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