is also the default used for TCP connections.
-n	Prints hop addresses numerically, rather than symbolically and numerically. This saves a name server address-to-name lookup for each gateway found on the path.
-p *port*	Sets the base user datagram protocol (UDP) port number used in probes. Default is 33434. traceroute hopes that nothing is listening on UDP ports *base* to *base*+*nhops*–1 at the destination host (so that an ICMP PORT_UNREACHABLE message is returned to terminate the route tracing). If a daemon is listening on a port in the default range, use this option to select an unused port range.
-q *nqueries*	Specifies the number of probes allowed.

`-r` Bypasses the normal routing tables and sends directly to a host on an attached network. If the host is not on a directly attached network, an error is returned. Use this option to ping a local host through an interface that has no route through it (for example, after the interface was dropped by `gated`(8)).

`-s` *src_addr* Uses the IP address *src_addr* (which must be given as an IP number, not a host name) as the source address in outgoing probe packets. On hosts that have more than one IP address, you can use this option to force the source address to be something other than the IP address of the interface on which the probe packet is sent. If the IP address is not one of this machine's interface addresses, an error is returned and nothing is sent.

`-S` *tos* Sets the Type-of-Service (*tos*) in probe packets to the following value. Default is 0. The value may be an integer in the range 0 to 255, or a symbolic TOS name found in the `/etc/iptos` file. You can use this option to determine whether different types of service result in different paths. Not all values of *tos* are legal or meaningful; for complete information, see the IP specification for definitions. Values may be either numeric or names listed in the `/etc/iptos` file; useful values include `-S delay`, `-S 16` (low delay), and `-S throughput` or `-S 8` (high throughput). For historic compatibility, the `-t` option is accepted as a synonym.

`-v` Specifies verbose output. When you specify `-v`, all received ICMP packets other than `TIME_EXCEEDED` and `UNREACHABLE` are listed.

`-w` *waittime* Sets the time, in seconds, to wait for a response to a probe. The default is 3 seconds.

host Specifies destination host.

packetsize Specifies a packet size that is an alternative to the default size of 38 bytes.

The `traceroute` command tries to trace the route that an IP packet would follow to some Internet host by launching UDP probe packets with a small ttl (time to live) and then by listening for an ICMP `time exceeded` reply from a gateway. `traceroute` starts the probes with a ttl of 1, and it increases by 1 until it receives an ICMP `port unreachable` message (which means `traceroute` reached *host*) or hits a maximum (which defaults to 30 hops and can be changed by using the `-m` option). Three probes (change the number by using the `-q` option) are sent at each ttl setting, and a line is printed showing the ttl, the gateway address, and round-trip time of each probe. If the probe answers come from different gateways, the address of each responding system is printed. If no response occurs within a 3-second time-out interval (changed with the `-w` option), an asterisk character (*) is printed for that probe.

The destination host should not process the UDP probe packets; therefore, the destination port is set to an unlikely value. If someone on the destination is using that value, it can be changed with the `-p` option.

CAUTIONS

This program is for use in network testing, measurement, and management. It should be used primarily for manual fault isolation. Because of the load it may impose on the network, it is unwise to use `traceroute` during normal operations or from automated scripts.

EXAMPLES

Example 1: A sample use of `traceroute`, including output, is as follows.

```
[yak 71]% traceroute nis.nsf.net
traceroute to nis.nsf.net (35.1.1.48), 30 hops max, 56 byte packet
 1  helios.ee.lbl.gov (128.3.112.1)  19 ms  19 ms  0 ms (19.0/12.667)
 2  lilac-dmc.Berkeley.EDU (128.32.216.1)  39 ms  39 ms  19 ms (39.0/32.333)
 3  lilac-dmc.Berkeley.EDU (128.32.216.1)  39 ms  39 ms  19 ms (39.0/32.333)
 4  ccngw-ner-cc.Berkeley.EDU (128.32.136.23)  39 ms  40 ms  39 ms (39.0/39.333)
 5  ccn-nerif22.Berkeley.EDU (128.32.168.22)  39 ms  39 ms  39 ms (39.0/39.0)
 6  128.32.197.4 (128.32.197.4)  40 ms  59 ms  59 ms (59.0/52.667)
 7  131.119.2.5 (131.119.2.5)  59 ms  59 ms  59 ms (59.0/59.0)
 8  129.140.70.13 (129.140.70.13)  99 ms  99 ms  80 ms (99/92.667)
 9  129.140.71.6 (129.140.71.6)  139 ms  239 ms  319 ms (239.0/232.333)
10  129.140.81.7 (129.140.81.7)  220 ms  199 ms  199 ms (199.0/206.000)
11  nic.merit.edu (35.1.1.48)  239 ms  239 ms  239 ms (239.0/239.0)
```

Lines 2 and 3 are the same. This is due to a bug in the kernel on the second hop system (`lbl-csam.arpa`) that forwards packets with a 0 ttl (this is a bug in the distributed version of 4.3 BSD UNIX systems). You must guess the path that the packets are taking across the network, because the NSFNet (129.140) does not supply address-to-name translations for its NSSs. The two numbers at the end of each line are the median and mean times for the answers received from that host.

Example 2: A more interesting example is as follows.

```
[yak 72]% traceroute allspice.lcs.mit.edu.
traceroute to allspice.lcs.mit.edu (18.26.0.115), 30 hops max
 1  helios.ee.lbl.gov (128.3.112.1)  0 ms  0 ms  0 ms (0.0/0.000)
 2  lilac-dmc.Berkeley.EDU (128.32.216.1)  19 ms  19 ms  19 ms (19.0/19.0)
 3  lilac-dmc.Berkeley.EDU (128.32.216.1)  39 ms  19 ms  19 ms (19.0/25.667)
 4  ccngw-ner-cc.Berkeley.EDU (128.32.136.23)  19 ms  39 ms  39 ms (39.0/32.333)
 5  ccn-nerif22.Berkeley.EDU (128.32.168.22)  20 ms  39 ms  39 ms (39.0/32.667)
 6  128.32.197.4 (128.32.197.4)  59 ms  119 ms  39 ms (59/72.333)
 7  131.119.2.5 (131.119.2.5)  59 ms  59 ms  39 ms (59.0/52.333)
 8  129.140.70.13 (129.140.70.13)  80 ms  79 ms  99 ms (80.0/86.000)
 9  129.140.71.6 (129.140.71.6)  139 ms  139 ms  159 ms (139.0/145.667)
10  129.140.81.7 (129.140.81.7)  199 ms  180 ms  300 ms (199.0/226.333)
11  129.140.72.17 (129.140.72.17)  300 ms  239 ms  239 ms (239.0/259.333)
12  * * *
13  128.121.54.72 (128.121.54.72)  259 ms  499 ms  279 ms (279.0/345.667)
14  * * *
15  * * *
16  * * *
17  * * *
18  ALLSPICE.LCS.MIT.EDU (18.26.0.115)  339 ms  279 ms  279 ms (279.0/299.000)
```

The gateways that are 12, 14, 15, 16, and 17 hops away either do not send ICMP `time exceeded` messages or send them with a ttl too small to reach `traceroute`. Gateways 14 through 17 are running the Massachusetts Institute of Technology (MIT) C Gateway code that does not send `time exceeded` messages. No information is available for gateway 12.

The silent gateway 12 in example 2 may be the result of a bug in the 4.2 BSD or 4.3 BSD network code (and its derivatives); the 4.0, 4.1, 4.2, and 4.3 versions of UNIX send an unreachable message, using whatever ttl remains in the original datagram. Because, for gateways, the remaining ttl is 0, the ICMP `time exceeded` is guaranteed to not make it back to `traceroute`. The behavior of this bug is slightly more interesting when it appears on the destination system, as in example 3.

Example 3:

```
 1 helios.ee.lbl.gov (128.3.112.1)  0 ms  0 ms  0 ms (0.0/0.000)
 2 lilac-dmc.Berkeley.EDU (128.32.216.1)  39 ms  19 ms  39 ms (39.0/32.333)
 3 lilac-dmc.Berkeley.EDU (128.32.216.1)  19 ms  39 ms  19 ms (19.0/25.667)
 4 ccngw-ner-cc.Berkeley.EDU (128.32.136.23)  39 ms  40 ms  19 ms (39.0/32.667)
 5 ccn-nerif35.Berkeley.EDU (128.32.168.35)  39 ms  39 ms  39 ms (39.0/39.000)
 6 csgw.Berkeley.EDU (128.32.133.254)  39 ms  59 ms  39 ms (39.0/45.667)
 7 * * *
 8 * * *
 9 * * *
10 * * *
11 * * *
12 * * *
13 rip.Berkeley.EDU (128.32.131.22)  59 ms !  39 ms !  39 ms ! (39.0/45.667)
```

There are 12 gateways (13 is the final destination), and exactly the last half of them are missing. Actually, `rip` (a Sun-3 workstation running SunOS 3.5 system) is using the ttl from the arriving datagram as the ttl in its ICMP reply. Therefore, the reply times out on the return path (with no notice sent to anyone because ICMPs are not sent for ICMPs) until `traceroute` probes with a ttl that is at least twice the path length; that is, `rip` is really only 7 hops away. If a reply returns with a ttl of 1, this problem probably exists. If the ttl is ≤ 1, `traceroute` prints a `!` character after the time. Because vendors ship some obsolete (DEC ULTRIX, Sun 3.*x*) or nonstandard (HPUX) software, expect to see this problem frequently and/or take care when you select the target host of your probes.

Other possible annotations after the time are `!H`, `!N`, `!P` (host, network, or protocol unreachable, respectively), `!S`, or `!F` (source route failed or fragmentation needed); these should be seen only if you specified a source route, or the `-f` flag; otherwise, neither of these should occur, and the associated gateway is broken if you see one. If most probes result in some kind of unreachable message, `traceroute` exits.

SEE ALSO

`ping`(8)

`netstat`(1B) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

NAME

`trcollect` – Collects trace information for TCP/IP, and NFS

SYNOPSIS

`/etc/trcollect` [`-b` *readbuffers*] [`-f` *tracefile*] [`-i` *interface*} [`-r` *recvspace*] [`-s` *timeout*] [`-t` *connection*] [`-u` *connection*] [*generic_info*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `trcollect` command is a trace collecting program for TCP/IP, and NFS protocols. It collects trace information by creating a trace socket and connecting to various entities in the protocol stacks. The options and arguments on the command line determine these entities. The following types of connections are available:

Generic	Traces all information through a specfic type.
Interface specific	Traces information through a specific interface.
Connection specific	Traces information through a particular TCP or UDP connection.

`trcollect` accepts the following options and operand:

`-b` *readbuffers* Specifies the size of the trace buffers in bytes; if the number is followed by a `k` or `K`, the size is in Kbytes. This is the amount of data that TCP/IP buffers before queuing it to satisfy the read request from `trcollect`. The default is 1024 bytes.

`-f` *tracefile* Specifies the name of the trace file to which `trcollect` writes the trace data. If `-f` is not specified, trace data is written to standard output; therefore, the same effect is accomplished by redirecting standard output to a file.

`-i` *interface* Specifies that interface-specific tracing be performed on the specified interface. For example, `-i hy0` traces data through interface `hy0`.

`-r` *recvspace* Specifies the receive space for the socket in bytes; if the number is followed by a `k` or `K`, the size is in Kbytes. This is the amount of data that TCP/IP buffers, while waiting for `trcollect` to issue a read, before dropping data. The default is 32 Kbytes. If the specified size exceeds the maximum for a socket, it can be truncated.

`-s` *timeout* Specifies a time-out period (in seconds) on read operations. This indicates the amount of time that `trcollect` waits for TCP/IP to fill one read buffer before putting data that has been received onto the receive queue. By default, this occurs every 5 seconds.

`-t` *connection* Specifies TCP connection-specific tracing. All TCP-related information for the specified connection is traced. The following four parameters, which must be surrounded by single or double quotation marks, define a connection:

`-fa` *foreign address* Remote host's Internet address or name.

`-fp` *foreign port* Remote host's port number or name, as given in `/etc/services` (see `services`(5)).

`-la` *local address* Local host's Internet address or name.

`-lp` *local port* Local host's port number or name, as given in `/etc/services`.

If you omit any one of these parameters, it will be used as a wildcard, that is, all of the types defined by the other three parameters are traced. TCP connection-specific tracing requires at least one option.

Examples:

To trace any TCP connection with foreign address *host1*, foreign port *port1*, local address *host2*, and local port *port2*, you should specify the following as the operand for the `-t` option:

```
-t '-fa host1 -fp port1 -lp port2 -la host2 '
```

To trace any TCP connection being established from port *port1* on foreign host *host1*, you should specify the following as the operand for the `-t` option:

```
-t '-fa host1 -fp port1 '
```

`-u` *connection* Specifies UDP connection-specific tracing. All UDP-related information for the specified connection is traced. The following four parameters, which must be surrounded by single or double quotation marks, define a connection:

`-fa` *foreign address* Remote host's Internet address or name.

`-fp` *foreign port* Remote host's port number or name, as given in `/etc/services` (see `services`(5)).

`-la` *local address* Local host's Internet address or name.

`-lp` *local port* Local host's port number or name, as given in `/etc/services`.

If you omit any one of these parameters, it will be used as a wildcard, that is, all of the types defined by the other three parameters are traced. UDP connection-specific tracing requires at least one option.

Examples:

To trace any UDP connection with foreign address *host1*, foreign port *port1*, local address *host2*, and local port *port2*, you should specify the following as the operand for the `-u` option:

```
-u '-fa host1 -fp port1 -lp port2 -la host2 '
```

To trace any UDP connection being established from port *port1* on foreign host *host1*, you should specify the following as the operand for the `-u` option:

```
-u '-fa host1 -fp port1 '
```

generic_info Specifies the type of tracing information. The valid types and their meanings are as follows:

`icmp` Specifies tracing information through the ICMP layer.

`idmap` Specifies tracing information through the NFS `idmap` scheme.

`if` Specifies tracing information on all interfaces.

`ip` Specifies tracing information through the IP layer.

`mbuf` Specifies tracing information on all mbuf allocations and deallocations. (This feature is not yet implemented.)

`nfs` Specifies tracing information through NFS. Note: `trcollect` does not support the Cray-to-Cray NFS protocol, CNFS. See NOTES for more information.

`profile` Specifies tracing information through the kernel subroutines, which include the driver interface subroutine, the socket send subroutine, and so on.

`rawip` Specifies tracing information through the raw IP layer.

`rpc` Specifies tracing information on all remote procedure call (RPC) requests. (Deferred implementation.)

`socket` Specifies tracing information on all socket changes. (Deferred implementation.)

`tcp` Specifies tracing information through all TCP sockets that have the `SO_DEBUG` flag set.

`udp` Specifies tracing information through all UDP sockets that have the `SO_DEBUG` flag set.

NOTES

NFS read and write traces are frequently lost. To use the `nfs` option to trace NFS read and write requests, the `-b readbuffers` option must be specified with a size of at least 8192. The size should be the same as the value specified for `rsize` and `wsize` on the `mount`(8) command.

Also, it might be necessary to specify the `-r recvspace` parameter to trace NFS. The value of `recvspace` for the `-r` parameter should be a multiple of the size specified for the `-b` parameter. Increase the size of the `-r` parameter until NFS traces are no longer missing.

BUGS

The `-i` option is not supported on the EL series.

SEE ALSO

`trformat`(8)

NAME

trformat – Formats trace information obtained from trcollect(8)

SYNOPSIS

/etc/trformat [-c] [-d] [-f *tracefile*] [-h] [-m *hex_bitmap*] [-n] [-s *nfs_bitmap*] [-t] [-v] [-x] [*types*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The trformat command is a trace formatting program for TCP/IP and NFS protocols. It formats the trace information collected by the trace collecting program, trcollect(8).

The trformat command accepts the following options and operand:

-c Checks the sequence numbers in the entry to determine whether entries are missing.

-d Prints the ack and seq fields in tcp entries in decimal format instead of hexadecimal format.

-f *tracefile* Specifies the name of the trace file produced by trcollect. If you omit -f, trace data is read from standard input; therefore, the same effect is accomplished by redirecting the data file to standard input.

-h (Header) Displays only the trace entry header information.

-m *hex_bitmap* Specifies the types of ID map entries to be formatted. hex_bitmask is a hexadecimal number that uses a value of 1 in a bit position if that type will be formatted and uses a value of 0 if that type will be skipped. The default is to format all entries.

-n Displays addresses in dot format and ports as integers.

-s *nfs_bitmap* Specifies the types of NFS entries to be formatted. nfs_bitmap is a hexadecimal number that uses a value of 1 in a bit position if that type will be formatted and uses a value of 0 if that type will be skipped. The default is to format all entries.

-t In the rtc field, prints the elapsed time (in milleseconds) instead of the real-time clock value.

-v Verbose mode. Displays detailed information for each entry.

-x (Hexidecimal) Displays each trace entry in hexadecimal format.

types Displays only the entries of the specified type or types. You can use this operand to filter the necessary entries from a trace file. The valid types and their meanings are as follows:

- icmp Displays trace information obtained through the ICMP layer.
- idmap Displays trace information obtained through the NFS idmap scheme.

`if`	Displays trace information obtained from all interfaces.
`ip`	Displays trace information obtained through the IP layer.
`mbuf`	Displays trace information obtained on all mbuf allocations and deallocations. (Deferred implementation.)
`nfs`	Displays trace information obtained through NFS.
`profile`	Displays trace information obtained through the kernel subroutines. These include the driver interface subroutine, the socket send subroutine, and so on.
`rawip`	Displays trace information obtained through the raw IP layer.
`rpc`	Displays trace information obtained on all RPC requests. (Deferred implementation.)
`socket`	Displays trace information obtained on all socket changes. (Deferred implementation.)
`tcp`	Displays trace information obtained on all TCP sockets with the `SO_DEBUG` flag set.
`udp`	Displays trace information obtained on all UDP sockets with the `SO_DEBUG` flag set.

SEE ALSO

`trcollect`(8)

NAME

trpt – Transliterates protocol trace

SYNOPSIS

/etc/trpt [-a] [-f] [-j] [-k] [-p *hex-address*] [-s] [-t] [*system* [*core*]]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The trpt procedure interrogates the buffer of Transmission Control Protocol (TCP) trace records created when a socket is marked for debugging (see setsockopt(2)) and prints a readable description of these records. When you omit all arguments, trpt prints all of the trace records found in the system grouped according to TCP connection Protocol Control Block (PCB). trpt accepts the following options to alter this behavior:

-a In addition to the normal output, prints the values of the source and destination addresses for each packet recorded.

-f Follows the trace as it occurs, waiting a short time for additional records each time the end of the log is reached.

-j Gives a list of only the those PCB addresses for which trace records exist.

-k The -k option is provided for debugging purposes. It forces trpt to use /dev/kmem to obtain data, rather than use system calls.

-p *hex-address*
Shows only trace records associated with the PCB whose address follows the block.

-s In addition to the normal output, prints a detailed description of the packet sequencing information.

-t In addition to the normal output, prints the values for all timers at each point in the trace.

system [*core*]
Allows you to specify alternate kernel and core dump files. The default is /unicos and /dev/kmem. These arguments allow trpt to be used on a core dump.

The recommended use of trpt is as follows:

1. Isolate the problem and enable debugging on the sockets involved in the connection.
2. Find the address of the PCBs associated with the sockets by using the -A option to netstat(1B).
3. Run trpt with the -p option, supplying the associated PCB addresses. You can use the -f option to follow the trace log after the trace is located. If many sockets are using the debugging argument, the -j option can be useful in checking to see whether any trace records are present for the socket in question.

MESSAGES

`no namelist` The system image does not contain the proper symbols to find the trace buffer.

BUGS

The `trpt` procedure should also print the data for each input or output, but this is not saved in the `trace` record.

The output format is inscrutable and should be described.

FILES

`/dev/kmem`	Kernel memory
`/unicos`	Default core dump file

SEE ALSO

`netstat`(1B) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`getsockopt`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

NAME

tsar – System data processing language

SYNOPSIS

/usr/bin/tsar [-c] [-d *flags*] [-v] [*source_files*]

/usr/bin/tsar [-d *flags*] [-e *end_time*] [-p *playback_file*] [-r *record_file*] [-s *start_time*] [-v] [-D *name*[=*def*]] [*source_files*]

/usr/bin/tsar [-d *flags*] [-e *end_time*] [-h *host* [-i *interval*] [-n *sample_count*] [-r *record_file*] [-P *path*]] [-s *start_time*] [-v] [-D *name*[=*def*]] [*source_files*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The tsar command reads system activity data records that have been produced on a UNICOS system by the sdc(8) command. By default, the programs are taken from stdin. tsar uses programs written in a language based on nawk to process the data. The *source_files* operand is a command file which specifies the names of these programs.

The tsar command operates in one of three modes: compilation_only, playback, or online. Use the -c, -p, and -h options to enter the respective modes. compilation_only mode is used to debug source files. In playback mode, tsar processes data from an existing system activity data file (*playback_file*). In the online mode, tsar processes the data as it is generated on the host system by the sdc(8) command. By default, tsar executes in playback mode.

The tsar language is described in detail in *UNICOS Resource Administration*, Cray Research publication SG–2302.

The tsar command accepts the following options and operands:

-c Compiles only the source files. Use this option to debug source files.

`-d` *flags* Specifies the debug flags. The flags are as follows:

0001 Lexical scanning
0002 Expression compilation
0004 Table entry
0010 Code execution
0020 Stack contents
0040 Playback file parsing
0100 Symbol table searching
0200 Table allocation

Multiple flags can be specified by adding the numerical values. For example, to enable lexical scanning and code execution debugging, set the flag to 011. Debug output is written to `stderr`.

`-e` *end_time* Specifies the end time of the report, using the form *hh*[:*mm*[:*ss*]]. This option can be used only in playback mode.

`-h` *host* Specifies the name of the remote Cray Research host system where data sampling is to occur, and places `tsar` in the online mode.

`-i` *interval* Specifies the data sampling interval. *interval* is of the form *xx*`h`*yy*`m`*zz*`s` (*xx* hours, *yy* minutes, *zz* seconds) or *zz* seconds. By default, data is sampled every 5 minutes. This option can be used only in online mode.

`-n` *sample_count* Specifies the number of samples to be taken when `tsar` is in the online mode. By default, `tsar` collects five data samples.

`-p` *playback_file* Specifies the file from which `tsar` extracts data. *playback_file* is the output from `sdc`(8). Use of this option places `tsar` in playback mode. The default playback file is the `dcf` file in the current directory.

`-r` *record_file* Specifies the file where the data samples are to be recorded. In online mode, `sdc`(8) writes its output to *record_file*. In playback mode, *record_file* will be a copy of the playback file.

`-s` *start_time* Specifies the start time of the report. The time is specified using the form *hh*[:*mm*[:*ss*]]. This option can be used only in playback mode.

`-v` Specifies verbose output when `tsar` processes a source file. Output is written to `stderr`.

`-D` *name*[=*def*] Defines a symbol *name* to be used in the source file during execution. *def* can be a number or a character string. Character strings must be delimited by escaped double quotation marks. By default, *def* is defined as the number 1.

`-P` *path* Specifies the path where the `sdc`(8) binary is to be found. This option can be used only with the `-h` option. By default, if `sh` or `ksh` is used, the `PATH` in `/etc/profile` is searched. If `csh` is used, the path in your `.cshrc` file is searched.

source_files A command file which specifies the names of `tsar` programs.

NOTES

When using the `-h` option, ensure that the appropriate `.rhosts` file on the remote host is set up properly. Otherwise, `sdc`(8) may abort with a `Permission denied` error.

If more than one *source_file* is specified, `tsar` executes the files in sequence. The preambles, bodies, and postambles are processed as if they came from a single source file.

EXAMPLES

Example 1: The following example shows `tsar` being executed in online mode. `tsar` collects ten data samples at a rate of one sample every 5 minutes. The data is collected from a machine called `haze`, and data is written to the file `dcf`. As the data is collected, it is formatted according to the source file `mem.ts`. The formatted output is written to `stdout`.

```
$ tsar -h haze -n 10 -i 5m -r dcf mem.ts
```

Example 2: The following example shows `tsar` being executed in playback mode. The data in the file `dcf` is formatted according to the source file `rpt.ts`. The symbol `CPU_RPT` is defined as 2.

```
$ tsar -D CPU_RPT=2 rpt.ts
      or
$ tsar -p dcf -D CPU_RPT=2 rpt.ts
```

SEE ALSO

`sdc`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

A. V. Aho, B. W. Kernighan, P. J. Weinberger, *The AWK Programming Language*, Addison-Wesley, 1988.

NAME

turnacct – Controls process accounting

SYNOPSIS

```
/usr/lib/acct/turnacct  on
/usr/lib/acct/turnacct  off
/usr/lib/acct/turnacct  switch
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The turnacct shell script is an interface to the accton(8) command to turn process accounting on or off or to switch files for maintainability. A super user or a user who is in the group, adm, and has permission bit acct set in their user database (UDB) entry (see udbgen(8)) must invoke turnacct.

The turnacct shell script accepts the following mutually exclusive operands:

on Turns on process accounting. If the process accounting file /usr/adm/acct/day/pacct exists, accounting records are appended to it; otherwise, turnacct creates a new file and accounting records are written to it.

off Turns off process accounting.

switch Specifies that a new process accounting file should be created to maintain manageable files. turnacct moves the current process accounting file /usr/adm/acct/day/pacct to the next available file /usr/adm/acct/day/pacct* and then calls accton by using the file operand /usr/adm/acct/day/pacct. In this way, process accounting data is not lost and /usr/adm/acct/day/pacct remains the current file.

NOTES

The shell script, ckpacct(8), invokes turnacct to maintain process accounting files of a reasonable size. You should run ckpacct(8) periodically by using cron(8).

If the super user has run any accounting commands or shell scripts, you may set up the group ID and permissions of the accounting files so that any user in the group, adm, with the permission bit acct set in his or her UDB entry cannot run accton. To change all of the accounting file's group ID and permissions as necessary, use the csaperm(8) command.

FILES

`/usr/adm/acct/day/pacct`	Current process accounting file
`/usr/adm/acct/day/pacct*`	Switched process accounting files

SEE ALSO

`acct`(8), `accton`(8), `acctsh`(8), `ckpacct`(8), `cron`(8), `csa`(8), `csaperm`(8), `udbgen`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

turndacct – Controls daemon accounting

SYNOPSIS

```
/usr/lib/acct/turndacct  on daemon
/usr/lib/acct/turndacct  off daemon
/usr/lib/acct/turndacct  switch daemon
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The turndacct shell script is an interface to the csaswitch command (see csaswitch(8)) to turn daemon accounting on or off or to switch files for maintainability. A super user or a user who is in the group, adm, and has permission bit acct set in their user database (UDB) entry (see udbgen(8)) must invoke turndacct.

The turndacct shell script accepts the following mutually exclusive operands:

on — Turns on daemon accounting for the specified *daemons*.

off — Turns off daemon accounting for the specified *daemons*.

switch — Specifies that a new daemon accounting file should be created for each of the specified daemons to maintain manageable files. turndacct moves the current daemon accounting file /usr/adm/acct/day/daemonacct to the next available file, /usr/adm/acct/day/daemonacct#, and then calls csaswitch by using the file operand /usr/adm/acct/day/daemonacct. In this way, daemon accounting data is not lost, and /usr/adm/acct/day/daemonacct remains the current file.

daemon — Valid daemon names are nqs, tape and socket. This operand is required in each of the three command formats described previously.

NOTES

The ckdacct(8) shell script invokes turndacct to maintain process accounting files of a reasonable size. You should run ckdacct periodically by using cron(8).

If the super user has run any accounting commands or shell scripts, you may set up the group ID and permissions of the accounting files so that any user in the group, adm, with the permission bit acct set in his or her UDB entry cannot run accton. To change all of the accounting file's group ID and permissions as necessary, use the csaperm(8) command.

EXAMPLES

The following example turns off Network Queuing System (NQS) accounting:

```
turndacct off nqs
```

FILES

`/usr/adm/acct/day`	Directory that contains current daemon accounting files

SEE ALSO

`acct`(8), `accton`(8), `acctsh`(8), `ckdacct`(8), `cron`(8), `csa`(8), `udbgen`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

udbgen – Generates or maintains the user database

SYNOPSIS

```
/etc/udbgen [-n] [-q] [-u] [-v] [-p UDB_path] [-t type] [infile]
/etc/udbgen [-n] [-q] [-u] [-v] [-p UDB_path] [-c command]
/etc/udbgen -s [-q] [-u] [-p UDB_path]
/etc/udbgen -a [-A] [-R] [-p UDB_path]
/etc/udbgen -h
/etc/udbgen -m
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The udbgen command creates and maintains information in the user database (UDB). The UDB contains comprehensive information about each user and is the primary source of that information in the UNICOS operating system. For more information on the UDB, see the udb(5) man page.

The udbgen command is intended for use by administrators. The UNICOS system may enforce further restrictions on various types of administrators concerning who may change sensitive fields in the UDB.

After updating the UDB, udbgen enters the associated file maintenance phase and updates /etc/acid files, /etc/group files, and /etc/passwd files, because some commands assume that they exist. (See the acid(5), group(5), and passwd(5) man pages for more information on these files.) If an account or group ID appears in the UDB without being specified in the existing /etc/acid or /etc/group files, a name is created for it by appending the ID number to the prefix A- (for /etc/acid) or G- (for /etc/group). The ID suffix is a minimum of five digits, such as A-00921.

A new /etc/passwd file, created entirely from the UDB, replaces the existing /etc/passwd file.

The number of names created are displayed in the statistics output if the -q option has not been selected.

There are three possible sources for the information udbgen places in the UDB, as follows:

- The input file, *infile*. This file can be either in UDB source format, in the form of the /etc/passwd file, or in the form of the /etc/uentry file. UDB source format is discussed in the Creating Input for udbgen subsection. When reading /etc/passwd or /etc/uentry type source, udbgen either creates or updates entries. If the user name already exists in the UDB, the name is updated; otherwise, the name is created. This decision is made on every line of input.

  WARNING: Support for /etc/passwd and /etc/uentry formats is included only for upgrading from previous releases of UNICOS (before release 5.0); do not use these formats for any other purpose.

- A directive statement used with the -c option. This statement must be in UDB source format.

- Statements entered interactively. `udbgen` enters the interactive mode if an input file is not specified on the command line and `stdin` is connected to a terminal.

The `udbgen` command accepts the following options and operand:

`-a` Analyzes the fair-share resource assignments in the UDB (using the `shrtree`(8) utility) and reports any problems. A nonzero exit status occurs if any potential problems are found. This report is for use by the administrator as determined by site policy; no changes to the UDB are made. Only the `-A`, `-p`, or `-R` options can be used with the `-a` option.

`-A` In conjunction with the `-a` option, this option causes fair-share resource group analysis to be based on the `acids` field rather than the `resgrp` field in the UDB. This is to be used if the fair-share scheduler is run in its Share by Account mode. The `-A` option can be used only with the `-a` option and not in combination with the `-R` option.

`-c` *command*
Specifies command-line input. This option allows simple database changes to be made without an input file. The *command* argument is a UDB source format directive or set of directives. When this option is used, the `-q` option is automatically invoked and no input file is read (specifying *infile* is an error when this option is specified). For example, the following command line sets the shell for user `xyz` in the database to `/bin/csh`:

```
udbgen -c 'update:xyz: shell:/bin/csh:'
```

For a list of available directives and possible field values, see the Creating Input for `udbgen` subsection.

`-h` Displays the command syntax and quits. All other options are ignored.

`-m` Resets the UDB maximum user ID (UID) value to the highest non-deleted UID. The UDB maintains a record of the highest UID value ever assigned. This value is then used to determine the UID when creating a new record using the `:uid:next:` syntax. Since this value is never decremented, even when records are deleted, it is possible to reach large UID values while leaving large ranges of UIDs unused. This option allows you to reset this value without having to rebuild the entire UDB from source.

All other options are ignored.

NOTE: Take care when using this option. For example, if the UDB record with the highest UID is deleted and then the `-m` option is used to reset the maximum UID value, the deleted UID value might be assigned to another user. If the original user had any files remaining on the system, they would come under the ownership of the newly created user.

`-n` Adds new user names only and does not affect existing records. Without this option, `udbgen` adds new user names and creates, deletes, and updates existing records in the UDB.

`-p` *UDB_path*
Sets the path name for access to the UDB files. The default path name is `/etc`. This option allows private or test versions of the UDB files to be created and maintained. The path name must end with a directory name. For example, if you want to specify that the UDB files are to be in the `/c/abc/UDBtest` directory, you would enter the following command line:

```
udbgen -p /c/abc/UDBtest
```

If you use the `-p` option and you are not appropriately authorized, `udbgen` eliminates all of its special abilities and runs with the normal abilities of the user. This behavior means that such users are constrained by the normal access controls of UNICOS.

The path names for files in the FILES section can be changed with this option.

`-q` Specifies quiet mode. This option prevents informative messages from appearing.

`-R` In conjunction with the `-a` option, this option causes fair-share resource group analysis to be based on the `resgrp` field in the UDB. This is to be used if the fair-share scheduler is run in its Share by User mode. This is the default mode if the `-a` option appears alone. The `-R` option can be used only with the `-a` option and not in combination with the `-A` option.

`-s` Updates `/etc/acid`, `/etc/group`, and `/etc/passwd` files to accurately reflect the contents of the UDB. Only the `-p`, `-q`, and `-u` options can be used with `-s`.

`-t` *type*
Specifies the format of the data in *infile*. The *type* argument may be one of the following values:

Value	Description
`passwd`	Password format. This type is provided only for upgrading from previous releases of UNICOS (before release 5.0); do not use this type for any other purpose, because it accommodates only a small part of the information contained in the UDB.
`uentry`	`uentry` format. This type is provided only for upgrading from previous releases of UNICOS (before release 5.0); do not use this type for any other purpose, because it accommodates only a small part of the information contained in the UDB.
`udb`	UDB format.

Ordinarily it is not necessary to specify the input format because `udbgen` analyzes the first line of the file to determine which format is in use. This option bypasses this analysis and must be used if the following message is displayed:

```
udbgen cannot determine input style
```

If *infile* is associated with a terminal device, this option is ignored and *type* is set to UDB format.

`-u` Reports database access statistics.

`-v` Verifies the accuracy of the input file only. The database is not updated with this option.

infile Name of the source data file. The default is `stdin`. If the file is associated with a terminal device, `udbgen` enters interactive mode and displays a prompt (`>`) to `stdout` before every read. The end of interactive input must be specified by a new line that contains `quit` or `q`.

The `udbsee`(1) command can be used to produce a file suitable as a source data file.

Creating Input for udbgen

A source data file (or directive used with the `-c` option or used interactively) contains the following information:

- Block introduction field name, usually consisting of a command to `udbgen` and a user name, separated by colons. See the Block Introduction Field Names subsection.
- Field names other than the user name, sometimes with editing suffixes, separated by colons. See the Field Names and Values subsection.
- Field values. See the Field Names and Values subsection.

To create an input file for `udbgen` to use in updating the UDB, you must follow these file format rules:

- Comments can be placed in the source code by using the character # (similar to shell script comments). Comments continue from the # to the end of the line on which the # appears. Comments are not recognized within colon-delimited fields. Blank lines are allowed in the source code.
- White space (blanks, tabs, and so on, as determined by `isspace` (see `ctype`(3C)) may appear freely in the source, but it is removed during parsing, except within some colon-delimited fields where white space is part of the field value. (The Field Names and Values subsection states where white space is retained in fields.)
- Each default block must begin on a new line with the field name `default`. (See the Block Introduction Field Names subsection.)
- Each user description block must begin on a new line with one of the field names `create`, `delete`, or `update`. This is how one user description is separated from the next. A user ID must be specified when `create` is used. (See the Block Introduction Field Names subsection.)
- Each field name must be followed by a colon-delimited field, according to the following format:

  field_name : *field_value* :

  This name/value pairing is implicit in the source definition and must, but for one exception, be followed exactly for the input file to be accepted. The exception occurs with the special word `quit` or `q`, which is recognized only in interactive mode as an input terminator.
- Fields that have batch and interactive components must have the appropriate indicator (`[b]` or `[i]`) following the field name (separating white space is optional). For example, the following line sets the interactive tape limit of tape type 0 to 1.

```
jtapelim[i][0] :1:
```

Block Introduction Field Names

The special block introduction field names update the UDB and are used in the input file, with the `-c` option as directive statements, and with `udbgen` interactively. The field names consist of a command to `udbgen` and a user name, separated by colons. All field names must be entered in lowercase in the source file. Fields not included in a user description are set to an appropriate default value. Strings (except for `passwd`) are null, fields that have a default value are set to that value (for a description of `default`, see the Global Default Table subsection), and other numeric and bit fields are set to `0`. An unspecified list of accounts (`acids`) or groups (`gids`) means that every entry in the associated array is set to `-1`. An undeclared password is set to the string `*`.

`create :`*user_name*`:uid :`*n*`:`

Adds the specified user's definition to the UDB. This is one of the recognized initial reserved words for a definition block and must appear first. If a record of this name is already present, a warning message is displayed and the existing record is not changed. The user ID must be specified with the `create` command.

User names are limited by the UNICOS operating system to 8 characters; `udbgen` issues a warning if this restriction is violated.

`delete :`*user_name*`:`

Removes the specified user's definition from the UDB. This is one of the recognized initial reserved words for a definition block and must appear first. While other information may be present in this definition block, it is ignored. A warning message is displayed if the specified user name does not appear in the database.

`global :default:`

Changes the values of the default fields named in the block. For a description of `default`, see the Global Default Table subsection.

`global :tmap:`

Changes the names of the eight tape types defined in the global tape map. For a description of `tmap`, see the Global Tape Name Map subsection.

`quit`
`q`

Simulates the end of a file. This field name is recognized only in interactive mode (see the previous description of *infile*) and is recognized only if it is alone on a line with no extraneous white space. This is the single exception to the pairing rule.

`see :`*user_name*`:`
`see` *user_name*

Displays a UDB entry by executing the `udbsee`(1) command. This field name is recognized only in interactive mode (see the previous description of *infile*) and must be alone on a line. This action does not disturb any ongoing input to `udbgen`. This field accepts either the colon delimiters or white space separating the field name from the field value.

`update :`*user_name*`:`

Updates fields defined in this block in the UDB. The user name must exist in the UDB. This is one of the recognized initial reserved words for a definition block and must appear first. Fields not mentioned retain their original values. Fields that are either bit or numeric arrays have the values supplied in an `update` block processed as directed by the editing suffix immediately following the field name. For a detailed explanation of the editing suffixes, see the Field Names and Values subsection.

If, during an `update`, the user ID is changed, the record is first deleted from the database and then added with the new user ID. No checking is done to ensure that no other user is assigned to the new user ID.

Field Names and Values

Field names describe various types of data contained in the UDB, including user limits, quotas, and privileges. Editing suffixes immediately follow the field name. The editing suffix `=` may be added to a field name to indicate that the value following is the value to be entered in the UDB. The `=` suffix is optional in all cases. The suffixes `+` and `-` allow you to update a field value that is either bit or numeric arrays using the `update` block. Field editing is allowed only with `update`.

In `update` mode, the numeric arrays can be altered with the editing suffixes `+` or `-`. Assuming a field is specified more than once in a definition block, processing is done as follows:

1. If replacement values are provided, they are entered in the array; otherwise, the values from the existing record are used.
2. All values appearing with a `+` suffix are appended to the array.
3. Any duplicates are deleted.
4. All values appearing with a `-` suffix that exist in the array are removed.
5. The array is packed to move all empty space to the end. At no time may the result array exceed its limit of entries; therefore, in full or nearly full arrays it is possible to get space-exceeded warnings even when the final result of the editing would not exceed the maximum array size. To avoid this, you can split the `update` block into two separate operations and then delete before appending, as in the following example:

```
update :name: acids- :23,45,87: acids+ :1,2,3,4:
```

To remove all values from an array (such as `acids`), use the following method:

```
update :name: acids= :1: acids- :1:
```

The value of `1` is arbitrary; any legal value may be used.

For example, if, for the user `anne`, the `acids` field contains the values 2, 34, and 45, you can remove 34 from the list by entering the following command line:

```
update :anne: acids-:34:
```

To add 97 to the list of account IDs, enter the following command line:

```
update :anne: acids+:97:
```

To change the account ID list to 1, 3, and 5, enter one of the following command lines:

```
update :anne: acids:1,3,5:
update :anne: acids=:1,3,5:
```

Field values are alphabetic and numeric, depending on the field. Numeric values are assumed to be decimal unless specifically noted as octal. Negative values are not allowed in any numeric field. Certain numeric fields also allow a special alphabetic notation for maximum values (all limits fields are in this category). This notation uses the words `none` or `unlimited` in lowercase or uppercase. (The words `infinite` and `infinity` in lowercase or uppercase are also supported, for compatibility reasons.) The appropriate numeric value is then stored in the UDB.

For example, to set an unlimited batch job CPU limit, enter the following command line:

```
jcpulim[b] :unlimited:
```

To prohibit interactive SDS usage, enter the following command line:

```
jsdslim[i] :none:
```

Internally, `none` results in a value of 0 and `unlimited` is a large value dependent on the data type of the field.

The following list contains field name / field value pairs, their descriptions, and possible values:

`acids =|+|- :`*n1*`, `*n2*`, ..., `*nn*`:`
`acids =|+|- :`*a1*`, `*a2*`, ..., `*an*`:`
: Account IDs. This is a list of up to 64 (set by MAXVIDS) numeric account IDs or account names separated by commas. If account names are used, they must be found in the `/etc/acid` file as it existed before `udbgen` was executed.

`age :`*max*`, `*min*`:`
`age :force:`
`age :superuser:`
: This field name is obsolete, but it continues to be supported for compatibility reasons. For a description of its replacement, see the `pwage` field. Only an appropriately authorized user can change this field.

`archlim :`*n*`:`
: The maximum amount of the user's disk space protected from data migration. Users can protect files from migration by listing them in their `.keep` file. The files so listed are protected up to the number of disk blocks specified here.

`archmed :`*n*`:` Archive media selector for data migration. The value of this field determines to which media migrated files owned by the user are written. The media associated with the value 0 is the default media. The table of media is established in the DMF run-time configuration file. For complete information on establishing archive media, see "Data Migration" in the *Cray Data Migration Facility (DMF) Administrator's Guide*, Cray Research publication SG–2135.

`batchhost :`*text*`:` Host name for the last batch request. Up to 31 characters can be specified; white space is not removed.

`batchtime :`*seconds*`:`
Time of last batch request (GMT seconds).

`comment :`*text*`:` A comment consisting of a maximum of 39 characters; white space is not removed.

`comparts =|+|- :`*name1*`, `*name2*`, ..., `*namen*`:`
Valid security compartment names. Only an appropriately authorized user can change this field.

`cpasswd :`*password*`:`
The clear text password to be encrypted and stored in the user's record. The password content is not validated. Only an appropriately authorized user can change this field.

NOTE: When creating a new account, see the `pwage` field. (Setting the `pwage` field to `force` ensures that the user will be required to enter a new password.)

`cpuquota :`*vv.v*`:` User CPU quota in seconds. If this field is nonzero, CPU quota is enforced to the value *vv.v* seconds. This value is stored internally in tenths of seconds but is expressed externally in seconds.

`cpuquotaused :`*vv.v*`:`
Amount of CPU quota used in seconds. This value is stored internally in tenths of seconds but is expressed externally in seconds.

NOTE: If this field is changed while the system is in multiuser mode, execute the command `shrsync -q` to synchronize the information in active lnodes with the changes in the UDB. For more information, see `shrsync`(8).

`defcomps =|+|- :`*name1*`, `*name2*`, ..., `*namen*`:`
Default security compartment names. Only an appropriately authorized user can change this field.

`deflvl :`*n*`:` Default security level. Only an appropriately authorized user can change this field.

`dir :`*directory*`:` Default login directory consisting of a string of up to 63 characters.

`disabled :`*n*`:` User disabled indicator. If *n* is `0`, the user is enabled; if *n* is `1`, the user is disabled. Only an appropriately authorized user can change this field.

`gids =|+|- :`*n1*`, `*n2*`, ..., `*nn*`:`
`gids =|+|- :`*g1*`, `*g2*`, ..., `*gn*`:`
Group IDs. This is a list of up to 64 (set by MAXVIDS) numeric group IDs (GIDs) or group names separated by commas. If group names are used, they must be found in the `/etc/group` file as it existed before `udbgen` was executed. A warning message is issued if no GIDs are defined in a record. The UNICOS operating system is not intended to be operated without a GID for each user, but this is not a fatal error in order to be compatible with previous systems. This is especially important if the site uses network information systems (NIS) (formerly called yellow pages) software, because NIS does not work if a user does not have a GID.

`intcat =|+|- :`*name1*`, `*name2*`, ..., `*nameN*`:`
Default categories. Only an appropriately authorized user can change this field.

`intcls :`*n*`:` Default class. Only an appropriately authorized user can change this field. This field is obsolete.

`jcpulim[b] :`*n*`:`
`jcpulim[i] :`*n*`:` Job CPU time limit, in seconds, for batch (`[b]`) or interactive (`[i]`) jobs. If *n* is `unlimited`, no restriction on the amount of CPU usage is applied. If this field is not specified, the appropriate value from the default table is used. The release default is `unlimited`.

`jfilelim[b] :`*n*`:`
`jfilelim[i] :`*n*`:` Per-job file allocation limit in 512-word blocks for batch (`[b]`) or interactive (`[i]`) jobs. If *n* is `zero` (default), there is no limit.

`jmemlim[b] :`*n*`:`
`jmemlim[i] :`*n*`:` Job memory limit, in 512-word blocks, for batch (`[b]`) or interactive (`[i]`) jobs. If *n* is `unlimited`, no restriction on the amount of memory usage is applied. If this field is not specified, the appropriate value from the default table is used. The release default is `unlimited`.

`jmppbarrier[b] :`*n*`:`
`jmppbarrier[i] :`*n*`:`
NOTE: Not implemented. Job massively parallel processing (MPP) barrier limit for batch (`[b]`) or interactive (`[i]`) jobs. (This field is present but not functional for Cray Research systems without a Cray MPP system.) If *n* is `unlimited`, no restriction on the number of barriers is applied. If this field is not specified, the appropriate value from the default table is used. The release default is `none`.

`jmpptime[b] :`*n*`:`
`jmpptime[i] :`*n*`:` Job MPP maximum reservation time limit, in wall-clock seconds, for batch (`[b]`) or interactive (`[i]`) jobs. (This field is present but not functional for Cray Research systems without a Cray MPP system.) If *n* is `unlimited`, no restriction on the MPP reservation time is applied. If this field is not specified, the appropriate value from the default table is used. The release default is `none`.

`jpelimit[b] :`*n*:
`jpelimit[i] :`*n*: Job MPP processing elements (PE) limit, for batch (`[b]`) or interactive (`[i]`) jobs. (This field is present but not functional for Cray Research systems without a Cray MPP system.) If *n* is `unlimited`, no restriction on the number of PEs is applied. If this field is not specified, the appropriate value from the default table is used. The release default is `none`.

`jproclim[b] :`*n*:
`jproclim[i] :`*n*: Job process limit for batch (`[b]`) or interactive (`[i]`) jobs. If this field is not specified, the default table entry will be used. The default table value for `jproclim` of the appropriate run class is either the release default returned from `sysconf(_SC_CHILD_MAX)` or the previously specified default table value. Setting *n* to `zero` or `none` indicates unlimited.

`jsdslim[b] :`*n*:
`jsdslim[i] :`*n*: Job secondary data segment limit, in 512-word blocks, for batch (`[b]`) or interactive (`[i]`) jobs. (This field is present but not functional for Cray Research systems without an SSD.) If *n* is `unlimited`, no restriction on the amount of secondary data segment usage is applied. If this field is not specified, the appropriate value from the default table is used. The release default is `none`.

`jshmsegs[b] :`*n*:
`jshmsegs[i] :`*n*: Maximum number of created shared memory segments for batch (`[b]`) or interactive (`[i]`) jobs. This field is always present, but is functional only for CRAY T90 systems supporting shared memory. If *n* is `unlimited`, the number of segments is limited by a configured upper limit (see `sys/config.h`). If *n* is `none`, shared memory cannot be used.

`jshmsize[b] :`*n*:
`jshmsize[i] :`*n*: Total shared memory segment size, in 512-word blocks, for batch (`[b]`) or interactive (`[i]`) jobs. This field is always present, but is functional only for CRAY T90 systems supporting shared memory. If *n* is `unlimited`, the number of segments is limited by a configured upper limit (see `sys/config.h`). If *n* is `none`, shared memory cannot be used.

`jsocbflim[b] :`*n*:
`jsocbflim[i] :`*n*: Total socket buffer space, in 512-word blocks, for batch (`[b]`) or interactive (`[i]`) jobs. If *n* is set to 0, unlimited use is allowed. (This is the default in the released system.)

`jtapelim[b][`*t*`] :`*n*`:`
`jtapelim[i][`*t*`] :`*n*`:`
Job tape unit limit for batch (`[b]`) or interactive (`[i]`) jobs. The value of *t*, the tape type, can be an integer in the range 0 through 7 or one of the names defined in the global tape name map. (For more information, see the Global Tape Name Map subsection.) By convention and for compatibility with previous releases, the value 0 corresponds to round tape types (`TAPE`) and the value 1 corresponds to cartridge tape types (`CART`). No names are defined by default in `tmap`. The maximum value of *n* is `unlimited` (254 for this type) and the released value is `none`. If this field is unspecified, the release default or the value previously specified in the default table will be used.

`limflags =|+|- :`*octal*`:`
`limflags =|+|- :`*name*`, `*name*`, ...:`
This field name is obsolete, but it continues to be supported for compatibility reasons.

`logfails :`*n*`:` Number of consecutive login attempt failures since the last successful attempt. Only an appropriately authorized user can change this field.

`loghost :`*text*`:` Host name for the last login. Up to 31 characters can be specified, and white space is not removed.

`logline :`*text*`:` Line name used for last login. Up to 15 characters can be specified, and white space is not removed.

`logtime :`*seconds*`:` Time of last login (GMT seconds).

`maxcls :`*n*`:` Maximum class. Only an appropriately authorized user can change this field. This field is obsolete.

`maxlvl :`*n*`:` Maximum security level. Only an appropriately authorized user can change this field.

`minlvl :`*n*`:` Minimum security level. *n* may be any integer from 0 through 16; the default is 0. The value of `minlvl` should be less than, or equal to, the value of `maxlvl`. Only an appropriately authorized user can change this field.

`mincomps =|+|- :`*name1*`, `*name2*`, ..., `*namen*`:`
Minimum security compartment names. Only an appropriately authorized user can change this field.

`nice[b] :`*n*`:`
`nice[i] :`*n*`:` Nice bias in the range $0 \le n \le 19$ for batch (`[b]`) or interactive (`[i]`) processes. If this field is not specified, the value from the default table or the released default value of 0 is used.

`parentuid :`*n*`:` UID of the group administrator for this entry.

passwd :*encrypted_password*:
The encrypted password to be stored in the user's record. The password content is not validated. Only an appropriately authorized user can change this field.

NOTE: When creating a new account, see the pwage field. (Setting the pwage field to force ensures that the user will be required to enter a new password.)

pcorelim[b] :*n*:
pcorelim[i] :*n*: Per-process maximum core file limit in units of 512 words for batch ([b]) or interactive ([i]) processes. This represents the maximum size of a core file that the process can create. If the size of the process is larger than this limit, a partial core file will be created. A partial core file contains just the user and user common structures. If *n* is unlimited, no restriction on the size of the core file is applied. If this field is not specified, the appropriate value from the default table is used. The release default is unlimited.

pcpulim[b] :*n*:
pcpulim[i] :*n*: Per-process CPU limit, in seconds, for batch ([b]) or interactive ([i]) processes. If *n* is unlimited, no restriction on the amount of CPU usage is applied. If this field is not specified, the appropriate value from the default table is used. The release default is unlimited.

permbits =|+|- :*octal*:
permbits =|+|- :*name*, *name*, ...:
User permission bits supported by Cray Research. The following names are recognized values, the octal equivalents, and the corresponding libudb(3C) bit names (for a description of valid permbits, see libudb(3C)):

Value	Octal Value	libudb(3C) Bit Name
bypasslabel	00000000001	PERMBITS_BYPASSLABEL
realtime	00000000002	PERMBITS_REALTIME
nobatch	00000000004	PERMBITS_NOBATCH
noiactive	00000000010	PERMBITS_NOACTIVE
yp	00000000020	PERMBITS_YP
CAUTION: If yp is on, password aging no longer functions (see pwage field).		
plock	00000000040	PERMBITS_PLOCK
cpu-dedicate	00000000100	PERMBITS_DEDIC
acct	00000000200	PERMBITS_ACCT
suspend-resume	00000000400	PERMBITS_SUSPRES
resource	00000001000	PERMBITS_RESLIM
mount	00000002000	PERMBITS_MOUNT
system-param	00000004000	PERMBITS_SYSPARAM
chroot	00000010000	PERMBITS_CHROOT
sigany	00000020000	PERMBITS_SIGANY

Value	Octal Value	libudb(3C) Bit Name
tape-manage	00000040000	PERMBITS_TAPEMANAGER
mknod	00000100000	PERMBITS_MKNOD
diag	00000200000	PERMBITS_DEVMAINT
nice	00000400000	PERMBITS_NICE
id-change	00001000000	PERMBITS_ID
acctid	00002000000	PERMBITS_ACCTID
system-restricted	00004000000	PERMBITS_RESTRICTED
chown	00010000000	PERMBITS_CHOWN
mlsmount	00020000000	Unused. This permbit is available to assign to user accounts, but it no longer grants special abilities.
guard	00040000000	PERMBITS_GUARD
wrunlab	00100000000	PERMBITS_WRUNLABEL
askacid	00200000000	PERMBITS_ASKACID
guest	00400000000	PERMBITS_GUEST
guestadm	01000000000	PERMBITS_GUESTADM
groupadm	02000000000	PERMBITS_GROUPADM
ipc-persist	04000000000	PERMBITS_IPCPERSIST

`permits =|+|- :`*name1*`, `*name2*`, ..., `*namen*`:`
UNICOS security permissions. Only an appropriately authorized user can change this field. For a description of valid `permits`, see `libudb`(3C).

`pfdlimit[b] :`*n*`:`
`pfdlimit[i] :`*n*`:` Per-process maximum open file limit for batch (`[b]`) or interactive (`[i]`) processes. This represents the maximum number of file descriptors that a process belonging to this user can allocate. At minimum, *n* must be at least the value of `OPEN_MAX` (64); `udbsee`(1) does not accept a value less than `OPEN_MAX`. In addition, the system imposes an upper limit of `K_OPEN_MAX`; `udbgen` allows values greater than `K_OPEN_MAX`, but the system forces the process' open file limit to `K_OPEN_MAX`. If this field is not specified, the appropriate value from the default table is used. The released default is 255.

`pfilelim[b] :`*n*`:`
`pfilelim[i] :`*n*`:` Per-process file allocation limit, in 512-word blocks, for batch (`[b]`) or interactive (`[i]`) processes. If *n* is 0 (default), there is no limit.

`pmemlim[b] :`*n*`:`
`pmemlim[i] :`*n*`:` Per-process memory limit, in 512-word blocks, for batch (`[b]`) or interactive (`[i]`) processes. If *n* is `unlimited`, no restriction on the amount of memory usage is applied. If this field is not specified, the appropriate value from the default table is used. The release default is `unlimited`.

`pmpptime[b] :`*n*`:`
`pmpptime[i] :`*n*`:` Per-process MPP maximum reservation time limit, in wall-clock seconds, for batch (`[b]`) or interactive (`[i]`) processes. (This field is present but not functional for Cray Research systems without a Cray MPP system.) If *n* is `unlimited`, no restriction on the MPP reservation time is applied. If *n* is `none`, the MPP system is inaccessible by the process. If this field is not specified, the appropriate value from the default table is used. The release default is `none`.

`psdslim[b] :`*n*`:`
`psdslim[i] :`*n*`:` Process secondary data segment limit in 512-word blocks for batch (`[b]`) or interactive (`[i]`) jobs. (This field is present but not functional for Cray Research systems without an SSD.) If *n* is `unlimited`, no restriction on the amount of secondary data segment usage is applied. If this field is not specified, the appropriate value from the default table is used. The release default is `unlimited`.

`pwage :force, superuser,` *max*`,` *min*`,` *time*`:`
`pwage :force, superuser,` *max*`,` *min*`, +`*age*`:`
`pwage :`*max*`,` *min*`:`

`pwage ::`

Password age control fields are manipulated with `pwage`. The keywords `force` and `superuser` are used to set or clear the `PWFL_FORCE` and `PWFL_SUPERUSER` flags. If either keyword is preceded with a minus sign, the flag is turned off. If a keyword has a plus sign or no sign, the flag is turned on. For example, to set `PWFL_FORCE`, use the keyword `+force` or `force`. To turn off `PWFL_SUPERUSER`, use the keyword `-superuser`. If the `superuser` flag is turned on, password aging is to be enabled and only the super user is allowed to change the user's password. Both the force and super-user flags cannot be set on at the same time.

Flags that need not be changed should not be named with a keyword, because a keyword will force the on or off state; omitting the keyword will leave the flag as it is in the record. If a keyword is omitted, also omit its separating comma.

The *max*, *min*, and *time* fields control how old a password can become (*max*), how long it has to exist before being changed (*min*), and when it was changed (*time*). Neither *max* nor *min* may exceed 64 weeks. Either of these values may be expressed in units other than the default [`w`]eeks, namely [`d`]ays or [`s`]econds, although it must be possible to express the value as an integer. For example, a minimum of one week and a maximum of 12 weeks could be written: "`12, 1`", "`12w, 1w`", "`84d, 7d`", or "`7257600s, 604800s`". It is not necessary that both values have the same unit designator, although it is wise to use uniform units whenever possible, for readability. Internally the values are maintained in seconds. The value of *max* must never be less than the value of *min*. The *time* field is the value of the system time of day clock expressed as an integer.

The second form of `pwage` shown above is distinguished by a plus sign preceding the last numeric value. This causes age to be interpreted as the amount of time to subtract from the time "now" to result in a value of the time of day clock which is *age* units in the past and then store that value in the time field. Usually, this is intended to make it easy to set the current time in the field by using the value +0 as the age. Units are allowed on this value just as with *max* and *min*. For example, to set a time 14 days ago, write: "`,,+2`", "`,,+14d`", or "`,,+1209600s`". Note that two commas must precede the time if *max* and *min* ages are not specified, since this part of the directive is position-dependent, requiring all left context in order to determine the meaning of the particular value string.

The third form of `pwage` shown above is used to alter the *max* and *min* age fields.

The fourth form of `pwage` shown above is used only to remove age control from a record. All age control fields are set to a zero or null state, which removes age control totally. Once this has been done, all historical information is lost from the record.

Note that when the YP permbit is set (see the `permbits` field) and the password is being accessed from the database, password aging is disabled.

`resgrp :`*name*`:`
`resgrp :`*n*`:` The user name or user ID of the resource group to which this user belongs. That a user name has been specified is an initial assumption. If there is no existing record in the database with that name and the field is a number, the value is assumed to be a user ID. Care must be taken when using names during creation to ensure that any names used here have already been created in the database. The `udbsee`(1) command always provides user IDs to avoid this problem.

`root :`*directory*`:` Login root directory; the root directory of the user's login process is set to *directory* with `chroot`(2). A string of up to 63 characters may be specified.

`shares :`*n*`:` Allocated shares for the fair-share scheduler. The default for *n* is 0, which means that no shares are allocated.

`shcharge :`*vv.vv*`:` Long-term accumulated costs for the fair-share scheduler. This is a floating-point value field.

NOTE: The UDB floating-point value fields are stored in Cray Research format. If these fields are accessed from an IEEE CPU, the UDB library performs the necessary conversion.

`shell :`*sh_name*`:` Default login shell. Up to 63 characters may be specified. The default value for *sh_name* is `/bin/sh`.

`shextime :`*n*`:` Time last lnode was freed (for the fair-share scheduler).

`shflags =|+|- :`*octal*`:`
The fair-share scheduler `l_flags`.

`shusage :`*vv.vv*`:` The fair-share scheduler decaying accumulated costs. This is a floating-point value field.

NOTE: The UDB floating-point value fields are stored in Cray Research format. If these fields are accessed from an IEEE CPU, the UDB library performs the necessary conversion.

`sitebits =|+|- :`*octal*`:`
`sitebits =|+|- :`*name*`, `*name*`, ...:`
User permission bits reserved by the site. Names `site1` (octal 01) through `site32` (octal 020000000000) are recognized values.

`trap :`*n*`:` UNICOS security trap field. The default for *n* is 0, which means the user login is not trapped. If *n* is `1`, the user login is trapped. Only an appropriately authorized user can change this field.

`uid :`*n*`:`

`uid :next:` If a number is specified, that value is assigned to the user ID. If the value is `next`, the next higher user ID from the UDB is assigned to this user. The maximum value of a user ID is one less than the number returned from `sysconf(_SC_UID_MAX)` (a system call). This is presently 59999 (60000 – 1). Only an appropriately authorized user can change this field.

`valcat=|+|- :`*name1*`, `*name2*`, ..., `*namen*`:`
Authorized categories. Only an appropriately authorized user can change this field.

Global Default Table

The `global :default:` block introduction phrase indicates that the subsequent field names and field value pairs are to be assigned to the global default table. There are default table entries for the fields listed in the following table:

Field Name	Released Default
`jcpulim[b|i]`	`unlimited`
`pcpulim[b|i]`	`unlimited`
`jmemlim[b|i]`	`unlimited`
`pmemlim[b|i]`	`unlimited`
`jsdslim[b|i]`	`none`
`psdslim[b|i]`	`none`
`jfilelim[b|i]`	`unlimited`
`pfilelim[b|i]`	`unlimited`
`jtapelim[b|i][t]`	`none`
`nice[b|i]`	`0`
`jproclim[b|i]`	System's default
`jpelimit[b|i]`	`none`
`jmpptime[b|i]`	`none`
`jmppbarrier[b|i]`	NOTE: Not implemented. `none`
`pmpptime[b|i]`	`none`
`pcorelim[b|i]`	`unlimited`
`pfdlimit[b|i]`	`255`
`jshmsegs[b|i]`	`0`
`jshmsize[b|i]`	`0`

The release defaults are applied by udbgen when it updates a UDB that has a default table containing all zeroes. To create a default table in an existing UDB, execute udbgen -c'#'. This is an empty modification request, but it causes the default table to be created with the released defaults. To change one or more entry, write the appropriate directive line.

To set the interactive job CPU limit to 300 seconds and the interactive process memory limit to 8000 clicks, enter the following command:

```
global :default: jcpulim[i] :300: pmemlim[i] :8000:
```

The defaults from the table above are supplied when new records are created and the fields that have defaults are not named or have empty value fields (::). After the defaults have been applied to a record, there is no way to determine that the value has been derived from the default. This means that existing records will not track changes to the default values. To apply the defaults to an existing record, an update directive can be written as follows, where user is a known name in the UDB:

```
update :user: jcpulim[i] :: pmemlim[i] ::
```

Global Tape Name Map

The global :tmap: global tape name map provides a common place to store the names of the tape devices so the same names can be used everywhere. After names have been placed in the map, udbgen will accept the names in place of the ordinal (*t*) in the jtapelim[b|i][*t*] directives. Tape names are from 1 to 8 characters in length and are case-sensitive. Make sure these names match those known to the tape subsystem. Access to the map information is through the library routine getudbtmap (see libudb(3C)) and the interface is defined in include/udb.h. No names are defined by default, but the example below shows how to define the names TAPE and CART as ordinals 0 and 1:

```
global :tmap: tmap[0] :TAPE: tmap[1] :CART:
```

Once this has been done, jtapelim directives can be written as follows:

```
update :user: jtapelim[b][CART] :2:
      jtapelim[b][TAPE] :1: jtapelim[i][CART] :1:
      jtapelim[i][TAPE] :none:
```

CAUTION: After the names have been created and are used, changing them can cause problems, because the names may become part of many users' files.

NOTES

If this command is installed with a privilege assignment list (PAL), a user who is assigned the following privilege text upon execution of this command is allowed to perform the actions shown:

Privilege Text	Action
chgany	Allowed to change all UDB fields.
nosec	Allowed to change all UDB fields, except sensitive fields.

If this command is installed with a PAL, a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`	Allowed to change all UDB fields.
`sysadm`	Allowed to change all UDB fields, except sensitive fields.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to change all UDB fields.

MESSAGES

Many error messages are possible from `udbgen`. Generally, they fall into the following five categories:

- Warning and informative messages
- Usage errors
- Source errors
- I/O errors
- Messages from the UDB manager functions

The messages pinpoint the problem and should be self-evident. Errors found in the input file give the line number and, when possible, the user name and field in error. Source errors and warnings prohibit updating of the record in error on the database. All messages are sent to the `stderr` file. Usage messages exit with an error code of 1. If any warning messages occur, the exit code is 2 and fatal errors exit with a value of 3.

BUGS

The `-v` option does not catch all problems because the database is not actually updated.

It is possible to specify encrypted passwords that contain characters outside of the defined alphabet. This is allowed in order to handle old `/etc/passwd` and `/etc/uentry` files where deleted users remain in the file but with an encrypted password that is impossible to match.

When the special *uid* value `next` is used to assign the user ID, and multiple `udbgen` processes are active, the same user ID could be assigned to more than one user.

Because of the definition of the source language, a block is not fully processed in interactive mode until the next `create`, `delete`, `update`, or `quit` block is seen. If this happens, error messages for the previous block are issued after the first line of the next block is entered.

EXAMPLES

Example 1: Batch UDB creation. The following example is taken from the output of udbsee(1), which is designed to create an output file acceptable as input to udbgen. This example builds a new UDB if one does not exist or places the named definitions in an existing database, replacing any entries with a new definition.

```
#
#      Created by udbsee -a
#      Thu Apr 16 16:39:30 1992
#
create :fil:  uid    :64:
       comment      :F. I. Ling:
       passwd :TI..ekLKRkiZI:
       pwage  :-force, -superuser,
               12w, 1d, 703460104: #Max, min age, changed 04/16/92 16:35:04
       gids   :102,  64,    1,     2,     3,
               14,   15,    16:
       acids  :61,   16,    14,    13,    12,
               1:
       dir    :/w/fil:
#
create :jed:  uid    :66:
       comment      :John E. Doe:
       passwd :9VrQEF/E1MI.C:
       pwage  :+force, -superuser,
               4w, 3d, 703460104: #Max, min age, changed 04/16/92 16:35:04
       gids   :102,  66:
       acids  :317:
       dir    :/w/jed:
#
create :fff:  uid    :77:
       comment      :Frank F. Frost:
       passwd :jpAB9T5FZKnQA:
       pwage  :-force, +superuser,
               12w, 12w, 703460104: #Max, min age, changed 04/16/92 16:35:04
       gids   :101,  1027,  116:
       acids  :48:
       dir    :/u/fff:
```

Example 2: Batch UDB update. To change the maximum password age to four weeks counting from the present, write the source file shown below. The minimum value for user `fff` must be altered to satisfy the rule that maximum age must not be less than minimum age. For the other two users, minimum age is not changed.

```
update :fil: pwage:4w,,+0:
update :jed: pwage:4w,,+0:
update :fff: pwage:4w,4w,+0:
```

Example 3: Interactive entry creation. To add a new user named `jjj` to the UDB, either use an input file, as in the preceding example, or do this interactively from the keyboard, as follows:

```
$udbgen
udbgen: 1>create:jjj: uid:109: comment:Jane J. Jones:
udbgen: 2>passwd:EfgTYUnnjuKP:
udbgen: 3>pwage:+force,4,1:
udbgen: 4>dir:/w/jjj: gids:64,1: acids:297:
udbgen: 5>quit
Added 1 record
$
```

Example 4: Report generation (`-a` option). Running `udbgen` with the `-a` option invokes the `shrtree`(8) evaluation command. Sample output from the `-a` option is shown in this example. For an explanation of the output, see `shrtree`(8).

```
gust.1-> udbgen -aR

DISPLAY OF SHARE TREE
---------------------
UDB path:      DEFAULT
Analyzed:      By UID
Format:        Groups only
Maxgroups:     4
Node:          ALL
Group Count:   17
Account Count: 0
User Count:    1402
Warnings:      10
Errors:        0

Warning Count: 4     (Nc) Group has no references
Warning Count: 2     (Zs) User has zero shares
Warning Count: 4     (Zs) Group has zero shares

  Type      Name    ID    Status  Description
--------- -------- ----- -------- --------------------------------
**WARN*** Serv      8001       10 Nc: Group has no references
**WARN*** Unknown   8393     1000 Zs: Group has zero shares
**WARN*** unknown     12     1001 Zs: User has zero shares
**WARN*** Country   8359     1010 Nc: Group has no references
**WARN*** Country   8359     1010 Zs: Group has zero shares
**WARN*** Region    8385     1010 Nc: Group has no references
**WARN*** Region    8385     1010 Zs: Group has zero shares
**WARN*** TechOps   8390     1010 Nc: Group has no references
**WARN*** TechOps   8390     1010 Zs: Group has zero shares
**WARN*** Tstusr01 33205     1001 Zs: User has zero shares
```

FILES

`/etc/acid`	Account name/ID file
`/etc/group`	Group name/ID/membership file
`/etc/passwd`	Password file
`/etc/udb`	User database file
`/etc/udb.public`	Copy of `/etc/udb` with public read permission. Sensitive information (such as the encrypted password and security fields) has been removed.
`/etc/udb_2/udb.index`	Public extension index file
`/etc/udb_2/udb.priva`	Private field extension file
`/etc/udb_2/udb.pubva`	Public field extension file

SEE ALSO

`shrsync`(8), `shrtree`(8)

`privtext`(1), `udbsee`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`libudb`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`acid`(5), `group`(5), `passwd`(5), `udb`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

Cray Data Migration Facility (DMF) Administrator's Guide, Cray Research publication SG–2135

General UNICOS System Administration, Cray Research publication SG–2301

NAME

udbpl – Prints administrative information for designated users

SYNOPSIS

```
/etc/udbpl -a [-g] [-p udb_path]
/etc/udbpl [-p udb_path] [logins] [uid[-uid]...]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The udbpl command displays administrative information for the specified list of login names from the user database (UDB). Only authorized users can obtain sensitive information from the private UDB, /etc/udb. All other users are restricted to the public UDB, /etc/udb.public.

The udbpl command accepts the following options:

-a
: Prints information on all currently active users. If the -g option is also specified, the printed information is restricted to real users.

-g
: This option must not appear without the -a option. It restricts the information to be printed to real users (in other words, it does not print groups).

-p *udb_path*
: Directs udbpl to use an alternate UDB having the path name *udb_path*.

logins
uid[-*uid*]...
: If the -a option is not specified, one or more *logins*, *uids*, or *uid* ranges may be specified. The logins and user IDs must be separated by white space. A user ID range cannot include white space. For example, the range 150 to 500 must be entered as 150-500. To determine whether a *login* or *uid* has been specified, udbpl attempts to access the UDB record by name. If that attempt is unsuccessful, it examines the string for a minus sign indicating an inclusive range. If udbpl finds no range indicator, it attempts to access a record by user ID. If it finds a range indicator, the bounds of the range are determined and all records in the range are accessed by user ID.

If no options are specified, udbpl prints permitted administrative information for the login name of the person who executed the command.

Information Format

The information printed by udbpl is divided into four lists, as follows:

List	Description
Shares	The number of shares the user has within the scheduling group, the nominal share of the resources, the current effective share of the resources (taking into account the user's recent usage of the resources), and the normalized usage of the resources (in the range 1 through 1000)

Privileges	The user's scheduling group and access groups, and a set of flags denoting account privileges and status
General	The encrypted password and the time the password was last changed (valid only if the user is privileged to see nonpublic administrative information), the total time the account has used the computer, the time the account was last used, the groups of terminals the account may log in, and other information that is site-dependent
Strings	The login name, its initial shell and directory, and a description of the owner of the account

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to obtain sensitive information from the private UDB.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to obtain sensitive information from the private UDB.

For `udbpl` to display accurate information, `share` must be turned on.

FILES

`/etc/udb*`	The user database. The path name can be changed by using the `-p` option.

SEE ALSO

`udbsee`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`libudb`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`lnode`(5), `passwd`(5), `share`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

udbrstrict – Enables and disables user access

SYNOPSIS

```
/etc/udbrstrict -r [-d udb_dir] [-m RIB] [-f file]
/etc/udbrstrict -r [-d udb_dir] [-m RIB] [-F file]
/etc/udbrstrict -r [-d udb_dir] [-m RIB] [-l users]
/etc/udbrstrict -r [-d udb_dir] [-m RIB] [-L users]
/etc/udbrstrict -u [-d udb_dir] [-m RIB] [-f file]
/etc/udbrstrict -u [-d udb_dir] [-m RIB] [-F file]
/etc/udbrstrict -u [-d udb_dir] [-m RIB] [-l users]
/etc/udbrstrict -u [-d udb_dir] [-m RIB] [-L users]
/etc/udbrstrict -p [-d udb_dir] [-m RIB]
/etc/udbrstrict -P [-d udb_dir] [-m RIB]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The udbrstrict command allows a privileged user (such as an operator) to enable and disable user access to the system. This includes both interactive access and batch access or some combination of the two. This capability is used to control access to the system during dedicated time.

Root (user ID 0) accounts are never restricted by udbrstrict. The udbrstrict command accepts two types of options: mode options, which control access and restriction, and user specification options, which specify users to be restricted or enabled.

The udbrstrict command accepts the following mode options and user specification options.

Mode Options

-r Restricts access. When used with no other options, it restricts access for all accounts except root. When used with the -f or -l options, it disables access for specified users. When used with the -F or -L options, it restricts access to specified users.

-u Removes access restrictions. When used with no other options, it removes access restrictions for all accounts (root is not affected). When used with the -f or -l option, it removes access restrictions for specified users. When used with the -F or -L option, it removes access restrictions for all users except those specified.

-p Displays a list of all restricted users.

-P Displays a list of all unrestricted users.

`-d` *udb_dir*

Sets the path name for access to the user database (UDB) files. The default name is `/etc`. This option allows private or test versions of the UDB to be maintained. The path name must end with a directory name. Use of this option has no effect on the file specified using the `-f` or `-F` option.

`-m` *RIB* Specifies the restriction/unrestriction mode that is to be performed. The available modes are:

`R` Restrict/unrestrict all access to the system

`I` Restrict/unrestrict interactive sessions

`B` Restrict/unrestrict batch sessions

User Specification Options

`-f` *file* Specifies an input file to be used as input. The names in the input file identify those users on whom the restrict/unrestrict operation is to be performed. The specified file must be formatted with one name per line. Input files may have white space and comments; white space (spaces and tabs) is ignored. A comment is any string preceded by a # character; it is in effect until the end of the current line.

`-F` *file* Specifies an input file to be used as input. The names in the input file identify those users on whom the restrict/unrestrict operation is not to be performed. The specified file must be formatted with one name per line. Input files may have white space and comments; white space (spaces and tabs) is ignored. A comment is any string preceded by a # character; it is in effect until the end of the current line.

`-l` *users* Specifies a list of users to be used as input. The user names in the list identify those users on whom the restrict/unrestrict operation is to be performed. The names must be separated by spaces. If the `-l` option is used, it must be the last option on the command line, and it must be followed by a list of one or more users.

`-L` *users* Specifies a list of users to be used as input. The names in the list identify those users on whom the restrict/unrestrict operation is not to be performed. The names must be separated by spaces. If the `-L` option is used, it must be the last option on the command line, and it must be followed by a list of one or more users.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to use this command.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to use this command.

WARNINGS

The udbrstrict utility also disables Network Queuing System (NQS) and cron jobs. If NQS is started while the udbrstrict -r (or udbrstrict -r -m B) option is set, all checkpointed and all queued NQS jobs of all access-disabled users will be deleted.

MESSAGES

The following message appears when a user is prevented from logging in because the failing routine setlimits from librsc has only one possible exit.

```
setlimits: login restricted
```

EXAMPLES

Example 1: The following example restricts the use of the system to root.

```
udbrstrict -r
```

Example 2: The following example removes restrictions for all accounts.

```
udbrstrict -u
```

Example 3: The following example restricts the use of the system to users mary and bob.

```
udbrstrict -r -L mary bob
```

Example 4: The following example removes restrictions for all users except bob.

```
udbrstrict -u -L bob
```

Example 5: The following example restricts all batch access to the system.

```
udbrstrict -r -m B
```

Example 6: The following example displays a list of all users who have their interactive sessions disabled.

```
udbrstrict -p -m I
```

SEE ALSO

udbgen(8), udbpl(8)

udb(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

unitap – Provides online testing of magnetic tapes

SYNOPSIS

/etc/diag/unitap

IMPLEMENTATION

All Model E based Cray Research systems

DESCRIPTION

The unitap command provides online testing of magnetic tapes, either under the control of the tape daemon (up mode) or without the tape daemon (down mode). unitap can test up to eight tape units at one time. It is a menu-driven test that can also be run from the command line or by using an input file that directs input into unitap.

NOTE: The unitap command is not supported on GigaRing based Cray Research systems. Instead of unitap, use vtt(8) on GigaRing based systems. For more information, see the vtt(8) man page.

The unitap testing options are as follows:

- Quick confidence tests. This option executes selected tape confidence tests.
- All confidence tests. This option executes all of the tape confidence tests.
- Two- through eight-path conflict tests. This option executes a selection of tape tests in parallel to exercise from two through eight tape paths. The tests verify that the paths can withstand conflict.
- Canned tests. This option executes a user-selected test (from the Canned Tests menu).
- Test loop. This option executes a user-defined test (defined with the Programming tool).

When running unitap, you must be prepared either to mount the tape when requested or to start the test with a scratch tape in the device.

The unitap environment can be modified by setting several environment variables. If the UNITAP_TPMNT_OPTION variable is defined, its contents are appended to the tpmnt(1) command, allowing the addition of site-specific tpmnt options. The UNITAP_TRACE_FILENAME variable specifies the default name of the trace file created by unitap. The name will still default to unitap.trace.

The UNITAP_SPECIAL_NAME and UNITAP_SPECIAL_NAME_*n* variables specify the actual name of the tape group when you use the SPECIAL option to set the tape group name. If these variables are not used, the tape group will actually be SPECIAL, just like for TAPE, CART, SILO, QIC, TEST, EXP, 3490, and ER90. The value of UNITAP_SPECIAL_NAME is used for all paths where UNITAP_SPECIAL_NAME_*n* is not set. The value of *n* determines which path (1 through 8) uses the UNITAP_SPECIAL_NAME_*n* value as the actual name of the tape group.

Tape units can be up or down when they are tested with unitap. When you are planning to test in down mode, enter the following form of the tpconfig(8) command to down a device: tpconfig *dev* DOWN. To test in down mode, you must have diagnostic privileges granted by your system administrator in your user database (UDB) or be running as super user. To test in up mode, you must have tape read and write privileges granted by your system administrator. However, you can practice using unitap without these privileges by running the test in learn mode or error mode. These modes simulate the passing and failing of test execution without removing the tape device from normal system operations. Choose learn mode by entering L or error mode by entering E.

By default, unitap runs in up mode. Entering the CH option from the Variable menu toggles unitap to run in down mode. Entering DOWN will also toggle unitap to down mode. Entering UP toggles unitap to up mode.

Each unitap menu option has a help window. To display a help window for a particular option, enter HELP followed by the option (for example, help DBM). unitap also has a program notes feature, which you can access by entering W from any menu. The program notes explain how to use unitap.

Whenever unitap is run, all input and output are stored in a file called unitap.trace. This file is useful for looking at failure history.

Entering the /ce/bin/unitap command displays the unitap Main menu. From this menu, you can display all the other menus in the unitap menu system.

MN Displays the unitap Main menu. The options on this menu include the following:

- D Displays the Debug menu.
- T Displays the Test menu.
- V Displays the Variable menu.
- Q Runs a quick test on the currently displayed path.
- A Runs all tests on the currently displayed path.
- DUMP Dumps all error data to the screen and to a trace file.
- L Toggles learn mode.
- W Displays a brief description of how to use unitap.
- HELP *option* Displays help information for *option* (for example, HELP DBM).
- G Displays the Global Options menu. The options on this menu are valid from all unitap menus.
- EXIT Exits unitap and releases channels dedicated to online diagnostic testing. You can also enter bye or quit to exit unitap.

C Displays the Canned Tests menu in up or down mode. This menu contains the tests that run on a single channel. These are the same tests used by the Q, A, and 2 through 8 options on the Test menu. The Canned Tests menu includes the following options:

AC All basic commands test. Tests the rewind, write, write tape mark, forward block, backward block, forward tape mark, and backward tape mark commands. This option does not test the read command. Applies only to down mode.

BS Bus test. Writes and reads 8-bit test patterns to the tape. Applies to both up and down mode.

DT Data buffer tests. Writes and reads 64-bit test patterns to the tape. Applies to both up and down mode.

ET End-of-tape test. Writes 32,768-byte blocks up to the value of ETCOUNT (the default is 20000). When the end-of-tape is detected, the tape is rewound and the data is verified. For this test, unitap goes into very fast mode, which disables the recording of system call history and the display of tape actions as they are processed. However, to help you determine that progress is being made, every hundredth read and write command is displayed. For this test, a block is 32,768 bytes. Applies only to down mode.

ETCOUNT *n*
Specifies how many blocks are written to the tape for the ET test. The default is 20000. Applies only to down mode.

MV Multivolume test. Writes 32,768-byte blocks up to the value of MVCOUNT (the default is 20000). After completing the writes, the data is read back and verified. For this test, unitap goes into very fast mode, which disables the recording of system call history and the display of tape actions as they are processed. However, to help you determine that progress is being made, every hundredth read and write command is displayed. For this test, a block is 32,768 bytes. Applies only to up mode.

MVCOUNT *n*
Specifies how many blocks are written to the tape for the MV test. The default is 20000. Applies only to up mode.

BC Byte counter test. Writes and reads 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, and 4096 bytes to the tape. Applies to both up and down mode.

NB Next byte counter test. Writes and reads 1 sector plus 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, and 4096 bytes to the tape. Applies to both up and down mode.

BP Tape block positioning commands test. Writes patterns to the tape, issues tape block positioning commands, and then reads the patterns to verify that the positioning commands work. Applies only to down mode.

LA Ladder tests. Writes and reads 1, 2, 3, 4, 5, 6, 7, and 8 sectors to the tape. Applies to both up and down mode.

LW Long write test. Writes 32768-byte blocks up to the value of LWCOUNT (the default is 5000). After completing the writes, the data is read back and verified. For this test, unitap goes into very fast mode, which disables the recording of system call history and the display of tape actions as they are processed. However, to help you determine that progress is being made, every hundredth read and write command is displayed. For this test, a block is 32,768 bytes. Applies to both up and down mode.

LWCOUNT *n*
Specifies how many blocks are written to the tape for the LW test. The default is 5000. Applies to both up and down mode.

WC Write compatibility test. Rewinds and writes a pattern to the tape. Applies to both up and down mode.

RC Read compatibility test. Rewinds and reads a pattern from the tape. Applies to both up and down mode.

TP Tape mark positioning commands test. Writes blocks and tape marks, issues tape-mark level positioning commands, then reads and verifies the patterns. Applies only to down mode.

D Displays the Debug menu, which contains troubleshooting tools. The options on this menu include the following:

B Displays the Breakpoint tool, which allows you to set a breakpoint immediately preceding or following a system call in a test.

CB Displays the Command Buffer tool, which allows you to build a string of unitap commands and save them to an internal variable.

CD Displays the Compare Data tool, which allows you to display the read and write data buffers, and exclusive ORs (logical differences) for the write and read address comparisons.

DS *n* Displays the Device Status menu, which shows information returned by the tpbmx(8) command. *n* can be a value from 1 through 15.

E Selects error mode, which is a subset of learn mode. In error mode, all tests fail (unlike learn mode, in which all tests pass). Error mode is useful for learning how to read failure conditions. Error mode is displayed in the upper left-hand corner of the screen.

H Displays the System Call History tool, which allows you to display a history of the last 15 system calls that preceded the current event.

L Selects learn mode. Learn mode simulates a passing test execution without removing the tape device from normal system operations. Learn mode is displayed in the upper left-hand corner of the screen.

LO Displays the Hardware Layout menu, which is a diagram that illustrates the typical hardware path from a Cray Research mainframe to a tape unit.

M — Displays the Memory tool, which allows you to display the read and write data buffers and modify the write data buffer.

PG — Displays the Programming tool, which allows you to build a test loop with up to 100 steps and up to 8 channels performing read, write, rewind, positioning, and compare operations.

S — Displays the Packet Status tool, which allows you to display the status of the last packet sent for each channel at the time of the current event.

SB *n* — Displays the Sense Byte menu. *n* is a value from 0 through 63. (You can also enter SB without a sense byte value to display a condensed version of sense byte information or enter SBS to display a summary of sense bytes.) This menu displays the meanings of additional bytes of information about a specific tape device.

U — Displays the User Tool menu, which enables you to issue various UNICOS commands without exiting unitap.

G — Displays the Global Options menu. This menu displays all the options available in unitap in alphabetical order. These options are valid from all unitap menus. Entering G goes to the next page of the menu. At the end of the menu screens, you are returned to the first screen. The following options are unique to the Global Options menu:

MAGIC — Causes unitap to display the name of each internal routine as it is entered

T2 *x* — Specifies that unitap should create a second trace file named *x*

TIMESTAMP — Prefixes most unitap output with the correct date and time

TPBMX — Displays operator information about tape devices

TPCLR — Clears the tape drive

TPCONFIG — Configures tape devices up and down

TPFRLS — Forcibly releases tape reservation and associated devices

TPGSTAT — Displays user reservation status for all users

TPLABEL — Labels magnetic tape reel or cartridge

TPMLS — Displays loader status

TPMQL — Displays the tape daemon’s mount request list

TPU — Unloads tape drives

T — Displays the Test menu. The options on this menu include the following:

Q — Quick confidence tests. Runs selected tape tests.

A — All confidence tests. Runs all of the confidence tape tests.

2 – 8 — Two- through eight-path confidence tests. Runs a selection of tape tests in parallel to exercise two through eight tape paths.

C Canned tests. Executes a user-selected test.

V Displays the Variable menu in up or down mode. The options on this menu include the following:

CH *n* Sets the channel number (20-37 octal). This option is required. Applies only to down mode. By default, `unitap` runs in up mode. Entering the `CH` option automatically toggles `unitap` to run in down mode.

CO *n* Sets the controller number (O-F hexadecimal). Applies only to down mode. This option is required.

CART Selects the 3480 cartridge tape group for up mode testing on the displayed path. By default, `unitap` selects the CART tape group. Applies only to up mode.

3490 Selects the 3490 cartridge tape group for up mode testing on the displayed path. Applies only to up mode.

TAPE Selects the round-tape tape group for up mode testing on the displayed path. Applies only to up mode.

SILO Selects the SILO tape group for up mode testing on the displayed path. Applies only to up mode.

TEST Selects the TEST tape group for up mode testing on the displayed path. Applies only to up mode.

QIC Selects the quarter-inch cartridge tape group for up mode testing on the displayed path. Applies only to up mode.

ER90 Selects the ER90 IPI tape group for up mode testing on the displayed path. Applies only to up mode.

SPECIAL Selects the SPECIAL tape group for up mode testing on the displayed path. This option supports the use of a site-specific tape group name. Applies only to up mode.

EXP Selects the EXP tape group for up mode testing on the displayed path. Applies only to up mode.

DV *n* Specifies the name of the tape device to be tested (required for down mode; optional for up mode). *n* can be either the absolute path name (for example, `/dev/tape/300`) or just 300. When 300 is used, `unitap` will prefix it with `/dev/tape` when needed. Applies to both up and down mode.

P*n* Initializes the path under test (channel, controller, and device). *n* is a value in the range 1 through 8. The default for *n* is 1. Applies to both up and down mode.

PC *n* Sets the pass count (decimal). The default for *n* is 1. Applies to both up and down mode.

RL Releases the dedicated path for the tape unit. Applies to both up and down mode.

`VSN` *x* Changes the volume serial number on which to test. *x* can be up to a 6-digit volume serial number. Applies to both up and down mode.

`LB` *x* Specifies the type of label on the tape. *x* is the 2-digit tape label type. Applies only to up mode.

`DOWN` Toggles up and down mode.

`UP` Toggles up and down mode.

`NW` Toggles no write ring installed. Applies only to up mode.

`MD` *x* Determines how `unitap` writes data to the tape. Applies only to up mode. *x* can be one of the following:

`TU` Transparent unbuffered

`TB` Transparent buffered

`IDRC` Toggles IDRC (data compression). Applies to both up and down mode.

`F` Toggles fast mode, which runs `unitap` faster than normal. This selection disables the Breakpoint tool and the System Call History tool. Fast mode status is displayed in the upper left-hand corner of the screen. Applies to both up and down mode.

`VF` Toggles very fast mode, which runs `unitap` at maximum speed. This selection disables the Breakpoint tool and the System Call History tool. It also disables the echoing of `unitap` commands to the screen; no history trail will be available, and no activity will occur on the screen until the test completes or an error occurs. Applies to both up and down mode.

`R` Returns to the previous menu.

`VSN1` through `VSN20`
Determines the list of VSNs on the `tpmnt`(1) command. (This menu option is not displayed on the Variable menu.) This option applies only to up mode.

`VOL1` through `VOL20`
Determines the list of VSNs on the `tpmnt`(1) command. (This menu option is not displayed on the Variable menu.) This option applies only to up mode.

NOTES

If this command is installed with the default privilege assignment list (PAL), you must have an active `secadm`, `sysadm`, or `diagadm` category to use this command.

EXAMPLES

Example 1: Starts `unitap` in down mode; tests the path to the device from channel 20, controller 3, to device `tape100`; and runs a quick test (`q`) on device `tape100`. (With the exception of the *n* entry in the `DV` *n* option, `unitap` is not case-sensitive. It will accept entries in either uppercase or lowercase.)

```
/ce/bin/unitap ch 20 co 3 dv /dev/tape/tape100 q
```

Example 2: Starts `unitap` in up mode (the default) and runs a quick test (`q`) on the CART tape group (the default).

```
/ce/bin/unitap q
```

Example 3: Runs the two-channel conflict test in devices `00` and `01` and then exits the program. (For formatting purposes, the command in this example is shown on two lines; it should be entered as a one command.)

```
/ce/bin/unitap ch 20 co 0 dv /dev/tape/00 p2 ch 21 co 1
  dv /dev/tape/01 2 exit
```

SEE ALSO

`vtt`(8) for information on the online tape exerciser for GigaRing based Cray Research systems

Online Maintenance Tools Guide for Cray PVP Systems, Cray Research publication SD–1012. (This document contains information private to Cray Research, Inc. It can be distributed to non-CRI personnel only with approval of the appropriate Cray manager.)

NAME

urmd – Starts the Unified Resource Manager (URM) daemon

SYNOPSIS

urmd [-D] [-l *directory*] [-p *directory*] [-s *socket*] [-t] [-u *directory*] [-c *directive*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The following options and arguments to the urmd command are used for testing purposes only:

-D	Debug mode. Sends error messages to the stderr file.
-l *directory*	Alternate directory for the URM log file. By default, *directory* is /usr/adm/urm. This option overrides a directory specified with the -p option. When URM is running, use the rmgr(1) subcommand set log directory to change the location of the log file.
-p *directory*	Directory in which to run. This option is especially useful during development, so the daemon can run without interfering with the UNICOS operating system. Generally, use -p after moving to a working directory. Using the -l option overrides this directory.
-s *socket*	Socket name or port number to open as the service name for URM. The socket name will be passed as a -s *socket* option to the *command* named with the -c option. The default service name is urm.
-t	Test mode. This prevents urmd from creating new processes and no initiation command is run. (Even if -c is present, it is ignored). Use this only if you must use cdbx and want to prevent urmd from becoming a true daemon.
-u *directory*	UDB directory. This is usually not necessary because urmd only accesses the user database (UDB) in public-read mode. The reason it is present is to allow debugging such parts of urmd as the share evaluator without having to depend on the running UDB.
-c *directive*	The name of the directive used to initialize urmd. Defaults to rmgr. No command is run if either the -t option is used or the directive is no. Anything entered after the -c option is passed on as an argument to the -c *directive*.

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
system, secadm, sysadm	Allowed to use this command.

If the PRIV_SU configuration option is enabled, the super user is allowed to use this command.

FILES

/etc/config/daemons	File to invoke urmd at system startup.
/etc/config/urm/	Directory containing configuration defaults for urmd. Do not attempt to alter any files in the directory, except the configuration file.
/etc/config/urm/configuration	File updated by the UNICOS installation and configuration menu system. Any site-specific URM configuration changes should be added at the end of this file.
/etc/config/urm/init	File containing the default urmd startup configuration directives. On startup of the urmd, rmgr(1) is automatically executed to include this file.
/usr/adm/urm/Urm.*yymmdd*	URM log file containing a record of URM activites for the day *yymmdd*.

SEE ALSO

rmgr(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

urmsnap – Captures the current URM configuration

SYNOPSIS

/etc/urmsnap [-M *rmgr_cmd*] [-S *socket*]

/etc/urmsnap [*input_file*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The urmsnap command captures the current Unified Resource Manager (URM) configuration. It reads the URM configuration, converts the information to rmgr(1) input format, and writes the result to standard output.

Changes to URM made on a running system with rmgr(1) are not permanent. urmsnap can be used to capture these changes for testing or for transfer to the permanent configuration of URM.

Options allow you to select a different version of the rmgr(1) command, a different socket, or an input file of rmgr(1) information (the output of the command rmgr -c 'View /*'). These options are useful only for internal testing purposes; they are not needed in normal use.

The urmsnap command accepts the following options and operands:

-M *rmgr_cmd* Specifies an alternate pathname for the rmgr(1) command. This option is provided for internal testing; it is not recommended for general use. If the -M option is not used, the standard rmgr(1) command (/bin/rmgr) is used.

-S *socket* Specifies the name of the socket connecting to URM. This option is provided for internal testing; it is not recommended for general use. If -S is not specified, the default URM socket (urm) is used.

input_file Specifies an input file containing output from rmgr(1). This option is provided for internal testing; it is not recommended for general use. Use this option to save temporary URM configuration information for later examination. The content of this file must be identical to the output of the rmgr(1) View command (rmgr -c 'View /*'). This option cannot be used with the -M or -S option.

EXIT STATUS

If no error occurs, the urmsnap command exits with a return value of 0. Errors detected by urmsnap, such as option errors, result in an error message and an exit code of 1. Errors detected by rmgr(1) display the message text on standard error and result in a urmsnap exit code of 3. The exit code for rmgr(1) is reported in the urmsnap error message that follows the rmgr(1) message text.

SEE ALSO

`rmgr`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

usetjob – Changes minimum rank field in URM job table entries

SYNOPSIS

usetjob -r *minrank* [-D] [-M *rmgr_command*] [-p *urmpath*] [-S *socket*] [-W *seconds*] *nqsjsn*[.*machine*] ...

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The usetjob command changes the minimum rank field in the Unified Resource Manager (URM) job table. You must be a URM administrator to alter this field. usetjob is an interface to the rmgr(1) command, which can also be used to accomplish this task.

By default, URM uses the priority determined by the Network Queuing System (NQS) for batch jobs. The minimum rank field of non-batch jobs is not used by URM; these jobs contain the default value of 0 in the minimum rank field.

The minimum rank, or NQS job priority, allows banding of batch jobs in the job backlog. Jobs are organized into categories that are independent of NQS queues. Jobs with higher minimum rank are guaranteed a better priority than jobs lower minimum rank. The usetjob command allows URM administrators to change the minimum rank outside of NQS.

A job is identified by the NQS sequence number, *nqsjsn*. Many IDs may be presented at once, each separated by white space. (The number of IDs is limited by the maximum length of the command line.) If jobs are being submitted from a number of machines, *nqsjsn* may not be unique. To further qualify the job name, you can specify either the machine name as reported by NQS or the numeric machine ID as shown by URM. The usetjob command translates machine names to machine IDs, using the translation file /usr/adm/urm/midfile, which is created by URM. This file is created the first time that NQS connects to URM, and it is updated on subsequent connections. If the translation file is not present, machine names are not accepted by usetjob.

The usetjob command accepts the following options and operands:

-r *minrank* Specifies minimum rank. The *minrank* argument must be a positive integer, 32 bits or less. This option is required.

-D Specifies the debug option. Internal information is displayed on standard output (stdout). This option is useful only for internal testing purposes.

-M *rmgr_command*
Specifies the full path name of the rmgr(1) command. By default, the path name /bin/rmgr is used. This option is useful only for internal testing purposes.

`-p` *urmpath* Specifies the directory containing the machine-name-to-MID (machine identifier) translation file. (The translation file must be named `midfile`.) By default, the directory `/usr/adm/urm` is used. This option is useful only for internal testing purposes.

`-S` *socket* Specifies the socket name to pass to `rmgr`(1). By default, `rmgr`(1) uses the name specified by the environment variable `RMGR_SOCKET`, if set, or the default socket name `urm`. This option is useful only for internal testing purposes.

`-W` *seconds* Specifies the wait time in seconds for a child process to respond on a pipe. By default, the wait is 20 seconds. This option is useful only for internal testing purposes.

nqsjsn[`.`*machine*]

Specifies the job identification field, which is composed of a NQS job sequence number (JSN) with an optional machine name or machine identifier (MID). Only the JSN is required unless more than one machine is submitting jobs and the JSN values overlap; in this case, the machine name or MID is required. (The `usetjob` command issues a warning if overlapping JSNs are detected.)

Multiple job identification fields can be specified, separated by spaces. Jobs are processes in left-to-right order. An error for a job causes that field to be abandoned, and `usetjob` processes the next the next job identification field.

NOTES

Future releases of the UNICOS operating system will allow a URM administrator to change more fields in the URM job table.

ENVIRONMENT VARIABLES

The `RMGR_SOCKET` environment variable can be used to specify an alternate socket name for `rmgr`(1). By default, `rmgr`(1) uses the default socket name `urm`.

The `-S` option of `usetjob` overrides both `RMGR_SOCKET` and the default.

EXAMPLES

The following examples shows several basic uses of the `usetjob` command.

Example 1: This example changes the minimum rank for NQS job 12345.

```
usetjob -r 7 12345
```

Example 2: This example changes the minimum rank for NQS job 54321 from machine name `fred`. If multiple machines submit jobs, the machine name is necessary to prevent misinterpretation of the JSN.

```
usetjob -r 15 12345.fred
```

The machine ID (for example, 999) could be used instead of the machine name, giving a job identification field of `12345.999`.

FILES

`/usr/adm/urm/midfile`	Translation file; translates machine names to machine IDs

SEE ALSO

`urmd`(8)

`rmgr`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

vht – Exercises SPN-based HIPPI devices (GigaRing based systems) or performs a test of a HIPPI channel (Model E systems)

SYNOPSIS

GigaRing based systems:

/etc/diag/vht [-a *action*] [-b *blocksize-max*[:*blocksize-min*]] | pass] [-e *echo-file-name*] [-l *logfile*] [-o *options*] [-P *pattern*] [-p *pass-end*[:*pass-start*[:*pass-step*]]] [-T *timeout*] [-t *tracefile*] -I *I-Field* -u *ULP-id device-name*

/etc/diag/vht -h

/etc/diag/vht -V

Model E systems:

/etc/vht [-i *idev*] [-o *odev*] [-s *spass*] [-c *epass*] [-h *hname*] [-e *filename*] [-b *blocksize*] [-D] [-d] [-f] [-I *I-Field*] [-L] [-m *errlim*] [-n *streams*] [-P *pattern*] [-p] [-q] [-S] [-t *timeout*] [-x] [-y] [-l *lpath*] [-r] [-w]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

GigaRing based systems:

The online exerciser for single-purpose node (SPN)-based HIPPI devices, vht, will exercise any SPN-based HIPPI network devices that have been configured into the operating system. Because vht uses raw channel HIPPI, the HIPPI interface must be configured "down" with ifconfig(8) if the device has been configured for use with TCP. (If you are not familiar with raw channel HIPPI, you are encouraged to use vst(8) to test the HIPPI device. vst requires that the TCP interface be configured "up" with ifconfig.)

Model E systems:

The vht command tests a HIPPI channel pair. The test creates a second process, by either a fork(2) operation or a remote shell command. The second process writes to the output device while the original one reads the input device. In each pass of the test, the I/O parameters (buffer lengths; synchronous, asynchronous, or listio; the number of buffers to make a block; and the data contents) are determined by pseudo-random numbers. The random number seed is the pass number, so that a given pass always executes the same test.

When running multiple instances of vht, the logical path option (-l) must be unique for each instance; either when in loopback mode (testing from the Cray host to the HIPPI switch) or when testing between two hosts. When testing between two hosts, their -l *lpath* options must match.

GigaRing Based Synopsis

The GigaRing based `vht` command accepts the following options:

`-a` *action* Specifies the action to be performed. Valid values are as follows:

`rw`	Read and write
`read`	Read
`write`	Write

The default is `rw`.

`-b` *blocksize-max*[`:`*blocksize-min*] | `pass`
Specifies the block size to be used for I/O operations. When the *blocksize-min* value is used, the actual block size will be randomly set to a value between *blocksize-min* and *blocksize-max* for each pass. When `pass` is specified, the block size will be set to the current pass count value.

`-e` *echo-file-name*
Writes the first data buffer to the specified file. No actual I/O to the HIPPI device is performed.

`-h` Displays the command-line synopsis on `stderr`. `vht` then exits.

`-I` *I-Field* Required option. Specifies the I-Field value. (If you are not familiar with the term "I-Field," you are encouraged to use `vst` instead.)

`-l` *logfile* Specifies where the exerciser's log file should be written. The file specification can be either absolute or relative. By default, logging is disabled.

`-o` *options* Specifies a colon-separated list of options to be turned on. Valid values are as follows:

`burst`	Deferred implementation. Uses the HIPPI short-burst-first feature for `listio`(2).
`debug`	Displays large amounts of information related to what the exerciser is doing.
`disconnect`	Causes HIPPI to disconnect after each packet.
`lock`	Locks the process(es) into memory to prevent swapping.
`fast`	Does not compare data.
`old`	Uses the old (pre GigaRing) `vht` command-line syntax.
`panic`	When a data miscompare occurs, panics the system.
`squeeze`	Deferred implementation.
`verbose`	Displays information about the progress of the exerciser.

By default, all options are turned off.

`-P` *pattern* Specifies the data pattern to be used. Valid values are as follows:

`bits` Each 64-bit word has a random sequence of consecutive 1-bits.

`slide0` The first 64-bit word has all bits set except bit 0, which is cleared. Subsequent words are circularly left-shifted by one bit position.

`slide1` The first 64-bit word has all bits cleared except bit 0, which is set. Subsequent words are circularly left-shifted by one bit position

`random` Each 64-bit word is set to a random value.

`all` All of the built-in data patterns are used, one per pass, in a circular fashion.

Instead of selecting one of the built-in patterns, you can specify a numeric constant that will be used to set each 64-bit word of the data buffer. The default is `all`.

`-p` *pass-end*[`:`*pass-start*[`:`*pass-step*]]
Specifies the number of passes to be performed. The number of passes must be the same on both the receiving and sending host. *pass-end* specifies the ending pass number; for example, `-p 1000` executes `vht` up to pass 1,000. *pass-start* specifies the starting pass number; for example, `-p 1000:500` executes `vht` starting at pass 500, and continuing up to pass 1,000. *pass-step* specifies how pass steps should be incremented; for example `-p 1000:500:100` executes `vht` starting at pass 500 and then steps up to pass 600, and so on, up to pass 1,000. *pass-start* and *pass-step* are most useful when specified with the `-b pass` option.

`-T` *timeout* Specifies the timeout value to be used when asynchronous I/O is performed.

`-t` *tracefile* Specifies where the exerciser's trace file should be written. The file specification can be either absolute or relative. By default, tracing is disabled.

`-u` *ULP-id* Required option. Specifies the ULP id value. (If you are not familiar with the term "ULP id," you are encouraged to use `vst` instead.)

`-V` Displays version information on `stderr`. `vht` then exits.

device-name Required option. Specifies the HIPPI device upon which the I/O operations are to be performed; for example, `/dev/hippi0/u0`. Note that only `/dev` is guaranteed to be the same on all machines.

Model E Synopsis

The Model E `vht` command accepts the following options:

`-i` *idev* Specifies the input path name of the channel to be tested. The default is `/dev/hippi0/i00`.

`-o` *odev* Specifies the output path name of the channel to be tested. The default is `/dev/hippi0/o00`.

NOTE: The path name to the channel pair to be tested can be changed by setting the environmental variable `VHTPREFIX`.

`-s` *spass* Specifies the starting pass number, *spass*.

`-c` *epass* Specifies the ending pass number, *epass*.

`-h` *hname* Specifies a remote host for the slave process that writes the output device. If the value is not specified, the local host is used, and the channels must be connected in loopback configuration.

`-e` *filename* Echoes data pattern to file (no test, does only one pass).

`-b` *blocksize* Overrides the normal random selection of packet size and forces all packets sent by `vht` to a given size. Specify *blocksize* in 64-bit words.

`-D` Disconnects after each packet (HIPPI).

`-d` Enables debug printouts; `vht` prints the parameters of each pass.

`-f` Sets fast-run mode; disables the exhaustive validation of received data.

`-I` *I-Field* Specifies the *I-Field* to use (HIPPI). The default *I-Field* is that given for the local host in the file `hycf.hippi`. The existence of the file is searched for in the following order: `./hycf.hippi`, `$HOME/hycf.hippi`, `/etc/hycf.hippi`, or, if no such file exists, then zero is used for the *I-Field*. Sample *I-Fields* are as follows: `-I 0x1300b00b` (hexadecimal) and `02300130013` (octal).

`-L` Locks the process in memory; that is, the `vht` test will not be swapped out of memory. You must have super-user privilege to use the `-L` option.

`-m` *errlim* Limits the number of data compare errors to print. The default is 10.

`-n` *streams* Specifies the number of streams. The default is 1. `vht` creates additional copies of itself by a `fork`(2) operation if multiple streams are requested. The input and output path names must end in a number, which `vht` increments each time it makes additional copies of itself, so each copy runs on a different logical path. When this option is specified, the output display may be garbled.

`-P` *pattern* Specifies the alternate data pattern to be used. Valid values are as follows:

`1` Random bits (same as `-q`): each data bit has a 50% chance of being set or clear.

`2` 0x5555555555555555

`3` 0xAAAAAAAAAAAAAAAA

`4` 0x5555555555555555 0xAAAAAAAAAAAAAAAA

`5` 0x0000000000000000

`6` 0xFFFFFFFFFFFFFFFF

`7` 0x00000000FFFFFFFF

`8` 0xFFFFFFFFFFFFFFFF 0x0000000000000000

`9` 0xFFFFFFFFFFFFFFFF 0xFFFFFFFFFFFFFFFF
0x0000000000000000 0x0000000000000000

`10` 0x0001000100010001

`11` 0x0000010000000000 (position of 1-bit is random)

`12` 0x0000000000000001 0x0000000000000002 0x0000000000000003 etc.

`13` 0x55555555AAAAAAAA

`14` 0xAC89ACACAC89ACAC

`15` 0x870e1c3870e1c387 0x0e1c3870e1c3870e 0x1c3870e1c3870e1c 0x3870e1c3870e1c38 0x70e1c3870e1c3870 0xe1c3870e1c3870e1 0xc3870e1c3870e1c3

`16` 0x0043004300430043

`17` 0x0001020304050607 0x08090A0B0C0D0E0F 0x1011121314151617 etc.

`-1` Select one of the above at random, different for each pass.

other Default data pattern consisting of path number and 15 bits each of pass count, segment number, stride number and word number.

`-p` Panics the system on error. This option is ignored unless a `panic set` super-user command is active.

`-q` Randomizes test data. The default is to build test data words from the path number, pass count, segment number, stride number, and word number.

`-S` Uses SSD buffers. Can be used only if the system has an SSD.

`-t` *timeout* Specifies the time-out value in seconds for driver and HIPPI connection. Sixty seconds is the maximum.

`-x` Forces the data length to be equal to the pass number; pass 1 sends 1 word, pass 2 sends 2, and so on.

`-y` Tests sending a short burst first. By default, the HIPPI driver sends a short burst last.

The following options are used by `vht` when it calls itself by remote shell command, or when the companion process is started manually:

`-l` *lpath* Specifies a value, *lpath*, to be used in the *to* header field in output messages. Specify *lpath* as an 8-bit value. When running multiple instances of `vht`, `-l` must be unique for each instance; either when in loopback mode (testing from the Cray host to the HIPPI switch) or when testing between two hosts. When testing between two hosts, their `-l` *lpath* options must match.

`-r` Reads only the input device; does not spawn a companion process.

`-w` Writes only the output device; does not spawn a companion process.

NOTES

On any data error, `vht` writes two files in `/tmp`: one containing the expected data, and one containing the data actually received. These can be compared to help in failure analysis.

It is not recommended to use `vht` in multi-user mode unless the HIPPI interface is configured down or is in shared mode.

MESSAGES

Error messages are self-explanatory. For example, the following message appeared because the `-i` option specified a nonexistent device:

```
cray%  vht -i/dev/hippi/01
vht /dev/hippi/01:  No such file or directory
```

The next example failed because the `-r` option suppressed the second process, which would have written the output side, and the read timed out on pass 1. The error message lists the parameters selected for this pass.

The read was a `listio`(2) system call with two elements (segments) in the list, reading a single packet into two buffers of 370 and 1675 words. The write for pass 1 is a simple write of 17130 words, which is longer than the combined input buffers. The read process received timeout errors on both `listio` elements. It expected to recieve full byte counts on both buffers, with a nonfatal error on the second buffer indicating that excess data was discarded. `vht` dumped both the expected and recieved data to `/tmp/vht.3853o` and `/tmp/vht.3853i`, respectively.

```
cray% setenv VHTPREFIX /dev/hippi1
cray% vht -r
Selected logical device: /dev/hippi1/i00

------------------------------------------------------------
HIPPI test pass=1, path=128, pattern=0
Received - Input:  listio  370    1675 words
Sent - Output: write   17130 words
Path 128 pass 1 input segment 1:  I/O request timeout
/dev/hippi1/i00 HIPPI error code:  HIST_RTMO, Read request timeout
Path 128 pass 1 input segment 2:  I/O request timeout
/dev/hippi1/i00 HIPPI error code:  HIST_RTMO, Read request timeout
Input request unexpected status (nonfatal)
/dev/hippi1/i00 HIPPI error code:  HIST_RTMO, Read request timeout
(expected 01113, HIST_LONG, Long packet excess discarded)
dumping data buffers to /tmp/vht.3853i and /tmp/vht.3853o
vht 128 failed on pass 1
```

EXAMPLES

GigaRing based systems:

Example 1: The following example executes a loopback test for 1 pass only. On GigaRing systems the *I-Field* and *ULP-id* are required.

```
vht -a rw  -I 0x1300b00b -u 128
```

Example 2: The next example is almost the same, but it executes passes 1 through 10,000 with random data.

```
vht -a rw  -I 0x1300b00b -u 128 -p 10000  -P random
```

Example 3: The following example executes 1000 passes between `crayhost1` and `crayhost2`. These channels are connected. This example bypasses the exhaustive validation of input data.

```
crayhost1$     vht -a read -p 1000 -o fast

crayhost2$     vht -a write -p 1000 -o fast
```

Model E systems:

Example 1: The following example executes a loopback test for 1 pass only. It reads `/dev/hippi0/i01` and writes `/dev/hippi0/o01` using the default *I-Field* value.

```
cray%  vht -ii01 -oo01
```

Example 2: The next example is almost the same, but it executes passes 1 through 10,000 with random data.

```
cray%  vht -q -c 10000 -ii01 -oo01 -I 0x1300b00b
```

Example 3: The following example executes passes 100 through 1000 between `/dev/hippi0/i16` on `crayhost1` and `/dev/hippi0/o20` on `crayhost2`. These channels are connected. This example bypasses the exhaustive validation of input data.

```
crayhost1%  vht -s100 -c1000 -i/dev/hippi0/i16 -o/dev/hippi0/o20 -f -h crayhost2
```

FILES

GigaRing based systems:

Data miscompares are written to a file `/tmp/jtmp.*/*` in the following format:

```
A : expected
B : actual
x : logical difference

A (     0) + 7f 06 3c 38 7c fc 1f 06 06 0c 7f ff ff ff 1c 3e *..<8|..........>*
B (     0) + fe fd fb f7 ef df bf 7f fe fd fb f7 ef df bf 7f *................*
x (     0) + 81 fb c7 cf 93 23 a0 79 f8 f1 84 08 10 20 a3 41 *.....#.y.......A*
```

Model E systems:

`/dev/hippi*/*` received. These can be compared to help in failure analysis.

EXIT STATUS

The following exit codes apply to GigaRing based systems only:

Exit Code	Description
`0`	The command completed successfully with no I/O errors and no data miscompares.
`1`	The `close`(2) call failed. The error message will be displayed on `stderr`.
`2`	The `creat`(2) call failed. The error message will be displayed on `stderr`.
`3`	The `fcntl`(2) call failed. The error message will be displayed on `stderr`.
`4`	The `ioctl`(2) call failed. The error message will be displayed on `stderr`.
`5`	The `lseek`(2) call failed. The error message will be displayed on `stderr`.
`6`	The `open`(2) call failed. The error message will be displayed on `stderr`.
`7`	The `read`(2) call failed. The error message will be displayed on `stderr`.
`8`	The `reada`(2) call failed. The error message will be displayed on `stderr`.
`9`	The `stat`(2) call failed. The error message will be displayed on `stderr`.
`10`	The `write`(2) call failed. The error message will be displayed on `stderr`.
`11`	The `writea`(2) call failed. The error message will be displayed on `stderr`.
`12`	The `fclose`(3C) call failed. The error message will be displayed on `stderr`.
`13`	The `fflush`(3C) call failed. The error message will be displayed on `stderr`.
`14`	The `fgetpos`(3C) call failed. The error message will be displayed on `stderr`.
`15`	The `fopen`(3C) call failed. The error message will be displayed on `stderr`.
`16`	The `fprintf`(3C) call failed. The error message will be displayed on `stderr`.
`17`	The `fread`(3C) call failed. The error message will be displayed on `stderr`.
`18`	The `fscanf`(3C) call failed. The error message will be displayed on `stderr`.
`19`	The `fseek`(3C) call failed. The error message will be displayed on `stderr`.

20 The fsetpos(3C) call failed. The error message will be displayed on stderr.

21 The ftell(3C) call failed. The error message will be displayed on stderr.

22 The ftruncate(3C) call failed. The error message will be displayed on stderr.

23 The fwrite(3C) call failed. The error message will be displayed on stderr.

24 The ffbksp(3C) call failed. The error message will be displayed on stderr.

25 The ffclose(3C) call failed. The error message will be displayed on stderr.

26 The fffcntl(3C) call failed. The error message will be displayed on stderr.

27 The fflistio(3C) call failed. The error message will be displayed on stderr.

28 The ffopen(3C) call failed. The error message will be displayed on stderr.

29 The ffpos(3C) call failed. The error message will be displayed on stderr.

30 The ffread(3C) call failed. The error message will be displayed on stderr.

31 The ffreada(3C) call failed. The error message will be displayed on stderr.

32 The ffseek(3C) call failed. The error message will be displayed on stderr.

33 The ffsetsp(3C) call failed. The error message will be displayed on stderr.

34 The ffweod(3C) call failed. The error message will be displayed on stderr.

35 The ffweof(3C) call failed. The error message will be displayed on stderr.

36 The ffwrite(3C) call failed. The error message will be displayed on stderr.

37 The ffwritea(3C) call failed. The error message will be displayed on stderr.

38 The accept(2) call failed. The error message will be displayed on stderr.

39 The bind(2) call failed. The error message will be displayed on stderr.

40 The connect(2) call failed. The error message will be displayed on stderr.

41 The gethostbyname(3C) or hostGetByName() call failed. The error message will be displayed on stderr.

42 The gethostname(2) call failed. The error message will be displayed on stderr.

43 The getsockopt(2) call failed. The error message will be displayed on stderr.

44 The listen(2) call failed. The error message will be displayed on stderr.

45 The recv(2) call failed. The error message will be displayed on stderr.

46 The send(2) call failed. The error message will be displayed on stderr.

47 The setsockopt(2) call failed. The error message will be displayed on stderr.

48 The shutdown(2) call failed. The error message will be displayed on stderr.

49 The socket(2) call failed. The error message will be displayed on stderr.

50	The `fork`(2) call failed. The error message will be displayed on `stderr`.
51	The `kill`(2) call failed. The error message will be displayed on `stderr`.
52	The `signal`(2) call failed. The error message will be displayed on `stderr`.
100	The data read does not match the expected data. The differences are written to the log file.
101	A read request was made but not all data was received.
102	A write request was made but not all data was sent.
255	An error was encountered while parsing the command-line options. The command synopsis will be displayed on `stderr`.

SEE ALSO

GigaRing based systems:

`ifconfig`(8)

Concurrent Maintenance Tools User's Guide, publication SD–2627

Model E systems:

`panic`(8)

`listio`(2) in the *UNICOS System Calls Reference Manual*, Cray Research publication SR–2012

`hippi`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

vst – Exercises network devices capable of running TCP/IP

SYNOPSIS

All Cray Research systems:

/etc/diag/vst [-a *action*] [-b *blocksize-max*[:*blocksize-min*] | pass] [-e *echo-file-name*] [-H *host-name*[:*port-number*]] [-l *logfile*] [-n *nprocs*] [-o *options*] [-P *pattern*] [-p *pass-end*[:*pass-start*][:*pass-step*]] [-T *timeout*] [-t *tracefile*]

/etc/diag/vst -h

/etc/diag/vst -V

Cray system workstation (SWS):

/opt/CYRIccmt/bin/vst [-a *action*] [-b *blocksize-max*[:*blocksize-min*] | pass] [-e *echo-file-name*] [-H *host-name*[:*port-number*]] [-l *logfile*] [-n *nprocs*] [-o *options*] [-P *pattern*] [-p *pass-end*[:*pass-start*][:*pass-step*]] [-T *timeout*] [-t *tracefile*]

/opt/CYRIccmt/bin/vst -h

/opt/CYRIccmt/bin/vst -V

IMPLEMENTATION

All Cray Research systems
Cray system workstation (SWS)

DESCRIPTION

The online, socket-based, device-independent network exerciser, vst, will exercise any network device that supports TCP/IP and has been configured into the operating system.

vst can be used to exercise TCP/IP network devices between two Cray systems, or between the System Workstation (SWS) and the Cray system (GigaRing only), or between the SWS and the multipurpose node (MPN) (GigaRing only). The general procedure is to open a listening connection on one platform (using the -a read option) and then attempt to send data to that connection (using the -a write option). With the exception of the MPN, it is possible to send and receive data on the same host.

Block size, patterns, and the number of passes must be the same on both the receiving and sending hosts.

The vst command accepts the following options:

-a *action*
Specifies the action to be performed. Valid values are as follows:

`ping` Uses the Internet Control Message Protocol's (ICMP) `ECHO_REQUEST` packets to elicit `ECHO_RESPONSE` packets from the specified host, similar to the `ping`(8) command. The block size is limited to 16,376 bytes. SWS, UNICOS, or UNICOS/mk systems only (not valid on the MPN).

`rw` Reads and writes data on the same machine. SWS, UNICOS, or UNICOS/mk systems only (not valid on the MPN).

`read` Listens for and reads data from the cooperating process. This action must be started before the `vst -a write` action is started.

`write` Writes data to the host specified by the `-H` option (the default host is the machine on which `vst` is run). The cooperating `vst -a read` process must have been started first.

The default is `rw` on the SWS, UNICOS, and UNICOS/mk systems. On the MPN, the default is `read`.

`-b` *blocksize-max*[`:`*blocksize-min*] | `pass`
Specifies the block size to be used for I/O operations. When the *blocksize-min* value is used, the actual block size will be randomly set to a value between *blocksize-min* and *blocksize-max* for each pass. The maximum block size must be the same on both the receiving and sending host. When `pass` is specified, the block size will be set to the current pass count value. The block size is limited to 16376 bytes for the `ping` action.

`-e` *echo-file-name*
Specifies that the first data buffer is to be written to the specified file. No actual I/O to a socket is performed.

`-H` *host-name*[`:`*port-number*]
Specifies the name of the host on which the cooperating process should be located. The optional port number specifies the TCP/IP port to be used.

`-h` Displays the command-line synopsis on `stderr`. `vst` then exits.

`-l` *logfile*
Specifies where the `vst` log file should be written. The file specification can be either absolute or relative. By default, logging is disabled.

`-n` *nprocs*
Specifies how many `vst` processes are to be started. The default is one process. This option can be used to stress a device by increasing data throughput to or from the device. This option is not available in VxWorks.

`-o` *options*
Specifies a colon-separated list of options to be turned on. Valid values are as follows:

`debug` Displays large amounts of information related to what `vst` is doing.

`fast` Does not compare data.

`lock` Deferred implementation. Locks the process(es) into memory to prevent swapping. UNICOS or UNICOS/mk systems only.

`panic` When a data miscompare occurs, panics the system. UNICOS or UNICOS/mk systems only.

`squeeze` Deferred implementation. UNICOS or UNICOS/mk systems only.

`ssd` Deferred implementation. Places the data buffers in SSD (Cray PVP systems only). UNICOS or UNICOS/mk systems only.

`verbose` Displays information about the progress of `vst`. This argument is valid only with the `rw` or `read` action. It has no effect on the `write` action.

By default, all options are turned off.

`-P` *pattern*

Specifies the data pattern to be used. Valid values are as follows:

`bits` Each 64-bit word has a random sequence of consecutive 1-bits.

`slide0` The first 64-bit word has all bits set except bit 0, which is cleared. Subsequent words are circularly left-shifted by one bit position.

`slide1` The first 64-bit word has all bits cleared except bit 0, which is set. Subsequent words are circularly left-shifted by one bit position.

`random` Each 64-bit word is set to a random value.

`all` All of the built-in data patterns are used, one per pass, in a circular fashion.

Instead of selecting one of the built-in patterns, you can specify a numeric constant that will be used to set each 64-bit word of the data buffer. The default is `all`. Patterns must be the same on both the receiving and sending host.

`-p` *pass-end*[`:`*pass-start*[`:`*pass-step*]]

Specifies the number of passes to be performed. The number of passes must be the same on both the receiving and sending host. *pass-end* specifies the ending pass number; for example, `-p 1000` executes `vst` up to pass 1,000. *pass-start* specifies the starting pass number; for example, `-p 1000:500` executes `vst` starting at pass 500, and continuing up to pass 1,000. *pass-step* specifies how pass steps should be incremented; for example, `-p 1000:500:100` executes `vst` starting at pass 500, and then steps up to pass 600, and so on, up to pass 1,000. *pass-start* and *pass-step* are most useful when specified with the `-b pass` option.

`-T` *timeout*

Specifies the time-out value to be used when asynchronous I/O is performed. The default is 10 seconds.

`-t` *tracefile*

Specifies where the `vst` trace file should be written. The file specification can be either absolute or relative. By default, tracing is disabled.

`-V` Displays version information on `stderr`. `vst` then exits.

NOTES

For the SWS, the path to this command and the appropriate environment variable settings are provided by the `craydiag` and `crayadm` modules.

EXIT STATUS

Exit Code	Description
`0`	The command completed successfully with no I/O errors and no data miscompares.
`1`	The `close`(2) call failed. The associated error message text will be displayed on `stderr`.
`2`	The `creat`(2) call failed. The associated error message text will be displayed on `stderr`.
`3`	The `fcntl`(2) call failed. The associated error message text will be displayed on `stderr`.
`4`	The `ioctl`(2) call failed. The associated error message text will be displayed on `stderr`.
`5`	The `lseek`(2) call failed. The associated error message text will be displayed on `stderr`.
`6`	The `open`(2) call failed. The associated error message text will be displayed on `stderr`.
`7`	The `read`(2) call failed. The associated error message text will be displayed on `stderr`.
`8`	The `reada`(2) call failed. The associated error message text will be displayed on `stderr`.
`9`	The `stat`(2) call failed. The associated error message text will be displayed on `stderr`.
`10`	The `write`(2) call failed. The associated error message text will be displayed on `stderr`.
`11`	The `writea`(2) call failed. The associated error message text will be displayed on `stderr`.
`12`	The `fclose`(3C) call failed. The associated error message text will be displayed on `stderr`.
`13`	The `fflush`(3C) call failed. The associated error message text will be displayed on `stderr`.
`14`	The `fgetpos`(3C) call failed. The associated error message text will be displayed on `stderr`.
`15`	The `fopen`(3C) call failed. The associated error message text will be displayed on `stderr`.
`16`	The `fprintf`(3C) call failed. The associated error message text will be displayed on `stderr`.
`17`	The `fread`(3C) call failed. The associated error message text will be displayed on `stderr`.
`18`	The `fscanf`(3C) call failed. The associated error message text will be displayed on `stderr`.
`19`	The `fseek`(3C) call failed. The associated error message text will be displayed on `stderr`.

20 The `fsetpos`(3C) call failed. The associated error message text will be displayed on `stderr`.

21 The `ftell`(3C) call failed. The associated error message text will be displayed on `stderr`.

22 The `ftruncate`(3C) call failed. The associated error message text will be displayed on `stderr`.

23 The `fwrite`(3C) call failed. The associated error message text will be displayed on `stderr`.

24 The `ffbksp`(3C) call failed. The associated error message text will be displayed on `stderr`.

25 The `ffclose`(3C) call failed. The associated error message text will be displayed on `stderr`.

26 The `fffcntl`(3C) call failed. The associated error message text will be displayed on `stderr`.

27 The `fflistio`(3C) call failed. The associated error message text will be displayed on `stderr`.

28 The `ffopen`(3C) call failed. The associated error message text will be displayed on `stderr`.

29 The `ffpos`(3C) call failed. The associated error message text will be displayed on `stderr`.

30 The `ffread`(3C) call failed. The associated error message text will be displayed on `stderr`.

31 The `ffreada`(3C) call failed. The associated error message text will be displayed on `stderr`.

32 The `ffseek`(3C) call failed. The associated error message text will be displayed on `stderr`.

33 The `ffsetsp`(3C) call failed. The associated error message text will be displayed on `stderr`.

34 The `ffweod`(3C) call failed. The associated error message text will be displayed on `stderr`.

35 The `ffweof`(3C) call failed. The associated error message text will be displayed on `stderr`.

36 The `ffwrite`(3C) call failed. The associated error message text will be displayed on `stderr`.

37 The `ffwritea`(3C) call failed. The associated error message text will be displayed on `stderr`.

38 The `accept`(2) call failed. The associated error message text will be displayed on `stderr`.

39	The `bind`(2) call failed. The associated error message text will be displayed on `stderr`.
40	The `connect`(2) call failed. The associated error message text will be displayed on `stderr`.
41	The `gethostbyname`(3C) or `hostGetByName`() call failed. The associated error message text will be displayed on `stderr`.
42	The `gethostname`(2) call failed. The associated error message text will be displayed on `stderr`.
43	The `getsockopt`(2) call failed. The associated error message text will be displayed on `stderr`.
44	The `listen`(2) call failed. The associated error message text will be displayed on `stderr`.
45	The `recv`(2) call failed. The associated error message text will be displayed on `stderr`.
46	The `send`(2) call failed. The associated error message text will be displayed on `stderr`.
47	The `setsockopt`(2) call failed. The associated error message text will be displayed on `stderr`.
48	The `shutdown`(2) call failed. The associated error message text will be displayed on `stderr`.
49	The `socket`(2) call failed. The associated error message text will be displayed on `stderr`.
50	The `fork`(2) call failed. The associated error message text will be displayed on `stderr`.
51	The `kill`(2) call failed. The associated error message text will be displayed on `stderr`.
52	The `signal`(2) call failed. The associated error message text will be displayed on `stderr`.
100	The data read does not match the expected data. The differences are written to the log file.
101	A read request was made but not all data was received.
102	A write request was made but not all data was sent.
255	An error was encountered while parsing the command-line options. The command synopsis will be displayed on `stderr`.

SEE ALSO

SWS-ION Administration and Operations Guide, publication SG–2204

NAME

vtt – Provides tape device testing

SYNOPSIS

/etc/diag/vtt [-a *action*] [-b *blocksize-max*[:*blocksize-min*] | pass] [-e *echo-filename*] [-l *logfile*] [-o *options*] [-P *pattern*] [-p *pass-end*[:*pass-start*[:*pass-step*]]] [-T *timeout*] [-t *tracefile*] *device-name*

/etc/diag/vtt -h

/etc/diag/vtt -V

IMPLEMENTATION

GigaRing based Cray Research systems

DESCRIPTION

The online tape exerciser, vtt, will exercise any online tape devices that have been configured into the operating system. The devices may be configured either down or up. When using devices that are configured down, the block-special node in /dev/tape will be accessed directly. For devices that are configured up, device access will go through the tape daemon. For up mode tests, the *device-name* is assumed to be configured up via rsv(1) and tpmnt(1).

The vtt command accepts the following options:

-a *action*

Specifies the action to be performed. Valid values are as follows:

rw Write a number of blocks, rewind the tape, read back the same number of blocks, and compare them to expected data values.

read Read a number of blocks and compare them to expected data values.

write Write a number of blocks.

thrash

Write a single block, rewind the tape, read back the block, and compare to expected data values. This sequence is repeated for each pass.

eot Write data until the end-of-tape is reached, rewind the tape, read back all the blocks, and compare them to expected data values. This test is only available if operating on a tape device that has been configured down (that is, /dev/tape/*).

mv Write data until a volume switch is performed, rewind the tape, read back all the blocks, and compare them to expected data values. This test is only available if operating on a tape device that has been configured up (that is, obtained via rsv(1) and tpmnt(1)).

`block` Use block positioning operations to skip around the tape to write, read, and compare data values. This test is only available if operating on a tape device that has been configured up (that is, obtained via `rsv`(1) and `tpmnt`(1)).

`tapemark`
Use tapemark positioning operations to skip around the tape to write, read, and compare data values. This test is only available if operating on a tape device that has been configured down (that is, `/dev/tape/*`).

The default is `rw`.

`-b` *blocksize-max*[`:`*blocksize-min*] | `pass`
Specifies the block size to be used for I/O operations. When the *blocksize-min* value is used, the actual block size will be randomly set to a value between *blocksize-min* and *blocksize-max* for each pass. When `pass` is specified, the block size will be set to the current pass count value.

`-e` *echo-file-name*
Specifies that the first data buffer is to be written to the file. No actual I/O to the device is performed.

`-h` Displays the command-line synopsis on `stderr`. `vtt` then exits.

`-l` *logfile*
Specifies where the exerciser's log file should be written. The file specification can be either absolute or relative. By default, logging is disabled.

`-o` *options*
Specifies a colon-separated list of options to be turned on. Valid values are as follows:

`debug` Displays large amounts of information related to what the exerciser is doing.

`fast` Does not compare data.

`panic` When a data miscompare occurs, panics the system.

`verbose` Displays information about the progress of the exerciser.

By default, all options are turned off.

`-P` *pattern*
Specifies the data pattern to be used. Valid values are as follows:

`bits` Each 64-bit word has a random sequence of consecutive 1-bits.

`slide0` The first 64-bit word has all bits set except bit 0, which is cleared. Subsequent words are circularly left-shifted by one bit position.

`slide1` The first 64-bit word has all bits cleared except bit 0, which is set. Subsequent words are circularly left-shifted by one bit position.

`random` Each 64-bit word is set to a random value.

`all` All of the built-in data patterns are used, one per pass, in a circular fashion.

Instead of selecting one of the built-in patterns, the user can specify a numeric constant that will be used to set each 64-bit word of the data buffer. The default is `all`.

`-p` *pass-end*[`:`*pass-start*[`:`*pass-step*]]
Specifies the number of passes to be performed (`thrash`) or blocks to be written/read (`rw`, `read`, `write`). This option is ignored for all other actions. The specification used when reading a tape should be the same as the specification used when writing the tape. *pass-end* specifies the ending pass number; for example, `-p 1000` executes `vtt` up to pass 1,000. *pass-start* specifies the starting pass number; for example, `-p 1000:500` executes `vtt` starting at pass 500, and continuing up to pass 1,000. *pass-step* specifies how pass steps should be incremented; for example `-p 1000:500:100` executes `vtt` starting at pass 500 and then steps up to pass 600, and so on, up to pass 1,000. *pass-start* and *pass-step* are most useful when specified with the `-b pass` option.

`-T` *timeout*
Specifies the timeout value to be used when asynchronous I/O is performed.

`-t` *tracefile*
Specifies where the exerciser's trace file should be written. The file specification can be either absolute or relative. By default, tracing is disabled.

`-V` Displays version information on `stderr`. `vtt` then exits.

device-name
Required option. Specifies the tape device upon which the I/O operations are to be performed.

EXIT STATUS

Exit Code	Description
`0`	The command completed successfully with no I/O errors and no data miscompares.
`1`	The `close`(2) call failed. The associated error message text will be displayed on `stderr`.
`2`	The `creat`(2) call failed. The associated error message text will be displayed on `stderr`.
`3`	The `fcntl`(2) call failed. The associated error message text will be displayed on `stderr`.
`4`	The `ioctl`(2) call failed. The associated error message text will be displayed on `stderr`.
`5`	The `lseek`(2) call failed. The associated error message text will be displayed on `stderr`.
`6`	The `open`(2) call failed. The associated error message text will be displayed on `stderr`.
`7`	The `read`(2) call failed. The associated error message text will be displayed on `stderr`.
`8`	The `reada`(2) call failed. The associated error message text will be displayed on `stderr`.
`9`	The `stat`(2) call failed. The associated error message text will be displayed on `stderr`.
`10`	The `write`(2) call failed. The associated error message text will be displayed on `stderr`.
`11`	The `writea`(2) call failed. The associated error message text will be displayed on `stderr`.

12 The `fclose`(3C) call failed. The associated error message text will be displayed on `stderr`.

13 The `fflush`(3C) call failed. The associated error message text will be displayed on `stderr`.

14 The `fgetpos`(3C) call failed. The associated error message text will be displayed on `stderr`.

15 The `fopen`(3C) call failed. The associated error message text will be displayed on `stderr`.

16 The `fprintf`(3C) call failed. The associated error message text will be displayed on `stderr`.

17 The `fread`(3C) call failed. The associated error message text will be displayed on `stderr`.

18 The `fscanf`(3C) call failed. The associated error message text will be displayed on `stderr`.

19 The `fseek`(3C) call failed. The associated error message text will be displayed on `stderr`.

20 The `fsetpos`(3C) call failed. The associated error message text will be displayed on `stderr`.

21 The `ftell`(3C) call failed. The associated error message text will be displayed on `stderr`.

22 The `ftruncate`(3C) call failed. The associated error message text will be displayed on `stderr`.

23 The `fwrite`(3C) call failed. The associated error message text will be displayed on `stderr`.

24 The `ffbksp`(3C) call failed. The associated error message text will be displayed on `stderr`.

25 The `ffclose`(3C) call failed. The associated error message text will be displayed on `stderr`.

26 The `fffcntl`(3C) call failed. The associated error message text will be displayed on `stderr`.

27 The `fflistio`(3C) call failed. The associated error message text will be displayed on `stderr`.

28 The `ffopen`(3C) call failed. The associated error message text will be displayed on `stderr`.

29 The `ffpos`(3C) call failed. The associated error message text will be displayed on `stderr`.

30 The `ffread`(3C) call failed. The associated error message text will be displayed on `stderr`.

31 The `ffreada`(3C) call failed. The associated error message text will be displayed on `stderr`.

32 The ffseek(3C) call failed. The associated error message text will be displayed on stderr.

33 The ffsetsp(3C) call failed. The associated error message text will be displayed on stderr.

34 The ffweod(3C) call failed. The associated error message text will be displayed on stderr.

35 The ffweof(3C) call failed. The associated error message text will be displayed on stderr.

36 The ffwrite(3C) call failed. The associated error message text will be displayed on stderr.

37 The ffwritea(3C) call failed. The associated error message text will be displayed on stderr.

38 The accept(2) call failed. The associated error message text will be displayed on stderr.

39 The bind(2) call failed. The associated error message text will be displayed on stderr.

40 The connect(2) call failed. The associated error message text will be displayed on stderr.

41 The gethostbyname(3C) or hostGetByName() call failed. The associated error message text will be displayed on stderr.

42 The gethostname(2) call failed. The associated error message text will be displayed on stderr.

43 The getsockopt(2) call failed. The associated error message text will be displayed on stderr.

44 The listen(2) call failed. The associated error message text will be displayed on stderr.

45 The recv(2) call failed. The associated error message text will be displayed on stderr.

46 The send(2) call failed. The associated error message text will be displayed on stderr.

47 The setsockopt(2) call failed. The associated error message text will be displayed on stderr.

48 The shutdown(2) call failed. The associated error message text will be displayed on stderr.

49 The socket(2) call failed. The associated error message text will be displayed on stderr.

50 The fork(2) call failed. The associated error message text will be displayed on stderr.

51 The kill(2) call failed. The associated error message text will be displayed on stderr.

52 The signal(2) call failed. The associated error message text will be displayed on stderr.

100 The data read does not match the expected data. The differences are written to the log file.

`101`	A read request was made but not all data was received.
`102`	A write request was made but not all data was sent.
`255`	An error was encountered while parsing the command-line options. The command synopsis will be displayed on `stderr`.

NAME

wall – Writes to all users

SYNOPSIS

/etc/wall [-d *secs*] [-g *group*] [*file*]

IMPLEMENTATION

All Cray Research systems

STANDARDS

POSIX, XPG4

DESCRIPTION

The wall command reads standard input, or a specified *file*, until an end-of-file (EOF) is reached. It then sends the message written to standard input to all users currently logged-in, preceded by the following text:

Broadcast Message from *user* at *nodename date*

A typical use of the wall message is to warn all users before shutting down the system.

The wall command accepts the following options and arguments:

-d *secs* The -d option specifies the number of seconds wall will wait for the message to be sent to all logged-in users. wall will exit after this amount of time even though the message was not delivered to all logged-in users. The default is 60 seconds.

-g *group* The -g option allows a message to be sent only to members of the *group* specified. The *group* is defined in the /etc/group file (see group(5)).

The sender must have super-user status to override any protections the users may have invoked (see mesg(1)).

NOTES

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active category	Action
system, secadm, sysadm	Allowed to write to all users.

If the PRIV_SU configuration option is enabled, the super user is allowed to write to all users.

MESSAGES

`Cannot send to` *user*`...`
This message occurs when the `open`(2) system call on a user's tty file fails.

FILES

`/etc/group`	Group file containing groups names and group IDs
`/dev/tty*`	Login devices
`/etc/udb`	User validation file containing user control limits
`/etc/utmp`	User information file

SEE ALSO

`mesg`(1), `uname`(1), `write`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`group`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

xadmin – Manages user login accounts through a graphical user interface

SYNOPSIS

xadmin [X11 *options*] [-D *t:d*] [-p *dir*] [-u *user*] [-x *dir*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The xadmin command is a graphical user interface (GUI) that replaces the nu(8) command. It allows administrators to update or view the user database (UDB) on a UNICOS system. You can create, modify, or delete user accounts. All the functionality of nu(8) is available using xadmin.

The GUI is intended to be self-explanatory. Extensive online help is available from the main menu bar. The help button accesses context help or the tutorial. The tutorial button on the main menu bar lets you view the help topics either by clicking your mouse button on the selected help option or by paging through the text, using the scroll bar at the right of the tutorial screen. Help topics describe X11 Window System settings and the xadmin displays.

When invoking xadmin, the command line can include the following options:

X11 *options* Sets the X11 default values if the *options* argument is missing. Alternatively, you can specify any valid DISPLAY shell environment variables as *options*.

-D *t:d* Sets message tracing and debugging levels. *t* is a trace message level integer value between 0 and 9 (inclusive). *d* is a debug message level integer value between 0 and 9 (inclusive). Generally, the higher the value you set, the more detailed will be the message and debugging information you receive.

-p *dir* Sets the path name of the directory that contains the udb, udb.public, acid, group passwd, and nu.cf60 files. The default directory is /etc.

-u *user* Sets read-only mode. Specifies the user name or the user ID whose data will be displayed initially. Read-only mode also is invoked if you do not have access privilege to the UDB when you execute the xadmin command.

-x *dir* Sets the path name of the directory that contains the files for the field computations and the prototypes. The default directory is /etc/nulib.

NOTES

The xadmin command traps interrupt signals (such as <CONTROL-c>) and does not end if you try to stop it in the middle of a critical section. Critical sections are primarily the updates of the UDB. A list of changes is recorded in a log file, usually /usr/adm/xa.log.

If this command is installed with a privilege assignment list (PAL), a user with one of the following active categories is allowed to perform the actions shown:

Active Category	Action
`system`, `secadm`	Modify all UDB fields.
`sysadm`	Modify all UDB fields, except sensitive fields.

If the `PRIV_SU` configuration option is enabled, the super user may modify all UDB fields.

CAUTIONS

Because any change to the `xadmin` or the `nu`(8) command executes as `root` user, a poorly written script potentially could make changes anywhere in the system.

WARNINGS

When you are experimenting with `xadmin`, you should use copies of the system UDB files from the `/etc` directory in a local directory, reducing risk to the UDB. For more information on creating test files, see the online `xadmin` help menu.

BUGS

The extensive use of shell scripts for doing sensitive duties (such as deleting and changing accounts) means that someone may be able to make `xadmin` fail in destructive ways without having access to the source code.

FILES

For the following files the default directory is `/etc`; you can change it by using the `-p` option.

`/etc/acid`	Account file
`/etc/group`	System group file
`/etc/nu.cf60`	Configuration file
`/etc/nulib/*.sh`	Shell scripts to perform the work
`/etc/nulib/nu.cf60`	Release default configuration file
`/etc/udb`	User validation file that contains user control limits
`/etc/udb.public`	Public version of the user database shell scripts to perform the work
`/usr/include/udb.h`	User database structure

SEE ALSO

nu(8), udbgen(8)

udbsee(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

getgrent(3C), getpwent(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

acct(5), group(5), udb(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

xdi – Graphical user interface for X Window System based diagnostic applications

SYNOPSIS

All GigaRing based Cray Research systems:

/etc/diag/xdi [-help] [-machine *host-of-diagnostic-client*] [-trace] [*diagnostic-client-name*]

Cray system workstation (SWS):

/opt/CYRIccmt/bin/xdi [-help] [-machine *host-of-diagnostic-client*] [-trace] [*diagnostic-client-name*]

IMPLEMENTATION

All GigaRing based Cray Research systems
Cray system workstation (SWS)

DESCRIPTION

The xdi graphical user interface (GUI) makes available a generic X Window System interface for client/server diagnostic applications.

xdi can be started by using default or user-supplied command-line options. Default resources can be defined in the application-defaults file /usr/lib/X11/app_defaults/XDi.

The xdi command accepts the following options:

-help Specifies that the command-line synopsis should be written to stderr, after which xdi immediately exits.

-machine *host-of-diagnostic-client*
Specifies the machine on which the diagnostic client should be running.

-trace
Specifies that a trace file be created that will contain a variety of information reflecting the flow of execution.

diagnostic-client-name
Specifies the diagnostic client to which to connect.

To start xdi using default options, enter the following on the command line:

```
xdi
```

To start xdi using the command-line options, enter a command similar to the following:

```
xdi -m cool diagsio
```

A pop-up window is displayed that prompts you for the diagnostic client and the host machine. If you specified this information on the command line, or if default options are defined in the application-defaults file, the information is displayed in the appropriate input fields of the pop-up window (to allow you to

modify the settings if necessary). Click on the OK button to initiate the connection to the diagnostic client.

After the appropriate information is gathered, xdi tries to connect to the specified diagnostic client. If the diagnostic client is not running, xdi attempts to start the client (not yet implemented).

When the xdi interface appears, text and data specific to the requested diagnostic client also appear.

To exit the xdi interface, click on the Exit option from the File pull-down menu (which is displayed by clicking on the File button in the menu bar at the top of the interface window).

NOTES

You must be in the craydiag group to execute this command.

The path to this command and the appropriate environment variable settings are provided by the craydiag and crayadm modules.

MENUS

xdi has four menus:

- File
- Logs
- Connections
- Help

The following are the options on the File menu:

Open...
: Brings up a file selection dialog that enables you to navigate the file system on which xdi is executing. From this dialog you can select a file whose contents will be displayed in a scrolled text window.

Save
: Saves the contents of all xdi regions and windows to the file xdi.save.

Save As...
: Saves the contents of all xdi regions and windows to the file name of your choice.

Print
: Prints the contents of all xdi regions and windows.

Trace
: Toggles whether or not the trace file facility is active.

Exit
: Exits xdi after displaying a confirmation window.

The following are the options on the Logs menu:

Version
: Displays a scrolled text window containing UNICOS source manager (USM) version information from xdi and the diagnostic client.

Error
: Displays a scrolled text window containing any error messages xdi needs you to see. When an error occurs, the keyboard bell is rung.

`Message`
Displays a scrolled text window containing messages from the diagnostic client.

The following are the options on the `Connections` menu:

`Create...`
Displays a window asking for information about which diagnostic client to connect to and on which machine that diagnostic client shoule be running. This menu item is active only when there are no current connections.

`Delete`
Deletes the connection to the diagnostic client. This menu item is active only when an active connection exists.

The following are the options on the `Help` menu:

`Overview`
Deferred implementation.

`Diagnostic`
Requests help information from the diagnostic client. Because the diagnostic client knows what is being displayed by `xdi`, the help is typically context-sensitive.

`On_Item`
Turns the mouse pointer into a question mark. Selecting a region of `xdi` using this pointer displays a scrolled text window that contains information about how `xdi` is treating that region.

`Using_Help`
Deferred implementation.

`About_xdi`
Deferred implementation.

LAYOUT

The `xdi` interface is divided into several regions. Each region is independent of the other regions. Any relationship between information in one region and information in another is defined by the diagnostic client. The regions are as follows:

Title bar
Used by the diagnostic client to display short messages to the user.

Lists A, B, and C
These lists can be independently configured in three different ways: (1) as a list that allows a single selection and immediately informs the diagnostic client; (2) as a list that allows a single selection without the immediate action; and (3) as a list that allows multiple selections. These configurations are completely controlled by the diagnostic client. For the latter two configurations, the selected information is sent to the diagnostic client when a button from the parameter region is pressed.

The left-most list is *list A*. Just to the right of list A is *list B*. If *list C* is being used at all, it will be directly under list B, and list B and list C will be half the size of list A.

Typically, list A contains a list of actions that the diagnostic client can perform, and list B contains the devices or files on which to perform those actions.

Parameters
: This region can contain any number of parameter descriptions and entry fields, configured 8 or 16 to a page. These entry fields can be configured to accept a general character string, only numeric values, or a true/false toggle. The diagnostic client can, and usually does, place a variety of buttons across the bottom of the parameters. `xdi` does not receive information about what these buttons do, but you can try the `Help/Diagnostic` menu item for an explanation.

Output
: This is a scrolled output text area that is of arbitrary length. The visible size is 21 rows and 80 columns. The diagnostic client can place a variety of buttons across the bottom of the text area.

Command-line
: This region allows you to send an arbitrary character string to the diagnostic client. Some diagnostic clients treat the string as a command to be processed by the operating system in which it is running, but this is not the only purpose.

X WINDOW SYSTEM RESOURCES

The following X Window System resources apply to `xdi`:

Resource	Description
`diagClientMachine` (class DiagClientMachine)	Specifies the name of the machine on which the diagnostic client should run.
`diagClientName` (class DiagClientName)	Specifies the name of the diagnostic client to which to connect.
`help` (class Help)	If true, the command-line synopsis is displayed to `stderr`, and `xdi` immediately exits. The default is false.
`printCommand` (class PrintCommand)	Specifies the local print command to be used when the `File --> Print` menu item is chosen.
`trace` (class Trace)	If true, a trace file is produced as `xdi` executes. This feature is helpful for debugging purposes, but can create large files if left on for an extended period of time. The default is false.

ENVIRONMENT VARIABLES

The following environment variables apply to `xdi`:

Variable	Description
`DISPLAY`	Specifies to which X11 bit map display to attach.

XAPPLRESDIR	Specifies a directory in which to look for X Window System resource files in addition to the system-defined locations.
XKEYSYMDB	Specifies the location of an alternate keyboard symbol table. All X11R5 installations install a file named XKeysymDB, but this file does not always contain the newest definitions needed by a Motif application like xdi. The package that xdi was installed from should contain a file named XKeysymDB to be used with xdi.

FILES

tracefile.*	File that contains trace information.
xdi.save	Default file name to be used when the File/Save menu item is chosen.
XDi	Resource file for xdi.

SEE ALSO

diagccmt(8) for information on the diagnostic client for online I/O exercisers

Concurrent Maintenance Tools User's Guide, publication SD–2627
SWS-ION Administration and Operations Guide, publication SG–2204

NAME

xdms – Provides online disk maintenance and initialization routines for Cray GigaRing attached SCSI and fibre channel devices, which are driven by the xdd driver

SYNOPSIS

/etc/xdms -a disk [-i | -u] *iopath* | *iopath.unit*

/etc/xdms -a info *iopath.unit*

/etc/xdms -a init [-m *mode*] [-A *alternate path*] [-M *minor*] *iopath.unit*

/etc/xdms -a init [-m *mode*] [-M *minor*] *iopath*

/etc/xdms -a init [-M *minor*] [-f *filename*]

/etc/xdms -a flaw -b blkno *iopath.unit*[-*spindle*]

/etc/xdms -a format -b block size *iopath.unit*[-*spindle*]

/etc/xdms -a readft *iopath.unit*[-*spindle*]

/etc/xdms -a read [-b *blkno range* | -C] [-p *pattern*] [-r *request size*] [-l *loop*] [-s *seed*] [-vz] *iopath.unit*[-*spindle*]

/etc/xdms -a write [-b *blkno range* | -C] [-m *mode*] [-p *pattern*] [-s *seed*] [-r *request size*] [-l *loop*] [-vz] *iopath.unit*[-*spindle*]

/etc/xdms -a surf [-b *blkno range* | -C] [-m *mode*] [-p *pattern*] [-s *seed*] [-r *request size*] [-l *loop*] [-vz] *iopath.unit*[-*spindle*]

/etc/xdms -a reconstruct *iopath.unit*

/etc/xdms -a disable *iopath.unit*-*spindle*

/etc/xdms -a scrub *iopath.unit*

/etc/xdms -a spinup *iopath.unit*[-*spindle*]

/etc/xdms -a spindown *iopath.unit*[-*spindle*]

/etc/xdms -a lducode -f filename *iopath.unit*

/etc/xdms -a ppage [-p *pagenum*] -r *iopath.unit*

/etc/xdms -a inquiry *iopath.unit*

IMPLEMENTATION

Cray Research systems that support GigaRing based multipurpose nodes (MPNs) and fibre channel nodes (FCNs) running SCSI disks

DESCRIPTION

The `xdms` (eXtent Disk driver Maintenance System) command provides disk maintenance, initialization, and information gathering for SCSI or Fibre/SCSI devices attached to a multipurpose node (MPN) and the fibre channel node (FCN-1).

The command line is made up of the command to run (`xdms`), the `-a` option defining the action, and the desired options for the action, followed by the *iopath.unit* to be run to. If the action is to be run against an individual spindle of an array device, a spindle indicator may also be supplied to the *iopath.unit* field (for example, `020204.11-3`, which is ring 20, node 20, channel 4, unit 11, and spindle 3).

To execute `xdms`, a character device node must be set up in `/dev/xdd/xdiag`. This device node must be of major type `xdd` with a minor number of 0.

To run `xdms`, you must either have `diag` privilege set in the user database (UDB) or be running as `root`. In addition, to have permission to write to customer data areas, your UDB must also contain `guard` privilege or be running as `root`.

The `xdms` command accepts the following options:

`-a` Specifies the action to be performed. Valid actions are as follows:

`disk` Generates output that defines the device mode, worldwide name (if a fibre channel device), vendor identification, model, serial number, I/O path defining the ring and node number given on the command line with the appropriate channel and unit number, and an alternate path if one can be determined. This default output may be used as input to the `init` action. See the EXAMPLES section for output from the `disk` action.

`info` Provides output for a given disk device. The output consists of the Disk Unit Identification information of a Report Characteristics request as well as Disk Block information if the device has been initialized.

`init` Defines the mode of access to be used by an FCN-1 device. MPN devices are automatically set to single-spindle devices so an `init` action is not required. However, for both FCN-1 and MPN devices, an `-M` option may be specified with a minor number to force the creation of a character special file in the `/dev/ddd` directory. See the description of the `-M` option.

Device initialization may take various forms with the `init` action: (1) One device initialization where an *iopath* and *unit* identify the device to be initialized. This form supports all device modes. If a Redundant Arrays of Independent Disks (RAID) device mode is given, the *iopath.unit* must identify the parity or most significant bit in the spindle mask. This must also be the lowest unit in the subrack comprising the RAID device. A sufficient number of devices must fill the subrack to make up the rest of the array.

(2) Full SPN initialization where the *iopath* is given minus the *unit* field, indicating to `init` that all uninitialized devices are on the SPN. If the `-m` option is given with an argument of 1, ALL devices will be initialized as single-spindle devices. This form only supports device mode 01 (single-spindle drive).

(3) Initialization from an input file. This file must be in the form of the output of the `disk` action. The file will be read in and checked for validity. Each device is then initialized with the device mode. This action will overwrite currently set device modes. If a RAID device is to be initialized, each spindle of the device must be entered in order beginning with the parity or most significant spindle in the spindle mask.

Valid device modes are:

00	Uninitialized drive
01	Single-spindle drive
35	RAID-3 4+1 array
54	RAID-5 3+1
55	RAID-5 4+1
56	RAID-5 5+1
57	RAID-5 6+1
58	RAID-5 7+1
59	RAID-5 8+1

Note: Each RAID device requires that there be enough devices consecutively placed to complete the RAID configuration. Each device must be in the same subrack.

`flaw` Reassigns a failing block to the next available spare block in the spare blocking scheme. This action results in the loss of data to the target block. This action is typically only required on unrecoverable block errors because auto reassignment is done by the small computer system interface (SCSI) devices on recoverable error correction code (ECC) read errors. When flawing a block of an array, the spindle value must be supplied with the *iopath.unit*.

`format` Initiates the device formatting process. The block size will default to the current format size of the device. The `-b` option may be used to select a different block size. The only value currently accepted is 512, which indicates 512 Cray words or 4096 bytes. When formatting an array, the spindle value must be supplied with the *iopath.unit*.

`readft` Reads and displays the defect list in block form and displays the defect type in bit mask form. If this action is specified to an array, the spindle value must be supplied with the *iopath.unit* field to indicate from which spindle to read the flaws.

`read` Reads the specified block range given in the `-b` option or the CE blocks if a `-C` option is specified. If the `-p` *pattern* option is specified, the data read will be compared against that pattern. Use the `-r` option to select the size of each request made. If a request size is not given, the request size will be the optimal I/O value given in the disk unit block fields of the report characteristics request. The `-s` *seed* value is used with the `-p` *rand* option to generate consistent pseudo random data. The spindle value may be supplied to run to an individual spindle of an array.

`write` Writes data across the block range given in the `-b` option or the CE blocks if `-C` is used. Data written is defined by the `-p` *pattern* option. The *mode* option is used by users with special privileges to access data areas otherwise restricted by general `xdms` users. The spindle value may be supplied to run to an individual spindle of an array.

`surf` Performs a combination of the `write` then the `read` action and verifies the data read. The `surf` action imposes the same restrictions as the `write` action, which requires the `-m` option to bypass. The spindle value may be supplied to run to an individual spindle of an array.

`reconstruct` Reinstates a downed spindle of an array and rewrites the data of each array block to the full complement of spindles in an array. This action is used when a failing spindle has been either replaced or fixed.

`disable` Allows the manual disabling of a spindle in a disk array. This may be necessary where intermittent disk failures do not cause automatic disabling of the spindle.

`scrub` Writes a pattern across an entire array to set data parity across the array.

`spinup` Remotely spins up a spindle that has been spun down. The spindle value may be used to spin up an individual spindle of an array.

`spindown` Remotely spins down a spindle. Typically this action is used to spin down a downed spindle of an array prior to removing it. The spindle value may be used to spin down an individual spindle of an array.

`lducode` Reads in a file identified by the `-f` option into memory and then transfers it to the device specified by the *iopath.unit*. This file should contain the microcode that is to be used to update the microcode level of the disk drive. If the load microcode action is run on an array, each spindle of the array will be loaded with the new code.

`ppage` Prints the SCSI mode pages from the device. If the mode page is a recognizable mode page, the output will define the values. Otherwise, the output will be in raw hexadecimal form. The `-r` option displays all pages in raw hexadecimal form. The `-p` option may be used to display individual mode pages. This option is currently available only for MPN devices.

`inquiry` Prints the contents of the SCSI `inquiry` command. This option is currently available only for MPN devices.

`-A` Use with the `init` action to provide an alternate path. May be used when a device node is to be created. See the description of the `-M` option.

`-b` Use with the `read`, `write`, and `surf` actions to indicate the range of blocks to be read/written to. The block range is specified by giving a lower block value followed by a dash (-), followed by the upper block value. For example, `-b 5-10` executes on blocks 5 through 10.

Use with the `flaw` action to indicate which block to be assigned to an alternate block.

Use with the `format` action to indicate the desired block size to be used during formatting. The default is the current block size of the device.

`-C` May be used in place of the `-b` option on the `read`, `write`, and `surf` actions to indicate to run to the CE blocks only.

`-f` Where indicated, the `-f` option specifies the file name to be used for the appropriate action.

`-i` Use with the `disk` action to display only initialized devices.

`-l` Use with the `read`, `write`, and `surf` actions to indicate the desired pass count (the default is 1).

`-m` Use with the `init` action to indicate the device mode desired. The default device mode is 1 (single-spindle device).

Use with the `write` or `surf` action to allow a user to write customer data areas if an `unguard` option is specified. Only users with `perm guard` privilege in their user database (UDB) may allocate an `unguard` option (the default is `guard`).

`-M` Use with the `init` action to tell `xdms` to create a device node in `/dev/ddd` using the device characteristics obtained by the device. The argument given with this option should be a valid nonzero minor number. If the `-A` option is also specified, the device node will be created with an alternate path.

`-p` Use with the `read`, `write`, and `surf` actions to indicate the desired pattern to be used when testing. Valid options are zeros, ones, `hilo`, `peak`, `hole`, `bump`, `rand`, `seq`, and `addr`. The default is zeros, ones, `hilo`, `peak`, `hole`, `bump`, and `rand`. Use with the `ppage` action to identify the mode page (*pagenum*) to print.

`-r` Use with the `read`, `write`, and `surf` actions to indicate the desired size of each request to be issued. The default is the optimal IOU value given with the disk block information. Use with the `ppage` action to print mode pages in raw hexadecimal form.

`-s` Specifies the seed value to be used when generating random data. The default value is generated by the real time clock.

`-u` Use with the `disk` action to display only uninitialized devices.

`-v` Where indicated, runs `xdms` in verbose mode.

`-z` Where indicated, bypasses questions requesting a response to continue. The action will be run without prompting the user.

EXAMPLES

Example 1: Displays the disks attached to ring 20 and node 20. This information may be used as input to the `init` action.

```
# ./xdms -a disk 020200
*
*mode    wwname           model            s/n     iopath.unit   altpath
*
 01 1000002037000061  ST15150FC         88071     020202.1       ------
 01 1000002037000062  ST15150FC         88072     020202.2       ------
 01 1000002037000063  ST15150FC         88073     020202.3       ------
 01 1000002037000064  ST15150FC         88074     020202.4       ------
 01 1000002037000065  ST15150FC         88075     020202.5       ------
 01 ----------------  ST15150FC         88076     020201.1       ------
```

Note: MPN devices are identified by the "--" sequence in the `wwname` field, while FCN-1 devices are shown with valid hexadecimal values in the `wwname` field. The `altpath` field will have a valid path name if an alternate path can be determined. Currently, alternate paths are only determined by having valid channels on both port A and port B for a given device node.

Example 2: Shows a typical `info` action output for ring 20, node 02, channel 0, and unit 4.

```
# ./xdms -a info 020020.4
*
* Disk Information for   020020.4
*
* Mode: 01      Device Ordinal: FFFF      Channel PrtA: 00  PrtB: ff
* Disk Unit State: FF    Spindle Mask: 0000    Unit: 004    LUN: 000
* Interface Type: 02     Command Protocol: 01     Vendor ID: SEAGATE
* Model Number: ST151150C              Serial Number: 12345678
* Firmware Revision: 2500       World Wide Name:  ---------------
*
* Block Size: 512       ECC Field Length: 24   Optimal IOU: 10
* Number of Blocks: 250   CE Blocks: 240-249
```

Example 3: Initializes the device defined by ring number 020, node number 02, channel 4, and unit 21 as a (default) single-spindle device.

```
# ./xdms -a init 020024.21
```

Example 4: Initializes the device defined by ring number 020, node number 02, channel 4, and unit 21 as a RAID-3 array defined by the `-m` option. Units 22, 23, 24, and 25 make up the other spindles of the array. Unit 21 would be the most significant bit of the spindle mask 020 (octal) while unit 25 is the least significant bit 001 (octal). The full mask would be 037 (octal).

```
# ./xdms -a init -m 34 020024.21
```

Example 5: Initializes the device defined by ring number 020, node number 02, channel 4, and unit 21 as a RAID-3 array defined by the `-m` option. The `-A` option defines the alternate path to the physical array.

```
# ./xdms -a init -m 34 -A 020042 020024.21
```

Example 6: Initializes all of the devices on ring number 020 and node number 02 that have a current mode of 00 (uninitialized) as single-spindle devices. All other devices will be unaltered.

```
# ./xdms -a init 020020
```

Example 7: Initializes all of the devices on ring number 020 and node number 02 as 01 (single-spindle device) no matter what the current mode state setting. The `-m` option of 01 forces this state on ALL devices.

```
# ./xdms -a init -m 01 020020
```

Example 8: Creates a file named `myfile` using the `info` action. The file may then be updated by the user to select the desired device modes and alternate path selections. This file is then given as input to the `init` action with the `-f` option.

```
# ./xdms -a info 020020 > myfile
# ./xdms -a init -f myfile
```

SEE ALSO

`udbgen`(8) for information on setting up permbits in the UDB

NAME

`xfddimap` – X Window System utility to draw an FDDI ring map

SYNOPSIS

`/etc/xfddimap` [`-d`] [`-p` *port*] [`-i` *infc*] [`-f` *font*]

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

DESCRIPTION

The `xfddimap` utility transmits FDDI station management (SMT) frames onto an FDDI ring from a Cray Research system that is directly attached to that same FDDI ring, to draw a logical ring map of the FDDI network. Users may also monitor some of the FDDI counters within the FDDI stations connected to the ring and gather information about the stations on the network.

To perform its function, `xfddimap` uses the socket utility interface to the station management daemon (SMTD) running on the same Cray Research system. See `smtd`(8).

The command-line options are as follows:

`-d` Enables debug mode; verbose output.

`-p` *port* Sets the UDP port number to which socket requests are sent. Default is `3000`. See `smtd`(8).

`-i` *infc* Sets the interface ordinal of the FDDI interface to use when sending requests on the FDDI ring. Default is 0.

`-f` *font* Sets the X11 font to the specified font type.

The MAC Address File

You can create a file that contains the IEEE 48-bit media access protocol (MAC) addresses of the FDDI stations (similar to the `/etc/hosts` file) to give names to each of the FDDI stations on the ring. This file is called `/etc/ethers` and is standard on most UNIX-based systems. The format of the file is as follows:

```
#
#    The addresses in this file must be in
#    Canonical (Ethernet) form.
#
0:40:a6:0:0:e0         sn1703b-1
00:40:a6:00:00:10      sn1703b-2
ff:ff:ff:ff:ff:ff      broadcast
```

You can place comment lines anywhere in the file if the comments start in column 1 of the line. Comments are designated by a # character in column 1. Leading zeros within each byte are not required, but they do make it easier for the users to read.

The xfddimap utility consults this file to associate a physical MAC address to a station (or host) name so that names and addresses appear in the ring map.

Logical Ring Map Display

After it is drawn, the logical ring map consists of several rectangles, each representing a physical FDDI station on the ring. Inside each rectangle three items of information appear. The name of the station as found in the /etc/ethers file appears first. Next is the station's CFM state (WRAPPED or THRU). Finally, the station's 48-bit MAC address is displayed in IEEE (Canonical) form. The arrow that is drawn between the rectangles indicates the direction of token flow; therefore, stations to the right are downstream from the station to their left.

User Actions

Users can invoke some simple actions by placing the cursor inside one of the stations in the ring map and pressing any mouse button. Pressing a button causes xfddimap to draw an additional window on the screen, which displays the MAC counters from within the FDDI station. In the tty window from which the program was started, a summary of station information is displayed. This information consists of configuration station information (CSIF), operational station information (OSIF), and neighbor information (NIF). The information should be self-explanatory; however, users should read the station management (SMT) ANSI document (see the SEE ALSO section) for a more detailed explanation.

FILES

/etc/ethers	File containing MAC addresses of all FDDI stations on an FDDI ring

SEE ALSO

`fddimap`(8), `smtd`(8)

`fddi`(4) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

The ANSI documents for FDDI:

FDDI MAC (Media Access Protocol) Specification (FDDI-MAC), document number X3.139–1987, November 5, 1986

FDDI PHY (Physical Layer Protocol) Specification (FDDI-PHY), document number X3.148–1988, June 30, 1988

FDDI PMD (Physical Medium Dependent) Specification (FDDI-PMD), document number X3.166–1990, September 28, 1989

FDDI SMT (Station Management) Specification (FDDI-SMT), document number X3T9.5/84-49, Rev 6.2, May 18, 1990

Other documents related to FDDI:

RFC 1188 Proposed standard for the transmission of IP datagrams over FDDI networks. October, 1990: D. Katz

Logical Link Control Specification (802.2 LLC), document number 802.2–1985, July 16, 1984

NAME

xmppview – Displays Cray MPP system activity through a graphic user interface

SYNOPSIS

/usr/bin/xmppview [-*toolkitoption* ...]

IMPLEMENTATION

Cray MPP systems

DESCRIPTION

The xmppview command graphically displays system activity of a Cray Research massively parallel processing (MPP) system. xmppview has all the functionality of the mppview(8) command. The displays show changes in MPP system utilization in approximate real time. System administrators can use this utility on a continuous basis to monitor MPP system activity and to request specific reports. System users can check resource availability before running jobs on the MPP system and see the status of jobs already initiated.

The xmppview command accepts most of the standard X Toolkit command-line options, [-*toolkitoption* ...]. See the X man page for a complete list of available options. Examples of common options include the following:

```
-display <display>
-geometry <geometry>
-fg <color>
-bg <color>
-xrm <resource string>
```

The xmppview utility is implemented with the standard Motif widget set, so you can modify the appearance of xmppview to suit your preferences by using the X resource database. For general information about X resources, see the xrdb(1X) and X(7X) man pages. The following are the default fallback resources for xmppview:

```
*background: LightSteelBlue
*XmFrame.XmDrawingArea*background: #305C7D
*labelLineForm*background: #B7C2E5
*labelLineForm*foreground: Blue
*helpLabel.foreground: Blue
```

Before you start the xmppview utility, check the following items:

- Set your DISPLAY environment variable to your workstation. For more information, see the X(7X) man page.
- Verify that the machine on which you want to run xmppview is in the list of valid host names allowed to connect to your X server. For more information, see the xhost(1X) man page.

- Set up your `.rhosts` file on the remote Cray MPP hosts on which you plan to monitor system activity. For more information, see the `rhosts`(5) man page.

To reduce work load on the Cray Research system, you can use a SPARC workstation or server to execute `xmppview`. In order to bring up the `xmppview` window, you must have your workstation environment set up to use the X Window System (X11) and to connect to the host system you want to monitor.

A tutorial is available through the `help` pull-down menu. You can navigate this tutorial by using the scroll bar at the right side of the window. The tutorial covers the following subjects:

```
* Getting Started
  - Uses and users
  - Running xmppview
  - A quick tour
  - Error messages
* Connecting to a Host
* Viewing the 3D Animated Torus Display
  - Viewing different attributes
  - Using the fill options
  - Changing the view orientation
  - Options
* Viewing Dynamic Text Reports
* Customization
```

NOTES

Invoking `xmppview` from a mainframe or workstation launches the `mppview` command on the designated host Cray Research system with an MPP system, unless the `XMPPVIEW_HOSTCMD` environment variable is set other than to the default.

CAUTIONS

(`xmppview` running on CRAY Y-MP E, CRAY Y-MP M90, or CRAY C90 systems only)
Although the `xmppview` utility's output can be displayed on most X Window System servers, displaying it on an operator workstation (OWS) or other SPARC workstations running an OpenWindows version prior to 3.0 is not recommended, because of known adverse interaction between current Motif libraries and older versions of OpenWindows.

This problem does not exist when running `xmppview` directly from the OWS (because the SPARC `xmppview` binary is built with older Motif libraries).

ENVIRONMENT VARIABLES

When starting up, the `xmppview` utility looks for the following optional environment variables:

XMPPVIEW_INITHOST
The value of this variable should be the name of the host machine that you want xmppview to connect to upon startup of the command. By default, when running xmppview on a CRAY Y-MP E, CRAY Y-MP M90, or CRAY C90 system, xmppview will try to connect to the local host (that is, the machine on which xmppview is running).

The literal string <Local_Host> is recognized by xmppview to mean the machine on which xmppview is running.

XMPPVIEW_HOSTLIST
The value of this variable should be a list of host names, separated by blank spaces, that you want to appear first in the item list of the Host Setup dialog box. Host names in this list will be available for simple point and click operations in that dialog box.

XMPPVIEW_QUERYINT
The value of this variable should be an integer from 1 through 9. The integer value represents the number of seconds between updates. The value controls the rate at which xmppview queries for information from the Cray MPP system and updates the screen display.

If you are displaying on a slow X terminal (perhaps across a serial line connection), you may want to set this variable to a higher value. The default is 3.

XMPPVIEW_HOSTCMD
The value of this variable should be a command line for the mppview command or a similar program that the xmppview command can query for its information. The default is /usr/bin/mppview -t.

MESSAGES

To the Terminal

Error messages sent to your terminal have the following meanings:

Warning: cannot open display
This message indicates that you have attempted to execute xmppview with the display set to a host that is not running an X server. Typically, this occurs as a result of running the program on a machine with these conditions:

- no DISPLAY environment variable set, and
- no -display command-line option specified.

For information on how to correct these conditions, see the X(7X) man page.

Xlib: connection to "<workstation>:<display>.<screen>" refused by server
Xlib: Client is not authorized to connect to Server

`Warning: cannot open display`

These messages indicate that you have attempted to execute `xmppview` with the display set to an `X` server that is not set up to accept connections from the machine on which you are running `xmppview`. For information on how to add host names to the list of machines allowed to make connections to the `X` server, see the `xhost`(1X) man page.

To a Pop-up Window

The message `Error: Unable to connect to MPP`, when found in a pop-up window, is accompanied by an explanation, such as the following:

`Can't talk to samdaemon from <host>`

This message indicates that a current version of the `samdaemon` is not running on the currently selected CRAY Y-MP E, CRAY Y-MP M90, or CRAY C90 host system with the Cray MPP system. Check with your system administrator to resolve this error.

`No MPP system present`

This message indicates that `xmppview` has attempted to access a host system that is not running UNICOS MAX software.

SEE ALSO

`csam`(8), `mppview`(8), `samdaemon`(8), `xsam`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

xsam – Displays graphic data about system activity

SYNOPSIS

/usr/bin/xsam [*X11-options*] [-f *script_file*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The xsam facility is a graphics tool that monitors, in real time, the activity of a Cray Research system. It manages a set of displays that show data received from the system activity monitoring daemon, samdaemon(8). The displays are X Window System Version 11 windows; this man page assumes that you are familiar with the X Window System.

This description of xsam is only an introduction; for a more detailed description and lists of available X Window System options, see *UNICOS Resource Administration*, Cray Research publication SG–2302.

The xsam facility accepts the following options:

X11-options Regular X11 options to control X11 defaults.

-f *script_file* Name of a file that contains valid xsam commands. These commands are executed before control is passed to the calling user.

The xsam facility creates six types of display windows, as follows:

Window Type	Description
Console window	Controls all aspects of xsam. You can use it to request other xsam displays. It also contains a message section that displays all error and informational messages.
Configuration window	Shows aspects of the hardware and software configuration of the target system.
Graph window	Allows you to monitor various kernel counters over a long period of time. Possible graphs include CPU utilization, I/O statistics, and system call usage.
Map window	Displays a representation of memory. Within the map are several columns of boxes, each with its own name and each corresponding to a process in memory. The size of each box is proportional to the size of the process in main memory. As the process moves or grows, the box moves and grows. Command options allow you to display the process ID, view a subsection of memory, and view a snapshot of any process.

Snapshot window	Displays information about a specific process in text form. The data is extracted from the kernel's process structure for the target process. You can request a snapshot from a memory map or directly by using the snapshot command. The snapshot window contains several sets of statistics, including user information (such as the size and address of the process and user and process flags), scheduling parameters, and an area devoted to user identification.
Device I/O window	Displays device configuration and I/O performance on a system wide level, as well as on an individual device level. Possible device type are *disks* and *logical devices.*

SEE ALSO

`csam`(8), `sam`(8), `samdaemon`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`xtpldr` – Manages selected autoloader operations

SYNOPSIS

`xtpldr` [`-t` *trace_file*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `xtpldr` command allows you to enter tape cartridges into the domain of an autoloader, eject tape cartridges from the domain of an autoloader, and query autoloaders for volume serial number (VSN) information.

The `xtpldr` command accepts the following option:

`-t` *trace_file* Specifies the file into which trace information is written. If you experience a problem, you can try to reproduce it by reissuing the `xtpldr` command with the `-t` option specified. The resulting trace information may be helpful to your support staff in resolving the problem.

The `xtpldr` command brings up a graphical user interface with the main window having the following components:

Main Window **Description**

Menu bar Consists of three buttons:

Button	**Description**
`exit`	Exits the `xtpldr` program.
`Cray Research, Inc.` (logo)	Activates a pop-up window that displays the following `xtpldr` information: • Version number • Name of computer system on which `xtpldr` is executing • UNICOS release that the host computer is executing • Compilation date • Copyright statement
`help`	Activates a pop-up window that displays `xtpldr` help information.

Instruction/message line	Displays messages that show you completion status for the operation you have selected or instructions for the next step.
`Available autoloaders`	Shows you the autoloaders that meet the following criteria: • Are configured up (`UP`) in the tape configuration. • Are connected to the Cray Research system through a network that uses the TCP/IP communication protocol.

Selecting Autoloader and Action

Before you can perform any autoloader operations, you must select a Storage Technology Corporation (StorageTek), IBM, or EMASS autoloader and an action.

1. Select an autoloader from the `available autoloaders` list box.

   To make a selection, position the mouse pointer over one of the entries listed and press the left mouse button.

   If the list consists only of a `retry` entry, your site supports autoloaders that satisfied the criteria specified, but they are not configured `UP`. When you select `retry`, the tape daemon re-examines the tape configuration. If the state of one or more autoloaders has changed to `UP`, these autoloaders replace the `retry` entry. If the state of the autoloaders has not changed, the `retry` entry is displayed.

2. Click on one of the following action push buttons:

Push Button	Description
`enter`	Enters tape cartridges into the domain of the selected autoloader.
`eject`	Ejects tape cartridges from the domain of the selected autoloader.
`query`	Obtains information from an autoloader concerning tape cartridge VSNs.

   After you select an autoloader, the `xtpldr` window is subdivided into sets of left-handed and right-handed panes. The autoloader and action selections are part of a set of left-handed panes into which you enter information. The right-handed panes list your `enter`, `eject`, and `query` VSN selections. For each action, you can erase your selections and restart the process by clicking the `clear vsn list` button at the bottom of the right-hand pane.

   During autoloader operations, various pop-up windows provide status information. To make a pop-up window disappear, click the `dismiss` button.

3. Follow the Entering Tape Cartridges procedure that pertains to your action selection. The instructions in instruction/message line will outline the basic steps in the procedure.

Entering Tape Cartridges

After you have chosen an autoloader and selected `enter`, select the procedure from the following table that applies to your StorageTek or IBM autoloader.

Autoloader **Procedure**

StorageTek

To enter tape cartridges into the domain of a StorageTek autoloader, perform the following steps:

1. Click on the `cap info` button. Each Cartridge Access Port (CAP) holds tape cartridge storage cells and is used to enter tape cartridges into the domain of a Library Storage Modules (LSM) without opening the access door in the LSM outer wall. The door contains one or more CAPs.
2. Select a CAP from the `enter/eject stations in loader` *name* list box in the `ees information` pop-up window. For each available CAP, the list contains its address, capacity, mode, state, and status, as shown in the following example:

   ```
   0, 0,0  21 MANUAL   ON_LINE     STATUS_CAP_AVAILABLE
   ```

3. Enter the address of the selected CAP in `enter the cap id - a,l,c` input line below the CAP listing. The address is made up of the Automated Cartridge System (ACS) number, LSM number, and CAP number.

   After you enter a CAP, its address appears in the `cap info` button, and the instruction in the instruction/message line reads:

   ```
   cap selected: press ok to start enter
   ```

4. Click on the `ok` button below the `enter/eject stations in loader` *name* list box.
   - If the selected CAP is in `MANUAL` mode, the CAP is unlocked so that you can open it and put tape cartridges in it. After you close it, the autoloader robotics stores the cartridges in the LSM slots.
   - If the CAP is in `AUTOMATIC` mode, follow the instructions in the StorageTek documentation.

     The message in the instructions/message line reads:

     ```
     enter cartridges in the ees
     ```

5. Enter the tape cartridges in the enter/eject station (ees).
6. See the `vsns entered from loader` *name* list box in the `cartridge information` pop-up window for status information on each individual operation. For each cartridge, the display shows the CAP, VSN, and status, as shown in the following example:

   ```
   0, 0,0  000620  STATUS_SUCCESS
   ```

7. Click on the `vsns completed` push button when you are done entering tape cartridges.

8. Note the termination message in the instructions/message line:

```
no cartridges were in ees
```

IBM — Enter the tape cartridges in the enter/eject station (ees). The IBM autoloader server is not involved in this operation, and status information is not available.

EMASS — To enter tape cartridges into the domain of an EMASS autoloader, perform the following steps:

1. Select an archive from the `archives queried in loader` *name* list box in the `archive information` pop-up window.
2. Select a media class from the `media classes queried in loader` *name* list box in the `media class information` pop-up window.
3. Read the instructions in the instruction/message line:

```
enter: single vsns for logical import
```

   The `vsn` and `filename` buttons, which appear opaque, are not available.

4. Enter the VSNs one at a time in the `information to be entered` input area.

   After you enter a VSN, it is displayed in the right-hand pane called `vsns to be entered`.

5. Click on the `vsns completed` push button to instruct the autoloader to add the VSNs to the internal list of VSNs associated with the specified archive and media class.
6. Check the `vsns entered into loader` *name* list box in the `cartridge information` pop-up window for status on each VSN that is logically imported. For each cartridge, the display shows its VSN and status, as shown in the following example:

```
000002  volume_logically imported
```

7. Import the VSNs into the archive from the archive controlling display on the system that executes the autoloader server. For instructions, see your EMASS documentation.

Ejecting Tape Cartridges

After you have chosen an autoloader and selected `eject`, select the procedure from the following table that applies to your StorageTek or IBM autoloader.

Autoloader	Procedure
StorageTek	To eject tape cartridges from the domain of a StorageTek autoloader, perform the following steps:

1. Click on the `cap info` button.

2. Select a CAP from the `enter/eject stations in loader` *name* list box in the `ees information` pop-up window. For each available CAP, the list contains its address, capacity, mode, state, and status, as shown in the following example:

```
0, 0,0  21 MANUAL   ON_LINE     STATUS_CAP_AVAILABLE
```

3. Enter the address of the selected CAP in the `enter the cap id - a,l,c` input line below the CAP listing.

   After you select a CAP, its address appears in the `cap info` button, and the instruction in the instruction/message line reads:

```
select: select eject mode
```

4. Click on the `ok` button at the bottom of the `ees information` pop-up window.
5. Select the `single vsn` or `vsn ranges` button.
6. Select the `vsn` or `filename` button.
7. Enter the VSNs in the `information to be entered` input area using one of the following formats:
   - Single VSNs or VSN ranges for the `vsn` selection
   - Name of a file (80 characters or less) that contains VSN information in the format documented for the `-v` option of the `tpmnt`(1) command

   After you enter a VSN, it is displayed in the right-hand pane called `vsns to be ejected`.
8. Select the `vsns completed` push button.

   The autoloader robotics deposit the cartridges in the CAP you specified, and the status message in the instructions/message line reads:

```
autoloader ejecting cartridges.
```

9. Remove the cartridges from the CAP and close the CAP door.
10. See the `vsns ejected from loader` *name* list box in the `cartridge information` pop-up window for status information on each individual eject operation. After you have emptied the CAP and closed the CAP door, status information is displayed, as shown in the following example:

```
0, 0,0  000629  STATUS_SUCCESS
```

11. Note the termination message in the instructions/message line:

```
final server reply received: 1
```

The number 1 in this message indicates that your cartridges have been removed from the CAP. If your list of VSNs exceeds the CAP capacity, the autoloader refills the CAP as many times as necessary to complete your request. The number increases with each completed refill.

IBM

To eject tape cartridges from the domain of an IBM autoloader, perform the following steps:

1. Enter a station identifier by clicking one of the following push buttons:

Station Identifier	Procedure
`convenience`	Selects the convenience enter/eject station of the IBM autoloader to deposit the ejected tape cartridges. The optional IBM 3494 station handles 10 or fewer tape cartridges.
`high-capacity`	Selects the high-capacity enter/eject station of the IBM autoloader to deposit the ejected tape cartridges. The IBM 3494 station handles 40 or fewer tape cartridges. Remove the tape cartridges after you have paused the autoloader and opened the door.

2. Enter the VSNs in the `information to be entered` input area using one of the following formats:
   - Single VSNs for the `vsn` selection
   - Name of a file (80 characters or less) that contains VSN information in the format documented for the `-v` option of the `tpmnt`(1) command

   After you enter a VSN, it is displayed in the right-hand pane called `vsns to be ejected`.
3. Select the `vsns completed` push button.

   The autoloader robotics deposit the cartridges in the enter/eject station you specified.
4. See the `vsns ejected from loader` *name* list box in the `cartridge information` pop-up window for immediate status information on each individual eject operation. For each cartridge, the display shows its VSN and status, as shown in the following example:

```
003600  volume ejected
```

5. Remove the cartridges and close the door.
6. Note the termination message in the instructions/message line:

```
final server reply received: 1
```

The number 1 in this message indicates that your cartridges have been removed from the CAP. If your list of VSNs exceeds the CAP capacity, the autoloader refills the CAP as many times as necessary to complete your request. The number increases with each completed refill.

EMASS

To eject tape cartridges from the domain of an EMASS autoloader, perform the following steps:

1. Select the `vsn` or `filename` button.

   You can alternate between the `vsn` and `filename` buttons without loss of previously entered information; alternating clears the information that remains in the input area in which you enter the information.

2. Enter the VSNs in the `information to be entered` input area using one of the following formats:

   - Single VSNs for the `vsn` selection
   - Name of a file (80 characters or less) that contains VSN information in the format in the `-v` option of the `tpmnt`(1) command

   After you enter a VSN, it is displayed in the right-hand pane called `vsns to be ejected`.

3. Select the `vsns completed` push button to logically export the specified VSNs.

4. Check the `vsns ejected from loader` *name* list box in the `cartridge information` pop-up window for status information on each individual logical export operation. For each cartridge, the display shows its VSN and status, as shown in the following example:

   ```
   000003  volume_logically exported
   ```

5. Export the VSNs in the archive from the archive controlling display on the system that executes the autoloader server. For instructions, see your EMASS documentation.

Querying Autoloaders

After you have chosen an autoloader and selected `query`, select the procedure from the following table that applies to your StorageTek or IBM autoloader.

Autoloader **Procedure**

StorageTek

To obtain information from a StorageTek autoloader concerning tape cartridge VSNs, perform the following steps:

1. Select the `vsn` or `filename` button.
2. Enter the VSNs in the `information to be entered` input area using one of the following formats:

   - Single VSNs for the `vsn` selection

- Name of a file (80 characters or less) that contains VSN information in the format documented for the `-v` option of the `tpmnt`(1) command

After you enter a VSN, it is displayed in the right-hand pane called `vsns to be queried`.

3. After you have entered all the VSNs that you want to query, select the `vsns completed` push button.
4. Check the `vsns queried in loader` *name* list box in the `cartridge information` pop-up window for status information about the specified VSNs. For each cartridge, the display shows the slot address, VSN, and status, as shown in the following example:

```
0, 0,15, 9, 0  000621 volume_in_loader: home
```

IBM — The instructions for the StorageTek autoloader are identical to the instructions for the IBM autoloader. The IBM autoloader server does not return the slot address of the tape cartridges.

EMASS — The instructions for the StorageTek autoloader are identical to the instructions for the EMASS autoloader. The EMASS autoloader server does not return the slot address of the tape cartridges.

EXIT STATUS

If `xtpldr` fails, an exit status code in the range of 1 through 255 is returned; where possible, the number is normalized to the last three digits. Exit status values are documented in the *Tape Subsystem User's Guide*, Cray Research publication SG–2051.

SEE ALSO

`tpmnt`(1), `tpquery`(1)

Tape Subsystem User's Guide, Cray Research publication SG–2051

NAME

ypinit – Builds and installs the network information service (NIS) database

SYNOPSIS

```
/etc/yp/ypinit -m
/etc/yp/ypinit -s master_name
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The ypinit command sets up an NIS database on a network information service (NIS) server. You can use it to set up a master or a slave server. It asks a few, self-explanatory questions, and reports success or failure to the terminal.

The ypinit command accepts the following options:

-m Indicates that the local host is the NIS master.

-s Sets up a slave database.

master_name
Specifies the host name of the NIS server (either the master server for all of the maps or a server on which the database is current and stable).

Cray Research currently supports the NIS distribution of only the following databases (master or slave server):

- group
- passwd
- publickey
- ypservers

It sets up a master server such that the server is master to all maps in the database. Later, you can change the association of maps to masters. All databases are built from scratch, either from information available to the program at run time, or from the ASCII database files in the /etc directory. All such files should be in their traditional form, rather than the abbreviated form used on client machines.

ypinit sets up an NIS database on a slave server by copying an existing database from a running server.

See the ypfiles(5) and ypserv(8) man pages for an overview of the NIS.

SEE ALSO

`makedbm`(8), `yppush`(8), `ypserv`(8), `ypxfr`(8)

`ypfiles`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

yppasswdd – Handles password change requests from yppasswd(1)

SYNOPSIS

/etc/yppasswdd *file* [-m *arg1 arg2* ...]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The yppasswdd program is a server that handles password change requests from yppasswd(1). It changes a password entry in the specified *file*, which is assumed to be in the format of the /etc/passwd file (see passwd(5)). An entry in the *file* is changed only if the password presented by yppasswd(1) matches the encrypted password of that entry.

After *file* is modified, if the -m option is specified, a make(1) is performed in the /etc/yp directory. Any arguments that follow the option are passed to make.

The yppasswdd command accepts the following options:

file Specifies the file that contains the password entry to be changed.

-m *arg1 arg2* ... Specifies that a make function will be performed; the accompanying arguments are passed to the make command. This server is not run by default, and it cannot be started from inetd(8). If you want to enable remote password updating for network information service (NIS), place an entry for yppasswdd in the /etc/rc script (see brc(8)) of the host serving as the master for the NIS.

The yppasswdd program catches the SIGHUP signal and reregisters itself with portmap(8) when it receives the signal. This enables yppasswdd to continue running properly when portmap(8) must be restarted.

EXAMPLES

If the NIS password file is stored as /etc/yp/src/passwd, to propagate password changes immediately, invoke the server as follows:

```
/etc/yppasswdd /etc/yp/src/passwd -m -f yp.mk passwd
```

In this case, src is the NIS domain name.

FILES

`/etc/yp/yp.mk`	Make file
`/etc/passwd`	Password file

SEE ALSO

`brc`(8), `portmap`(8)

`yppasswd`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`passwd`(5), `ypfiles`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

yppoll – Finds network information service (NIS) map version at NIS server host

SYNOPSIS

/etc/yp/yppoll [-h *host*] [-d *domain*] *map*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The yppoll command asks a ypserv(8) process what the order number is, and which host is the master network information service (NIS) server for the specified *map*. If the server is a V.1 YP protocol server, yppoll uses the older protocol to communicate with it. In this case, it also uses the older diagnostic messages in case of failure.

The yppoll command accepts the following options:

-h *host* Asks the ypserv process at the specified host about the map parameters. If the host is not specified, the NIS server for the local host is used (that is, the default host is the one returned by the ypwhich(1) command).

-d *domain* Specifies a domain to be used instead of the default domain.

map Specifies the name of the map being queried.

SEE ALSO

ypserv(8)

ypfiles(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

yppush – Forces the propagation of a changed network information services (NIS) map

SYNOPSIS

/etc/yp/yppush [-d *domain*] [-v] *map*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The yppush command copies a new version of an NIS map from the master NIS server to the slave NIS servers. It is usually run only on the master NIS server by the makefile in /etc/yp/ after the master data bases are changed. It first constructs a list of NIS server hosts by reading the NIS map ypservers within the domain. Keys within the ypservers map are the ASCII names of the machines on which the NIS servers run.

A transfer map request is sent to the NIS server at each host, along with the information that the transfer agent needs (the program that actually moves the map) to call back the yppush. When the attempt has completed (successfully or not), and the transfer agent has sent yppush a status message, the results may be printed to stdout. Messages are also printed when a transfer is not possible; for instance, when the request message is undeliverable, or when the timeout period on responses has expired.

See ypfiles(5) and ypserv(8) for an overview of the NIS.

The yppush command accepts the following options:

-d *domain* Specifies a domain.

-v Specifies verbose mode. This causes a message to be printed for each response when each server is called. If you omit this option, only error messages are printed.

map Specifies the name of the map to be copied.

BUGS

In version 2 NIS, the transfer agent is ypxfr(8), which is started by the ypserv program. If yppush detects that it is speaking to a version 1 NIS protocol server, it uses the older protocol, sending a version 1 YPPROC_GET request and issuing a message to that effect. Unfortunately, you cannot know if or when the map transfer is performed for version 1 servers. yppush prints a message saying that a version 1 message has been sent. You should check to see that the transfer has actually occurred.

FILES

`/etc/yp/`*domainname*`/ypservers` NIS map

SEE ALSO

`ypserv`(8), `ypxfr`(8)

`ypfiles`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

ypserv, ypbind – Provides network information service (NIS) server and binder processes

SYNOPSIS

```
/etc/ypserv
/etc/ypbind [-v] [-h hostname [hostname] ...]
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The network information service (NIS) function provides a simple network look-up service that consists of databases and processes. The databases are dbm(3C) files in a directory tree rooted at /etc/yp, and they are described in ypfiles(5). The processes are /etc/ypserv, the NIS database look-up server, and /etc/ypbind, the NIS binder. The program interface to NIS is described in ypclnt(3C). Administrative tools are described in the yppush(8), ypxfr(8), yppoll(8), ypwhich(1), and ypset(8) man pages. Tools that are used to see the contents of NIS maps are described in the ypcat(1) and ypmatch(1) man pages. Database generation and maintenance tools are described on the makedbm(8) and ypinit(8) man pages.

Both ypserv and ypbind are daemon processes which are typically activated at system start-up time by the rc(8) script. The ypserv command runs only on NIS server machines that have a complete NIS database. The ypbind command runs on all machines that use NIS services, both NIS servers and clients.

The ypserv daemon looks up information in its local database of NIS maps. Communication to and from ypserv is accomplished by Remote Procedure Call (RPC) calls. Look-up functions are described in the ypclnt(3C) man page, and they are supplied as C-callable functions in libc. The following look-up functions are performed on a specified map within an NIS domain:

Get_all	Ships the entire map to the requester as the response to one RPC request
Get_first	Returns the first key-value pair from the map
Get_next	Enumerates all but the first key-value pair from the map
Match	Takes a key, and it returns the associated value

The Get_order_number and Get_master_name functions supply information about the map, rather than map entries. In fact, both order number and master name exist in the map as key-value pairs, but the server does not return either through the normal look-up functions. (However, if you examine the map by using the makedbm(8) command, they are visible.)

Do_you_serve_this_domain?, Transfer_map, and Reinitialize_internal_state functions are also used within the NIS subsystem, but they are not of general interest to NIS clients.

The ypbind program remembers information that lets client processes on one node communicate with a ypserv process.

This program accepts the following options:

-v Specifies that ypbind will not put itself in background mode and that it will print debugging messages to standard output. If you create the /etc/yp/ypserv.log file, debugging messages are also written to this file.

-h *hostname* [*hostname*] ...
Specifies a list of hosts that are NIS servers for the current NIS domain to attempt to bind. The default is for the Cray Research system to bind to itself, because the Cray Research system does not support broadcast mode over its networks.

The ypbind process must run on every machine that has NIS client processes; ypserv may not be running on the same node, but it must be running somewhere on the network.

The information ypbind remembers is the association of a domain name with the Internet address of the NIS server, and the port on that host at which the ypserv process is listening for service requests. The process that ypbind goes through to find and remember this information is called *binding*, and this process is driven by client requests. As a request for an unbound domain arrives, the ypbind process broadcasts on the net to find a ypserv process that serves maps within that domain. Because the binding is established by broadcasting, at least one ypserv process must be on every net. After a domain is bound by a particular ypbind, that same binding is given to every client process on the node. The ypbind process on the local node or a remote node may be queried for the binding of a particular domain by using the ypwhich(1) command.

Bindings are verified before they are distributed to a client process. If ypbind cannot speak to the ypserv process to which it is bound, it marks the domain as unbound, tells the client process that the domain is unbound, and tries to rebind the domain again. Requests received for an unbound domain fail immediately. Generally, a bound domain is marked as unbound when the node running ypserv crashes or is overloaded. In this case, ypbind binds to any NIS server available on the net (typically, one that is not as heavily loaded).

The ypbind command also accepts requests to set its binding for a particular domain. Usually, the request is generated by the NIS subsystem. The ypset(8) command is a command to access the Set_domain facility. It is only for troubleshooting.

The ypbind and ypserv programs catch the SIGHUP signal and reregister themselves with portmap when they receive the signal. This enables these commands to continue running properly when portmap must be restarted.

FILES

/etc/yp/ypserv.log	The ypserv program appends its log messages to this file; however, you must first create this file with write access for ypserv.

SEE ALSO

`portmap`(8), `yppush`(8), `ypset`(8), `ypxfr`(8)

`ypwhich`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`ypclnt`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`ypfiles`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

ypset – Points ypbind at a particular server

SYNOPSIS

/etc/yp/ypset [-V1] [-h *host*] [-d *domain*] *server*
/etc/yp/ypset [-V2] [-h *host*] [-d *domain*] *server*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The ypset command tells ypbind (see ypserv(8)) to get the network information service (NIS) for the specified *domain* from the ypserv(8) process running on *server*. If *server* is down, or is not running ypserv(8), this is not discovered until an NIS client process tries to get a binding for the domain. At this point, ypbind tests the binding set by ypset. If the binding is not valid, ypbind tries to rebind for the same domain.

The ypset command is useful for binding a client node that is not on a broadcast net, or that is on a broadcast net not running an NIS server host. It also is useful for debugging NIS client applications (for instance, where an NIS map exists only one NIS server host).

If you use ATM as your network interface, certain applications or daemons may not work as you expect if broadcast is necessary for that application or daemon. ATM is a point-to-point connection, and broadcast is not supported on ATM at this time.

When several hosts on the local net are supplying NIS services, ypbind can rebind to another host even as you are attempting to discover if the ypset operation succeeded. That is, you can type the following command lines:

```
ypset host1
ypwhich
```

The response is as follows:

```
host2
```

This response can be confusing. It is a function of the NIS subsystem's attempt to load balance among the available NIS servers, and it occurs when host1 does not respond to ypbind because it is not running ypserv(8) (or is overloaded), and host2, running ypserv(8), gets the binding.

The *server* argument indicates the NIS server to which to bind, and you can specify it as a name or an Internet Protocol (IP) address. If specified as a name, ypset attempts to use NIS services to resolve the name to an IP address. This works only if the node has a current valid binding for the domain in question. Usually, you should specify *server* as an IP address.

See the ypfiles(5) and ypserv(8) man pages for an overview of the network information service (NIS).

The ypset command accepts the following options:

-V1 Binds *server* for the V.1 YP protocol.

-V2 Binds *server* for the V.2 YP protocol. If no version is supplied, ypset first attempts to set the domain for the V.2 protocol. If this attempt fails, ypset attempts to set the domain for the V.1 protocol.

-h *host* Sets ypbind binding on *host* instead of locally. You can specify the *host* argument as a name or as an IP address.

-d *domain* Specifies *domain* instead of the default domain.

server Server name.

SEE ALSO

ypserv(8)

ypwhich(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

ypfiles(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

ypstart – Starts the network information service (NIS)

SYNOPSIS

/etc/ypstart

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The ypstart script starts the software necessary for the network information service (NIS) at system startup when executed by the netstart(8) script.

The ypstart script performs the following functions:

- Sets the NIS domain name to the name found in the /etc/config/ypdomain.txt file (see domainname(1)).
- Starts the NIS server process (ypserv(8)).
- Binds the BNIS service to the local host (ypbind (see ypserv(8)) and verifies the binding (ypwhich(1)).
- Updates the ypservers, passwd.byname, and passwd.byuid domains (ypxfr(8)).

FILES

/etc/config/ypdomain.txt Holds the NIS domain name

SEE ALSO

netstart(8), sdaemon(8), ypserv(8), ypxfr(8)

domainname(1), ypwhich(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

UNICOS Networking Facilities Administrator's Guide, Cray Research publication SG–2304

NAME

ypupdated – Updates NIS information

SYNOPSIS

/etc/ypupdated [-i] [-s]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The ypupdated command is an Remote Procedure Call (RPC)-based daemon that updates the network information service (NIS) (formerly called yellow pages (YP)), and it is started up by inetd(8). ypupdated is always registered as RPC program 100028. ypupdated consults the updaters(5) file in the /etc/yp directory to determine which NIS maps should be updated and how to change them.

By default, the daemon requires the most secure method of authentication available to it, either Data Encryption Standard (DES) (secure) or UNIX (insecure).

The ypupdated command accepts the following options:

-i Accepts also RPC calls that have the insecure AUTH_UNIX credentials. This allows programmatic updating of NIS maps in all networks.

-s Accepts only calls authenticated by using the secure RPC mechanism (AUTH_DES authentication). This disables programmatic updating of NIS maps unless the network supports these calls.

FILES

/etc/yp/updaters Map file

SEE ALSO

inetd(8), keyserv(8)

updaters(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

NAME

ypxfr, ypxfr1pdy, ypxfr1phr, ypxfr2pdy – Transfers network information service (NIS) map from NIS server

SYNOPSIS

```
/etc/yp/ypxfr [-f] [-h host] [-d domain] [-c] [-C tid prog ipadd port] mapname
/etc/yp/ypxfr1pdy
/etc/yp/ypxfr1phr
/etc/yp/ypxfr2pdy
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The ypxfr command moves a network information service (NIS) map to the local host by making use of normal NIS services. It creates a temporary map in the /etc/yp/domain directory (which must already exist), fills it by enumerating the map's entries, fetches the map parameters (master and order number), and loads them. It then deletes any old versions of the map and moves the temporary map to the real *mapname*.

If ypxfr is run interactively, it writes its output to the terminal. However, if it is invoked without a controlling terminal, and if you have created log file /etc/yp/ypxfr.log, it appends all of its output to that file. Because ypxfr is most often run from /usr/lib/crontab, or by ypserv, you can use the log file to retain a record of what was attempted, and what the results were.

For consistency between servers, you should run ypxfr periodically for every map in the NIS database. Different maps change at different rates. For example, the services.byname map may not change for months at a time; therefore, it may be checked only once a day in the early morning hours. The mail.aliases or hosts.byname file changes several times per day. In such a case, you may want to check hourly for updates.

You can use a crontab(1) file entry to perform periodic updates automatically. However, rather than having a separate crontab entry for each map, you can group commands in a shell script to update several maps. The /etc/yp/ypxfr1pdy, /etc/yp/ypxfr1phr, and /etc/yp/ypxfr2pdy scripts allow you to run one transfer per day, to transfer volatile maps hourly, and to run two transfers per day, respectively. You can modify these scripts to meet your site's requirements.

See the ypfiles(5) and ypserv(8) man pages for an overview of the network information service.

The ypxfr command accepts the following options:

-f Forces the transfer to occur even if the version at the master is not more recent than the local version.

`-h` *host* Gets the map from *host*, regardless of what the map says is the master. If you omit *host*, `ypxfr` asks the NIS service for the name of the master, and it tries to get the map from there. The *host* argument can be a name or an Internet address in the form *a.b.c.d*.

`-d` *domain* Specifies a domain other than the default domain.

`-c` Inhibits `Clear current map` request to the local `ypserv` process. Use this option if `ypserv` is not running locally at the time you are running `ypxfr`; otherwise, `ypxfr` complains that it cannot talk to the local `ypserv`, and the transfer fails.

`-C` *tid prog ipadd port*
For use by `ypserv`. When `ypserv` invokes `ypxfr`, it specifies that `ypxfr` should call back a `yppush`(8) process at the host that has Internet Protocol (IP) address *ipadd*, registered as program number *prog*, listening on port *port*, and waiting for a response to transaction *tid*.

mapname Specifies name of map to be transferred.

FILES

`/etc/yp/ypxfr.log` The `ypxfr` program appends its log messages to this file; however, you must first create this file with write access for `ypxfr`.

`/usr/lib/crontab` File from which `ypxfr` is run.

SEE ALSO

`yppush`(8), `ypserv`(8)

`crontab`(1) in the *UNICOS User Commands Reference Manual*, Cray Research publication SR–2011

`ypfiles`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014