NAME

`portmap` – Converts Remote Procedure Call (RPC) program numbers into DARPA protocol port numbers

SYNOPSIS

`/etc/portmap` [`-i`] [`-r`] [`-f` *filename*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `portmap` command is a server that converts Remote Procedure Call (RPC) program numbers into Defense Advanced Research Projects Administration (DARPA) protocol port numbers. It must be running to make RPCs.

When an RPC server is started, it tells `portmap` to which port number it is listening, and the RPC program numbers it is prepared to serve. When a client wants to make an RPC to a given program number, it first contacts `portmap` on the server machine to determine the port number to which RPC packets should be sent.

By default, `portmap` does not forward any non–null requests to the `MOUNTPROG` program; this would bypass security checks performed by `/etc/mountd.` This limitation can present a problem with some programs, especially old versions of `automount`. To get around this problem, `portmap` accepts the following option:

`-i` Allows nonsecure forwarding of requests to `MOUNTPROG`.

Use of the `-i` option is not supported on systems that are configured to support nonzero security labels.

The `-r` and `-f` options allow RPC servers to reregister themselves with `portmap`. This capability is useful if, for some reason, it is necessary to restart `portmap` while the RPC servers continue. If `portmap` were to die and then be restarted without either of these options, any active RPC process would lose its RPC registration. The options are as follows:

`-r` Specifies restart for standard RPC servers. This option causes `portmap` to send a `SIGHUP` signal to the following processes:

```
cnfsd
inetd
keyserv
mountd
nfsd
pcnfsd
ypbind
yppasswdd
ypserv
```

Each of these processes catches the `SIGHUP` signal and reregisters with `portmap` upon receiving the signal. In addition, any root process invoked with one of these process names is sent a `SIGHUP` signal if you invoke `portmap` with the `-r` option.

`-f` *filename*

Directs `portmap` to send a `SIGHUP` signal to each of the `process/uid` pairs found in the *filename* file in addition to the standard RPC servers that receive the signal when the `-r` option is used. (Thus, the `-f` option implies the `-r` option.)
The format of the information in *filename* is as follows:

program_1 [*uid1*]
program_2 [*uid2*]
.
.
.
program_n [*uidn*]

The *program* field is the name of the program (for example, `rpc_server`). The *uid* field is optional and specifies that `SIGHUP` be sent only to processes of the specified name running under the specified *uid*. If the *uid* field is blank, `SIGHUP` is sent to all processes of the specified name, regardless of their user ID. For example, assume that `portmap` has been invoked with the `-f` *rpcfile* option, and assume that *rpcfile* contains the following lines:

```
rpc_server      0
test_server     123
test_server     456
new_server
```

The following table indicates which processes will and will not be sent a `SIGHUP` signal:

Process name	UID	Sent `SIGHUP`?
`rpc_server`	0	Yes
`rpc_server`	123	No
`rpc_server`	456	No

Process name	UID	Sent SIGHUP?
`rpc_server`	789	No
`test_server`	0	No
`test_server`	123	Yes
`test_server`	456	Yes
`test_server`	789	No
`new_server`	0	Yes
`new_server`	123	Yes
`new_server`	456	Yes
`new_server`	789	Yes
`other_proc`	0	No
`other_proc`	123	No
`other_proc`	456	No
`other_proc`	789	No

The `-f` option is useful when site-specific RPC servers are running. However, to make this option useful, you must code the servers to catch the `SIGHUP` signal and to reregister with `portmap` when they receive it.

NOTES

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

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

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

SEE ALSO

`rpcinfo`(8)

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

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

NAME

`prctmp` – Prints login session file

SYNOPSIS

`/usr/lib/acct/prctmp` *files*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `prctmp` command prints the login session file, usually the `/usr/adm/acct/nite/ctmp` file created by the `acctcon1` command (see `acctcon`(8)). The login session file is generated as part of the daily run of standard UNIX System V accounting by using the `runacct`(8) command. The output has the following format:

```
MAJ/MIN                          CONNECT SECONDS START TIME      SESSION START
DEVICE  UID     LOGIN   PRIME   NPRIME  (NUMERIC)       DATE    TIME
```

FILES

`/usr/adm/nite/ctmp`	Login session file

SEE ALSO

`acct`(8), `acctcon`(8), `acctsh`(8), `runacct`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`prdaily` – Prints daily accounting report

SYNOPSIS

`/usr/lib/acct/prdaily` [`-c`] [`-l`] [*mmdd*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `prdaily` command prints the standard UNIX System V accounting daily report to standard output. The `runacct`(8) command invokes `prdaily` to generate a report at the end of the daily accounting run. This report is found in `/usr/adm/acct/sum/rprt`*mmdd*, where *mmdd* is the month and day of invocation.

The `prdaily` command accepts the following options and operand:

`-c`	Specifies that a report of exceptional usage, sorted by command, be generated. This option is available only for the current day's accounting data.
`-l`	Specifies that a report of exceptional usage, identified by login ID, be generated.
mmdd	Specifies that a report be generated for each combination of month and day listed. By default, `prdaily` generated only the current day's report.

NOTES

After you invoke the `monacct`(8) command, reports for dates prior to the invocation are inaccessible, because `monacct` creates a file that contains a summary of the previous daily reports and deletes the original reports.

EXAMPLES

The following example generates the daily accounting report for August 1 and 2:

```
    /usr/lib/acct/prdaily  0801 0802 >> rprt.0102
```

SEE ALSO

`acct`(8), `acctsh`(8), `monacct`(8), `runacct`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`privcmd` – Sets file privileges, security label, permissions mode, owner, owning group, and security flags

SYNOPSIS

`/etc/privcmd` [`-l` *label*] [`-h` *label*] [`-i`] [`-I` *directory*] [`-b` *alt_db*] [`-S` *grammar_objects*] [*files*]
`/etc/privcmd -R` [`-I` *directory*] [`-b` *alt_db*] [*files*]
`/etc/privcmd` [`-a` *plst*] [`-f` *plst*] [`-s` *plst*] [`-m` *label*] [`-d` *dac*] [`-p` *pentry*] *files*
`/etc/privcmd -t` *grammar_objects files*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `privcmd` command sets the security label, permissions mode, owner, owning group, and security flags of files and directories. This command also sets file privileges. `privcmd` reads the names of the files and directories, and the security attributes to be assigned, from the `/etc/privdb/main.db` database file.

If you choose to modify the default attribute values, you should specify your changes in the `/etc/config/localpriv.db` database file, which is included automatically in `/etc/privdb/main.db`. Do not modify any of the database files other than the `/etc/config/localpriv.db` file.

Your site may not want to install every file that is specified in `/etc/privdb/main.db`. `privcmd` checks the system configuration options in `/etc/config/config.mh` to determine which files are appropriate for your system's configuration.

You can specify file and directory names on the command line. `privcmd` assigns the attributes that are defined in `/etc/privdb/main.db` for the specified files and directories.

If no file or directory names are specified on the command line, attributes are assigned to all files and directories as defined in `/etc/privdb/main.db`.

You can also specify both names and attributes on the command line. In this case, the names and attributes defined in `/etc/privdb/main.db` are ignored.

The `privcmd` command accepts the following options:

`-l` Assigns the specified security label to the files and directories that would otherwise be assigned the `syslow` security label. If this option is used with the `-S` option, `MAC` must be one of the members of the argument list passed to the `-S` option.

`-h` Assigns the specified security label to the files and directories that would otherwise be assigned the `syshigh` security label. If this option is used with the `-S` option, `MAC` must be one of the members of the argument list passed to the `-S` option.

`-i` Compares the attributes that are actually assigned to files and directories against the attributes that are specified in the database entries for those files and directories. Output is produced using a `+-` format. A `+` means that the file or directory is assigned an attribute that is not specified in its database entry. A `-` means that the object is missing an attribute that is specified in its database entry.

`-I` Changes the algorithm for searching for `#include` files whose names do not begin with `/` to look in *directory* before looking in the directories on the standard list. `#include` files whose names are enclosed in double quotation marks in the `include` directive are searched for first in the directory of the input file. All `#include` files (those whose names are enclosed in `< >` or `" "`) are then searched for in directories named in `-I` options, and, finally, in the standard directories `/etc/config` and `/etc/privdb`.

Each additional directory is specified by a separate `-I` option. If multiple `-I` options are specified, the directories are searched in the order specified on the command line.

`-b` Forces `privcmd` to use the database specified by *alt_db* instead of `/etc/privdb/main.db`. If the `-b` option is not specified, the `/etc/privdb/main.db` file is used as the default.

`-S` Specifies the selection criteria for the database grammar objects to be used when setting or doing an integrity check. If this option is not specified, the default value is `ALL`. The list of valid grammar objects is defined by *grammar_objects*.

`-R` Clears all privileges from *files*. When the `-R` option is specified, no mandatory access control, discretionary access control, or flag attributes from the database file are applied to any files.

`-a` Sets the file's allowed privileges to the specified value. If this option is not specified, the old allowed privilege set associated with *files* does not change. To delete the set of allowed privileges, assign it `PRIV_NULL`.

`-f` Sets the file's forced privileges to the specified value. If this option is not specified, the old forced privilege set associated with *files* does not change. To delete the set of forced privileges, assign it `PRIV_NULL`.

`-s` Sets the file's set_effective privileges to the specified value. If this option is not specified, the old set_effective privilege set associated with *files* does not change. To delete the set of set_effective privileges, assign it `PRIV_NULL`.

`-m` Sets the file's security label to the specified value.

`-d` Sets the file permissions mode, owner, and owning group to the specified value.

`-p` Updates the file privilege assignment list (PAL) with the specified PAL category record. Multiple PAL category records can be set by specifying the `-p` option multiple times.

`-t` Displays the output in the format of a `privcmd` database entry. You can use this output to generate `privcmd` database entries for specific files.

The `privcmd` command accepts the following operands:

files	The name(s) of the file(s) to which attributes are assigned. Multiple file names must be separated by white space.
alt_db	The name of the database file to be used instead of `/etc/privdb/main.db`.
directory	The name of the directory in which to search for the database files prior to searching the `/etc/config` and `/etc/privdb` directories.
label	A character string that represents the security label to be assigned. It has the following format: *level_name*[,*compartment_name*[,*compartment_name*[...]]] *level_name* is a character sequence that represents the name of a security level (for example `level2).` *compartment_name* is a character sequence that represents the name of a compartment (for example, `comp39`). If no compartments are specified, or the text string `none` or 0 is used, then the compartment bit mask is set to 0. If one or more compartment names are specified, components of the specified security label must be separated by commas with no intervening white space.
plst	A character string that represents the privileges to be assigned. It has the following format: *privilege_name*[,*privilege_name*[,*privilege_name*[...]]] *privilege_name* is a character sequence that represents the name of a privilege (for example `PRIV_MAC_READ`). If one or more privilege names are specified, they must be separated by commas with no intervening white space. The `PRIV_ALL` character string represents the list of all privileges. Privileges can be cleared by specifying `PRIV_NULL`.
dac	A character string that represents the permissions mode, owner, and owning group to be assigned. It has the following format: *permissions_mode*,*owner_name*,*group_name* *permissions_mode* must be specified in an octal value. *owner_name* is a character string that represents the name of the file owner. *group_name* is a character sequence that represents the name of the file group.

pentry A character string that represents the PAL category record to be assigned. This entry has the following format:

category_name`[:[`*plst*`][:[`*ptext*`]]]`

category_name is a character string that represents the name of category (for example, `secadm`). *plst* is a character sequence that represents one or more privileges to be associated with this PAL category record. See the previous description of *plst* for more information. *ptext* is a character sequence that represents the privilege text that is to be associated with this PAL category record. A maximum of 8 alpha-numeric characters can be specified for this field.

To delete a PAL category record, specify the following:

category_name`::`

The following syntax clears the privileges and privilege text of a PAL category record, but it does not delete the record from the PAL:

category_name`:PRIV_NULL:TEXT_NULL`

grammar_objects
A character string that represents the desired grammar objects to be used when generating, setting, or verifying a database record. If one or more *grammar_objects* are specified, they must be separated by commas, with no intervening white space. The following are acceptable values for *grammar_objects*:

Value	**Description**
`MAC`	Processes mandatory access control (MAC) security information.
`DAC`	Processes discretionary access control (DAC) security information.
`FLAGS`	Processes security flags information.
`PAL`	Processes PAL-related information.
`ALL`	Processes all of the `MAC`, `DAC`, `FLAGS`, and `PAL`.

NOTES

When using the `privcmd` command, the `syshigh` and `syslow` security labels are not applied to the specified objects unless the `SECURE_MAC` configuration option is enabled.

When `privcmd` is executed, file attributes for any given file are set in the following specific order:

1. Flag attributes
2. DAC attributes
3. MAC attributes
4. Privileges and PALs

If a failure occurs when trying to set MAC attributes, DAC attributes, privileges, or PALs, `privcmd` returns an exit status of 1, and all file attributes that have already been set remain set. If a failure occurs when trying to set flag attributes, an error message is issued and privcmd continues.

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

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

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

You can specify file record fields in any order. Also, you can specify multiple categories within a PAL field. For example, if you want to assign the `secadm` and `system` categories the exact same privileges and privilege text, use a comma-separated list of categories as follows:

```
PAL=[
     secadm,system:PRIV_DAC_OVERRIDE,...,PRIV_RESOURCE:exec;

    ];
```

The previous example is equivalent to specifying two separate entries with the same exact privileges and privilege text as shown in the following example:

```
PAL=[
     secadm:PRIV_DAC_OVERRIDE,...,PRIV_RESOURCE:exec;
     system:PRIV_DAC_OVERRIDE,...,PRIV_RESOURCE:exec;
    ];
```

EXIT STATUS

The `privcmd` command exits with one of the following values:

0	`privcmd` was successful.
1	Failed to set file security information.
2	Integrity checking found discrepancy.

EXAMPLES

Example 1: The following example applies the `syslow` label on `file1` and `file2`:

```
$ /etc/privcmd -m syslow file1 file2
```

Example 2: The following example applies a security label that consists of level 0 and the `comp24` and `comp39` compartments on all files in the current directory:

```
$ /etc/privcmd -m level0,comp24,comp39 *
```

Example 3: The following example removes the PALs associated with `/bin/cat` and `/bin/chmod`, and sets the security label for both to level 0:

```
$ /etc/privcmd -R -l level0 /bin/cat /bin/chmod
```

Example 4: The following example removes all PALs found in `/etc/privdb/main.db`. Relabels every file entry in the database that currently has a `syslow` label with `level0`, and relabels every file entry that currently has a `syshigh` label with `level3`:

```
$ /etc/privcmd -R -l level0 -h level3
```

Example 5: For the following example, assume that `mydbfile` contains the following record:

```
file= { name = /bin/xyz;
                 allowed = [PRIV_NULL];
                 forced = [PRIV_DAC_OVERRIDE,
                               PRIV_MAC_READ,PRIV_MAC_WRITE];
                 set_effective = [PRIV_DAC_OVERRIDE,
                               PRIV_MAC_READ,PRIV_MAC_WRITE];
                 MAC = [syslow,none];
                 DAC = [0755 bin bin];
                 FLAGS = [priv_root,exec];
                 PAL = [
                        other :PRIV_NULL:TEXT_NULL;
                        secadm:PRIV_DAC_OVERRIDE,PRIV_FOWNER,
                               PRIV_MAC_READ,PRIV_MAC_WRITE:
                               TEXT_NULL;
                        sysadm:PRIV_DAC_OVERRIDE:showall;
                        system : PRIV_DAC_OVERRIDE,
                                 PRIV_MAC_READ,
                                 PRIV_MAC_WRITE:TEXT_NULL;
                    ];
      };
```

Executing the following command labels the `/bin/xyz` file with the allowed, forced, and set_effective privileges, and creates a PAL with a `secadm`, `sysadm`, `system`, and `other` entries. The `MAC`, `DAC`, and `FLAGS` values are also applied; the `MAC` values are applied only if the `SECURE_MAC` parameter is enabled:

```
$ /etc/privcmd -b mydbfile
```

The two following methods can be used for changing record entries from within the database:

- Create a duplicate record entry and change the attributes.
- Use a special record called an update record to redefine categories, create new categories, and/or assign privilege text.

The second method is the recommended way for changing record entries. The default privilege definition and assignment involved extensive code analysis by Cray Research. For this reason, it is not recommended that you change individual privileged for a category. Rather, if you want to create and define new categories, they should be defined in terms of an already-defined category. This ensures that the new category is assigned the right privileges.

Example 6: The following example shows the recommended method outlined previously.

Assume that the following special update record has been appended to `mydbfile`:

```
file.update = { name = /bin/xyz;
                     PAL = [
                             secadm:secadm:sysadm;
                             sysops:system:TEXT_NULL;
                             sysadm::;
                     ];
           };
```

Adding this record keeps the `secadm` privileges intact and changes the privilege text for `secadm` from `TEXT_NULL` to `showall`. It also defines a new entry for the `sysops` category, which has the same privileges as the `system` entry, but with `TEXT_NULL` for the privilege text. The last entry deletes the `sysadm` category. The sequence in which these operations are performed is important, because the definition of categories is modified and any further reference to that category name receives the updated or modified definition.

Example 7: The following example shows how to change individual privileges and categories "on the fly." Making changes in this way does not change the databases. In this example, the `PRIV_FOWNER` privilege is deleted and the `showall` privilege text is assigned for the `secadm` category:

```
$ /etc/privcmd -b mydbfile
$ /etc/getpal /bin/xyz
other:PRIV_NULL:TEXT_NULL
secadm:PRIV_FOWNER,PRIV_DAC_OVERRIDE,PRIV_MAC_READ,PRIV_MAC_WRITE:TEXT_NULL
sysadm:PRIV_DAC_OVERRIDE:showall
system:PRIV_DAC_OVERRIDE,PRIV_MAC_READ,PRIV_MAC_WRITE:TEXT_NULL
$
$ /etc/privcmd -p secadm:PRIV_DAC_OVERRIDE,PRIV_MAC_READ,\
> PRIV_MAC_WRITE:showall /bin/xyz
```

Note that there are no intervening spaces between continuation marks and the new line.

The following example shows the resulting PAL entry from the previous example:

```
$ /etc/getpal /bin/xyz
other:PRIV_NULL:TEXT_NULL
secadm:PRIV_DAC_OVERRIDE,PRIV_MAC_READ,PRIV_MAC_WRITE:showall
sysadm:PRIV_DAC_OVERRIDE:showall
system:PRIV_DAC_OVERRIDE,PRIV_MAC_READ,PRIV_MAC_WRITE:TEXT_NULL
```

Another way to compare PAL differences is to use the `privcmd -i` option, as shown in the following example. Use of this option produces verification and validation results between the `/bin/xyz` database entry and the actual object's PAL entry:

```
$ /etc/privcmd -i -b mydbfile
/bin/xyz:secadm:- PRIV_FOWNER
/bin/xyz:secadm: showall/TEXT_NULL
```

The `-` sign means the `secadm` PAL entry has `PRIV_FOWNER` while the `/bin/xyz` file does not have it. Also, `/bin/xyz` has a `showall` privilege text while the database entry shows `TEXT_NULL` as privilege text for that category.

Example 8: The following example shows how to generate a database entry with the modified `secadm` category from the previous example. Use the `-t` option to generate a formatted database record:

```
$ /etc/privcmd -t PAL,MAC /bin/xyz >> /etc/config/localpriv.db
```

The generated record is appended to your `localpriv.db` database. This new entry supersedes any previously existing entries with the same file name.

Example 9: The following example shows a PAL entry with the same privileges specified for two categories and an additional PAL entry for a different category:

```
file = { name = /bin/dog;
                 forced = [PRIV_DAC_OVERRIDE,
                                PRIV_MAC_READ,PRIV_MAC_WRITE];
                 set_effective = [PRIV_DAC_OVERRIDE,
                                PRIV_MAC_READ,PRIV_MAC_WRITE];
                 allowed = [PRIV_NULL];
                 DAC = [ 755 bin bin ];
                 PAL = [
                        other:PRIV_NULL:TEXT_NULL;
                        secadm,system:PRIV_DAC_OVERRIDE,
                                PRIV_MAC_READ,PRIV_MAC_WRITE:TEXT_NULL;
                        ];
                 PAL = [
                        sysadm:PRIV_MAC_READ:mytext;
                        ];
                 MAC = [syslow,none];
```

The following shows the output from the executing the `getprivs` command after this PAL has been set:

```
$ getprivs /bin/dog
a:PRIV_NULL
f:PRIV_DAC_OVERRIDE,PRIV_MAC_READ,PRIV_MAC_WRITE
s:PRIV_DAC_OVERRIDE,PRIV_MAC_READ,PRIV_MAC_WRITE
```

The following shows the output the `getpal` command after this PAL has been set:

```
$ getpal bin/dog
other:PRIV_NULL:TEXT_NULL
secadm:PRIV_DAC_OVERRIDE,PRIV_MAC_READ,PRIV_MAC_WRITE:TEXT_NULL
system:PRIV_DAC_OVERRIDE,PRIV_MAC_READ,PRIV_MAC_WRITE:TEXT_NULL
sysadm:PRIV_MAC_READ:mytext
```

FILES

`/etc/privdb/main.db`	Database that contains the default file and directory names and the security attributes to be assigned by `privcmd`.
`/etc/config/localpriv.db`	File that contains the site-modified security attributes to be assigned by `privcmd`.

SEE ALSO

`getpal`(8)

General UNICOS System Administration, Cray Research publication SG–2301

NAME

`prtacct` – Prints total accounting file

SYNOPSIS

`/usr/lib/acct/prtacct` *file* [`"`*heading*`"`]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `prtacct` shell script formats and prints any total accounting (`tacct.h` format) file generated by the standard UNIX System V accounting package by invoking the `acctmerg`(8) command. `prtacct` requires that you specify the name of the `tacct` file.

The `prtacct` script accepts the following operands:

file	Name of the `tacct` file. This operand is required.
`"`*heading*`"`	A string enclosed in double quotation marks that defines the page header.

EXAMPLES

The following example prints the total accounting file `daytacct` with a defined header:

```
/usr/lib/acct/prtacct daytacct "DAILY USAGE REPORT FOR TODAY’S DATE"
```

SEE ALSO

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

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`pwck`, `grpck` – Checks password and group files for consistency

SYNOPSIS

`/etc/pwck` [*file*]
`/etc/grpck` [*file*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `pwck` command scans the password file and notes any inconsistencies. The checks include validation of the number of fields, login name, user ID, group ID, and whether the login directory and optional program name exist. The default password file is `/etc/passwd`.

The `grpck` command verifies all entries in the group file. This verification includes a check of the number of fields, group name, group ID, and whether all login names appear in the password file. The default group file is `/etc/group`.

NOTES

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

Active Category	**Action**
`system`, `secadm`	Allowed to specify any file.
`sysadm`	Allowed to specify any file, subject to security label restrictions. Shell redirected output is subject to security label restrictions.

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

Although these commands exist on a UNICOS system, `/etc/udbgen` (see `udbgen`(8)) assures the synchronization of these files.

MESSAGES

Group entries in `/etc/group` without login names are flagged.

FILES

`/etc/group`

```
/etc/passwd
```

SEE ALSO

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

NAME

`quadmin` – Administers file quotas

SYNOPSIS

`/etc/quadmin -c` *request* [`-D`] [`-p` *fstab_path*] [`-s` *file_systems*]
`/etc/quadmin -c` *request* [`-D`] [`-d` *directive*] [`-p` *fstab_path*]
`/etc/quadmin -c` *request* [`-D`] [`-p` *fstab_path*] [*srcfile*]
`/etc/quadmin -m` [`-D`] [`-d` *directive*] [`-F`] [`-p` *fstab_path*] [`-s` *file_systems*] [`-V` *version*]
`/etc/quadmin -m` [`-D`] [`-F`] [`-p` *fstab_path*] [`-s` *file_systems*] [`-V` *version*] [*srcfile*]
`/etc/quadmin -Q` [`-D`]
`/etc/quadmin -v` [`-a` *request*] [`-D`] [`-p` *fstab_path*] [`-s` *file_systems*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `quadmin` command provides an interface to the file quota system that allows you to control the quota system (using the `-c` option), to modify the information used by the quota system (using the `-m` option) and to view status information (using the `-v` option). Any use of the command, except when the `-Q` or `-v` options are selected, requires that the file system names be specified with the `-d` or `-s` option or from a source file (*srcfile*) or `stdin`.

Either the `-c`, `-m`, `-Q`, or `-v` option is required. The `quadmin` command accepts the following options and operands:

`-a` *request* Displays quota association information. This option may appear only with the `-v` or `-D` options. It is used to see how the current quota file systems are configured.

`-c` *request* Performs *request* on the specified file systems. This option cannot be used with the `-m`, `-Q`, or `-v` option. Valid *requests* are `count`, `default`, `inform`, and `enforce`. The requests match the `quotactl`(2) requests to manage the enforcement level of the quota system. If a *srcfile* has been specified, only `filesystem` and `set` directives are meaningful, and other directives are silently ignored. This is done to allow the same source file to be used with all `quadmin` modes of operation needing a source file. If the `-s` option is selected, only those file systems named will have a control action; otherwise, all file systems named in the source are affected.

`-d` *directive* Supplies directive input on the command line. Valid directives are described later in the entry. This option presents the command with a *directive* as though it had been read from a file. The normal source rules are followed, allowing multiple directives separated by semicolons and escaped new lines. The only way to change a quota is through an explicit directive; omitting an explicit directive will **not** cause a quota to be reset to the default. This option must not be specified with the `-Q` or `-v` options or *srcfile*.

`-D`	Sets debug mode. This option is useful only for testing.
`-F`	Forces quota file access. This option bypasses any use of the `quotactl`(2) interface and directly accesses the quota control files. It is intended for use only when the quota system is inactive; otherwise, information in the files could be altered by the kernel at any time. An example of when the `-F` option is used is when `quadmin` is used to create a quota control file from `qudu`(8) output. This option cannot be used with the `-c`, `-Q`, or `-v` options.
`-m`	Changes quota information if it exists; otherwise, creates new quota control files. This option must not be specified with the `-c`, `-Q`, or `-v` options.
`-p` *fstab_path*	Reads */fstab_path*/*fstab* and/or the kernel mount table to find the configuration information.
`-Q`	Displays the default name of the quota file on `stdout`. Scripts that manipulate the quota files have need of this feature. Only the `-D` option can be used with this option.
`-s` *file_systems*	Defines the specified file systems. The *file_systems* argument is a list of one or more file system names separated by commas. When this option is specified, `quadmin` bypasses reading the kernel mount table; the specified file systems are the only ones known to the command. The root directory names rather than device names should be used with this option. If this option is used with the `-c` option, only those file systems listed have a control action, and no source file is read. If this option is used with the `-v` option, the quota file name associated with the named file system or systems is written to the `stdout` file. If other file systems share the same quota file, the names of those are appended to the same line of output. The file system that appears in `/etc/fstab` with the quota option defining the control file is always first, following the quota control file name. To get a display of all quota file systems, use the option `-s all`.
`-v`	Views status of quota control information. This option must not appear with the `-c`, `-m`, or `-Q` options. The output is written to the `stdout` file.
`-V` *version*	Controls conversion from previous source language forms. Only the values 5 and 6 are allowed for *version*. The value 5 signifies conversion from the first release (UNICOS 5.1) format; the value 6 signifies that the source is in UNICOS 6.0 format (this is the default). This option takes precedence over a version directive in the source file. The `-V` option is allowed only with the `-m` option.

Directives

Directives that can be specified in the input file for `quadmin` or with the `-d` option have the following functions:

- They establish the relationship between file systems and quota control files.
- They contain information to place in the quota files.

The following conventions are used in the descriptions of directives:

- Optional fields are indicated by square brackets (`[ ]`).
- *Italics* indicates data that you supply.
- The vertical bar (`|`) character indicates that you should enter one of the items separated by the symbol or symbols and enclosed in curly braces (`{ }`).
- An unescaped end of line, semicolon, or null ends a directive.

The directives are processed sequentially in the order in which they appear within the file. Words in `typewriter font` can be shortened to their minimum unique length as shown in the Abbreviations subsection.

The following directives are valid:

`#` Indicates a comment. All text from this character to the end of the current line is ignored.

`version` *ver*

Specifies which version of the software was used to generate directive files. The `quota`(1) and `qudu`(8) commands insert this directive when generating `quadmin`-style output. This directive must appear before anything version-specific is encountered in the file; it is effective for all subsequent lines of directive input. (Quota commands prior to UNICOS 6.0 did not generate or recognize this directive, so if files generated on release 5.1 are to be used, this directive should be placed at the beginning of each file with *ver* set to 5.) *ver* may take the value 5 or 6. If *ver* is 5, conversion of warning windows to warning values will be done in order to compensate for the warning mechanism change between UNICOS 5.1 and 6.0. The value 6 means that the directives are in 6.0 form, so no conversion needs to be done (this is the default).

`set` *vname string*

Associates the variable name *vname* with the string *string*. Only alphanumeric characters are allowed in *vname*. As with shell variables, the string may be substituted into other directives by using the notation $*vname* or ${*vname*}. The string begins at the first nonwhite-space character following *vname* and ends at the end of the line with trailing white space removed. (A comment (#) or semicolon is treated as an end of line for the purposes of editing trailing white space.) A `set` directive naming an already specified *vname* replaces the old definition with the new. The variable name `QFILE` is predefined to be the default name of the quota file. Shell variable names are also allowed but they will be accessed only if the name is not predefined and has not been defined through `set`.

`filesystem` *fsname* [*qfname*]
: Specifies a file system, *fsname*, and associates it with a quota control file, *qfname*. *qfname* should be omitted if the quota configuration is defined in `/etc/fstab` as recommended. If this is not the case, *qfname* may be `.` or omitted if the quota file is to have the default name and is to reside in the root directory of *fsname*; otherwise, the full name is required. Once established, this relationship is permanent within a directive file. This directive declares the file system and its association with a quota control file, but `open` directives must be used to make each file system eligible for use within the directive file. Root directory names rather than device names are recommended for *fsname*. A new *fsname* automatically becomes a member of the current list of open file systems.

`open` *fsname* [*fsname* . . .]
: Makes the listed file system or systems, *fsname*, eligible for subsequent `account`, `acid`, `default`, `gid`, `group`, `uid`, and `user` directives. The special *fsname* `*` may be used to refer to all file systems known through previous `filesystem` directives. Each `open` redefines the list of eligible file systems.

The `default` directives below are used to set *generic* file system parameters (those not involved with specific ID classes).

`default algorithm` {`exponential` | `linear` | `none` | `site1` | `site2`}
: Three soft-quota control algorithms are provided with the released software and two additional algorithms can be supplied by the installation. Algorithm selection is specified in each quota control file and is `none` by default. The algorithm named `none` is compatible with UNICOS 5.1.

`default {ef1 | ef2}` [+ | -]*number*
: These are algorithm-specific fields declared as `long` in the header structure. Signed values are acceptable. The algorithm-specific fields depend on the algorithm selected by the `default algorithm` directive.

`default level` {`count` | `enforce` | `inform` }
: Specifies the quota enforcement level that will be imposed by `mount`(8) or `quadmin` with the `-c default` option. The default enforcement level is `count`.

`default time` *number*[`s | m | h | d`]
: This feature is not currently implemented, although the field is present. The time field is used to specify a minimum migration preference threshold. The default unit is seconds, but a suffix selected from the list `s` (seconds), `m` (minutes), `h` (hours) or `d` (days) may be used. The suffix must immediately follow the final digit of the number. A value of 0, the default, disables this feature.

`default style [online | aggregate]`

Specifies whether quota counts are maintained only for files that are physically on disk (online), or whether files migrated offline by the Data Migration Facility (DMF) are included in the count (aggregate).

The default is online quotas. In this case, when a file is migrated offline by DMF, the disk blocks released are subtracted from the user, group, and account totals. This allows a user, group, or account to have more "virtual" disk blocks then allowed for by the quota limit. The quota limit only applies to files actually on the disk.

If aggregate quotas are selected, the space for both online and migrated (offline) files are counted as part of the user, group, or account usage. In this case, the quota limits apply to the sum of the online disk blocks and the offline disk blocks. Aggregate quotas would be selected in order to limit the amount of space used in the offline storage facility, e.g., a tape silo.

The remaining `default` directives are ID-class specific.

`default` {`account` | `group` | `user`} `flags` *flags*

Binds the enable *flags* in the quota file header associated with the specified `account`, `group`, or `user` to all currently eligible *fsname*s. If a flag is off, quotas are not controlled for that class and type. The following values can be used for *flags*:

`f`	File quotas only.
`i`	Inode quotas only.
`fi`	Both file and inode quotas. This value is the default.
`off`	Quota checking disabled by default for the named ID class.

The binding is permanent within a directive file.

`default` {`account` | `group` | `user`} `file quota` *integer*

Binds the default `account`, `group`, or `user` file quota to the number of Cray file blocks (4096 bytes) specified by *integer* to all currently eligible *fsname*s. The value specified as the default file quota for the respective account, group, or user in the `sys/quota.h` file is used if this directive is not present. The binding is permanent within a directive file.

`default` {`account` | `group` | `user`} `inode quota` *integer*

Binds the default `account`, `group`, or `user` inode quota to the number of inodes specified by *integer* to all currently eligible *fsname*s. The value specified as the default inode quota for the respective account, group, or user in the `sys/quota.h` file is used if this directive is not present. The binding is permanent within a directive file.

`default` {`account` | `group` | `user`} `file warning` *float*

Binds the default `account`, `group`, or `user` file quota warning to the value specified by *float* to all currently eligible *fsname*s. If the value is in the range 0.0 < *float* < 1.0, it is the fraction of the file quota at which a warning occurs. If the value is > 1, it is an absolute warning value. If the value is 0 or 1, the warning is disabled, and no warning is issued. The value specified as the default file warning for the respective account, group, or user in the `sys/quota.h` file is used if this directive is not present. The binding is permanent within a directive file.

`default` {`account` | `group` | `user`} `inode warning` *float*

Binds the default `account`, `group`, or `user` inode quota warning to the value specified by *float* to all currently eligible *fsname*s. If the value is in the range 0.0 < *float* < 1.0, it is the fraction of the inode quota at which a warning occurs. If the value is > 1, it is an absolute warning value. If the value is 0 or 1, the warning is disabled, and no warning is issued. The value specified as the default inode warning for the respective account, group, or user in the `sys/quota.h` file is used if this directive is not present. The binding is permanent within a directive file.

The directives listed previously set the defaults for the specified file system or quota file. You may specify the defaults individually for each file system or apply the same defaults to as many file systems as were eligible when the default directives were encountered.

`remove` {`acid` | `account` | `gid` | `group` | `uid` | `user` | `all`} `usage`

This directive will set file and inode usage values for all quota entries in the named category on all open file systems to 0. This directive is acted upon when it is encountered in the input file. You should use this whenever an entirely new set of usage values are to be set and no memory of previous usage is wanted. `qudu`(8) uses this directive when it writes a file of current usage information.

The remaining directives establish quota values for individual or groups of IDs. All file systems already opened with an `open` directive are affected. The special name `*` means all users, account IDs, user IDs, or group IDs.

`enable` {`uid` | `acid` | `gid`} [*range*]

Evaluates the specified ID with respect to the inclusive *range* and, if true, enables the directive (this is used only with `*`). Valid expressions are as follows: `n-nn`, `n-`, `-nn`, and `n`. If two values are separated by a dash (`-`), the left side must be less than or equal to the right side. All values are unsigned integers or named variables that resolve into unsigned integers or recognized range constructs. To select values 50 and above, you would use the expression 50-; for values 1023 or less, you would use -1023; values from 100 through 2049 would be selected with the expression 100-2049; a single value of 35 would be selected with 35 or 35-35. You can specify multiple ranges by separating one range from another with a comma. To enable ranges 10 through 50 and 1000 through 1099, you would enter 10-50, 1000-1099. When multiple ranges are specified, the first satisfied range reading from left to right makes the condition true. White space is ignored in the range description. Each `enable` directive redefines the range for the specified ID class and removes the previous range description. The default range is 0-. To restore the default range, leave the range field empty.

The directives used to set values in individual or groups of quota file entries are made up of the components in the following list. The EXAMPLES section shows how the parts may be combined.

When creating directives, you may shorten them by using *implicit value typing*, which is built into the input process. As the line is scanned from left to right, previous classifications are remembered and used if needed. Therefore, instead of typing `default account file quota 123 file warning .8 inode quota 12 inode warning .95`, you could type `default account file quota 123 warning .8 inode quota 12 warning .95`. This applies to each line individually and is equally valid with the following directives.

The classification part of the directive must be first in the statement and consist of one of the following:

`account` *aname*
: Sets the specified values in account *aname*. If *aname* is `*`, all accounts in the user database (UDB) whose `acid`s satisfy the conditions established through the `enable acid` directive are set.

`acid` *id*
: Sets the specified values in account *id*. If *id* is `*`, all accounts in the UDB whose `acid`s satisfy the conditions established through the `enable acid` directive are set.

`group` *gname*
: Sets the specified values in group *gname*. If *gname* is `*`, all accounts in the UDB whose `gid`s satisfy the conditions established through the `enable gid` directive are set.

`gid` *id*
: Sets the specified values in group *id*. If *id* is `*`, all accounts in the UDB whose `gid`s satisfy the conditions established through the `enable gid` directive are set.

`user` *uname*
: Sets the specified values for user *uname*. If *uname* is `*`, all accounts in the UDB whose `uid`s satisfy the conditions established through the `enable uid` directive are set.

`uid` *id*
: Sets the specified values for user *id*. If *id* is `*`, all accounts in the UDB whose `uid`s satisfy the conditions established through the `enable uid` directive are set.

The next part of the statement must consist of one or more of the following value specification parts (these may be combined as necessary to establish all of the information; implicit value typing is allowed).

`file quota` {*integer* | `default` | `unlimited` | `infinite`}
: Sets the file quota to a specific *integer* value, `default`, or `unlimited`. `infinite` is a synonym for `unlimited`. The *integer* value specifies the number of 4096-byte blocks, by default. The value 0 means that file allocation is prevented on the file system.

`file usage` *integer*
: Sets the file usage to a specific value, *integer*. Typically, this would be used with only one file system open.

`file warning` {*float* | `default`}
: Sets the file warning to the value specified by *float* or `default`. If the value is in the range 0.0 < *float* < 1.0, it is the fraction of the file quota at which a warning occurs. If the value is > 1, it is an absolute warning value. If the value is 0 or 1, the warning is disabled, and no warning is issued. If the value of *float* is between 0 and 1, the warning value is computed by multiplying the inode quota by *float*.

`inode quota` {*integer* | `default` | `unlimited` | `infinite`}
: Sets the inode quota to a specific *integer* value, `default`, or `unlimited`. `infinite` is a synonym for `unlimited`. The value 0 means that new inode allocation is prevented on the file system.

`inode usage` *integer*
: Sets the inode usage to a specific *integer* value. Typically, this would be used with only one file system open.

`inode warning` {*float* | `default`}
: Sets the inode warning to the value specified by the floating value, *float*, or `default`. If the value is in the range 0.0 < *float* < 1.0, it is the fraction of the inode quota at which a warning occurs. If the value is > 1, it is an absolute warning value. If the value is 0 or 1, the warning is disabled, and no warning is issued. If the value of *float* is between 0 and 1, the warning value is computed by multiplying the inode quota by *float*.

The fields described below apply to the ID class and are not related to file or inode quotas. These fields can be placed between any of the previous specifications but will be more easily read if they are placed before file and inode specifications are written.

`time` [`+ | -`]*number*[`s | m | h | d`]
: This sets the time when the warning threshold was exceeded. Typically, this field need not be specified, but it is included for testing or other special needs. Time may be specified in either absolute or relative (to the time the directive is processed) form. Absolute form expects time in internal notation (seconds since January 1, 1970), so it is not very easy to use. The relative form is a signed value, while the absolute form is unsigned. The default unit for both forms is seconds, but a suffix selected from the list `s` (seconds), `m` (minutes), `h` (hours), or `d` (days) may be used. If the relative form is used, minus (-) means the past and plus (+) means the future. If present, the sign must immediately precede and the suffix must immediately follow the number. Set the current time using the relative notation +0. The value of this field is important with respect to the algorithm in use. See the algorithm descriptions for more information.

{`ef1 | ef2 | ef3 | ef4`} *number*
: These four fields are 32-bit unsigned integers whose meaning is algorithm-dependent; the meaning depends on the algorithm selected by the `default algorithm` directive.

`ef5` [`+ | -`]*number*
: This is a signed `long` field whose meaning is algorithm-dependent; the meaning depends on the algorithm selected by the `default algorithm` directive.

`runquota` *number*
: If this field is nonzero, it is assumed to be less than `file quota` and becomes the currently enforced file quota. The soft-quota algorithms use this field to adjust the file quota enforcement value without altering the actual file quota.

`delete` Deletes the quota entry by setting the usage value to 0 and the limit and warning values to their default values.

Abbreviations

The following minimum abbreviations are allowed in directives. Square brackets surround the optional part of each word. If an abbreviation is too short to be unique, an `ambiguous word` fatal error occurs.

Statement introductory words:

`acc`[ount], `aci`[d], `d`[efault], `e`[nable], `f`[ilesystem], `gi`[d], `gr`[oup], `o`[pen], `r`[emove], `s`[et], `ui`[d], `us`[er] `v`[ersion].

Value typing and other modifier words:

`acc`[ount], `aci`[d], `ag`[gregate], `alg`[orithm], `all`, `c`[ount], `def`[ault], `del`[ete], `ef1`, `ef2`, `ef3`, `ef4`, `ef5`, `en`[force], `ex`[ponential], `fi`[le], `fl`[ags], `gi`[d], `gr`[oup], `infi`[nite], `info`[rm], `ino`[de], `le`[vel], `li`[near], `n`[one], `on`[line], `q`[uota], `r`[unquota], `site1`, `site2`, `st`[yle], `t`[ime], `ui`[d], `un`[limited], `usa`[ge], `use`[r], `w`[arning].

NOTES

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

Active Category	**Action**
`system`, `secadm`, `sysadm`	Allowed to specify any file or device name.

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

MESSAGES

Possible messages from `quadmin` fall into the following categories:

- Usage error messages caused by improper option selection, and warning or fatal messages from the source.
- Messages caused by errors in the source; these include the line number on which the error was detected.
- Source lines that consist of multiple input lines with escaped new-line terminators; these inherit the line number of the first line of the group.

EXAMPLES

Example 1: Assume that you have file systems `/u`, `/v`, and `/w` defined as user home file systems, and their quota control files reside on the respective file system. In your environment, account IDs greater than 10, group IDs between 10 and 300, and user IDs greater than 49 are to have quotas. Also, file system `/tmp` is to have only user file quotas of 10,000 blocks. To define quotas for these file systems and give all users, accounts, and groups default quotas, you need the following source file:

```
    version 6
    filesystem /u ; filesystem /v ; filesystem /w ; filesystem /tmp
    open /tmp

      default account flags off
      default group flags off
      default user flags fi
      default user file quota 10000 warning .9 inode quota 200 warning .9
      user * file quota unlimited inode quota unlimited

open /u /v /w
  default account flags fi
  default account group fi
  default account user fi
  default account file quota 35000 file warning 0.85
  default account inode quota 400 warning 0.85
  default group file quota 35000 warning 0.85
  default group inode quota 400 inode warning 0.85
  default user file quota 15000 warning 0.85 inode quota 200 warning 0.85

#       Set special ranges of IDs to unlimited quota
  enable acid -10       # all acids 0 through 10
  enable gid -10, 300-  # all gids 0 through 10 and 300 and up
  enable uid -49        # all uids 0 through 49
  account * file quota unlimited inode quota unlimited
  group * file quota unlimited inode quota unlimited
  user * file quota unlimited inode quota unlimited

  #  Give user 'def' a small file quota on file system /v
open /v
  user def file quota 5000 warning default
```

Example 2: To set the disk quota for multiple groups and users, on file system `/core`, and quota control file `q`, first create a source file (*srcfile*) such as the following:

```
version 6
filesystem /core q
open /core
  group 16 file quota 6000 inode quota 2000
  group 21 file quota 6000 inode quota 2000
  user 100 file quota 6000 inode quota 2000
  user 101 file quota 6000 inode quota 2000
  user 102 file quota 6000 inode quota 2000
```

The first three lines of this file are required. You can add any number of groups or users. Then execute `quadmin` using the following command-line option:

`/etc/quadmin -m` *srcfile*

The file warning and inode warning values remain unchanged.

Example 3: For a single user (`uid 100`), on file system `/core`, and quota control file `q`, to change the file usage to `800`:

```
/etc/quadmin -Fmd 'filesys /core q; uid 100 file usage 800'
```

This changes only the file usage field; other fields in the quota control file remain unchanged.

FILES

`*/.Quota60`	Quota control file

SEE ALSO

`qudu`(8)

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

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`qudu` – Reports file system quota usage information

SYNOPSIS

`/etc/qudu` [`-a`] [`-A`] [`-D`] [`-f`] [`-G`] [`-p` *fstab_path*] [`-q` *quota_file*] [`-U`] *device_names*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `qudu` command traverses all of the inodes on the specified *device_names* and formats usage information for input to the `quadmin`(8) command. Output is sent to the `stdout` file.

The manner in which information is recorded by this command matches the manner in which the kernel records file system space usage when quota enforcement is active; it may not exactly match the usage information derived from other usage analysis tools.

The `qudu` command accepts the following options and operand:

`-a`	Uses aggregate quota counting method. Files migrated offline by DMF are included in the block counter.
`-A`	Selects account IDs. If `-A`, `-G`, or `-U` is not present, the default is `-AGU`.
`-D`	Sets debug mode. `qudu` sends information helpful in determining its internal workings to the `stderr` file. The function of the command is not affected.
`-f`	Reads `/etc/fstab` to find the quota configuration information for *device_names*. This option is not allowed with the `-p` or `-q` option.
`-G`	Selects group IDs. If `-A`, `-G`, or `-U` is not present, the default is `-AGU`.
`-p` *fstab_path*	Reads `/`*fstab_path*`/`*fstab* to find the configuration information for *device_names*. This option is not allowed with the `-f` or `-q` option.
`-q` *quota_file*	Places the specified quota file name on the `filesystem` directive to be written to the `stdout` file. This option is intended for use when the output from `qudu` will be used as input to `quadmin`(8). For compatibility with the previous release of `qudu`, if `-f` or `-p` has not been specified, *quota_file* is placed on the `filesystem` directive as the name of the quota file (null if `-q` is not specified). If either `-f` or `-p` has been specified, the quota file name is derived from `fstab`, and `-q` is not allowed.
`-U`	Selects user IDs. If `-A`, `-G`, or `-U` is not present, the default is `-AGU`.

device_names	The names of the device nodes from which usage should be extracted. Multiple device names may be included only if the `-f` or `-p` option is not present. In this case, usage over all specified devices is summed in the output. If `-f` or `-p` is present, one of the devices in a quota group (the only device if the group is singular) must be named; `qudu` sums usage over all devices in the group, as configured in `fstab`.

NOTES

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

Active Category	Action
`system`, `secadm`, `sysadm`	Allowed to specify any file or device name.

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

EXAMPLES

The following example collects disk usage for the `/dev/dsk/slash_a` file system mounted on the `/a` directory. Assume that `/etc/fstab` is set up to configure the file systems `/dev/dsk/slash_a` mounted on `/a`, `/dev/dsk/slash_b` mounted on `/b`, and `/dev/dsk/slash_c` mounted on `/c` as a quota group, with the quota file defined to be `/a/$QFILE`. Sample output follows the command line.

```
/etc/qudu  -f  /dev/dsk/slash_a

# Usage report by qudu: (SN-1203) on Mon Feb 12 14:11:01 CST 1990
version 6
filesystem dsk/slash_a /a/$QFILE
filesystem dsk/slash_b /a/$QFILE
filesystem dsk/slash_c /a/$QFILE
open dsk/slash_a dsk/slash_b dsk/slash_c
remove all usage
uid 0 inode usage 2 file usage 610
uid 559 inode usage 13 file usage 91
gid 0 inode usage 2 file usage 610
gid 117 inode usage 13 file usage 91
acid 0 inode usage 19 file usage 713
acid 559 inode usage 13 file usage 91
```

FILES

`sys/quota.h`	Quota control definition file
`/etc/fstab`	File describing file system and swapping partitions used by UNICOS

SEE ALSO

`quadmin`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`rdump` – Performs a file system dump across a TCP/IP network

SYNOPSIS

`/etc/rdump -f` *machine*`:`*device* [`-a`] [`-b` *nbs*`,`*pbs*] [`-d` *density*] [`-D` *device*] [`-e`] [`-m` *capacity*] [`-n`] [`-t` *dump_level*] [`-T` *t_fmt*] [`-u`] [`-v` *vsn_list*] *filesystem*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `rdump` command is a shell script that establishes a data path from *filesystem* to *machine*`:`*device* for backup purposes. The path goes from *filesystem* through an `/etc/dump` process across a pipe into an `/etc/lrmt` process across a socket connection into a process running `/etc/rmt` on a remote machine and from there into *machine*`:`*device*.

You can use any pre-existing *device* on the remote *machine* to contain the output; *device* does not have to be a tape.

The parameters that are accepted are a combination of the parameters for `/etc/dump` and for `/etc/lrmt`.

The `rdump` command accepts the following options:

`-f` *machine*:*device*	A required parameter. This is passed to `lrmt` as the network destination.
`-a`	A flag that is passed to `dump`.
`-b` *nbs*`,` *pbs*	This option is passed without change to `lrmt`.
`-d` *density*	An option that is passed to `dump`.
`-D` *device*	An option that is passed to `dump`.
`-e`	A flag that is passed to `dump`.
`-m` *capacity*	This option provides for volume switching control on Cray Research systems. The value *capacity* is taken to be the number of megabytes in each output volume. This is passed to `dump` as `-m` *capacity* and to `lrmt` as `-c` *capacity*. If `dump` or `lrmt` report a short write error, reduce the *capacity* value. This condition can occur when many small files are being dumped or when the block size of tape records is small on the destination machine.
`-n`	A flag that is passed to `dump`.
`-t` *dump_level*	An option that is passed to `dump`.
`-T` *t_fmt*	An option that is passed to `dump`.
`-u`	A flag that is passed to `dump`.

`-v` *vsn_list*	An option that is passed to `dump`.
filesystem	The `/dev/dsk` name of a mountable file system to be passed to the `dump` command.

The following `dump` options are not supported through the `rdump` interface:

`-f`	`rdump` sets `-f` - when executing `dump`.
`-g`	Parameter for local `rsv` and `tpmnt`.
`-l`	Parameter for local `tpmnt`.
`-c`	Equivalent to `-g CART`.
`-s`	Input to capacity computation for round tapes.
`-M`	A format for `tpmnt`.
`-R`	A format for local `rsv`.
`-P`	A format for prompting operator before `tpmnt`.
`-w`	Use dump for this listing.
`-W`	Use dump for this listing.

The following `lrmt` options are not supported through the `rdump` interface

`-p`	Use a named pipe rather than `stdin` and `stdout`.
`-d`	This is set to `-d n`.

FILES

`/etc/dumpdates`	Dump date record
`/etc/fstab`	File systems description
`/etc/group`	Group-information file (to find group *operator*)

SEE ALSO

`dump`(8), `lrmt`(8), `restore`(8), `rmt`(8), `rrestore`(8)

NAME

`reduce` – Extracts, formats, and outputs UNICOS security event files

SYNOPSIS

`/etc/reduce` [[`-i`] | [`-I` *optfile*]] [`-s` *date*] [`-e` *date*] [`-t` *typelist*] [`-l` *uidlist*] [`-u` *uidlist*] [`-c` *functionlist*] [`-g` *gidlist*] [[`-r`] | [`-R` *rawfile*]] [`-f` *logfile*] [`-m` *num*] [`-v`] [`-j` *jidlist*] [`-b` *pidlist*] [`-n` *filename*] [`-p`] [`-x`] [`-o` *object_level_list*] [`-O` [*level*[,*compartment*[,*compartment*[. . .]]]]] [`-S`]

`/etc/reduce` [[`-i`] | [`-I` *optfile*]] [`-s` *date*] [`-e` *date*] [`-t` *typelist*] [`-l` *uidlist*] [`-u` *uidlist*] [`-c` *functionlist*] [`-g` *gidlist*] [[`-r`] | [`-R` *rawfile*]] [`-f` *logfile*] [`-m` *num*] [`-v`] [`-j` *jidlist*] [`-b` *pidlist*] [`-n` *filename*] [`-p`] [`-x`] [`-o` *object_level_list*] [`-O` [*level*[,*compartment*[,*compartment*[. . .]]]]] [`-L`]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `reduce` command extracts and formats entries collected by the security log daemon, `slogdemon`(8), and generates a report of that data. The default is to report all entries currently in the active security log file. The default path for the log file is `/usr/adm/sl/slogfile`; however, this path can be site-defined.

If the `SECURE_MAC` configuration option is enabled, the security log file is labeled at `syshigh`.

Alternatively, you can process one of the retired security log files that has been date-named in the security log directory (`/usr/adm/sl`) or any file that is in the correct security log file format. You can use one or more of the options to limit the report to specific entries that are of immediate interest. Using the `-r` or `-R` option to create a pre-reduced security log file can be followed by issuing `reduce` with the `-f` option to access that file.

The `-i` and `-I` options should not be used on the same command line; only one is needed for correct results. They have identical functions, except that `-i` uses a default file, and you must specify an input file for `-I`.

The `-r` and `-R` options also should not be used on the same command line. They have identical functions, except that `-r` uses a default file, and you must specify an output file for `-R`.

The `reduce` command accepts the following options:

`-i`	Reads program options from the `opt.reduce` file in the current directory. Any options occurring in the command line override those found in the option file.
`-I` *optfile*	Reads program options from the file *optfile*, which may be a relative or full path name. Any options specified in the command line override those found in the option file.

`-s` *date* Ignores all entries posted earlier than *date*; *date* has the format *mmddhhmm*[*yy*[*ss*]] and is consistent in meaning with the `date`(1) command. This option also causes `reduce` to select files for processing (using parameters in the include file `/uts/cf.SN/config.h` that define retired security log file names) to ensure that the requested start time is reached. The `reduce` command searches the security log directory for the first (earliest) retired or active log containing the start date. This log and all later logs residing in the directory are processed until an end-of-processing condition is reached. If `now` is specified for *date*, `reduce` begins reading new entries; this should be used with the `-x` option.

`-e` *date* Ignores all entries posted later than *date*; *date* has the format *mmddhhmm*[*yy*[*ss*]] and is consistent in meaning with the `date`(1) command. When an entry with a date later than *date* is encountered, processing ends.

`-t` *typelist* Selects only entries matching the types in the *typelist* (a list of type identifiers separated by spaces or commas). Valid types and their meanings are as follows:

`audit`	Security auditing criteria selection changes
`cchg`	System configuration change
`chdir`	Change directory
`crl`	Cray/REELlibrarian activity
`dac`	Discretionary access control changes
`disc`	Discretionary access event
`disk`	Disk I/O error
`eoj`	End-of-job
`go`	System startup
`logn`	Login validation process
`mand`	Mandatory access event
`nami`	File manipulation system calls
`netcf`	Network configuration changes
`netip`	IP layer security violations
`netw`	Network access
`nfs`	Cray NFS activity
`nqs`	NQS activity
`nqscf`	NQS configuration changes
`oper`	Operational access event
`priv`	Use of privilege
`secsys`	Enhanced security system calls
`setuid`	`setuid` system calls
`ssd`	SSD I/O error
`stop`	System shutdown
`suid`	`setuid` file manipulation system calls
`sulog`	`su`(1) attempts
`tape`	Tape I/O error
`tchg`	System time change
`trust`	Trusted process activity

`-l` *uidlist* Selects only entries whose login user ID (which cannot change) matches one in *uidlist* (a list of numeric user IDs or ASCII user name specifiers separated by spaces or commas). This option is the surest way of getting true accountability. All actions performed after a user logs in contain the login user ID in the security log entry for that action.

`-u` *uidlist* Selects only entries whose subject user ID (real or effective) matches one in *uidlist* (a list of numeric user IDs or ASCII user name specifiers separated by spaces or commas).

`-c` *functionlist* Selects only the function-related security log entries (for example, mandatory/discretionary access, enhanced security system calls, and so on) whose function fields match one of those in *functionlist* (a list of valid functions separated by spaces or commas).

Valid functions are as follows:

```
acctid
alloc
alloct
chdir
chmod
chown
cpselect
creat
exec
exit
fork
fsecstat
getacl
getfacl
getsysv
ialloc
kill
limit
link
mkdir
mknod
mount
nicem
open
read
reada
resume
rmacl
rmfacl
rmdir
scrub
secstat
```

```
select
setacls
setdevs
setfcmp
setfflg
setflvl
setregid
setsid
setsockopt
setsysl
setsysv
setucat
setucmp
setulvl
setusrv
shutdown
suspend
tabinfo
tabread
trunc
umount
unlink
write
writea
```

`-g` *gidlist*	Selects only entries whose subject group ID (real or effective) matches one in *gidlist* (a list of numeric group IDs or ASCII group name specifiers separated by spaces).
`-r`	Writes a raw dump of the selected entries to the file `raw.reduce` in the current directory. An existing `raw.reduce` file is overwritten.
`-R` *rawfile*	Writes a raw dump of the selected entries to the file *rawfile*; *rawfile* can be a full or relative path name. Any existing *rawfile* is overwritten.
`-f` *logfile*	Uses the file *logfile* instead of the default file defined in the kernal via `config.h`. This option results in only one file being processed.
`-m` *num*	Halts processing after a specified number of entries, *num*, have been selected for output.
`-v`	Turns on verbose mode. Default is off. When on, `reduce` echoes the options.
`-j` *jidlist*	Selects only entries whose job ID (session) matches those in *jidlist*, a numeric list of job IDs separated by spaces or commas.
`-b` *pidlist*	Selects only entries whose process ID matches those in *pidlist*, a numeric list of process IDs separated by spaces or commas.

`-n` *filename* Selects only the entries for which the object file name component of the relative path name is the same as *filename*. This is applicable only to entry types that contain a relative path name.

`-p` Reconstructs a path name. For each entry that contains a relative path name, `reduce` attempts to reconstruct the full path name being referenced from the previous path history. The `SLG_PATH_TRACK` parameter (in `/uts/cf.SN/config.h`) must be enabled to use this option.

`-x` Continues to read the log file past EOF in the manner of the `tail`(1) command, reporting entries according to the selection criteria as they are received. This option functions correctly only if the file being processed is the same as the file currently being written to by the security logging daemon, `slogdemon`. The `-x` option can be used in conjunction with the `-s now` option to select only events occurring after `reduce` is initiated.

`-o` *object_level_list*
Selects only the entries whose object level matches one of the levels in *object_level_list*, a list of decimal level values separated by commas or spaces. Cannot be used with the `-O` option.

`-O` Selects only entries whose object label matches the specified value. The object label is specified in the form *level*[,*compartment*[,*compartment*[. . .]]]. *level* or *compartment* is the name or numeric value that represents the appropriate *level* or *compartment*, respectively. If no *compartments* are specified, or the text string `none` or `0` is used, then the compartment bitmask is set to `0`. Cannot be used with the `-o` option.

`-S` Prints the complete security label of both the subject and the object as part of the record header. The form of the subject's security label is `S_Label`: *level,comp1,comp2,. . .,compn*. The form of the object's security label is `O_Label`: *level,comp1,comp2,. . .,compn*. When the subject's and object's security labels are printed in the header, no subject or object label information is printed in the record body. This option will become the default header format in a future UNICOS release. See the `-L` option.

`-L` Prints the subject's and object's security level only as part of the record header. This record header format is the default format for the UNICOS 8.0 and later systems; it is needed to maintain compatibility with previous UNICOS releases. It will not be supported in future releases of the UNICOS operating system. See the `-S` option.

See *General UNICOS System Administration*, Cray Research publication SG–2301, for information on the `reduce` header fields.

NOTES

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

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

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

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

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

Using the `-p` option may cause `reduce` to operate more slowly and consume more memory than usual. Because of this effect, if the `-s` option is specified, path name reconstruction does not begin until the start time is reached. This may cause incomplete path name reconstruction if the process started before the start time selected.

FILES

`/usr/include/sys/slrec.h`	Defines security log record format
`/usr/include/sys/slog.h`	Defines security log file internal (kernel) structure
`/usr/src/uts/cf.SN/config.h`	Defines security log file parameters
`/usr/adm/sl/slogfile`	Repository for security event records (defined in the kernal through `config.h`)

SEE ALSO

`slogdemon`(8)

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

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

General UNICOS System Administration, Cray Research publication SG–2301

NAME

`regpar` – Generates register parity errors

SYNOPSIS

`/etc/regpar -r` *register* `-p` *parity*

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `regpar` command allows you to generate register parity errors in Revision 4 and subsequent Cray PVP systems CPUs by using the `/dev/secded ioctl`(2) interface to set bad parity into particular registers. The CPU must be in maintenance mode (you must have the maintenance mode switches on the maintenance panel `on` for all CPUs in which register parity errors are to be artificially generated).

The `regpar` command must be specified with the following options:

`-r` *register*	Specifies the register in which the bad parity bit is to be stored. The *register* argument can be `v`, indicating the V register.
`-p` *parity*	Sets the parity to even (0) or odd (1).

NOTES

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

Active Category	**Action**
`system`, `secadm`	In a privileged administrator shell environment, allowed to write shell redirected output to any file.
`sysadm`	Shell redirected output is subject to security label restrictions.

If the `PRIV_SU` configuration option is enabled, the super user can write shell redirected output to any file.

SEE ALSO

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

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

NAME

`remove` – Removes temporary accounting files

SYNOPSIS

`/usr/lib/acct/remove`

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `remove` command removes the temporary files `/usr/adm/acct/sum/wtmp*`, `/usr/adm/acct/sum/pacct*`, and `/usr/adm/acct/nite/lock*`. It is invoked by `startup`(8), which is the accounting start-up script, to clean up any files that may have been left over from a previous run.

FILES

`/usr/adm/acct/nite/lock*`	Controls simultaneous invocations of `csarun`(8) or `runacct`(8).
`/usr/adm/acct/sum/wtmp.`*MMDD*	A copy of `wtmp` file for *MMDD*; generated by standard UNIX System V accounting.
`/usr/adm/acct/sum/pacct.`*MMDD*	A concatenated version of all `pacct` files for *MMDD*; generated by standard UNIX System V accounting.

SEE ALSO

`acct`(8), `acctsh`(8), `csa`(8), `startup`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`rep` – Allows the operator to respond to action messages

SYNOPSIS

`/usr/lib/msg/rep` *msg_number* [*reply_string*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `rep` command allows the operator to respond to action messages, such as tape mount messages. The *msg_number* operand indicates the number of the action message to which you want to reply. The *reply_string* operand indicates an optional string to be returned to the sender of the action message. If no string is specified, a null string is returned to the sender.

NOTES

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

Active Category	**Action**
`repall`	Replies are not subject to security label restrictions.

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

Active Category	**Action**
`system`, `secadm`, `sysadm`, `sysops`	Allowed to use this command to reply to any messages.

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

Enclose any special characters or strings containing special characters in quotation marks, so they are not misinterpreted.

SEE ALSO

`infd`(8), `msgd`(8), `msgdaemon`(8), `msgdstop`(8)

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

NAME

`restore` – Performs an incremental file system restore

SYNOPSIS

`/etc/restore -i` [`-c`] [`-d`] [`-D` *device_name*] [`-f` *file*] [`-F` *file_id*] [`-g` *device_group*] [`-h`] [`-l` *lt*] [`-m`] [`-q`] [`-S` *symfile*] [`-T`] [`-v`] [`-V` *vsn*] [`-y`]

`/etc/restore -r` [`-c`] [`-d`] [`-D` *device_name*] [`-f` *file*] [`-F` *file_id*] [`-g` *device_group*] [`-h`] [`-l` *lt*] [`-m`] [`-q`] [`-S` *symfile*] [`-T`] [`-v`] [`-V` *vsn*] [`-y`]

`/etc/restore -M` [`-c`] [`-d`] [`-D` *device_name*] [`-f` *file*] [`-F` *file_id*] [`-g` *device_group*] [`-h`] [`-l` *lt*] [`-m`] [`-q`] [`-S` *symfile*] [`-T`] [`-v`] [`-V` *vsn*] [`-y`]

`/etc/restore -t` [`-c`] [`-d`] [`-D` *device_name*] [`-f` *file*] [`-F` *file_id*] [`-g` *device_group*] [`-h`] [`-l` *lt*] [`-m`] [`-q`] [`-S` *symfile*] [`-T`] [`-v`] [`-V` *vsn*] [`-y`] *files*

`/etc/restore -x` [`-c`] [`-d`] [`-D` *device_name*] [`-f` *file*] [`-F` *file_id*] [`-g` *device_group*] [`-h`] [`-l` *lt*] [`-m`] [`-q`] [`-S` *symfile*] [`-T`] [`-v`] [`-V` *vsn*] [`-y`] *files*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `restore` command reads data written by the `dump`(8) command. Its actions are controlled by the options selected. Other arguments to the command are file or directory names that specify the files to be restored.

The `restore` command restores the original file attributes when invoked by an authorized user (see NOTES for a description of an authorized user). Otherwise, the command restores the files using the attributes of the invoking user.

You must choose exactly one of the following options: `-i`, `-M`, `-r`, `-t`, or `-x`. Unless you specify the `-h` option, the appearance of a directory name refers to the files and (recursively) subdirectories of that directory.

The `restore` command accepts the following options:

`-i` This mode allows interactive restoration of files from a dump tape. After reading in the directory information from the tape, `restore` provides a shell-like interface that lets you move around the directory tree, selecting files to be extracted. The available commands follow; for those commands that require an argument, the default is the current directory:

Only the first character of a command is significant. An incorrect command of `exit`, for example, would yield the same results as `extract`.

l`s` [*directory*]
: Lists the current or specified directory. Entries that are directories are appended with a slash (/). Entries that are marked for extraction are prefixed with an * (as in **file*). If you set the verbose option, the inode number of each entry also is listed.

c`d` [*directory*]
: Changes the current working directory to the specified *directory*. Use commands of the form `cd` *name* rather than the form `cd ./`*name*.

p`wd`
: Prints the full path name of the current working directory.

a`dd` [*directory*] | [*filename*]
: Adds the current directory or the specified *directory* or *filename* to the list of files to be extracted. If a directory is specified, it and all its descendents are added to the extraction list (unless you specify the `-h` option on the command line). Use `add` with the current directory as default, rather than commands of the form "`add .`". Files that are on the extraction list are prepended with a * symbol when they are listed by `ls`.

: The `restore` command ensures that all files and directories that are added to the list of files to be extracted have the requisite directory structure in place on the file system being restored. If any directories are missing, they are immediately re-created.

d`elete` [*directory*] | [*filename*]
: Deletes the current directory or specified *directory* or *filename* from the list of files to be extracted. If a directory is specified, it and all its descendents are deleted from the extraction list (unless the `-h` option is specified on the command line). To extract most of the files from a directory expediently, add the directory to the extraction list and then delete those files that are not needed. Use `delete` with the current directory as default, rather than commands of the form "`delete .`".

: The `restore` command does not remove anything from the actual file system being restored, but only from the list of files to be extracted. Any directories created by a previous `add` command are not removed from disk. Thus, the following commands ensure that directories `w`, `x`, and `y` exist after the commands are executed:

```
restore > add w/x/y/z
restore > delete w/x/y/z
```

e`xtract`
: Extracts all files that are on the extraction list from the dump tape.

v`erbose`
: Enables verbose mode. When set, the `verbose` command causes the `ls` command to list the inode numbers of all entries. It also causes `restore` to print out information about each file as it is extracted.

h`elp`
: Summarizes the available commands.

q`uit`
: The `restore` command immediately exits, even if the extraction list is not empty.

`-r` Rebuilds an entire file system by using a level 0 backup and a collection of one or more incrementals. The first `restore -r` invocation processes the level 0 backup. Each subsequent `restore -r` invocation processes the next incremental backup, as ordered by increasing incremental levels (`-t` option to `dump`). Data is passed between invocations of `restore -r` in a file named `restoresymtabl`. You should remove this file after the last invocation.

`-M` Sets ownership, times, modes (permissions), and security labels on directories. The `restore` command does not usually require this option because these labels are usually set at the end of a `restore` procedure. If a `restore` procedure was interrupted before completion, however, you can run `restore` again with the `-M` option to set them.

`-t` Lists the names of the specified files if they occur on the tape. If you omit the *file* argument, the root directory is listed, which results in the entire contents of the tape being listed, unless the `-h` option has been specified.

`-x` Extracts the specified *files* from the tape. If the specified file matches a directory whose contents had been written onto the tape, and the `-h` option is omitted, the directory is recursively extracted. The owner, modification time, and mode are restored (if possible). If the *file* argument is omitted, the root directory is extracted, which results in the entire contents of the tape being extracted, unless the `-h` option was specified.

`-c` Causes `restore` to use cartridge tapes rather than round tapes.

`-d` Causes `restore` to enter another type of verbose mode (debug).

`-D` *device_name*
Specifies *device_name* as a specific tape device to be used for the restore. This lets you know on which tape drive the tapes have been requested to mount. If the specified device is unavailable, another drive of the same device type is substituted. You cannot use the `-D` option with the `-f` option. Specific device names are installation-specified parameters.

`-f` *file*
Performs the restore from *file*, rather than tape. If the name of the file is `-`, `restore` reads from standard input. For example, you can use `dump`(8) and `restore` in a pipeline to move a file system by executing the following command:

```
cd /mnt
dump -t  0 -f - /dev/dsk/usr | restore -x -f -
```

`-F` *file_id*
Specifies the tape file ID for a labeled tape. By default, `restore` uses the volume serial number (VSN) of the first tape of the dump file.

`-g` *device_group*
Specifies the name of the group to which the requested device belongs. The default is `-g TAPE`. The `-c` flag is equivalent to `-g CART`. Other types of devices may be accessed (for example, `-g SILO`).

`-h` Causes `restore` to extract the actual directory, rather than the files that it references, which prevents hierarchical restoration of complete subtrees from the tape.

`-l` *lt* Specifies *lt* as the type of label to be read on the tape. Values for *label_type* can be one of the following:

`nl`	Nonlabeled tapes
`sl`	IBM standard labels
`al`	ANSI labels (default)

`-m` Causes `restore` to extract a file in your current directory. This is useful only if a few files are being extracted and you want to avoid regenerating the complete path name to the file.

You also must use the `-x` or `-i` option in addition to `-m`, and you must specify a file name. The `restore` command extracts the file and places it in your current working directory. The restored file has the name of its corresponding inode.

For example, the following command restores only *directory/filename*, rather than any subtree below *directory/filename*:

```
restore -x -m directory/filename
```

`-q` Suppresses all error messages for inode modification failures when `restore` executes with user authority. This "quiet" flag does not suppress messages for data failures.

`-S` *symfile*
Creates the restore symbol table file in *symfile*. If you omit *file*, `restore` leaves the symbol table in `restoresymtabl`.

`-T` Generates a list of all files that were modified while `dump`(8) was executing.

`-v` Causes `restore` to type the name of each file it treats, preceded by its file type. Usually, `restore` does its work silently.

`-V` *vsn*
Causes `restore` to use the specified *vsn* list as a list of volume serial numbers to be used for the restore. Each VSN is a set of 1 to 6 alphanumeric characters, separated by colons (:). You cannot use this option with the `-f` option (restore from a file). If you do not use the `-V` option, `restore` solicits you to type in a VSN list.

The following sample command line illustrates the usage:

```
restore -i -V root1:root2:root3
```

`-y` Prevents `restore` from asking whether it should abort the restore if it gets a tape error. It tries to skip over the bad tape block(s) and to continue as best it can.

To use other specific tape mount options that `restore` does not cover, you may reserve a tape drive by using `rsv`(1) and mount the tape by using `tpmnt`(1), as follows:

```
rsv
tpmnt -l al -v a1:a2 -M -p /tmp/tapedev
restore -x -f /tmp/tapedev filename
rls -a
```

NOTES

The `restore` command is slowed by synchronous writes which are used on the kernel for file system robustness. To make `restore` run faster, assign ldcache to the file system.

You must do a level 0 dump after a full restore. Because `restore` runs in user code, it cannot control inode allocation; thus, you must do a full dump to get the new inode numbering.

The security of the files on the dump tape depends on the ability to read the tape.

The interface between `restore` and the data in the inode contains several fields. The owner, group, mode, and timestamps fields also include account ID. The inode data includes compartments, flags, levels, privilege assignment lists (PALs), access control lists (ACLs), and categories.

Unauthorized users are not allowed to restore file attributes from the dump file. The first 100 such failures generate warning messages; however, the files are restored with the attributes of the invoking user. To suppress the warning messages, use the `-q` option.

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

Active Category	**Action**
`system`, `secadm`	Restores all attributes associated with the files in the dump.

If the `PRIV_SU` configuration option is enabled, the super user can restore all attributes associated with the files in the dump.

CAUTIONS

Usually, the `set owner/mode for '.'` question that appears in the `-i` or `-x` option should be answered `NO`. The answer to this question determines whether the `restore` command should try to set the owner/mode for the root directory of the file system being restored. Unless you are restoring the root directory of the file system, you do not have to do this. If you are restoring a few files to somewhere other than the file system that was dumped (for example, into `/tmp`) answering `YES` to this question could cause difficulties because it will set the owner and mode of the root directory of the file system you are restoring (for example, `/tmp`), to the owner and mode of the root directory of the file system that was dumped.

The `dump`(8) and `restore` commands share an internal inode number that is used to pass information between levels of an incremental restore. If an inode region is added to your file system, the internal number of a given inode may change, making an incremental restore impossible. If the super-block time stamps indicate that this may have happened, the `dump` command prints a warning. If the maximum inode number has changed, an incremental restore fails. You can use `restore -x` or `restore -i`.

When you do a full restore of a file system, disk quotas should not be running. This prevents overwriting an active quota control file.

MESSAGES

The `restore` command complains about bad option characters. The `restore` command complains if it gets a read error. If `y` was specified, or a user responds with a `y`, `restore` tries to continue the restore.

The `restore` command can list numerous consistency checks. Most checks are self-explanatory or can never happen. Common errors are as follows:

"*filename*`: not found on tape`"
: The specified file name was listed in the tape directory, but it was not found on the tape. This is caused by tape read errors while looking for the file, and from using a dump tape created on an active file system.

"`expected next file` *inumber*`, got` *inumber*"
: A file that was not listed in the directory showed up. This can occur when using a dump tape created on an active file system, or can be caused by trying to restore sockets and unlinked temporary files.

"`Tape read error while restoring` *filename*"
"`Tape read error while skipping over inode` *inumber*"
"`Tape read error while trying to resynchronize`"
: A tape read error has occurred. If a file name is specified, its contents are probably partially wrong. If an inode is being skipped, or the tape is trying to resynchronize, no extracted files were corrupted, although files may not be found on the tape.

"`resync restore, skipped` *num* `blocks`"
: After a tape read error, `restore` may have to resynchronize itself. This message lists the number of blocks that were skipped over.

EXAMPLES

Example 1: The following example does an interactive `restore` from IBM standard labeled cartridge tapes, using a specified volume serial number list:

```
restore -i -l sl -V a1:a2 -c
```

Example 2: The following example does a complete restore from ANSI labeled tapes, requesting specific device `tape00`:

```
restore -r -l al -D tape00
```

FILES

`/tmp/rstdir*`	File that contains directories on the tape
`/tmp/rstmode*`	Owner, mode, and time stamps for directories
`./restorsymtabl`	Information passed between incremental restores

SEE ALSO

`dump`(8), `mkfs`(8), `mount`(8), `rdump`(8), `rrestore`(8)

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

NAME

`rexecd` – Invokes the remote execution server

SYNOPSIS

`/etc/rexecd` [`-S` *tos*] [`-U` *umask*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `rexecd` command is the server for the `rexec`(3C) routine. The server provides remote execution facilities with authentication that is based on user names and encrypted passwords.

The `rexecd` command invokes the centralized identification and authorization library routines to validate the user ID and password.

The `rexecd` command accepts the following options:

`-S` *tos*	Directs `rexecd` to set the IP Type-of-Service (TOS) option on the connection to the value *tos*. This value can be a numeric TOS value or a symbolic TOS name found in the `/etc/iptos` file.
`-U` *umask*	Directs `rexecd` to set the umask value of the process to the octal value *umask*.

Usually, `rexecd` is called by `inetd`(8) to wait for service requests at the port that is indicated in the `exec` service specification (see `services`(5)). When `rexecd` receives a service request, it initiates the following actions:

1. The server reads characters from the socket or transport endpoint up to a null byte (\0). The resultant string is interpreted as a base 10 ASCII number.
2. If the number received in step 1 is nonzero, `rexecd` interprets it as the port number of a secondary stream to be used for standard error. A second connection to the specified port on the client's machine is then created.
3. A null-terminated user name of 16 characters maximum is retrieved from the initial socket.
4. A null-terminated, encrypted password of 16 characters maximum is retrieved on the initial socket.
5. A null-terminated string that is to be passed to a shell as a command is retrieved on the initial socket. The length of the command is limited by the upper bound on the size of the system's argument list.
6. The user is validated at login time; if the authentication is successful, `rexecd` changes to the user's home directory and establishes the user and the user's group protections. If any of these steps fail, the connection is aborted and a diagnostic message is returned.

7. A null byte is returned on the connection associated with standard error, and the command line is passed to the normal user’s login shell. The shell inherits the network connections that are established by `rexecd`.
8. The `rexecd` command performs a `setusrv`(2) system call with the user’s default security parameters that are found in `/etc/udb`. Users cannot change any security attribute during the `rexecd` session.

NOTES

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

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

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

MESSAGES

All diagnostic messages are returned on the connection that is associated with standard error. Then any network connections are closed. An error is indicated by a leading byte with a value of 1 (0 is returned in step 7 after all steps prior to the command execution complete successfully).

`/bin/sh:` ...	The user’s login shell could not be started.
`command too long`	The command line that is passed exceeds the size of the argument list (as configured into the system).
`Login incorrect`	Your name and the password may not match. This is a generic message for any of several login failure causes. No more information is given. The user name may not be valid or the password may be wrong. You may have been denied access by the WAL check. Your login may be locked, disabled, or invalidated for security violations.
`No remote directory`	The `chdir` command to the home directory failed.
`password too long`	The password consists of more than 16 characters.
`Try again`	A `fork` process by the server failed.
`username too long`	The name consists of more than 16 characters.

BUGS

No facility is available that allows all data exchanges to be encrypted.

SEE ALSO

`ia_failure`(3C), `ia_mlsuser`(3C), `ia_success`(3C), `ia_user`(3C), `rexec`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

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

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

NAME

`ripquery` – Queries routing information protocol (RIP) gateways

SYNOPSIS

`/etc/ripquery` [`-1`] [`-2`] [`-a` *authkey*] [`-n`] [`-p`] [`-r`] [`-v`] [`-w` *time*] *gateways*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `ripquery` command requests all routes known by another routing information protocol (RIP) gateway by sending a RIP request or `POLL` command. The routing information in any routing packets returned is displayed numerically and symbolically. The `ripquery` command is intended to be used as a tool for debugging gateways, not for network management. Simple network management protocol (SNMP) is the preferred protocol for network management.

The `ripquery` command, by default, uses the RIP `POLL` command, which is an undocumented extension to the RIP specification supported by `gated`(8). The RIP `POLL` command is preferred over the RIP `REQUEST` command because it is not subject to split horizon and/or poisoned reverse effects. See RFC 1058 for more information.

The `ripquery` command accepts the following options:

`-1`	Sends the query as a version 1 packet. This is the default for UNICOS systems.
`-2`	Sends the query as a version 2 packet.
`-a` *authkey*	Specifies the authentication password to use for queries. If specified, an authentication type of `SIMPLE` is used, otherwise the default is an authentication type of `NONE`. Authentication fields in incoming packets are displayed, but not validated. This option is valid only with version 2 packets.
`-n`	Prevents the address of the responding host from being looked up to determine the symbolic name.
`-p`	Uses the RIP `POLL` command to request information from the routing table. This is the default. If no response occurs to the RIP `POLL` command, the RIP `REQUEST` command is tried. The `gated`(8) daemon responds to a `POLL` command with all the routes learned through RIP.

`-r`	Uses the RIP `REQUEST` command to request information from the gateway's routing table. Unlike the RIP `POLL` command, all gateways should support the RIP `REQUEST`. If no response occurs to the RIP `REQUEST` command, the RIP `POLL` command is tried. The `gated`(8) command responds to a `REQUEST` command with all the routes being announced through the specified interface. The `REQUEST` command provides information about the interface used to route packets back to the sender. This can be avoided by running a `ripquery` on the host being queried.
`-v`	Displays version information about `ripquery` before querying the gateways.
`-w` *time*	Specifies the time (in seconds) to wait for the initial response from a gateway. The default value is 5 seconds.
gateways	Specifies the gateways to be queried.

SEE ALSO

`gated`(8), `gdc`(8)

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

RFC 1058 - *Routing Information Protocol*

COPYRIGHT INFORMATION

This package and associated documentation is Copyright (c) 1990, 1991, 1992, 1993, 1994 Cornell University, all rights reserved. This software contains code that is Copyright (c) 1988 Regents of the University of California, all rights reserved.

GateD is maintained and developed by Cornell University and its collaborators.

NAME

`rlogind` – Invokes the remote login server

SYNOPSIS

`/etc/rlogind` [`-d`] [`-I` *initid*] [`-r` *highpty-lowpty*] [`-S` *tos*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `rlogind` command is the server for the `rlogin`(1B) command. The server provides a remote login facility with authentication that is based on privileged port numbers.

The `rlogind` command accepts the following options:

`-d` Specifies that each socket that is created by `rlogind` has debugging turned on (`SO_DEBUG`). With debugging enabled, the system traces all TCP packets sent and received on a socket. You can then use the `trpt`(8) command to interpret the packet traces.

`-I` *initid* Specifies the ID used by `/etc/init`. For more information, see the `inittab`(5) file format.

`-r` *highpty-lowpty*
Invokes the `login` program with *highpty-lowpty* default range of 0 through 100.

`-S` *tos* Directs `rlogind` to set the IP Type-of-Service (TOS) option for the connection to the value *tos*. This value can be a numeric TOS value or a symbolic TOS name found in the `/etc/iptos` file.

The `inetd`(8) daemon starts `rlogind` to receive service requests at the port indicated by the `login` service specification; see `services`(5). When a service request is received, the following protocol is initiated:

1. The server checks the client's source port. When the port is not in the range of 0 through 1023, the server aborts the connection.
2. The server checks the client's source address. When the address is associated with a host for which no corresponding entry exists in the host name database (see `hosts`(5)), the server aborts the connection.

After the source port and address are checked, `rlogind` allocates a pseudo terminal (see `pty`(4)) and manipulates file descriptors so that the slave half of the pseudo terminal becomes standard input, standard output, and standard error for a login process. The login process is an instance of the `login`(1) program invoked with the `-r` option. The login process then proceeds with the authentication process as described in `rshd`(8), but if automatic authentication fails, it reprompts the user to log in as indicated on a standard terminal line. The default range (*highpty-lowpty*) of pseudo terminals to use with `rlogind` is 0 through 100.

The parent of the login process manipulates the master side of the pseudo terminal, operating as an intermediary between the login process and the client instance of the `rlogin`(1B) command. In normal operation, the packet protocol that is described in `pty`(4) is invoked to provide `CONTROL-s` and `CONTROL-q` facilities and propagate interrupt signals to the remote programs. The login process propagates the client's terminal types, as found in the `$TERM` shell variable and terminal band.

NOTES

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

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

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

MESSAGES

All diagnostic messages are returned on the connection that is associated with `stderr`; then any network connections are closed. An error is indicated by a leading byte that has a value of 1.

`Host name for your address unknown`	No entry in the host name database existed for the client's machine.
`Try again`	A `fork` process by the server failed.
`/bin/sh: ...`	The user's login shell could not be started.

BUGS

The authentication procedure assumes the integrity of each client machine and the connecting medium. The scheme relies on the concept of privileged ports (those numbered less than 1024, which is a nonstandard notion that applies only to 4.2 BSD, 4.3 BSD, and their descendants). This is not a good security practice, but it is useful in an open environment.

No facility is available that allows all data exchanges to be encrypted.

SEE ALSO

`ftpd`(8), `rshd`(8), `trpt`(8)

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

`pty`(4), `hosts`(5), `services`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

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

NAME

`rmt` – Invokes the remote magnetic tape protocol module

SYNOPSIS

`/etc/rmt`

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `rmt` command is a program that the remote dump and restore programs use in manipulating a magnetic tape drive through an interprocess communication connection (see `rdump`(8) and `rrestore`(8)). Usually, `rmt` is started with an `rcmd`(3C) or `rexec`(3C) call.

The `rmt` program accepts requests to manipulate the magnetic tapes, performs the commands, and then responds with a status indication. All responses are in ASCII, and they are in one of the following two forms (you must press <`RETURN`> after each response):

`A`*number* Response to a successful command. The *number* argument is an ASCII representation of a decimal number.

`E`*error-number* error-message
Response to unsuccessful commands. The *error-number* argument is one of the possible error numbers described in `intro`(2); *error-message* is the corresponding error string as printed from a call to `perror`(3C).

Commands

The `rmt` command accepts the following commands (a space is present between each token):

`O` *device flags* Opens the specified *device* by using the indicated *flags*. The *device* argument is a full path name, and *flags* is an ASCII representation of a decimal number suitable for passing to `open`(2). Because the `rmt` protocol was originally defined on Berkeley UNIX systems, you must specify the flag bits by using the Berkeley definitions; `rmt` converts them to the appropriate UNICOS equivalents. The flag bits and their equivalents are as follows:

Flag Bits	**Equivalents**
`00001`	`O_WRONLY`
`00002`	`O_RDWR`
`00004`	`O_NDELAY`
`00010`	`O_APPEND`
`01000`	`O_CREAT`

`02000`	`O_TRUNC`
`04000`	`O_EXCL`

If a device was already opened, it is closed before a new `open` is performed.

`C` *device*	Closes the currently open device and ignores the *device* specified.
`L` *whence offset*	Performs an `lseek`(2) operation by using the specified parameters. The response value is that returned from the `lseek`(2) operation.
`W` *count*	Writes data onto the open device. `rmt` reads *count* bytes from the connection, aborting if a premature end-of-file is encountered. The response value is that returned from the `write`(2) system call.
`R` *count*	Reads *count* bytes of data from the open device. If *count* exceeds the size of the data buffer (10 Kbytes), it is truncated to the data buffer size. If the read was successful, `rmt` then performs the requested `read`(2) and responds with A*count_read*; otherwise, an error in the standard format is returned. If the read is successful, the data read is sent.
`I` *operation count*	Tries to perform a `MTIOCOP ioctl`(2) call on the device by using the specified parameters. Because UNICOS tape drivers do not support this `ioctl` request, the only operation allowed is `MTFSF` (skip file forward), which is simulated in `rmt` by a series of `read` calls (see `read`(2)).
`S`	Returns the status of the open device, obtained with a `MTIOCGET ioctl` call. Because UNICOS tape drivers do not support this operation, an `ack` is sent, followed by a status buffer filled with binary 0's.

Any other command causes `rmt` to exit.

BUGS

Do not use the `rmt` command for remote file access.

SEE ALSO

`rdump`(8), `rrestore`(8)

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

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

NAME

`route` – Manipulates network routing tables manually

SYNOPSIS

`/etc/route` [`-d`] [`-n`] [`-q`] [`-t`] [`-v`] *commands* [`-net | -host`] *destination gateway* [*modifiers* [*arg*]]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `route` command manually manipulates the network routing tables.

A given route is uniquely identified by its destination address, netmask, and Type of Service (TOS) attributes. When deleting or changing a route, you must specify these values so the kernel can locate the route.

The `route` command accepts the following options:

`-d` Debug mode. Operates similarly to the `-t` option, but it also displays the `route` message that would have been sent to the kernel. Useful for debugging the `route` command.

`-n` Prevents attempts to print host and network names symbolically when reporting actions. This is useful, for example, when all name servers are down on your local network and you need a route before you can contact the name server.

`-q` Suppresses the output of the `flush` directive.

`-t` Test directives. The `route` command processes the directives, but it makes no changes to the kernel routing table. Useful for testing configuration changes.

`-v` Prints additional routing details.

commands
The `route` command accepts several directives. You can use one of the following:

`add` Adds a route.

`change` Changes the gateway or metrics of a route.

`delete` Deletes a route. If the `-s` *tos* or `-netmask` *mask* modifiers are specified when adding a route, they must also be specified when deleting a route.

`flush -inet`
Flushes the routing tables of all gateway entries or, by specifying an optional keyword modifier that describes the address family, flushes only those routes whose destinations are of the specified address family. Specify `-inet` for the Internet address family.

`get` Looks up and displays the route for a destination.

`monitor`	Reports changes to the routing information base, routing lookup misses, or suspected network partitioning.

`-net | -host`

Forces *destination* to be interpreted as a network or a host address, respectively. If you omit these optional keywords and *destination* has a local address portion of `INADDR_ANY`, or *destination* is the symbolic name of a network, the route is assumed to be to a network; otherwise, it is assumed to be to a host. For example, `128.32` is interpreted as `-host 128.0.0.32`; `128.32.130` is interpreted as `-host 128.32.0.130`; `-net 128.32` is interpreted as `128.32.0.0`; and `-net 128.32.130` is interpreted as `128.32.130.0`. To eliminate ambiguity, use the `-net` or `-host` option.

destination gateway

destination is the destination host or network. *gateway* is the next-hop gateway to which packets should be addressed. Routes to a particular host are distinguished from those to a network by the interpretation of *destination*.

If you are adding an Internet route, *destination* is interpreted as a dot-notation host address. If this fails, the algorithm tries to interpret *destination* as a dot-notation network address. A dot-notation representation of a subnetted network address appears to be a host address and is interpreted as such. This interpretation affects the implicit network mask generation in that no network mask is generated for a host route (see the `-netmask` modifier). To ensure that a subnetted network address in dot-notation format is interpreted correctly, include the `-net` option on the command line.

If *destination* cannot be interpreted as a dot-notation address, the algorithm tries to resolve it by using `getnetbyname`(3C). If this fails, it uses `gethostbyname`(3C).

modifiers [*arg*]

The `route` command accepts several modifiers, some with arguments (*arg*). You can use any of the following:

`-admmtu` *mtu*	Specifies the largest size packet that will be sent to the next hop gateway when this route is used; *mtu* is a numeric argument.
`-blackhole`	Silently discards packets during updates
`-cloning`	Generates a new link level route on use
`-gid` *gidlist*	Limits the use of a route to particular groups of users. *gidlist* is of the form +`|`-*group1,group2,group3*, ... The + indicates an inclusive list; that is, only the specified groups may use this routing entry. The `-` indicates an exclusive list that excludes the specified groups from using the routing entry. The groups may be either numeric or ASCII names as shown in the `/etc/group` file.
`-inet`	Specifies that all subsequent addresses are in the Internet family.

`-interface` Specifies that the destination is directly reachable through an interface rather than through an intermediary system as a gateway. The gateway given is the address of this host on the common network, indicating the interface to be used for transmission.

`-link` Specifies that all subsequent addresses are specified as link-level addresses, and the names must be numeric specifications rather than symbolic names.

`-llinfo` Validly translates proto address to link address

`-netmask` *mask* The optional `-netmask` qualifier is intended to be used when manually adding subnet routes with different netmasks from that of the implied network interface. The required *mask* is an address to be interpreted as a network mask. You can use this modifier to override the implicit network mask generated in the `AF_INET` case by making sure this option follows the destination parameter.

If you specify this modifier when adding a route, you must also specify it when either deleting or changing the route.

`-noforward` Disables forwarding for this route.

`-nomtudisc` Disables path mtu discovery for this route.

`-nostatic` Pretend route added by kernel or daemon

`-proto1` Sets protocol specific routing flag #1

`-proto2` Sets protocol specific routing flag #2

`-reject` Emits an ICMP `unreachable` when matched

`-S` *arg* | `-service` *arg*

Requires an argument (symbolic name for option bits), such as `delay` or `throughput`, to mark the specified route with an IP Type-of-Service. See `iptos`(5) for other possible arguments. `-S` and `-service` are functionally identical.

If you specify this modifier when adding a route, you must also specify it when either deleting or changing the route.

`-static` Specifies a manually added route

`-tosmatch` Affects whether the specified routine can satisfy certain routing lookups. If you specify `-tosmatch` with a route, this route can be used to satisfy only those lookups that specify all of the type of service bits that are set in the route (using `-S` or `-service` modifiers). Lookups with more bits set can use this route if it is the best match available.

`-xresolve` Emits message on use (for external lookup)

NOTES

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

Active Category	**Action**
`system`, `secadm`, `sysadm`	May manipulate routing tables

If the `PRIV_SU` configuration option is enabled, the super user may manipulate routing tables.

Only an appropriately authorized user can use the `route` command to modify the routing tables.

The kernel installs point-to-point interface routes with a network mask of 255.255.255.255. To delete a point-to-point interface route, specify this network mask.

The `change` directive cannot be used on destinations with more than one route.

MESSAGES

`add [ host | network ] %s: gateway %s flags %x`
: The specified route that is being added to the tables. The values printed are from the routing table entry entered into the kernel routing table. If the gateway address used was not the primary address of the gateway (the first one returned by `gethostbyname`), the gateway address is printed numerically and symbolically.

`delete [ host | network ] %s: gateway %s flags %x`
: The specified route that is being deleted from the tables. The values printed are from the routing table entry entered into the kernel routing table. If the gateway address used was not the primary address of the gateway (the first one returned by `gethostbyname`), the gateway address is printed numerically and symbolically.

`%s %s done`
: When the `flush` directive is used, a message of this form indicates the destination and gateway address of each routing table entry that is deleted. The `-q` option disables this function.

`network is unreachable`
: An attempt to add a route failed because the gateway listed was not on a directly connected network. You must specify the next-hop gateway.

`not in table`
: A delete operation was tried for an entry that was not present in the tables.

`routing table overflow`
: An add operation was tried, but the system was low on resources and could not allocate memory to create the new entry.

`too many routes for destination`
: The `change` operation cannot be used for destinations that have multiple routes.

SEE ALSO

`gated`(8)

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

`hosts`(5), `intro`(4), `networks`(5), `route`(4P) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

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

NAME

`rpcbind` – Maps universal addresses to an RPC program number

SYNOPSIS

`rpcbind` [`-d`] [`-f` *filename*] [`-i`] [`-r`] [`-w`]

IMPLEMENTATION

Cray Research systems licensed for ONC+™ and UNICOS 8.3 or later

DESCRIPTION

The `rpcbind` server converts Remote Prcedure Call (RPC) program numbers into universal addresses. It must be running on the host to be able to make RPC calls on a server on that machine.

When an RPC service is started, it tells `rpcbind` the address at which it is listening, and the RPC program numbers it is prepared to serve. When a client wants to make an RPC call to a program number, it first contacts `rpcbind` on the server machine to determine the address where the RPC requests should be sent.

The `rpcbind` server should be started before any other RPC service. Normally, port monitors start standard RPC servers, so `rpcbind` must be started before port monitors are invoked.

When `rpcbind` is started, it checks that certain name-to-address translation calls function correctly. If they fail, the network configuration databases may be corrupt. Since RPC services cannot function correctly in this situation, `rpcbind` reports the condition and terminates.

The `-r` and `-f` options allow RPC servers to reregister themselves with `rpcbind`. This capability is useful if, for some reason, it is necessary to restart `rpcbind` while the RPC servers continue. If `rpcbind` stopped executing and then was restarted without either of these options, any active RPC process would lose its RPC registration.

The `rpcbind` command accepts the following options:

`-d` Runs in debug mode. In this mode, `rpcbind` does not fork when it starts, prints additional information during operation, and aborts on certain errors. With this option, the name-to-address translation consistency checks are shown in detail.

`-f` *filename*
Directs `rpcbind` to send a `SIGHUP` signal to each of the process/user ID pairs found in the specified *filename* file, in addition to the standard RPC servers that receive the signal when the `-r` option is used. (Thus, the `-f` option implies the `-r` option.) The format of the information in *filename* is as follows:

```
    program_1  [uid1]
    program_2  [uid2]
        .
        .
        .
    program_n  [uidn]
```

The `program` field is the name of the program (for example, `rpc_server`). The `uid` field is optional and specifies that `SIGHUP` be sent only to processes of the specified name running under the specified `uid`. If the `uid` field is blank, `SIGHUP` is sent to all processes of the specified name, regardless of their user ID. For example, assume that `rpcbind` has been invoked using the `-f rpcfile` option, and assume that `rpcfile` contains the following lines:

```
    rpc_server      0
    test_server     123
    test_server     456
    new_server
```

The following table indicates which processes will and will not be sent a `SIGHUP` signal:

Process name	UID	Sent `SIGHUP`?
`rpc_server`	0	Yes
`rpc_server`	123	No
`rpc_server`	456	No
`rpc_server`	789	No
`test_server`	0	No
`test_server`	123	Yes
`test_server`	456	Yes
`test_server`	789	No
`new_server`	0	Yes
`new_server`	123	Yes
`new_server`	456	Yes
`new_server`	789	Yes
`other_proc`	0	No
`other_proc`	123	No
`other_proc`	456	No
`other_proc`	789	No

The `-f` option is useful when site-specific RPC servers are running. However, to make this option useful, you must code the servers to catch the `SIGHUP` signal and to reregister with `rpcbind` when they receive it.

`-i` Allows nonsecure forwarding of requests to the `MOUNTPROG` program.

`-r` Specifies the command `restart` for standard RPC servers. This option causes `rpcbind` to send a `SIGHUP` signal to the following processes:

```
cnfsd
inetd
keyserv
lockd
mountd
nfsd
pcnfsd
statd
ypbind
yppasswdd
ypserv
```

Each of these processes catches the `SIGHUP` signal and reregisters with `rpcbind` upon receiving the signal. Any root process invoked with one of these process names is sent a `SIGHUP` signal if you invoke `rpcbind` using the `-r` option.

`-w` Performs a warm start. If `rpcbind` aborts or terminates on receipt of a `SIGINT` or `SIGTERM` signal, it writes the current list of registered services to the `/tmp/portmap.file` and `/tmp/rpcbind.file` files. When the `-w` option is specified, `rpcbind` searches for these files and starts operation with the registrations found in them. This allows `rpcbind` to resume operation without requiring all RPC services to be restarted.

NOTES

Terminating `rpcbind` using `SIGKILL` prevents the warm-start files from being written.

All RPC servers must be restarted if the following occurs: `rpcbind` crashes (or is killed using `SIGKILL`) and is unable to to write the warm-start files; `rpcbind` is started without the `-w` option after a graceful termination; or, `rpcbind` does not find the warm-start files.

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

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

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

SEE ALSO

`portmap`(8), `rpcinfo`(8)

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

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

NAME

`rpcbstart` – Starts `rpcbind`(8) or `portmap`(8)

SYNOPSIS

`/etc/rpcbstart`

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `rpcbstart` command starts either an `rpcbind`(8) or `portmap`(8) server. The command uses the following criteria for determining which server to start:

1. If `rpcbind`(8) is on your system and your site has an Open Network Computing plus (ONC+^{TM}) license, `rpcbind`(8) is started.
2. Otherwise, `portmap`(8) is started.

SEE ALSO

`portmap`(8), `rpcbind`(8)

NAME

`rpcinfo` – Reports RPC information

SYNOPSIS

`/etc/rpcinfo -p` [*host*]
`/etc/rpcinfo -u` *host program-number* [*version-number*]
`/etc/rpcinfo -t` *host program-number* [*version-number*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `rpcinfo` command makes a Remote Procedure Call (RPC) to an RPC server and reports information according to the option you select.

The following options are available:

`-p` *host* Probes the portmapper on the specified *host* computer and prints a list of all registered RPC programs. If you do not specify the *host*, it defaults to the value returned by `hostname`(1).

`-u` *host program-number version-number*
Makes an RPC to procedure 0 of *program_number* by using the user datagram protocol (UDP) and reports whether a response was received.

`-t` *host program-number version-number*
Makes an RPC to procedure 0 of *program_number* by using Transmission Control Protocol (TCP) and reports whether a response was received.

SEE ALSO

`portmap`(8)

`rpc`(3C) in the *Remote Procedure Call (RPC) Reference Manual*, Cray Research publication SR–2089

NAME

`rrestore` – Restores a file system dump across a TCP/IP network

SYNOPSIS

`/etc/rrestore -f` *machine*:*device* [`-i`] [`-d`] [`-h`] [`-m`] [`-q`] [`-v`] [`-y`] [`-S` *symfile*] [`-b` *nbs*,*pbs*]

`/etc/rrestore -f` *machine*:*device* [`-M`] [`-d`] [`-h`] [`-m`] [`-q`] [`-v`] [`-y`] [`-S` *symfile*] [`-b` *nbs*,*pbs*]

`/etc/rrestore -f` *machine*:*device* [`-r`] [`-d`] [`-h`] [`-m`] [`-q`] [`-v`] [`-y`] [`-S` *symfile*] [`-b` *nbs*,*pbs*]

`/etc/rrestore -f` *machine*:*device* [`-t`] [`-d`] [`-h`] [`-m`] [`-q`] [`-v`] [`-y`] [`-S` *symfile*] [`-b` *nbs*,*pbs*] [*file1 file2 ...*]

`/etc/rrestore -f` *machine*:*device* [`-x`] [`-d`] [`-h`] [`-m`] [`-q`] [`-v`] [`-y`] [`-S` *symfile*] [`-b` *nbs*,*pbs*] [*file1 file2 ...*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `rrestore` command is a shell script that establishes a data path from file *device* at *machine* to a process running `/etc/restore`. The path comes from *machine*:*device* through a remote process running `/etc/rmt` across a socket connection into a local process running `/etc/lrmt` across a pipe and finally into a process running `/etc/restore`. *device* may be any file on the remote *machine*; it does not need to be a tape.

The options that are accepted are a combination of the parameters for `/etc/restore` and `/etc/lrmt`.

`-f` *machine*:*device*	A required parameter. This is passed to `lrmt` as the network destination.
`-i`, `-r`, `-M`, `-t`, `-x`	Passed without change to `restore`.
`-d`, `-h`, `-m`, `-q`, `-v`, `-y`	Passed without change to `restore`.
`-S` *symfile*	Passed without change to `restore`.
`-b` *nbs*,*pbs*	Passed without change to `lrmt`.
file1 file2 ...	Passed to `restore`.

The following `restore` options are not supported:

`-c`	Equivalent to `-g CART`.
`-D` *device_name*	Parameter for `tpmnt`.
`-f` *file*	Performs the restore from *file*, rather than round tape. If the name of the file is `-`, `restore` reads from `lrmt` through `stdin`.
`-F` *file_id*	Parameter for `tpmnt`.

`-g` *device_group*	Parameter for `tpmnt` and `rsv`
`-l` *lt*	Parameter for `tpmnt`.
`-V` *vsn*	Parameter for `tpmnt`.

The following `lrmt` options are not supported:

`-d` *dest*	This is set to `-d p`.
`-p` *pname*	A named pipe to be used for passing data on the local machine (rather than `stdin` or `stdout`).
`-c` *capacity*	The capacity, in megabytes, of a remote output tape device.

SEE ALSO

`dump`(8), `lrmt`(8), `orestore`(8), `rdump`(8), `restore`(8), `rmt`(8)

NAME

`rshd` – Invokes the remote shell server

SYNOPSIS

`/etc/rshd` [`-S` *tos*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `rshd` command is the server for the `rcmd`(3C) routine and the `remsh`(1B) and `rsh`(1) commands. The server provides remote execution facilities with authentication based on privileged port numbers. `inetd`(8) invokes `rshd`.

The `rshd` command invokes the centralized identification and authorization library routines to validate the user ID and password.

The `rshd` command accepts the following option:

`-S` *tos* Directs `rshd` to set the IP Type-of-Service (TOS) option on the connection to the value *tos*. This value can be a numeric TOS value or a symbolic TOS name found in the `/etc/iptos` file.

When a service request is received, the following protocol is initiated:

1. The server checks the client's source port. If the port is not in the range of 0 through 1023, the server aborts the connection.
2. The server reads characters from the socket up to a null ( ) byte. The resultant string is interpreted as a base 10 ASCII number.
3. If the number received in step 1 is nonzero, it is interpreted as the port number of a secondary stream to be used for `stderr`. A second connection to the specified port on the client's machine is then created. The source port of the second connection also must be in the range of 0 through 1023.
4. The server checks the client's source address. If the address is associated with a host for which no corresponding entry exists in the host name database `/etc/host` (see `hosts`(5)), the server aborts the connection.
5. A null-terminated user name that consists of 16 characters maximum is retrieved on the initial socket. This user name is interpreted as a user identity to use on the server's machine.
6. A null-terminated user name that consists of 16 characters maximum is retrieved on the initial socket. This user name is interpreted as the user identity on the client's machine.
7. A null-terminated command to be passed to a shell is retrieved on the initial socket. The upper limit of the size of the system's argument list determines the length of the command.

8. The `rshd` command then validates the user according to the following steps. The remote user name is searched for in the password file and a `chdir`(2) call is made to move to the user's home directory.
   - If either the lookup or `chdir`(2) call fails, the connection is terminated.
   - If the user is not the super user (user ID 0), the `/etc/hosts.equiv` file is consulted for a list of hosts that are considered equivalent; see `hosts.equiv`(5).
   - If the client's host name is in this file, the authentication is considered successful. Some systems may be configured to require extra authentication. See the NOTES section.
   - If the lookup fails, or the user is the super user, the `.rhosts` file (see `rhosts`(5)) in the home directory of the remote user is checked for the machine name and identity of the user on the client's machine.
   - If this lookup fails, the connection is terminated. Also, the `.rhosts` file must have read/write permissions only for the user.
9. A null byte is returned on the connection that is associated with `stderr` and the command line is passed to the normal login shell of the user. The shell inherits the network connections established by `rshd`.

NOTES

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

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

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

The system configuration may require the `/etc/hosts.equiv` and `.rhosts` files each to contain a match for the remote host, and also require the remote user and local user names to match.

MESSAGES

All of the following diagnostic messages are returned on the connection associated with `stderr`; then any network connections are closed. An error is indicated by a leading byte with a value of 1 (0 is returned in step 9 after successful completion of all steps preceding the command execution).

`locuser too long`	The name of the user on the client's machine is longer than 16 characters.
`remuser too long`	The name of the user on the remote machine is longer than 16 characters.
`command too long`	The command line passed exceeds the size of the argument list (as configured into the system).
`Hostname for your address unknown`	The host name database contains no entry for the client's machine.
`Login incorrect`	The password file entry does not exist for the user name.

`No remote directory`	The `chdir`(2) call to the home directory failed.
`Permission denied`	The authentication procedure that is previously described failed.
`Can’t make pipe`	The pipe that the `stderr` needs is not created.
`Try again`	The server’s `fork` process server failed.
`/bin/sh: ...`	The user’s login shell could not be started.

BUGS

The authentication procedure used in `rshd` assumes the integrity of each client machine and the connecting medium. Also, the scheme relies on the concept of privileged ports (those numbered less than 1024, which is a nonstandard notion that applies only to 4.2 BSD, 4.3 BSD, and their descendants). This is not a good security practice, but it is useful in an open environment.

No facility that allows all data exchanges to be encrypted is available.

SEE ALSO

`inetd`(8)

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

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

`ia_failure`(3C), `ia_mlsuser`(3C), `ia_success`(3C), `ia_user`(3C), `rcmd`(3C) in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

`hosts`(5), `hosts.equiv`(5), `rhosts`(5), `services`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

TCP/IP Network User’s Guide, Cray Research publication SG–2009

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

NAME

`rstatd` – Performs kernel statistics server function

SYNOPSIS

`/etc/rstatd`

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `rstatd` command is a Remote Procedure Call (RPC) server that returns performance statistics obtained from the kernel to the client process that issued the RPC request. It is always registered as RPC program number 100,001. `inetd`(8) usually invokes the `rstatd` daemon.

FILES

`/etc/inetd.conf`	Contains configuration information and listens for incoming service requests
`/etc/rpc`	File that makes remote procedure calls

SEE ALSO

`inetd`(8)

`havedisk`(3R), `rstat`(3R) in the *Remote Procedure Call (RPC) Reference Manual*, Cray Research publication SR–2089

NAME

`rsvportbm` – Creates a bit map of well-known reserved ports

SYNOPSIS

`/etc/rsvportbm [-c]`

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `rsvportbm` command updates the port bit map in the kernel memory, based on the well-known reserved port numbers defined in the `/etc/services` file. Each time the command runs, it creates a new bit map, based on the port numbers in the `/etc/services` file, compares it with the one in the kernel memory, and replaces the one in the kernel memory if the two bit maps are different.

The `rsvportbm` command accepts the following option:

`-c` Specifies that a check will be made only if the bit map of well-known reserved ports from the `/etc/services` file differs from the one set in the kernel memory. If the bit maps are different, the following message is displayed:

```
The kernel memory port bitmap is not up-to-date.
```

Only an appropriately authorized user can use this command.

NOTES

This command should be run only at multiuser startup and each time the `/etc/services` file is modified.

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

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

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

FILES

`/etc/services`	File that contains a list of port numbers

SEE ALSO

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

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

NAME

`runacct` – Runs daily accounting

SYNOPSIS

`/usr/lib/acct/runacct` [*mmdd* [*state*]]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `runacct` command is the main daily accounting shell procedure used by the standard UNIX accounting system. It is usually initiated by using `cron`(8). The `runacct` command processes connect, fee, disk, and process accounting files. It also prepares summary files for `prdaily`(8) or billing purposes.

The `runacct` command ensures that active accounting files or summary files are not damaged if errors occur. It records its progress by writing descriptive diagnostic messages into the `active` file. When an error is detected, a message is written to `/dev/console`, mail (see `mail`(1)) is sent to `root` and `adm`, and `runacct` terminates. To protect against reinvocation, `runacct` uses a series of lock files. The `lock` and `lock1` files prevent simultaneous invocation, and `lastdate` prevents more than one invocation per day.

The `runacct` command accepts the following options:

mmdd	specifies the month and day for which `runacct` will rerun the accounting.
state	Designates where processing should begin. Overrides the contents of `statefile`.

`runacct` breaks its processing into separate, restartable *states* by using `statefile` to remember the last *state* completed. It accomplishes this by writing the *state* name into `statefile`; it then looks in `statefile` to see what it has done and to determine what to process next. The *states* are executed in the following order:

`SETUP`	Moves active accounting files into working files
`WTMPFIX`	Verifies integrity of `/etc/wtmp` file (see `utmp`(5)), correcting date changes if necessary
`CONNECT1`	Produces connect session records in `ctmp.h` format
`CONNECT2`	Converts `ctmp.h` format records into `tacct.h` format
`PROCESS`	Converts process accounting records into `tacct.h` format
`MERGE`	Merges the connect and process accounting records
`FEES`	Converts output of `chargefee` into `tacct.h` format and merges with connect and process accounting records
`DISK`	Merges disk accounting records with connect, process, and fee accounting records

`MERGETACCT`	Merges the daily total accounting records in `daytacct` with the summary total accounting records in `/usr/adm/acct/sum/tacct`
`CMS`	Produces command summaries
`USEREXIT`	User exit that executes site-dependent accounting program or script
`CLEANUP`	Cleans up temporary files and exits

To restart `runacct` after a failure, first check the `active` file for diagnostics; then fix up any corrupted data files such as `pacct` or `wtmp`. You must remove the `lock` files and `lastdate` file before `runacct` can be restarted. If `runacct` is being restarted, you must specify the *mmdd* argument. The entry point for processing is based on the contents of `statefile`; to override this, include the desired *state* on the command line.

NOTES

The mail recipients (`root` and `adm`) can be changed by modifying the `MAIL_LIST` parameter in `/etc/config/acct_config`.

BUGS

You should not restart `runacct` in the `SETUP` *state*, because doing so could overwrite valid data. Run `SETUP` manually and restart by using the following command line:

`runacct` *mmdd* `WTMPFIX`

If `runacct` failed in the `PROCESS` *state*, remove the last `ptacct` file because it is not complete.

If `runacct` terminates abnormally and leaves the lock files in place, the next execution of `runacct` will remove these locks, but it will also terminate abnormally.

EXAMPLES

Example 1: To start `runacct`, use the following command line:

```
nohup runacct 2> /usr/adm/acct/nite/fd2log &
```

Example 2: To restart `runacct`, use the following command line:

```
nohup runacct 0601 2>> /usr/adm/acct/nite/fd2log &
```

Example 3: To restart `runacct` at a specific *state*, use the following command line:

```
nohup runacct 0601 MERGE 2>> /usr/adm/acct/nite/fd2log &
```

FILES

`/etc/config/acct_config`	Accounting configuration file
`/etc/wtmp`	Connect time accounting data
`/usr/adm/acct/day/pacct*`	Process-accounting files switched using `turnacct`(8)
`/usr/adm/acct/nite/active`	Used by `runacct` to record progress and to issue error messages
`/usr/adm/acct/nite/daytacct`	Total accounting records for one day in `tacct.h` format
`/usr/adm/acct/nite/lock`	Used to control serial use of `runacct`
`/usr/adm/acct/nite/lock1`	Used to control serial use of `runacct`
`/usr/adm/acct/nite/lastdate`	Last day `runacct` executed in *date* `+%`*m%*`d` format
`/usr/adm/acct/nite/statefile`	A record of the current state during execution of `runacct`
`/usr/adm/acct/nite/ptacct*.`*mmdd*	Process-accounting information in `tacct.h` format
`/usr/lib/acct/run.user`	Executable called from the `runacct` script

SEE ALSO

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

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

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

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

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`runsequence` – Performs automatic test sequencing

SYNOPSIS

`/etc/diag/scripts/runsequence` *seqfile* [*cpu*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `runsequence` program is used with the `crontab`(1) command to perform automatic test sequencing (scheduling and testing without operator intervention). Error messages are returned to specified users through email and can also be sent to the system operator. The CPU in which the error occurred is downed if the error is repeatable and the down-CPU feature is enabled.

When an error occurs and a CPU is downed, error messages alert service personnel, who then examine the error log to determine where the error occurred. The goal is to detect and isolate failures before a system or application failure occurs.

To initiate automatic test sequencing, do the following:

1. Set the shell variables in the `runsequence` shell script.
2. Customize (or create) the sequence files.
3. Create the input file for the `crontab` command.
4. Execute the `crontab` command.

Cray Research recommends that a site check with its customer service representative before running diagnostics on CRAY J90 and CRAY EL systems on a regular basis (for example, using `runsequence`).

`crontab` input file

The `crontab` input file is read by the `/etc/cron` utility to determine the tasks to be done and the times. `/etc/cron` is the mechanism for scheduling `runsequence` execution. The `crontab` input file contains the following information:

- Times at which the sequences are to be run.
- Calls to `runsequence` (Each call to `runsequence` must contain an appropriate sequence file name and, optionally, a CPU designator.)

The `runsequence` command accepts the following options:

seqfile Name of the file containing the sequence of diagnostic tests to be run, the test command options, and comments. Comments are the same as shell script comments; they start with a pound sign (#) and continue to the end of the line.

cpu CPU in which the tests are to be run. *cpu* can be a, b, c, and so on, up to the maximum number of CPUs, or a digit can be used to designate a CPU. If this option is specified, the diagnostic tests in the sequence file must be CPU tests. All of the log and core files are placed in subdirectories of the `DIAGLOG` directory. The name of the subdirectory corresponds to the CPU designator. The subdirectory is created if it does not already exist. For example, the log and core files from a CPU diagnostic test executing in CPU b will be placed in subdirectory `b`. If the *cpu* option is not specified, the log and core files will be placed in the `other` subdirectory in the `DIAGLOG` directory.

If the *cpu* option is specified and the `DOWNCPU` variable is set to ON, the CPU will be downed if *cpu* consists of a single character or digits and the error is repeatable.

The following is the suggested `crontab` input file for a system with four CPUs:

```
#
# cronfile: Crontab input file for running diagnostic testing
# sequences.  See CRONTAB(1) for a complete definition of the
# first 6 fields in the cronfile.
#
# The example below runs cpuseq once an hour in each cpu,
# every day of the year.
#
# findseq removes extraneous files once a day at midnight.
#
# dtseq or cfdtseq need to be set up at the site.
#
# This file should be installed as follows:
#     host$ crontab <filename>
#
# Note: To run olcfdt, set FSPATH and DT to appropriate values.
#1  * * * 0 FSPATH=/??? DT=???? $HOME/scripts/runsequence cfdtseq
# Note: To run oldt, set DEVNAME to the appropriate value.
#1  * * * 0 DEVNAME=devname $HOME/scripts/runsequence oldt
1  * * * * $HOME/scripts/runsequence cpuseq a
15 * * * * $HOME/scripts/runsequence cpuseq b
30 * * * * $HOME/scripts/runsequence cpuseq c
45 * * * * $HOME/scripts/runsequence cpuseq d
1  0 * * * $HOME/scripts/runsequence dailyseq a
15 0 * * * $HOME/scripts/runsequence dailyseq b
30 0 * * * $HOME/scripts/runsequence dailyseq c
45 0 * * * $HOME/scripts/runsequence dailyseq d
1  0 * * * FINDPATH=$HOME/log $HOME/scripts/findseq
```

The minute field is set to 1 to offset the diagnostic program execution to 1 minute after the hour. This allows scheduled system activities to be performed at the start of each hour.

Note that FSPATH, DT, and DEVNAME must be set to the appropriate values for your site when the crontab input file is first installed. For example, these values might be set as follows:

```
1  * * * 0 FSPATH=/tmp DT=DD49 $HOME/scripts/runsequence cfdtseq
```

The following is an optional addition to the suggested crontab input file for a system with four CPUs. Note that olsbt(8) will have large wall-clock completion times on heavily loaded systems. olsbt should only be run if problems with semaphore-related hardware are suspected. Regularly scheduled use of olsbt is not recommended.

```
# Optionally runs olsbt to test semaphore-related hardware
1  0 * * * $HOME/scripts/runsequence sbtseq a,b,c,d
15 0 * * * $HOME/scripts/runsequence sbtseq b,c,d,a
30 0 * * * $HOME/scripts/runsequence sbtseq c,d,a,b
45 0 * * * $HOME/scripts/runsequence sbtseq d,a,b,c
```

Sequence files

The sequence files contain a list of the diagnostic tests to be executed and their related command options. A recommended sequence file is supplied with the diagnostics. A site may determine that more or less testing is desirable, or circumstances may require that testing be varied. You can either customize the recommended sequence file for your site or create new sequence files. Place the sequence files in the directory specified by the DIAGSCRIPT shell variable.

The following are the recommended sequence files, which contain the maximum suggested diagnostic execution times:

cpuseq:

```
#
#  cpuseq: This sequence is a list of diagnostics to be
#  run by runsequence.  The cpu is selected in the crontab
#  file (cronfileX).
#
olcrit cputime 3 +getseed       Reads seed from olcrit.seed if available
olcsvc cputime 3 +getseed       Reads seed from olcsvc.seed if available
```

dailyseq:

```
#
#  dailyseq: This sequence is a list of diagnostics to be
#  run by runsequence once a day.  The cpu is selected in the crontab
#  file (cronfileX).
#
olcrit cputime 30 +getseed      Reads seed from olcrit.seed if available
olcsvc cputime 30 +getseed      Reads seed from olcsvc.seed if available
olibuf cputime 30 +getseed      Reads seed from olibuf.seed if available
olcfpt cputime 30 +getseed      Reads seed from olcfpt.seed if available
olcm   cputime 30 +getseed      Reads seed from olcm.seed if available
```

The following is an optional addition to the recommended sequence input file. It should be run only if semaphore-related problems are suspected.

`sbtseq`:

```
#
#  sbtseq: This sequence tests olsbt in all cpu’s available.
#  The cpu is selected in the crontab file (cronfileX).
#
olsbt cputime 2 +getseed
```

The following is a sample `cfdtseq` sequence file:

`cfdtseq`:

```
#
#  cfdtseq: This sequence tests a mass storage device.
#  FSPATH and DT are defined in the cronfile.  If more than
#  one copy of olcfdt is run, it must have a unique value
#  specified for fn.
#  cfdtseq should be run from the installed cronfile as follows:
#
olcfdt maxp 20 fn $FSPATH/workfil. sz 250 dt $DT
rm -f $FSPATH/workfil
```

The following is a sample `dtseq` sequence file:

`dtseq`:

```
#
#  dtseq: This sequence tests a mass storage device.
#  DEVAME is defined in the crontab.
#
oldt -nF2 -m1 -d $DEVNAME
```

The following is a sample `findseq` sequence file:

`findseq`:

```
#
# findseq: This sequence finds (starting at FINDPATH) and removes
# small log files, stderr files and log files older than TOO OLD.
# FINDPATH is defined in cronfile (default:  FINDPATH=$HOME/log).
#

TOO_OLD=180     # Number of days to save log files

#
# Remove small log files or stderr files that runsequence created.
#
find $FINDPATH   -name ’*.[0-9]*[0-9]’ -size -300c ) -o -name

#
#Remove any log file that has not been touched recently
#
find $FINDPATH -name ’*.[0-9]*[0-9]’ -type f -atime +$TOO_OLD -exec rm -f {} ;
```

`runsequence` **shell script**

The `runsequence` shell script runs under the POSIX shell and executes a series of diagnostic tests by reading a file containing a list of the tests to be run. The tests should be run with the verbose option disabled (`-verbose`; the default), because the size of each diagnostic output file is used to determine whether the test has failed.

The shell script maintains the diagnostic output and sends messages to a specified list of users when an error is detected. You can set the following variables in the `runsequence` shell script:

`CONSOLE=ON|OFF`
: Enables (`ON`) or disables (`OFF`) the option that sends error messages to the system operator. The default is `ON`.

`DIAGBIN=`*path*
: Indicates the full path name of the directory in which the executable binary files of the diagnostic tests reside. If they reside in more than one directory, enter colons between directories. The following entry defines a single directory:

```
DIAGBIN=/ce/bin
```

The following entry defines several directories:

```
DIAGBIN=/ce/bin:$HOME/bin
```

The default is `/etc/diag:/etc:/ce/bin`.

`DIAGLOG=`*path*
: Indicates the full path name of the directory in which the log files are saved when a diagnostic test detects an error. The default is `/usr/spool/diag`.

`DIAGSCRIPT=`*path*
: Indicates the full path name of the directory in which the sequence files reside. You can specify only one full path name. The default is `/etc/diag/scripts`.

`DOWNCPU=ON|OFF`
: Enables (`ON`) or disables (`OFF`) the option that downs a failing CPU. For a CPU to be downed, `runsequence` must be given a single character or digits in the *cpu* option, and the error must be repeatable. The default is `OFF`.

`MAILLIST="`*user ...user*`"`
: Provides a list of users to be notified when a diagnostic test detects an error. Enter a space between user names and enclose the list in double quotation marks. It is recommended that the list contain more than one user name. The default is `LOGNAME`.

`NICE=`*n*
: Indicates the amount by which the diagnostic test's priority in the execution queue is to be lowered. *n* can be any integer in the range 1 through 19. If a value greater than 19 is entered, it is processed as if it were 19. If a value of 0 or less is entered, it has no effect. The default is 4.

`RUNLOG=`*logfile*
: Indicates the name of the log file containing information on the sequence being run and any errors detected. The log file resides in the `DIAGLOG` directory.

`SAVECORE=ON|OFF`
: Enables (`ON`) or disables (`OFF`) the option that renames and saves each core file generated. If `SAVECORE` is set to `OFF`, any new core file overwrites an existing one. The default is `OFF`.

The default values for the variables in the `runsequence` shell script are as follows:

```
CONSOLE=ON      # Set to ON to send messages to the operator
DIAGBIN=/etc/diag:/etc:/ce/bin      # Location of the executable diagnostic tests
DIAGLOG=/usr/spool/diag      # Location of the diagnostic log files
DIAGSCRIPT=/etc/diag/scripts      # Location of the diagnostic sequence lists
DOWNCPU=OFF     # Down failing CPU if repeatable error
RUNLOG=$DIAGLOG/runlog  # Program log
NICE=4  # Amount by which to lower the diagnostic
          test's priority
SAVECORE=OFF    # Existing core file will be overwritten
MAILIST="$LOGNAME"      # List of people to receive error messages
```

EXAMPLES

The following example shows a `runsequence` error message, which is sent to the system operator and to users specified by the `MAILLIST` variable.

```
Mon May 20 11:44:37 CST 1996: diagnostic found an ERROR: olcrit cputime 1
 +getseed cpu a
Dump in /usr/spool/diag/a/olcrit.111931
```

SEE ALSO

`crontab`(1) for information on the `crontab` file

Online Maintenance Tools Guide for Cray PVP Systems, Cray Research publication SD–1012. (This document contains information private to Cray Research, Inc. It can be distributed to non-CRI personnel only with approval of the appropriate Cray manager.)

NAME

rusersd – Performs RPC-based network user name server function

SYNOPSIS

```
/etc/rusersd
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The rusersd command is a Remote Procedure Call (RPC) based server that returns a list of users on a host to a requesting RPC-based client. It is always registered as RPC program number 100002. Typically, inetd(8) invokes the rusersd daemon.

FILES

/etc/inetd.conf	Contains configuration information and listens for incoming service requests
/etc/rpc	File that makes remote procedure calls

SEE ALSO

inetd(8)

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

rnusers(3R) in the *Remote Procedure Call (RPC) Reference Manual*, Cray Research publication SR–2089

NAME

`rwall` – Writes to all users on a network

SYNOPSIS

`/etc/rwall` *host*. . .

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `rwall` command reads a message from standard input until the end-of-file (EOF). It then sends this message, preceded by the following line, to all users who are logged in on the specified host machines:

```
Broadcast Message at [Time] ...
```

A machine can receive a message only if it is running `rwalld`(8), which is started typically by the daemon `inetd`(8).

The `rwall` command takes the following argument:

host. . .	Specifies the host machines on which users must be logged in to receive the message that `rwall` sends.

BUGS

The time-out is fairly short so that messages can be sent to a large group of machines (some of which might be down) in a reasonable amount of time. Thus, the message might not get through to a heavily loaded machine.

FILES

`/etc/inetd.conf`	Contains configuration information and listens for incoming service requests

SEE ALSO

`inetd`(8), `rwalld`(8), `wall`(8)

NAME

`rwalld` – Performs RPC-based network `rwall`(8) server function

SYNOPSIS

```
/etc/rwalld
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `rwalld` command is a Remote Procedure Call (RPC) based server that processes `rwall`(8) requests. It is always registered as RPC program 100008. Usually, `inetd`(8) invokes the `rwalld` daemon.

FILES

`/etc/inetd.conf`	Contains configuration information and listens for incoming service requests
`/etc/rpc`	File that makes remote procedure calls

SEE ALSO

`inetd`(8), `rwall`(8), `wall`(8)

NAME

`sam` – Displays data about system activity

SYNOPSIS

`/usr/bin/sam` [*csam_options*]
`/usr/bin/sam` [*xsam_options*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `sam` command is a generic interface to the `csam`(8) and `xsam`(8) utilities. Both utilities display data received from `samdaemon`(8). `xsam` is a graphic user interface based on the X Window system, while `csam` is a menu driven display based on curses, which is useful on dumb terminals. `sam` invokes either program using the following rules:

- If the `-h` *hostname* option has been specified, `csam` is started.
- If the `DISPLAY` variable is part of the user's shell environment, `xsam` is started.
- If the `-display` *value* option is specified, `xsam` is started.
- In any other case, `csam` is started.

`sam` accepts the following options:

csam_options	Options accepted by the `csam` utility; refer to the `csam`(8) man page for more information.
xsam_options	Options accepted by the `xsam` utility; refer to the `xsam`(8) man page for more information.

SEE ALSO

`csam`(8), `samdaemon`(8), `xsam`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`samdaemon` – System activity data daemon

SYNOPSIS

`/usr/lib/sam/samdaemon` [`-?`] [`-d`] [`-f`] [`-l` *logfile*] [`-D`]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The system activity monitor daemon, `samdaemon`, reads kernel data structures in memory as requested by the display clients, `xsam`(8) and `csam`(8). `samdaemon` can run on a Cray Research mainframe or on a connected OWS operator workstation. The daemon and clients communicate through Remote Procedure Call (RPC).

`samdaemon` contains a list of kernel data structures. When a client requests a structure, the daemon reads that structure and passes the information back to the client. At fixed intervals, the daemon re-reads the requested information, but passes it to a client only upon a subsequent client request.

The `samdaemon` command accepts the following options:

`-?`	Prints a short synopsis of the command-line arguments.
`-d`	Turns on debugging mode.
`-f`	Forces a start up of the daemon, even when the RPC program number is still defined. `samdaemon` uses the RPC program number as a *lock* to ensure that only one daemon is running at any given time. If abnormal termination occurs, this *lock* is not cleared.
`-l` *logfile*	Specifies an alternative log file; the default is `/usr/lib/sam/samdaemon.log`.
`-D`	Prints a list of all built-in default values.

`samdaemon` provides access control to UNICOS data with the use of a validation file. If `samdaemon` finds a validation file (by default, `/usr/lib/sam/samdaemon.val`), it will return data only for validated users. The validation file consists of entries that take the following format:

```
host_name user_name
host_name *
host_name -user_name
```

The first format allows access for the specified user from the specified host. The second format allows access for all users from the specified host. The third format denies access for the specified user from the specified host.

On startup, `samdaemon` attempts to read the two configuration files, `./samdaemon.rc` and `/etc/config/samdaemon.rc`. If the first file is found, the second is not read. If neither file is found, `samdaemon` uses default values.

The rules governing the format of the `samdaemon.rc` are as follows:

- Lines starting with # are treated as comments.
- All other lines have the format *token value*. The following *tokens* are valid:

`AUTOGEN`	Name of the configuration binary; the default is `/usr/lib/sam/sama`.
`MAP_FILE`	Name of the map file generated by `AUTOGEN`; the default is `/usr/lib/sam/samdaemon.map`.
`VALID_FILE`	Name of the validation file; the default is `/usr/lib/sam/samdaemon.val`.
`NLIST_FILE`	Name of the *name list* file; the default is `/unicos`. This token is valid only for daemons running on Cray Research systems.
`HEARTBEAT`	Time in seconds between structure updates; the default is 5 seconds.
`MASTER`	Host name of the Cray Research system to monitor; the default is `cray`. This token is valid only for daemons running on an operator workstation. Another `samdaemon` must be running on the target Cray Research system.
`MASTER_MODE`	If set, `samdaemon` does not respond to data requests. You can use this token to restrict the use of `samdaemon` on a Cray Research system and force users to use a `samdaemon` resident on an operator workstation. The default value is 0. This token is valid only for daemons running on the Cray Research system.
`TABLE_HBEAT`	Number of structure updates from the kernel without any client request; the default is 10.
`PROC_HBEAT`	Heartbeat of the *proc table* updates from the kernel. Because the *proc table* is big, updates can be configured to occur at a slower pace. The unit of `PROC_HBEAT` is `HEARTBEAT`. The default value is 2.
`PROC_READS`	Number of *proc table* entries `samdaemon` is to read. To limit the amount of memory needed to hold a copy of the kernel *proc table*, `samdaemon` reads only `PROC_READS` entries at one time. If `PROC_READS` is set to a negative number, all entries of the kernel *proc table* are read simultaneously. You should use caution when changing the value of `PROC_READS`; too small a value results in higher system overhead, while too great a value significantly increases the size of `samdaemon`. The default value is 50.
`MEM_DEV`	Name of the device for memory access. For the UNICOS operating system, the default value is `/dev/mem`. For Sun Workstations, the default value is `/dev/fy0/xc0i0`.
`RPC_VERSION`	RPC version number; the default is 1. This token may be used for debugging.

`MODEL_E`	Used to configure nonstandard configurations between operator workstations and the IOS. When running on a Sun Workstation, the daemon assumes a connected IOS model E. This value can be used to override this default. This is only valid for daemons running on an operator workstation.

FILES

`/etc/config/samdaemon.rc`	System activity monitoring (`sam`) configuration file
`/usr/lib/sam/samdaemon.log`	System activity monitoring (`sam`) configuration file
`/usr/lib/sam/samdaemon.val`	System activity monitoring (`sam`) validation file

SEE ALSO

`csam`(8), `sam`(8), `xsam`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`sadc`, `sa1`, `sa2` – Generates system activity data on a routine basis

SYNOPSIS

`/usr/lib/sa/sadc` [*t n*] [*ofile*]

`/usr/lib/sa/sa1` [*t n*]

`/usr/lib/sa/sa2` [`-a`] [`-b`] [`-c`] [`-d`] [`-e` *time*] [`-g`] [`-h`] [`-i` *sec*] [`-j`] [`-k`] [`-l`] [`-p`] [`-q`] [`-s` *time*] [`-t`] [`-u`] [`-v`] [`-w`] [`-x`] [`-y`] [`-z`] [`-A`] [`-B`] [`-H`] [`-M`] [`-T`] [`-W`] [`-X`] [`-Z`]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

System activity data can be accessed upon special request of a user (see `sar`(1)); it can also be accessed automatically, on a routine basis, by invoking `sa1`, `sa2`, and `sadc`. The UNICOS operating system contains a number of counters that are incremented as various system actions occur. These counters include CPU usage, buffer usage, disk and tape I/O activity, tty device activity, switching and system call, file access, queue activity, and interprocess communication.

The `sadc`, `sa1`, and `sa2` commands sample, save, and process this data.

The shell collector, `sadc`, samples system data *n* times every *t* seconds and writes in binary format to *ofile*, if specified, or to standard output. If you do not specify *t* and *n*, `sadc` writes a special record. Use this facility at system boot time to mark the time at which the counters restart from 0. The following `/etc/rc` entry writes the special record to the daily data file to mark the system restart:

```
/usr/lib/sa/sadc /usr/adm/sa/sa‘date +%d‘
```

The `sa1` command, a variant of `sadc`, collects and stores data in the binary file `/usr/adm/sa/sa`*dd; dd* is the current day. The *t* and *n* operands specify that records are to be written *n* times every *t* seconds; the default is 1. The following entries in an appropriately authorized user’s `crontab` file (see `cron`(8)) produce records every 20 minutes during working hours and hourly otherwise:

```
0 * * * 0,6 /usr/lib/sa/sa1
0 8-17 * * 1-5 /usr/lib/sa/sa1 1200 3
0 18-7 * * 1-5 /usr/lib/sa/sa1
```

The shell script `sa2`, a variant of `sar`(1), writes a daily report in the `/usr/adm/sa/sar`*dd* file. The options are explained in the `sar`(1) man page. The following `crontab` entry reports important activities hourly during the working day:

```
5 18 * * 1-5 /usr/lib/sa/sa2 -A
```

NOTES

Any `ncyls` values reported are valid only on Cray Research systems with an I/O subsystem model E (IOS-E).

`sa1` must be executed by an appropriately authorized user to get all possible data collected. `sadc` collects `ldcache` statistics from `/dev/dsk`, which only appropriately authorized users can access.

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

Active Category	**Action**
`system`, `secadm`, `sysadm`	Allowed to collect all possible data.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to collect all possible data.

FILES

`/usr/adm/sa/sa`*dd*	Daily data file
`/usr/adm/sa/sar`*dd*	Daily report file
`/usr/adm/sa.adrfl`	Address file

SEE ALSO

`cron`(8)

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

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`sdaemon` – Starts or kills daemon processes

SYNOPSIS

`/etc/sdaemon -c` [`-f` *file*] [`-g` *daemongroup*] *daemon*
`/etc/sdaemon` [`-s`] [`-`[`qnv`]] [`-f` *file*] [`-h` *header*] *daemon* [...]
`/etc/sdaemon` [`-s`] [`-`[`qnv`]] [`-f` *file*] [`-h` *header*] `-g` *daemongroup*
`/etc/sdaemon -k` [`-`[`qnv`]] [`-f` *file*] [`-h` *header*] *daemon* [...]
`/etc/sdaemon -k` [`-`[`qnv`]] [`-f` *file*] [`-h` *header*] `-g` *daemongroup*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `sdaemon` script starts or kills daemons (and other commands) according to information that is contained in a tabular configuration file. It is invoked at system startup by many start-up scripts (including `/etc/rc`, `/etc/netstart`, `/etc/tcpstart` (invoked by `/etc/netstart`), and `/etc/nfsstart`) to start the daemons necessary for normal system operation.

Each daemon that is to be started or killed can be specified on the command line according to a tag name that is given for the daemon in the configuration file. Alternatively, you can start or kill all of the daemons listed in the configuration file as members of the group name *daemongroup* by specifying the `-g` option. As part of its configuration, the `sdaemon` script with the `-c` option can return a simple status, indicating whether the daemon is in the configuration file and the path given points to an executable file.

When starting or killing daemons, `sdaemon` prints a header line, which is not terminated by a newline character, and it prints a tag name after the header for each daemon that starts successfully or is killed. The list terminates with a period and a newline character. The default header strings are `Starting` [*group*] `daemon(s):` and `Stopping` [*group*] `daemon(s):`; these strings are used for starting and killing daemons, respectively. Use the `-q` option to suppress this information.

The `sdaemon` script accepts the following options:

`-c` Determines whether the specified daemon is listed in the daemon's file and if its path name is executable. The `sdaemon` script returns a true value (that is, exits with a status of 0) if the specified daemon is listed as a tag in the configuration file and if the path name that is specified for the tag exists and is executable; otherwise, `sdaemon` returns a false value (that is, exits with a nonzero exit status). When used together with the `-g` option, `sdaemon` returns a true value only when the specified daemon is listed as a tag in the configuration file with a group membership of *group*, a *groupstart* value of `YES` for the tag, and a path name for the tag that exists and is executable.

`-f` *file* Uses *file* as the configuration file from which to fetch information about daemons to be started or killed.

`-g` *daemongroup*
: Starts or kills all daemons in the specified *daemongroup*. When starting a group of daemons, start each daemon as directed by its *groupstart* action in the configuration file (see the File Format subsection).

daemon Specifies the name of the daemon to be started or killed.

`-s` Starts the specified daemon, or all of the daemons that are listed in the configuration file as part of the *daemongroup* specified with the `-g` option. This is the default action.

`-q` Specifies quiet mode. Does not print either a header or the tag names of daemons that are started or killed.

`-n` Specifies no execution commands. The `sdaemon` script processes arguments and prints headers and tags, as usual, but does not actually execute the commands to start or kill the specified daemons. (This is very useful, in conjunction with the `-v` option, to double-check the actions that `sdaemon` will perform before actually performing them.)

`-v` Specifies verbose mode. The `sdaemon` script prints the command that it would actually execute to start or kill the specified daemons. When the commands that kill a daemon are printed, each command is followed by the actual process ID of the daemon, which is determined by the `sdaemon` and enclosed in brackets.

`-h` *header* Uses the string *header* as the header that is printed before the list of daemons that are started or killed. White space in a header string must be escaped from the shell.

`-k` Kills the specified daemon that is currently running, or all daemons that are listed in the configuration file as part of the *daemongroup* specified with the `-g` option, as directed by the kill information supplied in the configuration file for the appropriate daemon.

File Format

A configuration file for `sdaemon` consists of a series of lines with the following format:

daemongroup tag groupstart kill pathname arguments ...

The fields on each line have the following meanings:

daemongroup The name of a group to which this daemon belongs.

tag The tag name by which the individual daemon can be named as a command-line argument, and which `sdaemon` prints to indicate successful starting or killing of the daemon. Adjacent lines that have the same *tag* have only a single *tag* printed after the last line is executed.

groupstart The group start-up action for this daemon, which may be `YES`, `NO`, or `ASK`, or a run-level specific designation of the three. If the *groupstart* action is `YES`, the daemon is started when this daemon's group is specified with the `-g` flag. If the *groupstart* action is `NO`, the daemon is not started when this daemon's group is specified with the `-g` flag (however, you can start the daemon by not using the `-g` flag but explicitly specifying the tag on the command line). If the *groupstart* action is `ASK`, the operator of the command is prompted about whether to start this daemon when this daemon's group is specified with the `-g` flag. The prompt line is as follows:

```
Do you want to start tag (y/n) ?
```

To start the daemon, press `<RETURN>`, or any response that begins with the characters `Y` or `y`.

To tailor a group start-up action for a daemon to specific system run levels, use a run-level *groupstart* designation that has the following format:

runlevels=*action*

In the preceding example, *runlevels* is a concatenation of the characters for which *action* is the appropriate group start-up action. Alternatively, you can use the asterisk character (`*`) as a synonym for all run levels; omitting *runlevels*= from the beginning of the string can be used as a synonym for all run levels.

To specify different group start-up actions for different run levels for the same daemon specification, use colon characters to separate multiple run-level *groupstart* designations. The `sdaemon` script interprets these multiple designations from left to right and uses the first action specified for the current run level.

As examples, `sdaemon` interprets the following *groupstart* actions as follows:

Action	**Interpretation**
`NO`	Do not start this daemon as part of the group.
`YES`	Always start this daemon as part of the group.
`ASK`	Prompt for whether to start this daemon as part of the group.
`2=YES`	Start this daemon as part of its group if the system is at run level 2. (Implicitly, the user is prompted as to whether this daemon will be started, if the system is at any other run level.)
`2346=YES`	Start this daemon as part of its group when the system is at run level 2, 3, 4, or 6.
`2=ASK:*=YES`	Prompt for whether or not to start this daemon as part of its group when the system is at run level 2, and always start this daemon when the system is at any other run level.

`2=YES:1=NO:ASK`
: Always start this daemon as part of its group when the system is at run level 2; do not start this daemon when the system is at run level 1, and prompt for whether to start this daemon when the system is at any other run level. (The string `ASK` implicitly functions as a synonym for `*=ASK`.)

(See `init`(8) for further description of system run levels.)

kill The following information describes how to kill a currently running copy of the daemon:

- If *kill* is one hyphen (`-`), `sdaemon` must not attempt to find or kill a currently running copy of this daemon.
- If *kill* is the name of an executable file, `sdaemon` executes the file, expecting the program or script that is executed to exit with a status of 0 after a successful kill of the running daemon.
- If *kill* is the name of a nonexecutable file, `sdaemon` assumes that it is a text file that contains the process identification number (PID) of a currently running copy of the daemon, to which `sdaemon` sends a `SIGTERM`.
- If *kill* is one asterisk (`*`), `sdaemon` sends a `SIGTERM` to all running processes (as determined by `ps`(1)) that have a name equal to the file name component (the base name) of *pathname*.
- Otherwise, `sdaemon` sends a `SIGTERM` to all running processes (as determined by `ps`(1)) that have a name of *kill*.

pathname The path name of the file to execute to start the daemon.

arguments The command-line arguments to pass to the executable file when the daemon is started.

An initial # character at the beginning of a line indicates that the line is a comment.

MESSAGES

If the daemon with the tag name *tag* is not listed in the configuration file *file*, the following message is issued:

```
sdaemon:  daemon 'tag' not found in 'file'
```

BUGS

The `sdaemon` script has no flow-control ability in its configuration file; that is, the successful execution of an earlier daemon or command must not be a condition of executing any given daemon or command.

Because `sdaemon` relies on output from `ps`(1) and the base name of commands to determine the process IDs of many running daemons, the possibility exists that it can kill unintended programs if they share the same base name as the daemon being killed.

When using the `-g` option to start more than one daemon, if a daemon reads from standard input when it is started, it clears the pipe used internally in the `sdaemon` script. The result is that no daemons are started after that, and `sdaemon` exits normally.

The daemons started by `sdaemon` may inherit some characteristics of the user evironment that is running `sdaemon`. This may influence how the daemon executes. For example, if the user’s operator has restricted CPU or memory limits when starting a daemon, that daemon may fail when those limits are reached.

FILES

`/etc/config/daemons`	Default configuration file

SEE ALSO

`init`(8)

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

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

NAME

`sdc`, `sdcx` – Collects system data

SYNOPSIS

`/usr/bin/sdc` [`-c` *path*] [`-d` *debug_file*] [`-h` *host*[`:`*port*]] [`-i` *interval*] [`-n` *sample_count*] [`-B` | `-S`] [`-R` *request_file*] [*output_file*]

`sdcx` [`-i` *interval*] [`-n` *sample_count*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `sdc` command generates, compiles, and executes a `C` program named `sdcx`, which samples system activity data from the kernel. The `sdcx` source code cannot be accessed, but the executable can be retained by using the `-c` option.

When the `-c`, `-B`, or `-S` option is specified, `sdc` writes only a header record to *output_file*. When these options are not used, data records are also written. If *output_file* is not specified, output is written to a file named `dcf` in the current directory.

Each time `sdc` is invoked, it generates and compiles code for `sdcx`, and writes a header record. `sdc` may then start a new session and use the `fork`(2) system call to create a child process, which executes `sdcx`. The parent `sdc` process exits, and the child process collects *sample_count* data samples at a rate of one sample each *interval*.

The `sdcx` command samples the system activity counters and writes data records. `sdcx` does not generate header records. To use `sdcx`, first save the `sdcx` binary, and write a header record to the output file by invoking `sdc` with the `-c` option. Next, sample the system data by executing `sdcx`. You must append the `sdcx` output, which is written to `stdout`, to the `sdc` output file. Failure to do so can cause the `tsar`(8) command to abort or can generate erroneous reports.

UNICOS Resource Administration, Cray Research publication SG–2302, contains the list of data items that `sdc` and `sdcx` collect. The `sdc -R` option allows you to sample a subset of the data items.

The `tsar`(8) command formats the output file into an ASCII report.

The `sdc` command accepts the following options:

`-c` *path* Specifies the path name of the directory where `sdcx`, the data collector executable, is placed. By default, the data collector is placed in `$TMPDIR` and is not accessible after the parent `sdc` process exits. This option writes a header record to the output file and does not gather data samples.

`-d` *debug_file* Specifies the name of the file where debug information is written.

`-h` *host*[`:`*port*]	Specifies the name of the remote *host* where the data sampling is to occur and the *port* which is used. The default *port* is 0. By default, data sampling is done on the host on which `sdc` is invoked.
`-i` *interval*	Specifies the data sampling interval. *interval* is of the form *xx*`h`*yy*`m`*zz*`s` (*xx* hours, *yy* minutes, *zz* seconds) or *zz* seconds. By default, data is sampled every 5 minutes.
`-n` *sample_count*	Specifies the number of samples to be taken. By default, `sdc` collects 5 data samples.
`-B`	Specifies that a system boot header record be written to the output file. No data is collected with this option. The `-B` and `-S` options are mutually exclusive.
`-S`	Specifies that a system shutdown header record be written to the output file. No data is collected with this option. The `-B` and `-S` options are mutually exclusive.
`-R` *request_file*	Specifies the name of the file that contains a list of data items to be sampled. By default, all of the data items sampled. A complete list of data items that `sdc` samples is found in *UNICOS Resource Administration*, Cray Research publication SG–2302.

The `sdcx` command accepts the following options:

`-i` *interval*	Specifies the data sampling interval. `interval` may be in the form *xx*`h`*yy*`m`*zz*`s` (*xx* hours, *yy* minutes, *zz* seconds) or *zz* seconds. By default, `sdcx` uses the interval specified to `sdc`.
`-n` *sample_count*	Specifies the number of intervals for which samples are to be taken. By default, `sdcx` uses the number of samples specified to `sdc`.

NOTES

The `sdc` and `sdcx` commands can be executed only by a user who has permission to read the `/unicos` and `/dev/kmem` files.

If you will not reconfigure the system during the sampling interval, use `sdcx` instead of `sdc` to collect the data samples. `sdcx` takes less time to execute than `sdc`, because it does not generate and compile `C` code. `sdc` always generates and compiles `sdcx`; thus, it may take significant time to execute on a busy system.

The `sdc` command creates a new session and forks a child process to perform the actual data collection and does not wait for the child process to terminate. Thus, the child process can continue to write data to the data file even though the parent process has exited. The name of the child process is generated by the `tmpnam`(3C) routine.

EXAMPLES

Example 1: The following example gathers data only for the items listed in the file `disk_data`. Twelve samples are collected at a rate of one sample every 5 minutes. The data is written to the `dcf` file.

```
    $ sdc -R disk_data -n 12 -i 5m dcf
```

Example 2: In the following example, `sdc` writes a header record to the `dcf` file and saves the `sdcx` binary in the `/usr/adm/tsar` directory. `sdcx` then is invoked to gather 12 data samples. The samples are collected at a rate of one sample every 5 minutes (300 seconds). Last, the data is appended to the `dcf` file.

```
$ sdc -c /usr/adm/tsar dcf
$ /usr/adm/tsar/sdcx -i 300s -n 12 >> dcf &
```

SEE ALSO

`tsar`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`sdconf` – Controls the state of a disk drive

SYNOPSIS

`/etc/sdconf` [*path control* `...`]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `sdconf` command controls the state of a disk drive. It is based on the `pddconf`(8) command, but allows all physical disk devices (`xdd`, `qdd`, `hdd`, and `pdd`) to be controlled. The `pddconf`(8) only works with `pdd` devices.

If no arguments are given on the command line, a summary display of the current state of all drives is shown.

The `sdconf` command accepts the following command line arguments:

path The I/O path of the device to be controlled. For `pdd`, `qdd`, and `xdd` devices, *path* takes the form *iopath.unit*. For network devices, *path* takes the form *iopath.unit.ifield*. `pdd` and `qdd` devices are controlled through per-device control nodes that reside in `/dev/ddd/`*path*. `xdd` devices are controlled through the single control node `/dev/ddd/xdiag` (IOS model E systems) or `/dev/xdd/xdiag` (GigaRing-based systems). `hdd` devices are controlled through the single control node `/dev/ddd/hdd` (IOS model E systems). Path may also be the token `all`; see the `qsort` control.

control The control operation to perform on *path*, followed by per-control arguments. The following controls are supported:

`rw`	Sets the physical device mode to read/write.
`ronly`	Sets the physical device mode to read-only. Read requests are permitted, writes are returned with an *errno* of `EIO`. Any mounted file system slice(s) residing on the device are marked as not available for allocation.
`noall`	Sets the physical device mode to nonallocatable. Any mounted file system slices residing on the device are marked as not available for allocation.
`up`	Sets the physical device state to up. Any previous setting of mode is still in affect.
`down`	Sets the physical device state to down and terminates all queued I/O requests with errors. Any mounted file system slices residing on the device are marked as not available for allocation.

`spinup`	Spinup a disk drive. Valid only for DD-60 disk drives, spinup is also done during device open.
`spindown`	Spindown a disk drive. Valid only for DD-60 disk drives.
`pripath`	Change disk device to primary path.
`altpath`	Change disk device to alternate path if available.
`reset`	Reset disk statistics.
`disable` *spindle*	Disable a given spindle of a disk array. A spindle number 0-4 must be given.
`qsort`	Turn on the disk queue sorting algorithm in the disk driver for the specified device; the device flag is on by default. Specifying `all` as the path turns on a global flag that enables queue sorting for all disks in the system for which the disk queue sorting algorithm is enabled; this global flag is off by default.
`noqsort`	Turn off the disk queue sorting algorithm for the indicated device; the device flag is on by default. Specifying all as the device turns off a global flag, disabling queue sorting for all disks in a system; this global flag is off by default.
`autoswitch on|off`	Enable/disable autopath switching during error recovery. The system will not switch to the alternate path if the switch is disabled. The device must be configured with an alternatepath path to work.
`racerron`	Routes Recovered Disk Error messages to be printed on the console. This is the default.
`racerroff`	Stops routing Recovered Disk Error messages to the console.

Not all controls are applicable for all device types. The following table lists valid device type controls:

Control	`xdd`	`pdd`	`hdd`	`qdd`
`rw`	Yes	Yes	Yes	Yes
`ronly`	Yes	Yes	Yes	Yes
`noall`	Yes	Yes	Yes	Yes
`up`	Yes	Yes	Yes	Yes
`down`	Yes	Yes	Yes	Yes
`spinup`	No	Yes	No	Yes
`spindown`	No	Yes	No	Yes
`pripath`	Yes	Yes	No	Yes
`altpath`	Yes	Yes	No	Yes
`reset`	No	Yes	No	Yes
`disable`	No	Yes	No	Yes

Control	xdd	pdd	hdd	qdd
qsort	No	Yes	No	Yes
noqsort	No	Yes	No	Yes
autoswitch	No	Yes	No	Yes
racerron	No	Yes	No	Yes
racerroff	No	Yes	No	Yes

If no path or control arguments are given, the following display is generated:

```
iopth.un[.ifld] type  drvr state mode  altp  flags  wstrm   rstrm  qon  q’ed
--------------- ----- ---- ----- ---- ------ ----- ------- ------- --- -------
   0230.0        DD61  pdd  up    rw    ----  01100 ------- ------- yes       0
   0230.1        DD61  pdd  up    rw    ----  01100 ------- ------- yes     169
   0232.0        DD60  pdd  up    rw    ----  01000 ------- ------- yes       0
   0232.1        DD62  pdd  up    rw    ----  01000 ------- ------- yes       0
   0234.2        DD61  pdd  up    rw    ----  01100 ------- ------- yes       0
   0234.3        DD62  pdd  up    rw    ----  01000 ------- ------- yes       0
   0236.3        DD61  pdd  up    rw    ----  01000 ------- ------- yes       0
   0236.4        DD61  pdd  up    rw    ----  01000 ------- ------- yes       0
   0236.5        DD61  pdd  up    rw    ----  01100 ------- ------- yes       0
   0236.6        DD61  pdd  up    rw    ----  01000 ------- ------- yes       0
```

The following are definitions of the headings in the preceding table:

iopth.un[.ifld]	Device I/O path having the format *iopath.unit* for pdd, qdd, and xdd devices, *iopath.unit.ifield* for network devices.
type	Device type.
drvr	Driver type.
state	Device state - driver dependent.
mode	Device mode (read-write (rw), read-only (ro), or no allocate (na)).
altp	Device alternate path if available.
flags	Device flags - driver dependent.
wstrm	Device write stream control flags (for disk arrays).
rstrm	Device read stream control flags (for disk arrays).
qon	Indicates that sort is enabled for a particuliar device.
q’ed	Total number of requests that have been sorted since the system was booted, or since device statistics were reset.

NOTES

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

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

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

SEE ALSO

`pddconf`(8), `sdstat`(8)

`ioctl`(2) in *UNICOS/mk System Calls Reference Manual*, Cray Research publication SR–2612

NAME

sdstat – Displays information about disk device I/O

SYNOPSIS

```
/etc/sdstat
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `sdstat` command displays information about disk device activity. It is based on the `pddstat`(8) command, but it displays information about all disk types (`pdd`, `hdd`, `qdd`, and `xdd`).

Currently, the `sdstat` output is limited to a summary display having the following format:

```
                                                 Sectors Moved    Errors
iopth.un[.ifld] type  drvr state mode req  reads    writes   rec  unr
--------------- ----- ---- ----- ---- --- -------- -------- ---- ----
  0130.20.13    HD32  hdd  up    rw     0        0        0    0    0
  0130.40.25    HD32  hdd  up    rw     0      512        0    0    0
  0230.0        DD60  pdd  up    rw     0       26        2    0    0
  0230.3        DD62  pdd  up    rw     0     4468     4377    0    0
  0230.4        DD60  pdd  up    rw     0       23        2    0    0
  0230.5        DD60  pdd  up    rw     0        4        0    0    0
  0232.0        DD60  pdd  up    rw     0    18587     2732    0    0
  0232.1        DD60  pdd  up    rw     0      868        4    0    0
  0232.2        DD60  pdd  up    rw     0     1054        2    0    0
  0232.3        DD62  pdd  up    rw     0      122        2    0    0
  0234.0        DD60  pdd  up    rw     0      320       38    0    0
  0234.1        DD60  pdd  up    rw     0     1157     1099    0    0
  0234.2        DD60  pdd  up    rw     0     2052      115    0    0
  0234.3        DD62  pdd  up    rw     0       92        2    0    0
  0236.2        DD62  pdd  up    rw     0     2115        1    0    0
  0236.3        DD61  pdd  up    rw     0     4343        0    0    0
  0236.4        DD60  pdd  up    rw     0     4737      282    0    0
  0236.5        DD60  pdd  up    rw     0     6806     1936    0    0
```

The following are definitions of the headings in the preceding table:

`iopth.un[.ifld]`	Device I/O path having the format *iopath.unit* for `pdd`, `qdd`, and `xdd` devices, *iopath.unit.ifield* for network devices. The path will be suffixed by `a` if the device has an alternate path, and the alternate path is active.
`type`	Device type.

`drvr`	Driver type.
`state`	Device state - driver dependent.
`mode`	Device mode (read-write (`rw`), read-only (`ro`), or no allocate (`na`)).
`req`	Number of currently outstanding requests.
`reads`	Number of sectors read (sector size is device type dependent).
`writes`	Number of sectors written.
`rec`	Number of recovered read and/or write errors.
`unr`	Number of unrecovered read and/or write errors.

SEE ALSO

`sdconf`(8), `pddstat`(8)

`ioctl`(2) in *UNICOS/mk System Calls Reference Manual*, Cray Research publication SR–2612

NAME

`secded` – Memory error correction interface SECDED hardware

SYNOPSIS

`/etc/secded` [`-u` *uncmax*] [`-n` *cormax*] [`-w` *mint*] [`-i` *disint*]
`/etc/secded` [`-l` *umelife*] [`-m` *umemax*] [`-t` *umedown*]
`/etc/secded -a` *address* `-d` *data*
`/etc/secded -a` *address* `-c` *check*
`/etc/secded -a` *address* `-c` *check* `-d` *data*

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `secded` command allows any memory location to be written with any pattern of bits by using the SECDED maintenance hardware. The command also provides access to the memory error processing control parameters and control of CPU downing due to memory errors.

The `secded` command accepts the following options:

`-a` *address* Specifies absolute octal address, *address*, of the word of memory to be altered. If you specify this option, you must also specify the `-d` or `-c` option.

`-d` *data* Specifies data bits to be toggled in the word of memory (interpreted as an octal mask). For example, to toggle bit 2^4, use the following command line:

```
/etc/secded -a address -d 20
```

Either the `-d` option or the `-c` option must be specified if the `-a` option is specified.

`-c` *check* Specifies check bits to be toggled in the word of memory (interpreted as an octal mask). For example, to toggle bit 2^4, use the following command line:

```
/etc/secded -a address -c 20
```

The mask is limited to the 8 check bits that exist. Either `-d` or `-c` must be specified if the `-a` option is specified.

`-u` *uncmax* Specifies the number of uncorrectable memory errors that the system will allow before forcing a panic. The default is 64.

`-n` *cormax* Specifies the number of single-bit errors that can occur in *mint* seconds with no intervals of longer than *mint*/*cormax* seconds without an error before single-bit errors are turned off, for all user processes, for *disint* seconds. The default for *cormax* is `16`. If `-1` is specified, single-bit interrupts are always disabled.

`-w` *mint*	Specifies the number of seconds that *cormax* single-bit errors can occur in before disabling single-bit errors. The default is 5.
`-i` *disint*	Specifies the number of seconds to keep memory errors disabled. The default is 300. If `-1` is specified, single-bit errors are permanently disabled.
`-l` *umelife*	Specifies the lifetime (in seconds) of uncorrectable memory errors. The default is 86400 (24 hours). This is related only to the downing of CPUs due to uncorrectable memory errors and not to the `-u` option.
`-m` *umemax*	Specifies the maximum number of uncorrectable memory errors before a CPU is downed. The default is 0. A value of zero disables downing of CPUs due to uncorrectable memory errors.
`-t` *umedown*	Specifies the time (in seconds) that a CPU will remain down after uncorrectable memory errors. The default is 0. A value of zero means that the CPU will remain down until restarted by the /etc/cpu command.

NOTES

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

Active Category	**Action**
`system`	Allowed to use this command.

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

BUGS

On Cray PVP systems, error maintenance switches must be on for the CPUs used to execute the `secded` command. Software cannot determine whether or not error maintenance is enabled.

Because of its ability to write memory errors anywhere in physical memory, the `secded` command should be used with caution. For instance, if a double-bit or multibit memory error is written in memory and the UNICOS kernel reads that word next (as opposed to a user process), the kernel panics. Generally, use of the SECDED maintenance feature should be done with the mainframe in single-user mode.

FILES

`/dev/secded`	SECDED maintenance special device

SEE ALSO

`cpu`(8), `errdemon`(8), `errpt`(8)

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

General UNICOS System Administration, Cray Research publication SG–2301

NAME

`sendmail`, `newaliases`, `mailq` – Sends mail over the Internet

SYNOPSIS

`/usr/lib/sendmail` [`-B`*type*] [`-ba`] [`-bd`] [`-bi`] [`-bm`] [`-bp`] [`-bs`] [`-bt`] [`-bv`] [`-C`*file*] [`-d`*X*] [`-F`*fullname*] [`-f`*name*] [`-h`*N*] [`-n`] [`-o`*x value*] [`-p`*protocol*] [`-q`[*time*]] [`-qI`*substr*] [`-qR`*substr*] [`-qS`*substr*] [`-r`*name*] [`-t`] [`-v`] [`-X`*logfile*] *address* ...

`/usr/bin/newaliases`

`/usr/bin/mailq` [`-v`]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `sendmail` command sends a message to one or more recipients, routing the message over whatever networks are necessary. It does internetwork forwarding, as necessary, to deliver the message to the correct place.

The `sendmail` command is not intended as a user interface routine. Other programs provide user-friendly front ends; `sendmail` is used only to deliver preformatted messages.

With no options specified, `sendmail` reads its standard input up to an end-of-file or a CONTROL-d or a line with a single dot (`.`) and sends a copy of the message found there to all of the addresses listed. It determines the network(s) to use based upon the syntax and contents of the addresses.

The `sendmail` command looks up local addresses in a file and determines aliases. To prevent aliases from being determined, precede the address with a backslash. Usually, the sender is not included in any alias expansions (for example, if `john` sends to `group`, and `group` includes `john` in the expansion, the letter will not be delivered to `john`).

The `sendmail` command accepts the following options:

`-B`*type*	Sets the body type to *type*. Current legal values are 7BIT or 8BITMIME.
`-ba`	Goes into ARPAnet mode. All input lines must end with a carriage return-line feed (CR-LF), and all messages are generated with a CR-LF at the end. Also, the `From:` and `Sender:` fields are examined for the name of the sender.
`-bd`	Runs as a daemon. `sendmail` will fork and run in background, listening on socket 25 for incoming SMTP connections.
`-bi`	Initializes the alias database.
`-bm`	Delivers mail in the usual way (default).
`-bp`	Prints a listing of the queue.

`-bs`	Uses the SMTP protocol as described in RFC 821 on standard input and output. This option implies that all operations are compatible with SMTP.
`-bt`	Runs in address test mode, which reads addresses and shows the steps in parsing. It is used for debugging configuration tables. This mode is used for debugging configuration tables.
`-bv`	Verifies names only; does not try to collect or to deliver a message. This option is usually used for validating user or mailing lists.
`-C`*file*	Uses an alternate configuration file. `sendmail` refuses to run as root if an alternate configuration file is specified.
`-d`*X*	Sets debugging value to *X*.
`-F`*fullname*	Sets the full name of the sender.
`-f`*name*	Sets the name of the sender of the mail. Only special users (`root`, `daemon`, and `network`) can use the `-f` option, unless the sender you are trying to set has the same status as yours.
`-h`*N*	Sets the hop count to *N*. The hop count is incremented each time the mail is processed. When it reaches the limit (*N*), the mail is returned with an error message, the victim of an aliasing loop. If not specified, `Received:` lines in the message are counted.
`-n`	Prevents alias determination.
`-o`*x value*	Sets option *x* to the specified *value*. Options are described below.
`-p`*protocol*	Sets the protocol used to receive the message. This can be a simple protocol, such as `UUCP`, or a protocol and hostname, such as `UUCP:ucbvax`.
`-q`[*time*]	Processes saved messages in the queue at given intervals. If you omit *time*, the queue is processed once. *time* is specified in seconds (`s`), minutes (`m`), hours (`h`), days (`d`), and weeks (`w`) (for example, `-q1h30m` or `-q90m` both set the *timeout* to 1 hour, 30 minutes). If *time* is specified, `sendmail` will run in background. This option can be used safely with `-bd`.
`-qI`*substr*	Limits processed jobs to those containing *substr* as a substring of the queue id.
`-qR`*substr*	Limits processed jobs to those containing *substr* as a substring of the recipients.
`-qS`*substr*	Limits processed jobs to those containing *substr* as a substring of the sender.
`-r`*name*	Indicates an alternate and obsolete form of the `-f` option.
`-t`	Reads message for recipients, and scans `To:`, `Cc:`, and `Bcc:` lines for recipient addresses. The `Bcc:` line is deleted before transmission. Any addresses in the argument list are suppressed.
`-v`	Goes into verbose mode. Alias expansions are announced, and so on.
`-X` *logfile*	Logs all traffic in and out of mailers in the indicated log file. This option should be used only as a last resort for debugging mailer bugs. This option will log large amounts of data very quickly.
address ...	Specifies the address to which the mail is sent.

Options

You can set several processing options. Usually, these are used only by a system administrator. Options can be set either on the command line by using the `-o` option or in the configuration file. The processing options are as follows:

`A` *file*	Uses an alternate alias file.
`b` *nblocks*	Indicates the minimum number of free blocks needed on the spool filesystem.
`c`	Indicates that for mailers that are considered expensive to which to connect, `sendmail` does not initiate immediate connection. This requires queueing.
`C` *N*	Checkpoints the queue file after every *N* successful deliveries (the default value is 10). This options avoids excessive duplicate deliveries when sending to long mailing lists and the delivery is interrupted by a system crash.
`d` *x*	Sets the delivery mode to *x*. Delivery modes are `i` for interactive (synchronous) delivery, `b` for background (asynchronous) delivery, and `q` for queue only; that is, actual delivery is done the next time the queue is run.
`D`	Tries to rebuild the alias database automatically if necessary.
`e` *x*	Sets error processing to mode *x*. Valid modes are `m` to mail back the error message, `w` to write back the error message (or mail it back if the sender is not logged in), `p` to print the errors on the terminal (default), and `q` to throw away error messages (only exit status is returned). If the text of the message is not mailed back by modes `m` or `w` and if the sender is local to this machine, a copy of the message is appended to the `dead.letter` file in the sender's home directory.
`f`	Saves UNIX-style `From` lines at the front of messages.
`G`	Matches local mail names against the `GECOS` portion of the password file.
`g` *N*	Specifies the default group ID to use when calling mailers.
`H` *file*	Specifies the SMTP help file.
`h` *N*	Specifies the maximum number of times a message is allowed to hop before it is considered to be in a loop.
`i`	Does not take dots on a line by themselves as a message terminator.
`j`	Sends error messages in MIME format.
`K`*timeout*	Sets the connection cache timeout.
`k`*N*	Sets the connection cache size.
`L` *n*	Specifies the log level.
`l`	Pays attention to the `Errors-To:` header.
`m`	Sends to ‘‘me’’ (the sender) also if the sender is in an alias expansion.

`n`	Validates the right hand side of aliases during a `newaliases`(1) command.
`o`	If set, specifies the message may have old style headers. If not set, the message is guaranteed to have new style headers (that is, commas instead of spaces between addresses). If set, an adaptive algorithm that correctly determines the header format is used.
`Q` *queuedir*	Selects the directory in which to queue messages.
`S` *file*	Saves statistics in the specified file.
`s`	Instantiates the queue file, even under circumstances where it is not strictly necessary. This provides safety against system crashes during delivery.
`T` *time*	Sets the *timeout* on messages in the queue to the specified time. After delivery has failed for this amount of time, they are returned to the sender; the default is 3 days.
`t` *stz, dtz*	Sets the name of the time zone.
`u` *N*	Sets the default user ID for mailers.
`Y`	Forks each job during queue runs. This option can be convenient on memory-poor machines.
`7`	Strips incoming messages to seven bits.

If the first character of the user name is a vertical bar (`|`), the remainder of the user name is used as the name of the program to which to send the mail. It might be necessary to enclose the name of the user in quotation marks to prevent `sendmail` from suppressing the blanks from between arguments. For example, a common alias is:

```
msgs: "|/usr/bin/msgs -s"
```

Aliases can also have the syntax `:include:filename`, asking `sendmail` to read the named file for a list of recipients. For example, the alias:

```
poets: ":include:/usr/local/lib/poets.list"
```

would read `/usr/local/lib/poets.list` for the list of addresses making up the group.

The `sendmail` command returns an exit status that describes what it did. The codes are defined in the `/usr/include/sysexits.h` include file.

Code	**Description**
`EX_OK`	Successful completion at all addresses
`EX_NOUSER`	User name not recognized
`EX_UNAVAILABLE`	Necessary resources are not available
`EX_SYNTAX`	Syntax error in address
`EX_SOFTWARE`	Internal software error, including bad arguments
`EX_OSERR`	Temporary operating system error, such as cannot perform fork process
`EX_NOHOST`	Host name not recognized

`EX_TEMPFAIL`	Message could not be sent immediately, but it was queued

If invoked as `newaliases`, `sendmail` rebuilds the alias database. If invoked as `mailq`, `sendmail` prints the contents of the mail queue.

NOTES

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

Privilege Text	**Action**
`daemon`	Allowed to start the `sendmail` daemon.
`dmstart`	Allowed to start the `sendmail` daemon.

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

Active Category	**Action**
`system`, `secadm`	Allowed to start the `sendmail` daemon.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to start the `sendmail` daemon.

For the root user's `.forward` file to be processed, the file must have the other-readable mode bit set.

All undefined command-line switches are silently ignored. No error message is printed to notify the user of the invalid command-line switch.

All unknown options are silently accepted, but ignored. No error message is printed to notify the user of the invalid option.

The name of the local host should be fully qualified in the `/etc/hosts` file. Otherwise, a delay will occur when initiating `sendmail`.

For information about `/usr/lib/sendmail.cf`, see the *UNICOS Networking Facilities Administrator's Guide*, Cray Research publication SG–2304. The documentation for `sendmail` is also available in the UNIX4.3 BSD System Manager's Manual, University of California at Berkeley.

BUGS

The `sendmail` command converts blanks in addresses to dots. This is incorrect according to the removed ARPAnet mail protocol RFC 733 (NIC 41952), but it is consistent with the new protocols (RFC 822).

FILES

The following path names are site-specific, except for `/usr/lib/sendmail.cf`. You must specify the path names in `/usr/lib/sendmail.cf`.

`/usr/lib/aliases`	Raw data for alias names
`/usr/lib/aliases.dir`	Alias database

`/usr/lib/aliases.pag`	Alias database
`/usr/lib/sendmail.cf`	Configuration file
`/usr/lib/sendmail.hf`	Help file
`/usr/spool/mqueue/sendmail.st`	Collected statistics
`/usr/spool/mqueue/*`	Temporary files
`/etc/sendmail.pid`	Process ID of the daemon
`$HOME/dead.letter`	Copy of message

SEE ALSO

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

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

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

`ia_mlsuser`(3C) for information about users' mandatory access control (MAC) attributes in the *UNICOS System Libraries Reference Manual*, Cray Research publication SR–2080

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

DARPA Internet Request For Comments RFC 819, RFC 821, and RFC 822

Allman, E. and M. Amos, ''SENDMAIL Revisited,'' *Proceedings of the Summer 1985 USENIX Conference*. USENIX Association

NAME

`setacid` – Sets default accounting ID

SYNOPSIS

`/usr/lib/acct/setacid` [`-v`] [`-z`] *directory*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `setacid` command assigns a default accounting ID to each file in a file system, or directory tree.

The `setacid` command accepts the following options and operand:

`-v`	Sets verbose mode on.
`-z`	Sets a file's account ID to the file owner's default account ID. This is done only on files that have an account ID of 0.
directory	Starting directory in which `setacid` starts when making changes.

NOTES

When running `setacid` on a directory that has account IDs set, always use the `-z` option to assure that the owner's default account ID is set for that directory.

FILES

`/etc/acid`	Account ID information file
`/etc/udb`	User database control information file

SEE ALSO

`diskusg`(8)

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

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

NAME

`setdate` – Sets the system date from the IOS

SYNOPSIS

`/etc/setdate`

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `setdate` command uses a date packet passed into central memory by the I/O subsystem (IOS) at boot time to determine the time of day.

The `setdate` command also sets the system time-zone information into the kernel extended time-zone structure as described in the `sys/time.h` file. This is usually done at boot time, but `setdate` may be run at any time to set or update this information. The time-zone information is obtained from the `TZ` environment variable. When `setdate` detects that the time-zone information has already been set, it updates the time-zone information as required, but does not modify the system time.

The `init`(8) command should execute `setdate` at boot time. To execute the procedure, use the following `inittab` command:

```
tz::sysinit:TZ=CST6CDT
sd::sysinit:/etc/setdate 1 >/dev/console 2>&1
```

For the time to be correct, you must change the time-zone string (`TZ=CST6CDT`) in the `inittab` file (see `inittab`(5)). The first line is usually set prior to the setdate in `/etc/inittab`.

Only an appropriately authorized user can set the system date.

NOTES

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

Active Category	**Action**
`system`, `secadm`	Allowed to set the system date.
`sysadm`	Allowed to set the system date. Shell-redirected I/O is subject to security label restrictions.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to set the system date.

SEE ALSO

`init`(8)

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

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

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

NAME

`seterr` – Set maximum user error counts

SYNOPSIS

`/etc/seterr` [`-e` *err*] [`-o` *ore*] [`-p` *pre*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `seterr` command allows values to be set in the kernel `maxerrint` table. These values specify the maximum number of Error Exits, Operand Range Errors or Program Range Errors that a process may accumulate before being aborted. The error counts are maintained on a per-CPU basis and are cleared each time a process is connected to a CPU. If any of the error counts exceeds the value in the `maxerrint` table during a connection interval, the connected process is terminated with a `SIGKILL` signal. The `seterr` command will print the values in the `maxerrint` table after changes specified by the parameters, if any, have been made.

The `seterr` command accepts the following options:

`-e` *err*	Specifies the maximum allowable count of Error Exits, *err*.
`-o` *ore*	Specifies the maximum allowable count of Operand Range Errors, *ore*.
`-p` *pre*	Specifies the maximum allowable count of Program Range Errors, *pre*.

NOTES

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

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

If the `PRIV_SU` configuration option is enabled, the super user is allowed to alter system values.

MESSAGES

`seterr` sends messages about parameter values that cannot be converted correctly, or about problems reading or setting the values in the system `maxerrint` table.

FILES

`/dev/cpu/any`	CPU control device

SEE ALSO

`cpu`(8)

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

General UNICOS System Administration, Cray Research publication SG–2301

NAME

`setfs` – Changes dynamic information in file system super block

SYNOPSIS

`/etc/setfs` [`-B` *bf*] [`-A` *bu*] [`-L` *sl*] [`-U` *sl*] [`-a` *al*] [`-b` *flaw_list*] [`-c`] [`-i`]
[`-s` *arbiter:semaphore_count*] [`-z`] [`-`*flag*] *special*

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `setfs` command makes changes to the file system super block without requiring you to make an entire new file system. You must unmount the file system before using this command to make alterations.

The `setfs` command accepts the following options and argument:

`-B` *bf*	Big file size (in bytes) for the file system.
`-A` *bu*	Big file allocation units (in 512-word blocks) used when a file exceeds the "Big file" size.
`-L` *sl*	Lower-level security for this file system.
`-U` *sl*	Upper-level security for this file system.
`-a` *al*	Allocation strategy for this file system. The values for *al* may be one of the following:

`rrd1`	Handles the first level directories in a round-robin manner.
`rrda`	Handles all directories in a round-robin manner.
`rrf`	Handles all files in a round-robin manner. Directories and inodes are allocated in the first partition of the multiple partition file system whenever possible.

`-b` *flaw_list* Reads a list of decimal tuples (pairs of numbers) from the file *flaw_list* that specifies the starting block number and the number of blocks in the bad sections of the file system. If the name of the file is –, `setfs` will read standard input.

The `setfs` command avoids using the specified areas, when possible: If information for which the location is critical (such as the super block) falls within the specified area, that area of the disk is used; if the specified area is not a critical area, the area specified is reserved when `mkfs` is executed and is not allocated as new file space. Reserving bad blocks with the `-b` option avoids using the bad block areas on disk that are remapped to spare cylinders. Using the spare cylinders may case an I/O performance degradation when reading or writing the bad block area; the tradeoff can be disk fragmentation caused by bad block avoidance, if there are a large number of flaws on a particular disk.

Blokcs that are already reserved when `setfs` declares them as suspect could still be in use by files. This does not affect prior allocation of these blocks.

`-c`	Clears any existing flaw list from the file system. Specifying both the `-b` and the `-c` options replaces an old flaw list with the flaw list specified by the file *flaw_list*.
`-i`	Toggles inode allocation preference placement.
`-s` *arbiter:semaphore_count*	Used to change `SFS` (Shared File System) file system parameters. The argument to the `-s` option provides the *arbiter* name or number, and the number of semaphores from that arbiter to be assigned to this file system at mount time. The *arbiter* name or number must match one of the valid configuration entries in the `/etc/config/sfs` configuration file. With the `-s` option, an unmounted shared file system may be changed to an `NC1FS` file system by specifying a *semaphore_count* of zero. Similarly, an unmounted `NC1FS` file system may be changed to a shared file system with the `-s` option.
`-z`	Toggles the setting of the file system panic flag. When enabled, the system panics on file system errors encountered. When disabled, the error is logged to `/dev/fslog` and handled by the `fslogd`(8) daemon.
`-`*flag*	Sets a test flag for the file system. *flag* can be 1, 2, or 3.
special	Specifies the name of the block special device on which the file system exists. If no options other than *special* are specified, the `setfs` command displays the current values in effect for the file system.

NOTES

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

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

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

When using either the `-b` or `-c` option, `fsck`(8) must be run on the file system.

SEE ALSO

General UNICOS System Administration, Cray Research publication SG–2301

NAME

`setpal` – Sets privilege assignment list (PAL) category entries of a file

SYNOPSIS

`setpal` [`-f`] [`-p` *privlist*] [`-t` *privtext*] *catlist* *files*. . .
`setpal -d` *catlist* *files*. . .
`setpal -c` *files*. . .

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `setpal` command adds or deletes category entries from the privilege assignment lists (PALs) of the specified regular file(s).

Only one PAL entry may exist for a specified category.

The `setpal` command accepts the following options and operands:

`-c`	Clears the PAL such that only the `other:PRIV_NULL:TEXT_NULL` entry remains.
`-d`	Deletes the PAL entries of categories specified in *catlist*. Specified categories that do not exist in the PAL are ignored. The `other` entry cannot be deleted. Attempting to delete the `other` category entry clears the privilege and privilege text values.
`-f`	Forces the overwrite of any PAL entries for privileges specified in *catlist*. If this option is not specified, and a PAL entry already exists for a category specified in *catlist*, the `setpal` command returns an unsuccessful error status.
`-p`	Specifies the set of privileges to associate with PAL entries of the specified categories. Privileges are specified in *privlist*. If this option is not specified, the privilege set associated with each PAL entry is `PRIV_NULL`.
`-t`	Specifies the privilege text to associate with PAL entries of the specified categories. Privilege text is specified by *privtext*. If this option is not specified, the privilege text associated with each PAL entry is `TEXT_NULL`.
catlist	A character string that represents one or more category names (for example, `secadm`). Multiple category names must be separated by commas, with no intervening white space.
privlist	A character string that represents one or more privilege names (for example, `PRIV_MAC_READ`). Multiple privilege names must be separated by commas, with no intervening white space. The `PRIV_ALL` character string represents the list of all privileges.
privtext	A sequence of one to eight alphanumeric characters, or the word `TEXT_NULL`, that represents privilege text.
files	Represents the name(s) of the file(s) whose PALs are to be changed.

NOTES

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

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

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

EXIT STATUS

The `setpal` command exits with one of the following values:

Value	**Description**
0	The specified file's PAL was successfully updated.
1	A badly formed option or an option that was not valid was supplied.
2	When multiple files are supplied, both failure and success occurred.
4	The PAL(s) for the specified file(s) could not be displayed.

EXAMPLES

Example 1: The following example clears the contents of the PAL associated with `testfile`, such that it contains only the PAL entry `other:PRIV_NULL:TEXT_NULL`:

```
$ setpal -c testfile
```

Example 2: The following example creates a `sysadm:PRIV_FOWNER,PRIV_KILL:adm` entry in PAL associated with `testfile`, overwriting a `sysadm` category entry if it exists:

```
$ setpal -f -p PRIV_FOWNER,PRIV_KILL -t adm sysadm testfile
```

Example 3: The following example creates a `secadm:PRIV_NULL:sec` entry in the PAL associated with `testfile`, overwriting a `sysadm` category if it exists:

```
$ setpal -f -t sec secadm testfile
```

Example 4: The following example creates a `secadm:PRIV_NULL:TEXT_NULL` entry in the PAL associated with `testfile`, overwriting a `secadm` category entry if it exists:

```
$ setpal -f -t TEXT_NULL secadm testfile
```

An alternative way of creating a `secadm:PRIV_NULL:TEXT_NULL` entry in the PAL associated with `testfile`, overwriting a `secadm` category entry if it exists, is as follows:

```
$ setpal -f secadm testfile
```

Example 5: The following example creates a `secadm:PRIV_ALL:TEXT_NULL` entry in the PAL associated with `testfile`, overwriting the `secadm` and `system` category entries if they exist:

```
$ setpal -f -p PRIV_ALL secadm,system testfile
```

Example 6: The following example deletes the `secadm` category entry from the PAL associated with `testfile`:

```
$ setpal -d secadm testfile
```

SEE ALSO

`getpal`(8)

General UNICOS System Administration, Cray Research publication SG–2301

NAME

`setprivs` – Modifies the file privilege sets of a file

SYNOPSIS

`setprivs` [`-a`] [`-f`] [`-s`] *privlist files*. . .

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `setprivs` command updates the allowed, forced, and set-effective privilege sets of the specified regular files.

Only an appropriately authorized user can use this command.

The `setprivs` command accepts the following options and operands:

`-a`	Updates the allowed privilege set to the value specified in *privlist*.
`-f`	Updates the forced privilege set to the value specified in *privlist*.
`-s`	Updates the set-effective privilege set to the value specified in *privlist*.
privlist	A character string that represents one or more privilege names (for example: `PRIV_MAC_READ`). Multiple privilege names must be separated by commas with no intervening white space. To clear privileges, the value of *privlist* should be the `PRIV_NULL` character string. The `PRIV_ALL` character string represents the list of all privileges.
files	Represents the name(s) of the file(s) whose privilege set(s) are to be updated.

If no options are specified, then all privilege sets are updated.

NOTES

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

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

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

EXIT STATUS

The `setprivs` command exits with one of the following values:

Value	**Description**
0	The specified privilege state was successfully updated.
1	A badly formed option or an option that was not valid was supplied.
2	When multiple files are supplied, failure and success both occurred.
4	The privilege state for the specified file(s) could not be updated.

EXAMPLES

Example 1: The following example sets the forced and set-effective privilege sets of `testfile` to `PRIV_MAC_READ,PRIV_MAC_WRITE:`

```
$ setprivs -fs PRIV_MAC_READ,PRIV_MAC_WRITE testfile
```

Example 2: The following example sets the forced and set-effective privilege sets of `testfile` to contain all of the defined privileges:

```
$ setprivs -fs PRIV_ALL testfile
```

Example 3: The following example sets the allowed, forced, and set-effective privilege sets of `testfile` to `PRIV_MAC_READ`:

```
$ setprivs -fsa PRIV_MAC_READ testfile
```

An alternative way to set the allowed, forced, and set-effective privilege sets of `testfile` to `PRIV_MAC_READ` is shown in the following example:

```
$ setprivs PRIV_MAC_READ testfile
```

Example 4: The following example clears the allowed privilege set of `testfile`:

```
$ setprivs -a PRIV_NULL testfile
```

Example 5: The following example clears all privilege sets of `testfile`:

```
$ setprivs -a -f -s PRIV_NULL testfile
```

An alternative way to clear all privilege sets of `testfile` is shown in the following example:

```
$ setprivs PRIV_NULL testfile
```

SEE ALSO

`getprivs`(8)

NAME

`sfsd` – Initializes and monitors the External Semaphore Device

SYNOPSIS

`/etc/sfsd` [`-v`] [`-F`]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `sfsd` command is used to initialize and monitor the External Semaphore Devices.

The `sfsd` command replaces the `esdaemon` command.

The `sfsd` command uses the special devices `/dev/sfsd...`, `/dev/smp...` and `/dev/smnt...` to interface to the respective device drivers. The actual names of the special devices is provided in the `/etc/config/sfs` Shared File System (SFS) configuration file. Usually, `sfsd` tries to determine the necessity of clearing and initializing the External Semaphore Device by scanning the state of all the semaphores. The presence of any errors usually indicates a *power-up* condition, and initialization will be done. Use the `-F` option to force unconditional initialization.

Executing the `sfsd` command requires super-user permissions.

`-v`	(Verbose) Causes additional information to be displayed.
`-F`	(Force) Causes the unconditional clearing and initialization of the External Semaphore Device.

FILES

`/etc/config/sfs`	The Shared File System configuration file.

SEE ALSO

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

NAME

`shradmin` – Changes and displays costs for fair-share scheduler usage calculations

SYNOPSIS

`/etc/shradmin` [`-b` *bio*] [`-c` *clicks*] [`-D` *h1*`,`*h2*] [`-E` *half_life*] [`-F` *flags*] [`-G` *maxgroups*] [`-K` *half-life*] [`-m` *click*] [`-n` *interval*] [`-P` *maxpri*] [`-p` *procs*] [`-Q` *maxupri*] [`-R` *delta*] [`-S` *maxusers*] [`-s` *syscall*] [`-t` *tick*] [`-U` *maxusage*] [`-v`] [`-X` *maxushare*] [`-Y` *mingshare*] [`-y` *tio*] [`-Z` *sharemin*] [*percent*]

`/etc/shradmin -r`

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `shradmin` command changes the costs associated with the usage calculations for the fair-share scheduler.

The `-r` option must be used by itself, because the order of option processing by `shradmin` is controlled by the `getopt`(3C) library routine).

The `shradmin` command accepts the following options and operand. You must specify at least one of these options; otherwise, `shradmin` does not execute.

`-b` *bio* Sets the charge for a logical I/O request.

`-c` *clicks* Specifies the maximum number of clicks of memory that can be used by the processes attached to an lnode. (This amount is divided among all the processes.)

`-D` *h1*`,`*h2* Sets the decay rate for process priorities to normal nice, so that they decay to half their initial value in *h1* seconds, and sets the decay rate for process priorities to maximum nice, so that they decay to half their initial value in *h2* seconds.

`-E` *half_life* Sets the decay rate for users' process rates so that they decay to half their initial value in *half_life* seconds.

`-F` *flags* Sets various global scheduling flags. The *flags* value must be in octal. The following flags are available:

Flag	**Description**
`NOSHARE`	(`001`) Turns off the fair-share scheduler. Leaves accumulated charges in the user database (UDB) unless they are cleared by the system administrator.
`ADJGROUPS`	(`002`) Specifies adjustments by group IDs (group share allocation).
`LIMSHARE`	(`004`) Specifies limits on maximum share.

`NOSCHED`	(`020`) Gathers fair-share charges and usage information, but does not use these values for CPU scheduling.
`SHAREBYACCT`	(`010`) Specifies Share by Account mode; share priorities are calculated based on active account IDs (shareholders) instead of active user IDs. The correct value for the `SHAREHOLDER` flag in the UDB is now required; see the NOTES section for more information.
`USRLEVLFSS`	(`0100`) Specifies *user-level fair-share mode*; in this mode, the fair-share daemon (`shrdaemon`(8)) performs the share calculations and updates the lnodes. This will be the default mode in future releases of the UNICOS operating system.

See the `share`(5) man page for more information on the global scheduling flags.

`-G` *maxgroups* Sets the maximum depth for the fair-share hierarchy. *maxgroups* is 4 by default.

`-K` *half-life* Sets the decay rate for users' usages so that they decay to half their initial value in *half-life* hours. If *half-life* has the suffix `s`, the decay rate is set to *half-life* seconds rather than hours.

`-m` *click* Sets the charge for a memory tick.

`-n` *interval* Specifies the interval, in seconds, to be used for copying lnode information to the UDB.

`-P` *maxpri* Specifies the absolute upper bound for the priority of a process. (The value must be less than the largest nonnegative integer.)

`-p` *procs* Specifies the maximum number of processes that can be attached to an lnode.

`-Q` *maxupri* Specifies the upper bound for normal processes' priorities. Idle processes run with priorities in the range *maxupri* < *pri* < *maxpri*.

`-R` *delta* Sets the run-rate for the fair-share scheduler in seconds.

`-r` Causes `shradmin` to reset to the default values. No other options may be used with this option.

`-S` *maxusers* Sets the maximum number of users and groups that can be active. This value cannot exceed the size of the lnode table configured in the kernel (set by the `NUSERS` parameter).

`-s` *syscall* Sets the charge for a system call.

`-t` *tick* Sets the charge for a CPU tick.

`-U` *maxusage* Sets the `MAXUSAGE` parameter (the upper bound for reasonable usages). Users with usages larger than this are grouped together and given a normalized usage that prevents them from interfering with users that do not have large usages.

`-v` Displays the scheduling feed-back parameters. Do not attempt to set the changed parameters.

`-X` *maxushare*	If the `LIMSHARE` scheduling flag is on, then this option limits the maximum effective share an individual user can have to *maxushare* times his or her allocated share. The default is 2.0, or 2%.
`-Y` *mingshare*	If the `ADJGROUPS` scheduling flag is on, the priority is adjusted for any group that is getting less than *mingshare* times its allocated share.
`-y` *tio*	Sets the charge for a stream I/O operation. This is dependent on the number of kernel buffer operations, so a "`write(1)`" operation costs the same as a "`write(64)`" operation to an ordinary stream, or a "`write(1024)`" operation to a pipe.
`-Z` *sharemin*	Specifies the minimum machine share allocated to a user. This option, combined with the `-X (`*maxushare*) option, provides a floor on individual priorities to provide all users with reasonable interactive response, regardless of share allocations or past usage. The default is 0.
percent	The percentage to apply to all the charges.

NOTES

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

Privilege Text	**Action**
`chgany`	Allowed to modify charges.

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

Active Category	**Action**
`system`, `secadm`, `sysadm`	Allowed to modify charges.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to modify charges.

As of UNICOS release 8.3, all entries in the UDB must be correct. The correct use of flags in the UDB is enforced. For example, in Share by Account mode, account ID entries (shareholders) must now have the the `SHAREHOLDER` flag set to 01000000. Lnodes are no longer generated for incorrect entries; users whose UDB entries reference the incorrect entries are unable to log in and their Network Queuing Environment (NQE) jobs are rejected. The `shrtree`(8) command can be used to analyze entries in the UDB; this command reports problems that can cause unexpected system behavior or prevent login for affected users. See the `shrtree`(8) man page for information on running this command.

BUGS

The percent value also affects any new constants, so bias them accordingly.

The defaults are rarely relevant.

EXAMPLES

Example 1: The following example invokes `shradmin` with the `-v` (view) option to show the current charges:

```
$ shradmin -v

sn1703a ice 9.0.0ab wdp.0 CRAY Y-MP
Scheduling flags     = ADJGROUPS,LIMSHARE
Charging percentage = 100%,
Usage decay rate    = 0.99807644 (half-life of 3600.0 seconds),
active users = 16, active groups = 10.
max. users = 200, max group nesting = 4.

Charge: syscall      0%, bio      0%, tio      0%, tick    100%, click      0%.
Costs:  syscall       0, bio       0, tio       0, tick     100, click       0.
Counts: syscall 3916030, bio  627083, tio   38762, tick 3858349, click       0.

Process priority decay rate biased by "nice":-
    high priority (nice -20) 0.4044 (half-life of   0.8 seconds),
    avg  priority (nice   0) 0.7039 (half-life of   2.0 seconds),
    low  priority (nice  19) 0.9885 (half-life of  60.0 seconds).
Run rate decay rate          0.8409 (half-life of   4.0 seconds).

Max. value for normal usage        = 1.000000e+12,
Max. value for normal p_sharepri   = 1.000000e+28,
Max. value for idle   p_sharepri   = 1.000000e+38.

High value of current normal usage = 9.902495e+08,
high value of current p_sharepri   = 1.041637e+01.
```

Example 2: The following command line changes the costs to 10% of the cost values:

```
shradmin 10
```

Example 3: The following command line alters the default parameters at system boot time in `/etc/config/daemons`; this is unnecessary if the default values set in the kernel are correct.

```
shradmin -F04 -K1 -R10 -Y0.90 -Z0.001 -b1666 -m3 -s523 -t600 -y4700
```

Example 4: The following example sets share hierarchy levels at 7:

```
shradmin -G 7
```

Example 5: The following example sets up Share by User mode by setting the `LIMSHARE` and `ADJGROUPS` flags:

```
shradmin -F 06
```

Example 6: The following example sets up Share by Account mode by setting the `ADJGROUPS` and `SHAREBYACCT` flags:

```
shradmin -F 012
```

Example 7: The following command line turns off the fair-share scheduler. (The fair-share daemon, `shrdaemon`(8), continues to accrue charges in the UDB until it is stopped by the system administrator.)

```
shradmin -F 01
```

FILES

`/usr/include/sys/share.h`	Definition of `shconsts` structure

SEE ALSO

`shrdaemon`(8), `shrtree`(8)

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

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

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

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

UNICOS Resource Administration, Cray Research publication SG–2302

Henry, G.J., "The Fair Share Scheduler," *Bell Labs Technical Journal*, LVIII-8b, 10-84, pp. 1845-1858.

NAME

`shrdaemon` – Performs system functions for the fair-share scheduler

SYNOPSIS

`/etc/shrdaemon [-t]`

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `shrdaemon` command cleans up kernel limits structures (lnodes) after users have logged off, and writes the usage and charge information to the user database (UDB). `shrdaemon` is invoked at system startup in the `/etc/config/daemons` configuration file and runs as a system daemon (with UID 0) in the background.

The `shrdaemon` command uses the `limits`(2) system call to initialize the default idle lnode, configured at system boot time, for use of the idle processes. `shrdaemon` goes into a loop scanning for dead user lnodes, updating them in the UDB as they appear, and checkpointing active lnodes every few minutes.

The `shrdaemon` command also maintains a checkpoint file of active users, `/etc/lnodes.chkpt`, so that after a system crash, charges accumulated by users active at the time of the crash are not lost. The `/etc/lnodes.chkpt` file, if present, is read when `shrdaemon` is first invoked, and active lnodes found in it are updated in the user database (UDB).

`shrdaemon` catches the system termination signal 27 (`SIGSHUTDOWN`), which causes it to remove `/etc/lnodes.chkpt`, update the UDB for any remaining active lnodes, and terminate.

If the user-level fair-share mode is enabled (that is, the `USRLEVLFSS` flag is set), `shrdaemon` performs the functions that the kernel would perform in kernel-level fair-share. `shrdaemon` applies CPU scheduling policy calculations to data in the lnodes, then updates the lnode table in the kernel. In addition, a user exit allows sites to substitute site-specific scheduling policy algorithms for the standard policies Share by User and Share by Account (`SHAREBYACCT`), and the `ADJGROUPS`, `NOSHARE`, `LIMSHARE`, and `NOSCHED` kernel calculations.

The `shrdaemon` command accepts the following option:

`-t` Creates a log file for debugging purposes; log information is written to standard error (`stderr`). This option is intended for internal debugging only; it is not recommended for use at a site because the log mechanism consumes significant CPU time and the log file can become quite large.

NOTES

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

Active Category	**Action**
`system`, `secadm`, `sysadm`	Allowed to start the daemon.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to start the daemon.

The checkpoint file used by `shrdaemon` is not in any way related to the checkpointing accomplished by the `chkpnt`(1) and `restart`(1) commands, and the `chkpnt`(2) and `restart`(2) system calls.

BUGS

If the `shrdaemon` process should die for any reason, then new users logging into the system gradually consume all available kernel lnodes. You can use the automated incident reporting (AIR) daemon, `aird`(8), to restart `shrdaemon`.

FILES

`/etc/udb`	User validation file containing user control limits
`/etc/lnodes.chkpt`	Checkpoint file for active lnodes

SEE ALSO

`aird`(8), `shradmin`(8)

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

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

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`shrdist` – Redistributes shares among a resource group for the fair-share scheduler

SYNOPSIS

`shrdist` [`-b`] [`-g` *group*] [`-p` *directory*] [`-s` *shares*] [`-u` *user*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `shrdist` command reassigns shares within a resource group. The point-of-contact (POC) for each resource group can control the allocation of resource shares in his or her group. This control is authorized by an authorization file, which `shrdist` checks before executing. If the POC administers more than one resource group, the particular group must be specified (by using the `-g` *group* option and argument) when the `shrdist` command is invoked.

The `shrdist` command accepts the following options:

`-b`	Enables batch mode.
`-g` *group*	Specifies a resource group.
`-p` *directory*	Specifies a path to a directory other than `/etc` (this option allows you to test the execution of `shrdist`).
`-s` *shares*	Specifies the new shares value (not valid in interactive mode).
`-u` *user*	Specifies the user name to be updated (not valid in interactive mode).

The `shrdist` command has two modes of execution: interactive and batch.

To use interactive mode, you must be able to communicate in full-screen mode (for example, `vi`).

When you execute `shrdist` in interactive mode, the command displays the current share distribution for each account under the resource group. You control the cursor position with the following keys:

Key	**Cursor Movement**
`<h>`	Left
`<j>`	Down
`<k>`	Up
`<l>`	Right
`<+>`	Next page
`<->`	Previous page
`</>`	Prompts for a search pattern

You can change (reallocate) the share values by positioning the cursor (which should always be positioned over the rightmost digit of the share allocation field) at the desired account and entering the new values. All other accounts are then automatically recalculated. Digits shift to the left as you enter them. The current value for the <ERASE> key can be used to erase characters.

After you have reallocated all new share values, enter `u` to update the share allocation database, and enter `q` to quit (exit) the `shrdist` command. If the share distribution has been updated and you exit (`q`) without having updated (`u`), `shrdist` prompts for verification.

To use `shrdist` in batch mode, use the `-b` option on the command line. In this mode, only one account may be updated per execution of `shrdist`. Specify the user name to be updated with the `-u` option, and the new shares value with the `-s` option. (The `-u` and `-s` options are valid only with the `-b` option.)

The `shrdist` command supports test mode execution (similar to `udbgen` and `udbsee`). You can specify the `-p` option to specify a path to a directory other than `/etc`, in which test versions of the `udb`, `group`, `acid`, and `shrdist.auth` files can be found.

The `shrdist` command uses `/etc/shrdist.auth` as the authentication file; therefore, all of the resource points of contact should have read access to `/etc/shrdist.auth`. The `shrdist.auth` file is a simple ASCII text file, consisting of two fields per line, separated by white space. The first field is the user identity of the resource point of contact, and the second field is the name of the resource group. The resource group name may be terminated by end-of-line or any white space. If white space follows the resource group name, anything beyond that white space is treated as a comment.

NOTES

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

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

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

EXAMPLES

Example `shrdist.auth` file:

```
user1 resource1
user1 resource-group2
user2 resource-group2         Comments
```

In this example, `user1` is allowed to function as point of contact for more than one resource group. If a resource group is not specified with the `-g` option, the first match (in this case, `resource1`) is used.

FILES

`/etc/shrdist.auth`	Share authentication file

SEE ALSO

`shradmin`(8), `shrsync`(8)

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

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`shrlimit` – Sets another user's limits for the fair-share scheduler

SYNOPSIS

`/etc/shrlimit` [`-u`] *user_name* [*command*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `shrlimit` command creates a new process (with limits per the values for the specified *user_name* in the user database (UDB)), changes to the user and group IDs *user_name*, prefixes *command* with the login shell for *user_name* (or `bin/sh -c` by default), and executes using `execv` (see `exec`(2)) with the resulting string. This command can be executed only by an appropriately authorized user.

The `shrlimit` command accepts the following option and operands:

`-u`	Does not change user and group IDs.
user_name	Specifies the user name; this operand is not optional.
command	Passes any optional extra arguments as one argument to the invoked shell's `-c` option.

NOTES

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

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

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

FILES

`/etc/udb`	User validation file containing user control limits

SEE ALSO

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

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

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`shrmon` – Monitors detailed system fair-share scheduling information

SYNOPSIS

`/etc/shrmon` [`-c`] [`-i` *n*] [`-l` *n*] [`-r` *n*] [`-u`] [`-v`]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `shrmon` command displays a columnar table of information in order by user and group.

The number of columns displayed depends upon the `-v` (verbose) option. Each usage of the `-v` option increases the level of verboseness by one. For example, the following command line specifies verbose level 2:

```
shrmon -vv
```

See the Column Descriptions subsection to determine the level of verboseness at which that particular column will appear.

If no options are specified, `shrmon` prints the minimum report and exits.

The `shrmon` command accepts the following options:

`-c` Specifies continuous operation, using `curses`(3) routines for full-screen update.

`-i` *n* Specifies the interval between updates as *n* seconds. The default is 4.

`-l` *n* Limits the display to the first *n* levels of the share hierarchy. For example, the following command line reports on only the first level groups:

```
shrmon -l 1
```

`-r` *n* Specifies repeat mode of operation. `shrmon` repeats the display *n* times, each separated by a form-feed, and then exits.

`-u` Prints the user and group names on each line, separated by a `/`. For example, user `work` in group `busy` would be displayed as follows:

```
System/busy/work
```

`-v` Increases the amount of information displayed. Each usage of the `-v` option increases the verboseness level by one.

Column Descriptions

The following columns are always displayed:

Column	Description
`CPUs`	The number of CPU seconds of time accumulated by this user during the sample period.
`charge`	A relative number indicating the amount charged to this user during the sample period. This number is a derivation of the cost factors described on the `shradmin`(8) man page.
`%Rate`	The percentage of all charges ascribed to this user during the sample period.
`%share`	The percentage of the effective share of the machine to which this user or group is entitled (as described by the allocation of shares to individual users and groups defined by the system administrator).
`%CPU`	The percentage of all available CPU resources that were allocated to this user or group during the sample period.
`Name`	The name of the user or group, as defined in the user database (UDB), with indentation to indicate hierarchy level.

The following additional columns are displayed at verbose level 1 (the `-v` option specified once):

Column	Description
`rate`	The `kl_rate` field, which is a floating-point number used as an indication of the rate of system usage by this user or group.
`nrun`	Estimate of the maximum amount of the `rshare` value (machine shares) that can be used by all runnable processes under the lnode.
`%rshare`	The percentage representing the dynamic share of the system that this user or group has been allocated in order that they achieve their effective share (see the `%share` column).

The following additional columns are displayed at verbose level 2 (the `-v` option specified twice):

Column	Description
`kids`	The number of group members currently active on the system.
`ref`	The number of processes that this user, or all of the users in this group, have currently active on the system.

The following additional columns are displayed at verbose level 3 (the `-v` option specified three times):

Column	Description
`usage`	The floating-point number that represents the `kl_usage` field, and relates to the system use by this user or group.
`temp`	The floating-point number that represents the `kl_temp` field, and that is an intermediate variable used in the calculation of `kl_usage`.

NOTES

The `shmon` command will be removed in a future UNICOS release. The `shrview`(1) command provides an improved method to monitor fair-share scheduling information.

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

Active Category	**Action**
`system`, `secadm`	Shell redirected I/O is not subject to security label restrictions.

If the `PRIV_SU` configuration option is enabled, for the super user, shell redirected I/O is not subject to security label restrictions.

EXAMPLES

The following command line displays the minimum report in continuous-update mode:

```
$ shrmon -c

Mon Oct 11 18:10:24 1993                   CPUs charge  %Rate %share   %CPU
    0.0    1.8  100.0  100.0    0.0 Root
    0.0    0.0    0.0    0.0    0.0  Idle
    0.0    0.0    0.0   10.0    0.0  System
    0.0    0.0    0.0   10.0    0.0   operator
    0.0    1.8  100.0   90.0    0.0  Users
    0.0    1.8   96.6    0.8    0.0   SoftDev
    0.0    0.0    0.3    1.4    0.0   Mktg
    0.0    0.1    3.2    0.0    0.0   CCN
    0.0    0.0    0.0    5.0    0.0   HardDev
```

SEE ALSO

`shradmin`(8), `shrtree`(8)

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

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

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`shrsync` – Synchronizes the UDB and fair-share information for active users

SYNOPSIS

`shrsync` [`-n`] [`-p` *directory*] [`-q`] [`-s`] [`-u`]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `shrsync` command synchronizes the fair-share information contained in the user database (UDB) and the active system. By default, `shrsync` examines the system to determine what active users and resource groups are present in the system. Then it updates the active users and resource groups on the system with the current value of allocated shares as read from the user database. This process allows you to make routine changes to share allocation, such as daytime/nighttime, or shift-to-shift, without users having to log out and log in again.

The `shrsync` command accepts the following options:

`-n` (Negative) Checks for negative values in the usage and charge fields of the lnode (the `l_usage` and `l_charge` fields, respectively), and replaces them with valid information from the UDB.

`-p` *directory*
(Path) Specifies a path name other than `/etc` so that test versions of `udb`, `group`, and `acid` files can be used.

`-q` (Quotas) Updates the CPU-quota-used information with the current values from the UDB. See the EXAMPLES section for more information.

`-s` (Silent) Suppresses the normal messages concerning the updating of a user's UDB entry.

`-u` (Update) Updates the usage, charge, and quota information from the active system in the UDB so that it is an accurate reflection of all users' usage and charges when `shrsync` was executed.

NOTES

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

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

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

Active Category	**Action**

`system`, `secadm`, `sysadm`	Allowed to use this command.

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

EXAMPLES

The CPU-quota-used information in the UDB is updated only when a user logs off the system. For this reason, you should ensure that the UDB contains current information before updating the active system from the UDB. This is accomplished by first updating the CPU-quota-used information in the UDB with information from the active system.

The following procedure shows the preferred sequence of events when clearing the CPU-quota-used information for a group of users.

1. Use the `shrsync -u` command to update the UDB with the information from the active system, as follows:

   ```
   $ shrsync -u
   ```

2. Clear the desired CPU-quota-used fields in the UDB with the `udbgen` command. (See the `udbgen`(8) man page for more information.)
3. Use the following command to update the CPU-quota-used information from the UDB:

   ```
   $ shrsync -q
   ```

FILES

`/etc/udb`	User database (UDB)

SEE ALSO

`shradmin`(8), `shrdist`(8), `udbgen`(8)

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

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

shrtree – Displays the fair-share hierarchy defined in the user database (UDB)

SYNOPSIS

/etc/shrtree [-e] [-f] [-F *mode*] [-G *maxgroups*] [-L *level*] [-l] [-m] [-p *udb_path*] [-s] [*id*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The shrtree command displays the share tree, or fair-share hierarchy, that is defined in the user database (UDB), highlighting problems that could prevent a user from logging in. This command can be executed by all users. The *share tree* is the set of all resource group chains starting from the base node (also called the *root entry*, labeled _ROOT_) and ending with the user entries. Any problems with the configuration of the fair-share scheduler in the UDB are marked in the shrtree output. In addition, each entry in the hierarchy is marked with a status indicating whether it is a resource group, account ID (shareholder), or user entry.

When invoked with no options, shrtree displays the short version of output (equivalent to the -s option).

The shrtree command accepts the following options and arguments:

-e	(Errors) Displays errors only. UDB entries that are set up incorrectly for use with the fair-share scheduler are displayed, along with a description of the problem. If a single UDB entry has more than one problem, the shrtree output displays an error for each problem. Entries marked with WARN are warnings; entries marked with ERROR are problems that will prevent logging in.
-f	(Full display) Displays the full share tree, including user entries. By default, only resource group entries are displayed.
-F *mode*	(Force fair-share mode) Overrides the current fair-share mode. Use ACCT to specify Share by Account mode; use UID to specify Share by User mode. By default, the mode specified in the kernel shconsts structure is used. This option is useful if you are analyzing a UDB that is in a different mode from the current system mode.
-G *maxgroups*	(Group depth) Sets the maximum depth of the hierarchy to the specified value. By default, the value of MAXGROUPS (as defined in the shconsts structure) is used to specify the maximum level of the hierarchy.
-L *level*	(Maximum level) Specifies the maximum level of the fair-share hierarchy to be displayed. By default, all levels of the hierarchy are displayed.
-l	(Long display) Produces a wider display than the default display, with additional columns for the number of shares and the usage value for each entry.
-m	(Medium display) Displays share usages values as decimal fractions rather than percentages.

`-p` *udb_path*	(UDB path) Specifies an alternate UDB as found in the directory specified by *udb_path*.
`-s`	(Short display) Displays the short version of the share tree. The `Share` and `Usage` fields are omitted; fair-share information is displayed as percentages; and only resource groups are listed (unless the `-f` option is specified). This display is the default.
id	Displays information on all possible resource group chains that contain an entry that matches *id*. The value *id* can be specified as either the entry name or the ID number.

Resource Group Hierarchy Display

The `shrtree` command displays the resource group hierarchy by default (with all options except `-e`). The column headings for this display and their meanings are as follows:

Heading	**Description**
`Lv`	Level of the entry in the resource group chain. The root entry (`_ROOT_`) is always at level 0.
`Name`	Name of the entry in the UDB.
`ID`	ID number of the entry in the UDB.
`Shares`	(Long listing only) Number of shares from the UDB.
`System Share`	System-wide percentage of shares for the entry. For example, if an entry had a resource group share of 20% and the resource group had a 30% share of the system, the system share value would be 6% (`0.20 * 0.30 = 0.06`). If the `-m` option is specified, this value is displayed as a decimal fraction.
`Group Share`	Relative percentage of shares for the entry within the resource group. For example, if an entry was assigned 10 shares and the total shares in its resource group was 100, the group share value would be 10% (`10 / 100 = 0.10`). If the `-m` option is specified, this value is displayed as a decimal fraction.
`Usage`	(Long listing only) Usage value from the UDB (`shusage` field). The share usage for `_ROOT_` is calculated by adding the share usages of the first-level resource groups.
`System Usage`	Accumulated system share usage for the entry based on the usage value in the `shusage` field of the UDB. The value is first decayed and then divided by the total system usage obtained from `_ROOT_`. If the `-m` option is specified, this value is displayed as a decimal fraction.
`Status`	Flags to indicate the entry type (user, resource group, or account ID) and to indicate problems with the configuration of the fair-share hierarchy in the UDB. Note that problems of type `ERROR` can prevent the affected entries from logging in. An entry can have several flags, separated by the `|` character. The following flags are available:

Flag	**Description**
`A`	Account; marks an account ID (shareholder) entry.
`G`	Resource Group; marks a resource group entry.

`U`	User; marks a user entry.
`Ba`	Bad Account. (Severity: `ERROR`) An entry is being used as an account ID without the `SHAREHOLDER` flag set. To enable this flag, set the `shflags` field in the UDB to 01000000; this marks the entry as an account ID. (See the `udbgen`(8) man page for more information).
`Bf`	Bad flag. (Severity: `ERROR`) For a resource group or account ID entry, this error indicates that the entry's share flags are incorrect for its position in the resource group chain (for example, if the entry appears to be used as a resource group but does not have the `NOTSHARED` flag set). For a user entry, this error indicates that either that the entry references a resource group when Share by Account mode is enabled, or that it references an account ID when Share by User mode is enabled. The referenced entry will be marked with the `Bg` flag.
`Bu`	Bad UID. (Severity: `ERROR`) The UID assigned to this UDB entry is greater than the maximum UID allowed by the system.
`Mg`	Maximum groups exceeded. (Severity: `ERROR`) The level of this resource group exceeds the maximum level allowed in the share tree hierarchy (as defined by `MAXGROUPS` in the `shconsts` structure, or as set by the `-G` option).
`Bg`	Bad group. (Severity: `ERROR`) This error indicates that the entry is being used as a resource group but does not have the `NOTSHARED` flag set. The user entries referencing this resource group will be marked with the `Bf` status. To correctly set the `NOTSHARED` flag for a resource group entry, set the `shflags` field in the UDB to 040000. (See the `udbgen`(8) man page for more information).
`Nc`	No children. (Severity: `WARN`) This resource group does not have any user entries (no references to the resource group).
`Ng`	No group. (Severity: `ERROR`) If this error appears for a resource group entry, it indicates that a user entry references this group as its resource group (or, if Share by Account mode is enabled, in its `acid` list) but the entry does not exist in the UDB. If this error appears for a user entry, it indicates that the user entry references an undefined resource group.
`Nr`	No root. (Severity: `ERROR`) Indicates that this resource group chain is circular and does not start at `_ROOT_`.
`Rl`	Root link. (Severity: `WARN`) A user or account ID entry is referencing `_ROOT_` directly. There should be at least one level of resource group or shareholder entries between `_ROOT_` and the user or account ID entries.
`Zs`	Zero shares. (Severity: `WARN`) A resource group or user entry does not have any shares. This problem can prevent login of the affected entry.

Error Display

The `shrtree` error display is produced by the `-e` option. The column headings for the error display and their meanings are as follows:

Heading	**Description**
`Type`	Severity of the problem; either `ERROR` or `WARN`. Problems of type `ERROR` can prevent login.
`Name`	Name of the entry in the UDB.
`ID`	ID number of the entry in the UDB.
`Status`	Numeric flag to indicate the type of error.
`Description`	Flag to indicate the type of error, with a brief description of the problem. See the "Resource Group Hierarchy Display" subsection for a list of the available flags and their meanings.

EXAMPLES

Example 1: The following example shows sample output for `shrtree` with no options (short form report). The system is running Share by User mode, with a maximum of four levels in the fair-share hierarchy. Note that in this example, the `Serv` entry has warning flag `Nc`; `Unknown` has warning flag `Zs`; and `Country`, `Region`, and `TechOps` have flags `Nc` and `Zs`.

```
$ shrtree

DISPLAY OF SHARE TREE
_____________________
UDB path:      DEFAULT
Analyzed:      By UID
Format:        Groups only
Maxgroups:     4
Node:          ALL
Group Count:   15
Account Count: 0
User Count:    1370
Warnings:      9
Errors:        0

Warning Count: 4     (Nc) Group has no references
Warning Count: 1     (Zs) User has zero shares
Warning Count: 4     (Zs) Group has zero shares

                                       System Group  System
Lv        Name                 ID      Share  Share  Usage  Status   Flags
__ __________________________ ________ ______ ______ ______ ________ ________
 0 _ROOT_                            0 100.0% 100.0% 100.0% G           40000
 1   Demos                        8367 100.0% 100.0%   0.0% G           40000
 2     Serv                       8001 100.0% 100.0%   0.0% G|Nc        40000
 1   System                       8389   0.0%   0.0%   0.0% G           40000
 2     Admin                      8306   0.0%   2.9%   0.0% G           40000
 1   Unknown                      8393   0.0%   0.0%   0.0% G|Zs        40000
 1   Users                        8395   0.0%   0.0%  28.8% G           40000
 2     CCN                        8354   0.0%   4.8%   0.2% G           40000
 2     Country                    8359   0.0%   0.0%   0.0% G|Nc|Zs     40000
 2     HardDev                    8372   0.0%   4.8%   0.0% G           40000
 2     Intl                       8374   0.0%  14.3%   0.0% G           40000
 2     Mktg                       8381   0.0%  28.6%   0.1% G           40000
 2     Region                     8385   0.0%   0.0%   0.0% G|Nc|Zs     40000
 2     SoftDev                    8386   0.0%  47.6%  28.5% G           40000
 2     TechOps                    8390   0.0%   0.0%   0.0% G|Nc|Zs     40000
```

Example 2: The following example shows the error display for the same system (`shrtree` with the `-e` option). Only the entries with errors are displayed.

```
$ shrtree -e

DISPLAY OF SHARE TREE
_____________________
UDB path:      DEFAULT
Analyzed:      By UID
Format:        Groups only
Maxgroups:     4
Node:          ALL
Group Count:   15
Account Count: 0
User Count:    1370
Warnings:      9
Errors:        0

Warning Count: 4     (Nc) Group has no references
Warning Count: 1     (Zs) User has zero shares
Warning Count: 4     (Zs) Group has zero shares

  Type      Name    ID    Status  Description
_________ ________ _____ ________ ____________________________________
**WARN*** Serv     8001       10 Nc: Group has no references
**WARN*** Unknown  8393     1000 Zs: Group has zero shares
**WARN*** unknown    12     1001 Zs: User has zero shares
**WARN*** Country  8359     1010 Nc: Group has no references
**WARN*** Country  8359     1010 Zs: Group has zero shares
**WARN*** Region   8385     1010 Nc: Group has no references
**WARN*** Region   8385     1010 Zs: Group has zero shares
**WARN*** TechOps  8390     1010 Nc: Group has no references
**WARN*** TechOps  8390     1010 Zs: Group has zero shares
```

FILES

`/usr/include/sys/share.h`	Definition of `shconsts` structure
`/usr/include/udb.h`	Definition of fields in the UDB

SEE ALSO

`shradmin`(8)

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

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

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`shutacct` – Shuts down accounting

SYNOPSIS

`/usr/lib/acct/shutacct` ["*reason*"]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `shutacct` shell script turns off process and daemon accounting and is usually called during a system shutdown in `/etc/shutdown`. `shutacct` invokes the `turnacct`(8) and `turndacct`(8) commands with the `off` operand.

The `shutacct` shell script accepts the following operand:

"*reason*" Specifies that *reason* enclosed in double quotation marks should be appended to `/etc/wtmp`, the login and logoff summary file. A maximum of 11 characters is allowed. By default, the reason is specified by the `ACCTOFF` parameter in the accounting configuration file `/etc/config/acct_config`.

FILES

`/etc/config/acct_config`	Accounting configuration file
`/etc/wtmp`	Login/logoff summary
`/usr/adm/acct/day`	Directory that contains current accounting files

SEE ALSO

`acct`(8), `acctsh`(8), `csa`(8), `shutdown`(8), `turnacct`(8), `turndacct`(8)

UNICOS Resource Administration, Cray Research publication SG–2302

NAME

`shutdown` – Terminates all processing

SYNOPSIS

`/etc/shutdown [-y]` [*grace*]

IMPLEMENTATION

Cray PVP systems

DESCRIPTION

The `shutdown` utility is part of the operation procedures on systems based on the UNIX system. As implemented in the UNICOS system, the `shutdown` utility uses the `killall`(8) command to terminate all currently running processes belonging to sessions other than that of the person who invoked `shutdown`. The `shutdown` utility is designed to bring the system from multiuser mode to single-user mode in an orderly and cautious manner.

The utility is designed to interact with the person who invoked `shutdown`, unless the `-y` option is specified.

The `shutdown` command accepts the following options:

`-y` Quiet mode (no interactive session); executes `shutdown` without waiting for responses from the person who invoked `shutdown`.

grace Specifies the grace period, in seconds, before the shutdown will begin. Default is 60 seconds. Unless the `-y` option is specified, the `shutdown` procedure may instruct you to perform some specific tasks or to supply certain responses before execution can resume.

The `shutdown` procedure goes through the following steps:

1. Checks for and reports on active guest systems. If there is an active guest system, asks whether you want to continue shutdown.

   A return to single-user mode in the host should not affect a multiuser guest, but a subsequent host dump or reboot will destroy the guest and probably will adversely affect its mounted filesystems. If possible, it is best to stop and release guests prior to host system shutdown.

2. If it exists, executes the `/etc/shutdown.pre` user exit.

3. All users logged on the system are notified by a broadcasted message to log off the system. If you did not specify the `-y` option, you may display your own message at this time. Otherwise, the standard file save message is displayed. The `shutdown` script then sleeps for *grace* seconds, to allow users to log out.

4. Shuts down daemons in the `SYS1` and `SYS2` groups (defined in the `/etc/config/daemons` file), using the `sdaemon`(8) command.

5. Terminates remaining processes, using the `killall`(8) command.

6. Releases logical device cache, using the `ldcache`(8) command. This ensures that all logical device buffers are flushed.
7. Shuts down all configured network interfaces (defined in the `/etc/config/interfaces` file), using the `ifconfig`(8) command.
8. If it exists, executes the `/etc/shutdown.pst` user exit.
9. Unmounts all file systems.
10. Enters single-user mode.

After the `shutdown` procedure completes, the system is in single-user mode and is essentially equivalent to the state of the system after a reboot procedure.

User Exits

To allow each site to modify the shutdown procedure, `shutdown` provides two user exits: `/etc/shutdown.pre` and `/etc/shutdown.pst`. The user exits are described as follows:

`/etc/shutdown.pre` This is the first user exit in `shutdown`. If an executable named `/etc/shutdown.pre` exists, it will be executed by `shutdown`.

At this point, nothing has been done in shutting down the system. All daemons are still running, all file systems are still mounted, and all users are still active and unaware that the shutdown processing has begun. This exit could be used to verify the user's permission to run the shutdown script or to run some system cleanup routines.

Shutdown will check the return status from the `shutdown.pre` program. If the return status is nonzero, and the `-y` option was not specified, the user will be queried as to whether or not to continue the shutdown processing. At this point, the shutdown can be stopped without any effect on the system.

`/etc/shutdown.pst` This is the second (and last) user exit in `shutdown`. If an executable named `/etc/shutdown.pst` exists, it will be executed by `shutdown`.

At this point, all processes (users and daemons) have been terminated, but all of the file systems are still mounted. This is virtually single-user mode, except that the file systems are still mounted.

NOTES

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

Privilege Text	**Action**
`valid`	Allowed to use this command.

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

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

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

Caution should be used in choosing what is executed in `/etc/shutdown.pst`. At this point, the various log daemons have been terminated. Since the log daemons are not available to free up space, the file systems containing the various system log files could fill up.

MESSAGES

The most common error message is *device busy*. This message is issued when a particular file system could not be unmounted.

SEE ALSO

`brc`(8), `ifconfig`(8), `init`(8), `killall`(8), `ldcache`(8), `mount`(8), `sdaemon`(8), `slogdemon`(8)

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

General UNICOS System Administration, Cray Research publication SG–2301

NAME

`slogdemon` – Security event logging daemon

SYNOPSIS

`/etc/slogdemon`

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The security auditable event logging daemon, `slogdaemon`, collects security-relevant audit records from the operating system by reading the character special pseudo device `/dev/slog` (defined by `SLG_DEV` in `sys/slog.h`) and places them in the security audit log file. At daemon startup time, the security audit log file is created if it does not already exist. Security audit records are appended to this file.

The directory in which the security audit log file exists is defined by the `SLG_DIR` configuration option. The name of the security audit log file is defined by the `SLG_FILE` configuration option. The default path name of the security audit log file is `/usr/adm/sl/slogfile`. The file's maximum size is defined by the `SLG_MAXSIZE` configuration option. The default value for the maximum size is 8192000 bytes. When the security audit log file reaches its maximum size, the daemon renames the file according to the following naming convention:

`SLG_DIR/s.`*yymmddhhmmss*

The `s.` prefix of the renamed file is defined by the `SLG_FPREFIX` configuration option. The suffix of the renamed file identifies the date and time at which the file exceeded its maximum size. After the file has been renamed, a new security audit log file is created by using the path name of the original security audit log file.

Analysis of security audit records can be performed using the `reduce`(8) utility. The daemon is stopped during system shutdown after all user processes have been terminated.

NOTES

Processes are placed into a sleep state when the operating system audit record buffer becomes greater than 50% full. If the buffer becomes full, the system panics to ensure that no security audit records are lost.

The daemon should be started for all valid `init` levels by an entry in `/etc/inittab` (see `inittab`(5)), or for multiuser mode by an entry in `/etc/rc` (see `brc`(8)).

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

Active Category	**Action**
`system`, `secadm`	Allowed to start the daemon.

If `PRIV_SU` is enabled, the super user is allowed to start the daemon.

FILES

`/dev/slog`	Source of security event records.
`/slogdemon_err`	Contains a copy of the error messages sent out by the current `slogdemon` process.
`/usr/adm/sl/slogfile`	Repository for security event records.
`/etc/slogd.pid`	Contains the process ID of the currently-running version of the security log daemon. Used by `shutdown`(8) to exclude the security log daemon from receiving the `SIGTERM` signal and to kill the user processes.

SEE ALSO

`brc`(8), `killall`(8), `reduce`(8), `shutdown`(8)

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

`inittab`(5), `slog`(4), `slrec`(5) in the *UNICOS File Formats and Special Files Reference Manual*, Cray Research publication SR–2014

General UNICOS System Administration, Cray Research publication SG–2301

NAME

`smpmon` – Interactively monitors the physical-layer External Semaphore Device

SYNOPSIS

`/etc/smpmon` [`-s` *SFS-Arbiter*]

IMPLEMENTATION

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

DESCRIPTION

The `smpmon` command is used to interactively monitor the physical-layer External Semaphore Device. `esdmon` uses the special device `/dev/smp` to interface to the device driver.

Executing the `smpmon` command requires super-user permissions.

`-s` *SFS-Arbiter*	Specifies the name of the Shared File System (SFS) arbitration service to be monitored. The *SFS-Arbiter* name must match one of the valid entries in the `/etc/config/sfs` configuration file.

`smpmon` always operates in full-screen mode.

At the end of each repeat interval, which is initially set to 2 seconds, keyboard input is sampled for one of the following commands:

Command	**Description**
`^l`	Clear and refresh the display.
`+`	Increase the repeat interval by 1 second.
`-`	Decrease the repeat interval by 1 second.
`h,?`	Display the *help* menu.
`q`	Quit, exit the program.

FILES

`/dev/smp`	Character Device Special used to interface to the channel-level device driver for the External Semaphore Device

NAME

`smtd` – FDDI station management daemon

SYNOPSIS

`/etc/smtd` [`-z` *device*] [`-s` *sec*] [`-n` *sec*] [`-v` *sec*] [`-c` *file*] [`-u`] [`-p` *port*] [`-d`] [`-i` *infc*]

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

DESCRIPTION

The `smtd` is the daemon process that handles all frame-based station management functions for the Cray FDDI network interface. Cray FDDI supports Station Management (SMT) version 6.2, which is comprised of two major components: frame-based SMT and Connection Management (CMT). Frame-based SMT performs three major functions: neighborhood notification (NN), duplicate address detection (DAD), and response to SMT request frames from the FDDI network such as neighbor information request frames (NIF), and station information request frames (SIF) requests. (For an explanation of Connection Management, see `fddi`(4).)

To provide a mechanism for other processes such as utility programs (see `fddimap`(8)) and network management applications (see `snmpd`(8)), `smtd` provides a UDP-based socket utility over which requests and replies can be sent between processes to obtain information about the FDDI network. The `smtd` acts as a type of proxy to these other processes. The socket utility is discussed in detail in the Socket Utility section of this man page.

The command-line options are as follows:

`-z` *device*	Specifies the UNICOS character special device name to open for communications on the FDDI network. Default is `/dev/fddi0/smt`.
`-s` *sec*	Specifies the status report threshold (in seconds) for sending SRF announcements. If 0 seconds is specified, the status reporting protocol is disabled. The default is 2 seconds, the minimum is 2 seconds, and the maximum is 32 seconds.
`-n` *sec*	Specifies the interval (in seconds) for sending NIF requests during the neighbor notification protocol (NNP). The default is 5 seconds and the minimum is 2 seconds.
`-v` *sec*	Specifies the number of seconds that is allowed between NIF requests from upstream and downstream neighbors before considering the UNA and DNA not valid. The default is 228 seconds.
`-c` *file*	Specifies the name of the configuration file to use.
`-u`	Enables communications across the socket utility. The default is disabled.
`-p` *port*	Sets the UDP port number of the socket utility. The default is 3000. See also `fddimap`(8).
`-d`	Enables debug mode; verbose output.

`-i` *infc* Specifies the interface ordinal of the FDDI network that will be managed by this copy of `smtd`. If the Cray Research system has more than one FDDI interface, and each interface is attached to a different ring, copy of `smtd` must exist for each ring.

Configuration File

Any parameters that you can set on the command line can also be set in a more permanent fashion by placing them in a configuration file; the exception is the `-d` debug flag, which can be entered only on the command line. The following shows the format of the configuration file. Only parameters that will change must appear in the file; others are set to their default values.

```
#
#    These three lines are comment lines.
#
device=path
interface=ordinal
udpport=portnumber
tnotify=seconds
tnnout=seconds
srthreshold=seconds
```

You can place comment lines anywhere in the file if the comment starts in column 1 of a line. Comments are designated by a # character in column 1.

Neighbor Notification Protocol

The `smtd` performs neighborhood notification (NN) by sending NIF request frames to the broadcast address by using next station addressing (SMT NSA) every `T_NOTIFY` seconds. The NSA mode of addressing lets only a single station on the FDDI ring copy the frame, even though it is addressed to the broadcast address. In this way, it should be the station's downstream neighbor (DNA) that copies the frame. The DNA sets its UNA to our address and sends a response to the NIF request. The DNA is set to the MAC address of the station that responds to our NIF request. All stations on the FDDI ring perform this same protocol; therefore, at some point in time, the same type of NIF request is received from our upstream neighbor (UNA). When received, set your UNA to the MAC address of the sender of that NIF request before replying to the frame. When the UNA gets your response, it sets its DNA to your address. This process continues forever and the stations on the FDDI ring eventually learn their UNAs and DNAs.

Another part of the Neighbor Notification Protocol involves the timing out of the receipt of NIF requests from our UNA and NIF responses from our DNA. If we do not hear from our neighbors for a period longer than `T_NN_Out`, we set the appropriate neighbor's address to the FDDI unknown address to indicate that the contact with that neighbor was lost.

Duplicate Address Detection Protocol

The `smtd` also performs duplicate address detection (DAD) by monitoring the NIF request frames that arrive at this station. Because our hardware can copy an NIF request that we send and one that is sent from our UNA, we can determine whether that may be a duplicate MAC address on the ring. By receiving our own NIF request frames back, we can check the Transaction ID of the frame to ensure that it is the same Transaction ID that we sent; if it is not, a station on the ring must have the same MAC address as ours.

Another way you can check for duplicates is by ensuring that when we receive a NIF response frame to one of our NIF request frames, the Address Recognized (A bit) is not set. If it is, this would also indicate a duplicate address, because the A bit being set indicates some other station on the ring has recognized the destination address of the response frame as their own.

If the `smtd` detects a duplicate address condition on the FDDI ring, it notifies the FDDI driver (`fd.c`) by using an `ioctl` request indicating that the duplicate address test results have changed and that the new state is `failed`. When the driver receives this notification, it will first notify the CMT process on the channel adapter of the duplicate address condition. The CMT process causes the FDDI interface to disconnect from the ring. The driver also marks the logical link control (LLC) services unavailable, which prevents any further traffic from leaving the Cray Research system. When the ring becomes operational again, the CMT process notifies the FDDI driver and LLC services are reenabled. During the time the ring is not operational, the `smtd` continues to try to send the NIF request frames. These attempts will fail because the ring is down, but will not fail after the ring comes back up.

Status Reporting Protocol

The `smtd` also supports the Status Reporing Protocol of the SMT standard. When one of several predefined conditions exists or events occur, the `smtd` sends a status report frame (SRF) to the ANSI-owned IEEE SRF multicast address (`80:01:43:00:80:00`). Each SRF frame contains all of the currently active conditions and any new events that have occurred since the last SRF was sent. The protocol limits the frequency of sending the SRF announcements to a maximum of one frame every 2 seconds. The protocol also defines a report threshold, which can vary from a minimum of 2 seconds to a maximum of 32 seconds. This threshold is the amount of time the FDDI station can wait before sending an SRF frame when conditions exist or events occur that require it. Setting this time to a higher value is useful if conditions persist or events occur at an unusually high rate because a higher value minimizes the number and frequency of SRF frames added to network traffic load.

Socket Utility

To provide a mechanism for other processes that want to obtain information about FDDI stations and FDDI station management information, `smtd` supports a UDP-based socket over which it can receive requests and send replies. You can use this socket for such things as sending SMT request frames, receiving SMT response frames, obtaining the MAC addresses of upstream and downstream FDDI stations, and obtaining a copy of all FDDI MIB variables for the station.

After a process has opened a utility socket connection to the `smtd`, it can then send requests to the daemon over the socket and receive the results of the requests over the socket. The format of the structures and the contents of the structures that are used across the UDP socket is specified in the `smtd` header files `defines.h` and `typedefs.h`. The following shows the format of the socket structures (for a more accurate definition of the structures, see the header files themselves):

```
/*
 *      Request-Response packet format
 */
typedef struct {
        QueueElement                    qe;
        int                             len;
        int                             fromlen;
        struct sockaddr_in              from;
        union {
                int     request;
                int     response;
        } rr;
        union {
                util_xmtframeparms      xmt;
                util_rcvframe           rcv;
                util_macs               macs;
                util_mibvar             mib;
                util_cancel             cancel;
        } arg;
} util_packet;

/*
 *      Transmit Frame packet format
 */
typedef struct {
        uchar   fc;
        uint    dest;
        uchar   class;
        uchar   type;
        uint    tid;
} util_xmtframeparms;

/*
 *      Receive Frame packet format
 */
typedef struct {
        QueueElement    qe;
        int             age;
        int             hold;
        int             tid;
        int             len;
        smt_frame       frame;
} util_rcvframe;
```

```
/*
 *      Get MAC addresses packet format
 */
typedef struct {
        uint    ME;
        uint    UNA;
        uint    DNA;
} util_macs;

/*
 *      Get FDDI MIB variables packet format
 */
typedef struct {
        uint    dummy;          /* To be determined */
} util_mibvar;
```

The `qe`, `len`, `fromlen`, and `from` fields of the `util_packet` are all used internally by `smtd` and do not have to be manipulated by the user.

The `rr` field of the `util_packet` is used for request and response codes.

The `arg` field holds the actual data associated with the request or response.

When requests are compiled, they can then be sent to `smtd` by using a `sendto`(2) system call. You can obtain the response to the request by using a `recvfrom`(2) system call.

To test the response for success, compare the `rr.response` field to `UTIL_RSP_OK`.

The following section describes the way in which the socket utility requests are formed and the contents of the responses for all of the different types of operations across the socket.

To Transmit an SMT Request Frame

- Set the `rr.request` field to `UTIL_REQ_XMT_FRAME`.
- Set the `arg.xmt.fc` field to the desired FDDI frame control.
- Set the `arg.xmt.dest` field to the destination station's MAC address (in MSB form).
- Set the `arg.xmt.class` field to the frame class.
- Set the `arg.xmt.type` field to the frame type.
- Set the `arg.xmt.tid` to the Transaction ID of the user's request. This ID should be a number that is unique to each request that is sent by the process and unique to the process itself. Because many programs can communicate with the `smtd` over the socket simultaneously, this is the only mechanism that is available to maintain order to all of the various requests and responses that will be seen by the `smtd`. The `smtd` uses the Transaction ID to differentiate response frames that it receives due to request frames that it sends, from those response frames that are received because of request frames sent by users communicating over the socket utility.

To Receive an SMT Response Frame

- Set the `rr.request` field to `UTIL_REQ_RCV_FRAME`.
- Set the `arg.rcv.tid` to the Transaction ID of the request that generated this SMT response frame. (This must be the same Transaction ID that was in the transmit request.)
- Set the `arg.rcv.hold` to the number of seconds the user wants to wait for a response frame to arrive. If no response frame arrives with the desired Transaction ID within this much time, the request will fail with a `UTIL_RSP_RCV_TIMEOUT` error. When a response frame with the designated Transaction ID is received, it will be copied into the `arg.rcv.frame` field of the request, the length of the frame will be placed into the `arg.rcv.len` field, and the response delivered to the user. If no receive request is posted, frames that arrive are queued. These frames are held until either a receive request with the proper Transaction ID is received or a time-out occurs. If the time-out occurs before the receive request is received, the frame(s) will be discarded.

To Cancel a Receive Request

- Set the `rr.request` field to `UTIL_REQ_CANCEL_RCV`.
- Set the `arg.cancel.tid` to the Transaction ID of the request to cancel.

To Obtain the MAC Addresses of This Station and Its Neighbors

Set the `rr.request` field to `UTIL_REQ_GET_MACS`. The addresses that are returned are in MSB form.

Get FDDI MIB Variables

This request is not yet defined.

SEE ALSO

`fddimap`(8), `xfddimap`(8)

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

The ANSI documents for FDDI:

FDDI MAC (Media Access Protocol) Specification (FDDI-MAC), document number X3.139–1987, November 5, 1986

FDDI PHY (Physical Layer Protocol) Specification (FDDI-PHY), document number X3.148–1988, June 30, 1988

FDDI PMD (Physical Medium Dependent) Specification (FDDI-PMD), document number X3.166–1990, September 28, 1989

FDDI SMT (Station Management) Specification (FDDI-SMT), document number X3T9.5/84-49 Rev 6.2, May 18, 1990

Other documents related to FDDI:

RFC 1188 Proposed standard for the transmission of IP datagrams over FDDI networks. October 1990: D. Katz

Logical Link Control Specification (802.2 LLC), document number 802.2–1985, July 16, 1984

NAME

`snmpd` – Simple network management protocol agent/server for Cray Research systems

SYNOPSIS

`/etc/snmpd` [`-b` *size*] [`-d`] [`-D`] [`-P` *portnumber*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `snmpd` command is a server that acts as a management agent, implementing the Simple Network Management Protocol (SNMP). After receiving a message, it authenticates the request, attempts the operation, and then returns a response.

The managed objects that `snmpd` manipulates are defined in the `snmpd.defs` file, which is kept in the system administrator's area. These objects conform to the Internet standard management information base (MIB), which is defined in RFC 1213 as MIB-II. The rules used for naming and describing objects are taken from the Internet standard structure of management information (SMI), which is defined in RFC 1155 and enhanced by RFC 1212.

The `snmpd` command reads the `/dev/kmem` file to obtain most objects. Some objects can be specified in the `/etc/snmpd.conf` configuration file instead, which can be read only when the daemon starts. The `/etc/snmpd.conf` configuration file provides a mechanism to list all management stations that can access `snmpd`.

The `snmpd` command accepts the following options and operands:

`-b` *size* Specifies the "too big" packet size. Using this option determines the largest packet size that `snmpd` will accept.

`-d` Turns on debugging. This option specifies that the daemon will not be run in the background; it enables debug messages to print to the screen.

`-D` Turns on debugging and packet dumps. Using this option enables the `-d` options and also prints out each packet that is received and sent.

`-P` *portnumber*
Listens on port *portnumber*. Using this option overides the default port 161 to which `snmpd` listens for requests.

Configuration

The `/etc/snmpd.conf` file contains the following customization directives:

`community` *name* [*address*] [*access*] [*view*]

Defines an SNMP community called *name* and indicates the level of *access*.

The *address* token is either a host name, an IP address, or a network address (using dot notation). If *address* is present and you use a value other than 0.0.0.0, incoming messages that claim to belong to the specified community must come from this address.

If the *access* token is present, its value is `readOnly`, `readWrite`, or `none`; it defaults to `readOnly`.

If the *view* token is present, it is an object identifier that names the corresponding view of MIB objects that this community can access; otherwise, the view contains all variables known to the agent.

`logging` *attribute=value*

Sets the logging parameters according to the values specified for attributes. Attributes are as follows:

`file`	Specifies the file name for the log, whose value is interpreted relative to the ISODE logging area unless the value starts with a slash.
`size`	Specifies the maximum file size (an integer value in Kbytes) that the log should be allowed to grow.
`slevel`	Specifies which one of the following events should be logged: `none`, `fatal`, `exceptions`, `notice`, `trace`, `pdus`, `debug`, or `all`.
`dlevel`	Specifies the events (listed under `slevel`) that should not be logged.
`sflag`	Specifies which one of the following logging options should be enabled: `close` (to close the log after each entry), `create` (to create the log if it does not already exist), `zero` (to reset the log if the size is exceeded), and `tty` (to log events to the user's terminal in addition to the file).
`dflags`	Specifies the logging options (listed under `sflag`) that should be disabled.

`trap` *name address* [*view*]

Defines a trap sink for the SNMP community called *name*, on the indicated address, for the indicated view. *address* can be a host name, an IP address, or a network address (using using dot notation). If you omit *view*, a view is not named for the trap sink. Note: UDP must be able to reach the trap sink.

`variable` *name value*

Sets the *name* variable to the indicated *value*. Following are the variables that can be set:

`sysDescr`	Can be set to a string value that describes the management agent.
`sysObjectID`	Can be set to an object identifier value that identifies this system.
`sysContact`	Can be set to a string value that identifies the person who is responsible for the node.

`sysName`	Can be set to a string value that gives an administratively assigned name for the node.
`sysLocation`	Can be set to a string value that describes the location of the node.
`sysServices`	Can be set to an integer that describes the services that are offered by the node.
`snmpEnableAuthTraps`	Can be set to either `enabled` or `disabled`, which enables or disables the generation of `authenticationFailure` traps, respectively.

See RFC 1213 for a more thorough explanation of these objects.

`variable interface` *name attribute*=*value*...

Sets attributes for the specified interface, *name*. The *name* token is the same interface name that results from the `netstat -i` command. You can set the following attributes for each interface:

`ifType`	Can be set to an integer value that identifies the type of interface.
`ifSpeed`	Can be set to an integer value that describes the speed of the interface.

See RFC 1213 for a more thorough explanation of these objects.

`view` *name* [*subtree* ...]

Determines the part of the MIB that can be accessed. *name* specifies for which view the MIB is being accessed. *subtree*... specifies the MIB variables that can be accessed. If no subtrees are listed, the view contains all variables known to the agent.

The following parameters are supported for compatability with previous versions of the configuration file.

`readall`;

Allows any management station to access the agent.

`community` *community_name address/hostname type*;

Specifies additional options for a community. The *community_name* argument specifies the name of a community (in ASCII); *address/hostname* is either the Internet address (in dot notation format) or the name of the host; *type* can be one of the following values:

`read-only`	Allows read-only access
`traps`	Specifies the hosts to which `trap` messages will be sent

A value of `0.0.0.0` in the *address/hostname* field allows any host in a community to access `snmpd`.

`traps enabled`;

Tells `snmpd` to send traps. You must define a community to specify the management station to receive traps.

`authentication_traps enabled`;

Sends authentication traps when an unknown community attempts to query `snmpd`. If you specify `readall`, no authentication traps are generated. If an unknown host tries to use a valid community, an authentication trap is sent.

Debug Operation

To enter debug mode, specify the `-d` or `-D` option. In debug mode, all logging activity is displayed on the user's terminal in verbose mode. In addition, when you specify the `-D` option, each SNMP packet that the agent receives and sends is dumped in hexadecimal format.

You can use the `-b` option to specify the maximum message size that is supported by the daemon. (This is useful for testing how management stations recover from `tooBig` errors.)

You can use the `-P` option to specify a port other than 161. You can use the same option with the SNMP commands to allow `snmpd` to be debugged without affecting the system `snmpd`.

NOTES

The names of the objects in the `snmpd.defs` file are case sensitive. This was necessary to improve the efficiency of the hashing algorithm used for object lookup.

FILES

`/etc/services`	Network service name database
`/etc/snmpd.conf`	Configuration file
`/etc/snmpd.defs`	MIB definitions
`/etc/snmpd.pid`	Daemon PID file
`/usr/spool/osi/snmpd.log`	Log file

SEE ALSO

RFCs 1155, 1212, 1213, and 1157

NAME

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

SYNOPSIS

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

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

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

The `snmproute` command accepts the following options:

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

SEE ALSO

`snmpd`(8)

NAME

`spaudit` – Changes security auditing criteria

SYNOPSIS

`/etc/spaudit` [`-c`] [`-e` *enableopts*] [`-d` *disableopts*] [`-l`] [`-s`]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `spaudit` command allows an appropriately authorized user to make a change in the security audit log edition. On a UNICOS system, it also allows an appropriately authorized user to change the selection of auditable events when the UNICOS operating system is running.

To change the security audit log edition automatically at specified time intervals, initiate a `cron` job that executes the `spaudit -c` command. If a security audit log exceeds its threshold before an edition change is requested, the security audit log daemon automatically creates a new edition of the security audit log. This prevents the possibility of an edition of the security audit log overflowing its disk space and causing the UNICOS operating system to panic.

On a UNICOS system, the `spaudit` command can be used in either single-user or multiuser mode to enable or disable logging options. This command cannot be used to both enable and disable security logging options at the same time. If both the `-e` and `-d` options are specified, an error message is issued and no action is taken. Any changes in logging options are recorded in the security audit log.

The `spaudit` command accepts the following options:

`-c` Forces a change in the security audit log edition.

`-e` *enableopts*
Enables the specified logging option(s).

`-d` *disableopts*
Disables the specified logging option(s).

`-l` Lists and describes the valid logging options.

`-s` Lists the state (on or off) of valid logging options.

The following list contains valid logging options:

Option	**Description**
`all_nami`	All `mkdir`, `rmdir`, `link`, and `rm` calls
`all_rm`	All remove requests
`all_valid`	All access requests
`audit`	All security auditing criteria changes

`chdir`	All change directory requests
`config`	All UNICOS configuration changes
`crl`	Cray/REELlibrarian activity
`dac`	Discretionary access control changes
`discv`	Discretionary access violations
`filexfr`	All file transfer requests
`ipnet`	All IP layer activities
`jend`	End-of-job
`jstart`	Start-of-job
`linkv`	All link (`ln`) violations
`mandv`	Mandatory access violations
`mkdirv`	All make directory (`mkdir`) violations
`netcf`	Network configuration changes
`netwv`	Network violations
`nfs`	All NFS activity
`nqs`	NQS activity
`nqscf`	NQS configuration changes
`object_path`	Object's full path name on accesses
`operator`	Operator actions
`physio_err`	Currently not used
`priv`	Use of privilege
`removev`	All remove violations
`rmdirv`	All remove directory (`rmdir`) violations
`secsys`	All security system call requests
`setuid`	All setuid requests
`shutdown`	System shutdown requests
`startup`	System start-up requests
`state`	Defines if security audit log is on (`1`) or off (`0`)
`sulog`	All `su` attempts
`tapes`	Tape activities
`time_change`	System time change

`trust`	Trusted process activity
`user`	User name for failed login attempts

NOTES

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

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

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

EXAMPLES

Example 1: The following is an example of a `cron` entry that allows `spaudit` to execute at midnight each night to force a security audit log edition change:

```
crontab
0 0 * * * spaudit -c
```

To interactively force a security audit log edition change, execute the `spaudit -c` command.

Example 2: The following example shows the resultant display when using the `-l` (which lists the valid logging options and `-s` (which lists the state of the options) options:

```
$ spaudit -l -s
state           ON      security auditing state
all_nami        OFF     all nami requests
all_valid       OFF     all valid accesses
all_rm          OFF     all rm requests
audit           ON      audit criteria changes
chdir           ON      chdir requests
config          ON      config changes (UNICOS)
crl             OFF     Cray/REELlibrarian
dac             ON      discretionary access control changes
discv           ON      discretionary violations
filexfr         ON      file transfer requests
io_err          OFF     I/O errors
ipnet           ON      IPnet layer activities
jend            ON      job end
jstart          ON      job initiation
linkv           ON      link violations
mandv           ON      mandatory violations
mkdirv          ON      mkdir violations
netcf           ON      network config changes
netwv           ON      network violations
nfs             OFF     NFS activity
nqs             ON      NQS activity
nqscf           ON      NQS config changes
operator        ON      operator actions
object_path     ON      object path tracking
priv            OFF     privilege use
removev         ON      remove violations
rmdirv          ON      rmdir violations
secsys          ON      security syscalls
shutdown        ON      system shutdown
startup         ON      system startup
time_change     ON      system time change
setuid          ON      setuid requests
sulog           ON      su requests
tapes           OFF     tape activity
trust           ON      trusted process activity
user            OFF     user password on login fail
```

Example 3: The following example shows the use of the `-d` option, which disables the `nqs` and `nqscf` options. The `-s` option is then executed to show the state of the options:

```
$ spaudit -d nqs,nqscf
$ spaudit -s
state           ON
all_nami        OFF
all_valid       OFF
all_rm          OFF
audit           ON
chdir           ON
config          ON
crl             OFF
dac             ON
discv           ON
filexfr         ON
io_err          OFF
ipnet           ON
jend            ON
jstart          ON
linkv           ON
mandv           ON
mkdirv          ON
netcf           ON
netwv           ON
nfs             OFF
nqs             OFF
nqscf           OFF
operator        ON
object_path     ON
priv            OFF
removev         ON
rmdirv          ON
secsys          ON
shutdown        ON
startup         ON
time_change     ON
setuid          ON
sulog           ON
tapes           OFF
trust           ON
user            OFF
```

Example 4: The following example shows the use of the `-e` option, which enables the `tapes`, `nqs` and `nqscf` options. The `-l` and `-s` options are then executed to show the description and the state of the options:

```
$ spaudit -e tapes,nqs,nqscf
$ spaudit -ls
state           ON      security auditing state
all_nami        OFF     all nami requests
all_valid       OFF     all valid accesses
all_rm          OFF     all rm requests
audit           ON      audit criteria changes
chdir           ON      chdir requests
config          ON      config changes (UNICOS)
crl             OFF     Cray/REELlibrarian
dac             ON      discretionary access control changes
discv           ON      discretionary violations
filexfr         ON      file transfer requests
io_err          OFF     I/O errors
ipnet           ON      IPnet layer activities
jend            ON      job end
jstart          ON      job initiation
linkv           ON      link violations
mandv           ON      mandatory violations
mkdirv          ON      mkdir violations
netcf           ON      network config changes
netwv           ON      network violations
nfs             OFF     NFS activity
nqs             ON      NQS activity
nqscf           ON      NQS config changes
operator        ON      operator actions
object_path     ON      object path tracking
priv            OFF     privilege use
removev         ON      remove violations
rmdirv          ON      rmdir violations
secsys          ON      security syscalls
shutdown        ON      system shutdown
startup         ON      system startup
time_change     ON      system time change
setuid          ON      setuid requests
sulog           ON      su requests
tapes           ON      tape activity
trust           ON      trusted process activity
user            OFF     user password on login fail
```

FILES

`/usr/adm/sl/slogfile`	Security log
`/usr/include/sys/slrec.h`	Format of security audit log record

SEE ALSO

`reduce`(8), `slogdemon`(8)

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

General UNICOS System Administration, Cray Research publication SG–2301

NAME

`spcheck` – Performs security checks on a UNICOS system

SYNOPSIS

`/etc/spcheck` [`-a`] [`-b`] [`-B`] [`-c`] [`-f` *filesystem*] [`-g`] [`-G`] [`-l`] [`-p`] [`-q`] [`-Q` *file*] [`-r`] [`-s`] [`-w`]
`/etc/spcheck` [`-a`] [`-b`] [`-B`] [`-c`] [`-g`] [`-G`] [`-l`] [`-p`] [`-q`] [`-Q` *file*] [`-r`] [`-s`] [`-w`] [`-u` *user*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `spcheck` command performs a variety of security-relevant checks on a UNICOS system. This command can be used only by an appropriately authorized user. `spcheck` makes administrative and logical file system checks according to the selected options. If you run `spcheck` with no options, the `-a`, `-b`, `-g`, `-l`, `-p`, `-q`, `-s`, and `-w` options are executed by default.

The `-f` and `-u` options cannot be used together.

The `spcheck` command accepts the following options:

`-a` Lists users who have administrative categories.

`-b` Lists all setuid and setgid files in the `/bin`, `/usr/bin`, `/usr/lib`, and `/etc` directories.

`-B` Performs the same function as the `-b` option, but uses the directory list in `/etc/bincheck`. The `/etc/bincheck` file is a one-record file and has the following format:

directory1 directory2 directory3 . . . directoryn

`-c` Runs the `spfilck`(8) program in check mode (with the `-c` option), using `/etc/permlist` as input to `spfilck`.

`-f` *filesystem*
Performs a security check on the logical *filesystem* (for example, `/usr`). This option should be used in conjunction with the `-r`, `-s`, and `-w` options, and it cannot be used with the `-u` option.

`-g` Reports users in groups `root`, `adm`, `bin`, and `sys`. This option also runs the `grpck` (see `pwck`(8)) command.

`-G` Performs the same function as the `-g` option but uses the group list in `/etc/grpcheck`. The `/etc/grpcheck` file is a one-record file and has the following format:

group1 group2 group3 . . . groupn

`-l` Reports infrequently used login IDs (users who have not logged in for at least 14 days and users who have not logged in for 180 days) and lists `.profile` and `.cshrc` files that are writable by anyone.

`-p`	Reports users with duplicate user IDs, users who cannot change their passwords, and users whose passwords do not expire. This option also runs the `pwck`(8) command.
`-q`	Reports all users who have accumulated 10 or more `su`(1) command failures in any one day, as logged in the `/usr/adm/sulog` file.
`-Q` *file*	Performs the same function as the `-q` option, but it uses *file* instead of `/usr/adm/sulog` as the search log.
`-r`	Reports files in the system that have the world read permission set (ignores files in `/tmp`, `/usr/tmp`, and `/usr/spool`). This option is to be used in conjunction with the `-f` and `-u` options.
`-s`	Reports list of programs with either the setuid or setgid bits set. If you are root and you specify this option, `spcheck` also reports block/character special files not in `/dev`. This option is to be used in conjunction with the `-f` and `-u` options.
`-w`	Reports files in the system that have the world write permission set (ignores files in `/tmp`, `/usr/tmp`, and `/usr/spool`). This option is to be used in conjunction with the `-f` or `-u` options.
`-u` *user*	Performs security check on the specified *user*'s home directory. See the `-r`, `-s`, and `-w` options for security check options. This option cannot be used with the `-f` option.

NOTES

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

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

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

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

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

FILES

```
etc/bincheck
etc/grpcheck
usr/adm/sulog
```

SEE ALSO

`grpck`(8), `pwck`(8), `spfilck`(8)

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

General UNICOS System Administration, Cray Research publication SG–2301

NAME

`spdev` – Sets, clears, and displays objects

SYNOPSIS

`/etc/spdev` [`-l` *minlvl*] [`-c` *mincmp*] [`-L` *actlvl*] [`-K` *actcmp*] [`-u` *maxlvl*] [`-v` *maxcmp*] [`-m`] [`-s`] [`-p`] *devf* ...
`/etc/spdev -C` [`-p`] *devf* ...

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `spdev` command sets, clears, and displays the security labels on objects. `spdev` can be executed by anyone; kernel-level restrictions limit the actions available to users who are not appropriately authorized.

The `spdev` command accepts the following options:

`-l` *minlvl*	Sets the minimum level for the requested devices to *minlvl*. The default minimum level is 0. It is not possible to set the minimum level on files, so this option is ignored for nondevice objects.
`-c` *mincmp*	This option is provided for future use in setting minimum compartments on devices. Because the device labeling mechanism currently does not support a nonzero set of minimum compartments on devices, this option is recognized but not implemented.
`-L` *actlvl*	Sets the active level for single-level objects to *actlvl* and sets the active level for multilevel files. The active level on multilevel devices is always set to the maximum level by the kernel, so this option has no effect on multilevel device labels. The default active level is 0.
`-K` *actcmp*	Sets the active compartments for single-level objects to *actcmp* and sets the active compartments for multilevel files. The active compartments for multilevel devices are always set to the authorized compartments by the kernel, so this option has no effect on multilevel device labels. The default active compartment set is the empty set.
`-u` *maxlvl*	Sets the maximum level for the requested devices to *maxlvl*. For multilevel devices, the kernel also sets the active level to *maxlvl*. The default maximum level is 0. It is not possible to set the maximum level for files, so this option is ignored for nondevice objects.
`-v` *maxcmp*	Sets the authorized compartments for the requested devices to *maxcmp*. For multilevel devices, the kernel also sets the active compartments to *maxcmp*. The default authorized compartment set is the empty set. It is not possible to set the authorized compartments for files, so this option is ignored for nondevice objects.
`-m`	Sets the requested objects to multilevel mode. The default mode is single-level.

`-s`	Sets the requested devices to the `ON` state. Setting the device state to `ON` enables nonprivileged access to single-level devices, and signifies to privileged software using multilevel devices that the device is now available for use. The default state is `OFF`. Once the state is set to `ON` (when using this option), the `-C` option must be used to clear the device and set its state to `OFF` again.
`-C`	Sets the device labels on the requested objects to the default settings described for each of the previously described options.
`-p`	Prints the device labels of the requested objects.

NOTES

When the path name supplied to the `spdev` command specifies a multilevel symbolic link (the name of a multilevel directory), the attributes are changed only on the root of the multilevel directory tree. In the case of access control lists (ACLs), this affects all subsequently created labeled subdirectories. In any case, the attribute change does not affect existing labeled subdirectories. To set attributes on the existing labeled subdirectories, you must specify the path names of the existing labeled subdirectory found in the root of the multilevel directory to the `spdev` command.

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

Active Category	**Action**
`system`, `secadm`	Allowed to set security attributes of any object.
`sysadm`	Allowed to set security attributes of any object, subject to security label restrictions on the object's path. Shell redirected I/O is subject to security label restrictions.

If the `PRIV_SU` configuration option is enabled, the super user is allowed to set security attributes for any object.

MESSAGES

`spdev` writes the following error messages to standard error:

`spdev: Stat failed on` *objects*
The device *devf* does not exist or user does not have access.

`spdev: Cannot secstat` *objects*
The user does not have the security privileges necessary to access the device.

`spdev: Invalid security level` or `Invalid compartments`
If the `MLS_OBJ_RANGES` configuration option is enabled, and the requested device security levels or compartment settings are not within UNICOS system security settings.

`spdev: Unable to set label on` *objects*
The user is not authorized to change the label on the object, or the requested label is not valid.

SEE ALSO

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

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

General UNICOS System Administration, Cray Research publication SG–2301

NAME

`spfilck` – Checks and sets file ownership, access, and security parameters

SYNOPSIS

`/etc/spfilck -c` [`-f` *file*] [`-q`]
`/etc/spfilck -s` [`-f` *file*] [`-q`]
`/etc/spfilck -m` [`-f` *file*] [`-q`]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `spfilck` command monitors and controls ownership, mode, and security parameters of files. It reads or creates a control file, `/etc/permlist` (see the `-f` option description), which defines the actual or desired file attribute (owner, mode, security) definition. `spfilck` can set from, check against, or make `/etc/permlist`.

The `spfilck` command operates in one of three modes:

- Checks the owner, group owner, access modes, security levels, security compartments, integrity class, categories, and integrity flags against a master list of files, and flags any discrepancies (with the `-c` option)
- Sets the owner, group owner, access modes, security levels, security compartments, integrity class, categories, and integrity flags for a list of files (with the `-s` option)
- Creates a master list of files (with the `-m` option)

The `spfilck` command accepts the following options. Only one of the `-c`, `-s`, and `-m` options may be used on a command line.

`-c` Checks file parameters against input from the `/etc/permlist` file. An input line from `/etc/permlist` is in the following format:

owner group mode level compartments class categories flags files

The components of the input line are the following:

owner	Owner name.
group	Group name.
mode	Octal representation of the user, group, and other permissions.
level	Decimal representation of the file's security level.
compartments	Either a list of valid compartments separated by commas or the word `none`.
class	Decimal representation of the file's integrity class. This value is no longer used.

categories Either a list of valid categories separated by commas or the word `none`. This value is no longer used.

flags Either a list of valid integrity flags separated by commas or the word `none`.

files A nonempty list of path names that are compared against the attributes previously described. The names can include the following substitution characters, whose meaning is analogous to their use in `ed`(1): `*`, `.`, `[`, and `]`.

`-s` Sets file attributes according to `/etc/permlist`. The input from `/etc/permlist` is in the same format as that for the `-c` option. `spfilck` attempts to set the file attributes for the *files*, the format of which is identical to the `-c` option.

`-m` Reads a list of files from standard input or from those specified with the `-f` option and writes the current file attributes to standard output in the `/etc/permlist` format to be used by `spfilck` with the `-c` and `-s` options. The input line used by `spfilck` is in the following format:

file [*file*]

This is identical to the corresponding `-c` file name definition.

`-f` *file* Causes the `-s` and `-c` options to use *file* instead of `/etc/permlist` for input against which to check or set file parameters, and `-m` to use *file* instead of standard input for reading file names.

`-q` Causes `spfilck` not to perform the character substitution described above in the explanation of the *file* field of the `-c` and `-s` options. This is useful when used with the `-c` and `-s` options, which use the output of an `-m` option run for input. The `-m` option output does not insert any special characters in the *file* field of its output, and this precludes the necessity of checking for them if `-c` or `-s` are used with `-m` output as their input. `-m` might contain '.', '*', '[', or ']' in its output. If it does, this means that the literal character is used in the actual path name. Therefore, you do not want character substitution (for example, 'file *' means that the file name is really 'file*.')

NOTES

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

Active Category	**Action**
`system`, `secadm`	Allowed to access all files.

On a UNICOS MLS system with `PRIV_SU` enabled, the super user is allowed to access all files.

`*` does not have the same meaning in `ed`(1) as in `sh`(1). For example, `/bin/*` matches `/bin/a` but not `/bin/aa`, and `/bin/.*` matches both `/bin/a` and `/bin/aa`. Standard output receives a list of any files that fail the checks performed.

FILES

`/etc/permlist`

SEE ALSO

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

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

General UNICOS System Administration, Cray Research publication SG–2301

NAME

`spmap` – Reads or generates and writes a physical disk spare map

SYNOPSIS

`spmap [-r]` *special*
`spmap [-w]` *special*

IMPLEMENTATION

Cray PVP systems with I/O subsystem model E

DESCRIPTION

With no options selected, the `spmap` command reads physical device flaw information from standard input and writes to standard output a formatted spare sector map for the specified special device. The flaw information read is of the same format written to the standard output by the `ift`(8) command. The options can be used to read or write the spare sector map from or to a specified slice.

The `spmap` command accepts the following options:

`-r` Reads the spare sector map from the slice described by the character special file *special*. In this mode, no data is read from standard input.

`-w` Writes the spare sector map to the slice described by the character special file *special*. The spare sector map is generated from flaw information read from standard input. The `-w` option initializes only the spare sector map on the disk. The incore spare map is initialized only at device open time, normally by means of a file system mount.

The `-r` and `-w` options are mutually exclusive.

By convention, the special files describing the spare disk slices are described by the character special files in the `/dev/spare` directory. Typically, they are named after the physical device on which they reside.

EXAMPLES

Example 1: The following two commands write a spare sector map on device 0130.

```
ift /dev/ift/0130 | spmap -w /dev/spare/0130

cat /etc/aft/0130 | spmap -w /dev/spare/0130
```

Example 2: The following command reads the spare sector map from device 0130:

```
spmap -r /dev/spare/0130
```

SEE ALSO

`ift`(8), `mkspice`(8)

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

NAME

`spnet` – Manages NAL, WAL, and IP Security Options (IPSO) maps stored in network security tables

SYNOPSIS

`/etc/spnet` [`-f` *filename*] [`-d`] [`-n`] [`-q`] [`-v`] *command* `-`*table* [*table_args*]

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `spnet` command is a combined tool for managing security network configuration information. The `spnet` command currently maintains the network access list (NAL), the workstation access list (WAL), and the Internet Protocol Security Options (IPSO) mapping tables. It converts between external human-readable text and kernel-internal binary forms of the information. It adds, deletes, and lists the binary information in the kernel.

The typical use of the `spnet` command is to load all of these tables from the default input file when the UNICOS system is started, but it can be used with a small input file that contains a few entries to make a run-time change to the configuration. The `spnet` command reads a text file (`/etc/config/spnet.conf` by default) that contains descriptions of any or all of these tables and performs the operations specified on the command line.

The `spnet` command recognizes shortened option names. The following options are accepted:

`-d`[`efaults`] When used with the `list` command for the NAL table, causes `spnet` to list all the NAL field values. Normally, fields using default values are not listed.

`-f`[`ile`] *filename*
Specifies text files other than `/etc/config/spnet.conf`.

`-n`[`umeric`] Prints IP addresses in dotted-decimal form.

`-q`[`uiet`] Suppresses display of entries that `spnet` is adding to or deleting from the kernel tables.

`-v`[`erbose`] When used with the `parse` command, displays the input file information in a canonical format. When used with the `spnet list` command, it provides a text display of the radix tree (kernel-internal form of the NAL and WAL tables).

command Specifies an operation to perform on the NAL, WAL, or IPSO maps. You can specify any of the following commands:

`add` Adds an entry or entries from `/etc/config/spnet.conf` or the specified text file to the kernel table. Specify the entries in the *table_args* argument.

`delete` Removes an entry or entries from the kernel table. Specify the entries in the *table_args* argument.

`export` Reads the `*.cfg` files (listed under the `import` command) and writes to the `stdout` file the `spnet` input language version of the information. This command is not intended for use outside of the installation and configuration menu system because compatibility might not be maintained in the next UNICOS release.

`help` Displays usage information.

`import` Reads the input file and produces the following files for the installation and configuration menu system: `ssnalset.cfg`, `sswalset.cfg` `sscipsomap.cfg`, and `ssspnet.cfg`. This command is not intended for use outside of the installation and configuration menu system because compatibility might not be maintained in the next UNICOS release.

`list` Lists the kernel table.

`parse` Parses the input file, but it performs no actions except to report errors.

`resolve` Lists individual NAL or WAL entries from the kernel table. Specify the entries in the *table_args* argument.

`-`*table* Specifies the type of table on which to operate. Valid types are as follows:

`-nal` Indicates NAL operations for Internet Protocol (IP) addresses

`-wal` Indicates WAL operations for IP addresses

`-map` Indicates IPSO map operations

table_args Specifies table arguments for the `add`, `delete`, `convert`, and `resolve` `spnet` commands. The format of *table_args* is as follows:

[*address_list*] [*ifile*] [*ofile*]

For the `add`, `delete`, and `resolve` commands, *table_args* is an address list based on the type of table specified with the *table* argument. If you specify more than one *table* argument, you must specify a separate *table_args* entry for each one. You must specify at least one of the following *address_list* items with the `add`, `delete`, and `resolve` commands:

address_list
Specifies one of the following:

- A list of addresses from the NAL or WAL definitions in the `/etc/config/spnet.conf` file or the kernel.
- A list of addresses from the alternate input file specified by the `-f` *filename* option.
- A list of DOI numbers for the IPSO maps defined in the input file.
- One or more of the following address indicators:

`-net`	Limits the selection of NAL or WAL addresses to network entries
`-host`	Limits the selection of NAL or WAL addresses to host entries
`-ip`	Limits the selection of NAL or WAL addresses to IP protocol family entries
`-station` or	
`-cipso`	Limits the selection of IPSO maps to CIPSO entries
`-basic`	Limits the selection of IPSO maps to basic entries
`-all`	Specifies all entries within the constraints set by the modifiers in the preceding list. Use of this indicator in an `add` command saves kernel space because several addresses can be specified to share a single NAL or WAL entry.

For the `convert` command, *table_args* is the following argument:

ifile	Specifies the name of the UNICOS 7.0 binary-format file to be converted to `spnet` input text format.

Following are examples of the `spnet` command:

```
spnet add -map -all
spnet add -nal -ip -all
spnet add -wal -net -all
spnet -file special.nal add -nal pecan poplar14 pecan0-net
spnet delete -map -basic
spnet list -map

spnet convert -wal /etc/hosts.wal >> /etc/config/spnet.conf
```

Input File

The input file for the `spnet` command (`/etc/config/spnet.conf` or a file specified by the `-f` *filename* option) contains a readable text language in which NAL, WAL, and IPSO map specifications are given. A simple grammar specification for the language is provided at the end of this man page.

Comments can be placed on any line. They begin with the `!` character and continue until the end of the line. Numeric values are scanned by using the `strtol`(3C) function, which means that numbers beginning with 0 are octal, numbers beginning with 0x are hexadecimal, and numbers beginning with a nonzero digit are decimal. Capitalized words in the following subsections describe rules in the grammar. The remaining subsections describe the following elements of the input language:

- Addresses
- Labels
- NAL entries
- WAL entries
- IPSO map entries

Addresses

One or more addresses are associated with each WAL or NAL entry. Each address applies to the IP family. In the `spnet.conf` file, a single NAL or WAL definition can have several addresses associated with it. In addition to enhancing readability and organization of the file, this can result in space savings in the kernel. Use of the `-all` indicator in an `add` operation causes `spnet` to load entries into the kernel in a way that retains the multi-address association so that only one record is stored per definition.

An IP address can apply to either a single host or to a network (default is host). An IP network address can have a subnet mask, which the NETMASK specifies in dotted-decimal form. The IP_ADDRESS can be either a name or numeric address in dotted-decimal form. Names are looked up in the system host database and then in the system network database. The special network name `default` applies to any host that is not explicitly addressed. Separate default entries exist for `ip net` and for `station`.

The following are examples of address specifications:

```
troll.cray.com
128.162.28.3
ip net cray-net
ip net 128.162.28.80 ( 255.255.255.240 )
ip host 127.0.0.1
localhost
station "NSC,V3,128,0"
```

Labels

Labels contain a level component and a compartments component. The level can be either a name (see `secnames`(3C)), including `syslow` and `syshigh`, or an integer from 0 through 16. The compartments can be names (see `secnames`(3C)), an integer preceded by the # character that represents a bit position as a power of 2, or an integer, ortal, or hex bit mask.

The following are examples of levels:

```
level1
16
syshigh
syslow
```

The following are examples of compartments:

```
secadm
sysadm
human_resources
director
#0, #1, #2, #3, #4
037
0x1f
```

NAL Entries

The network access list (NAL) defines the hosts that are granted access to a UNICOS system. The NAL is an essential mechanism for maintaining the UNICOS security policy in a network environment. Only those hosts that are listed in the NAL, either as a single node or as a member of a network, can communicate over the network with the UNICOS system. A packet from any other host is simply dropped. The NAL holds a collection of attributes for each host, which allows the UNICOS administrators to adjust for the various security implementations and levels of trust of each of the nodes on the network.

The following are the available NAL attributes:

- `name`
- `authority-in` or `auth-in`
- `authority-out` or `auth-out`
- `authority`
- `class`
- `domain` or `doi`
- `ipso`
- `label`
- `min label`
- `max label`
- `modes` or `mode`

The `name` attribute attaches an optional name to a NAL entry. It serves no function as input to the `spnet` command, but it can be useful for labeling the various security domains in a large network. The installation and configuration menu system requires this name, but a default name is generated automatically if it is missing when the `spnet` input file is imported. The name is used to associate NAL entries with network addresses because the creation of NAL entries is a separate step in the installation and configuration menu system.

The name is stored in kernel NAL entries and printed in the `list` output, but the kernel does not interpret the name. Only the first eight characters are saved.

The `authority`, `authority-in`, `authority-out`, `auth-in`, and `auth-out` attributes are used when the `ipso` attribute value is `basic`. These attributes specify the predefined protection-authority flags that are required to be included on input and output IP packets. The flags are specified as a comma-separated list of one or more of the following numeric values:

- 0x80 (`GENSER`)
- 0x40 (`SIOP-ESI`)
- 0x20 (`SCI`)
- 0x10 (`NSA`)
- 0x8 (`DOE`)

The `class` attribute specifies the level of trust of a remote host. The class value can be either uppercase or lowercase, and is one of the following values:

```
    class = D | C1 | C2 | B1 | B2 | B3 | A1;
```

The default value is `B1`.

The `ipso` attribute specifies the type of packet labeling in use between the remote host and the UNICOS system. It can be set to one of the following values:

Packet Type	**Description**
`none`	No packet labeling is specified. This is the default.
`basic`	Indicates that the IP Basic Security Option (IP BSO), defined in RFC 1108, will be used. Protection authority information is described in the preceding section for the `auth-in` and `auth-out` attributes. To select optional security level mapping, define and activate an IPSO map table with an identifier of 0 or the name `basic` (see the IPSO Map Entries subsection).
`cipso`	Indicates that the Common IP Security Option (CIPSO) defined by revision 2.2 of the Trusted Systems Interoperability Group CIPSO specification will be used. In this case, a Domain of Interpretation (DOI) is required in the `domain` attribute. An IPSO mapping table definition is also required (see "IPSO Map Entries" in a succeeding subsection).

The `label`, `min label`, and `max label` attributes specify the single label, or the label range at which a remote host is allowed access to the UNICOS system. When the `label` form is used, both the `min label` and `max label` are set to that value. The default `min` and `max` labels are level 0 and no compartments.

The `mode` or `modes` attribute specifies flags that have the following effects:

Mode	**Description**
`receive`	Allows Transmission Control Protocol (TCP) connections from the local system to the remote system to complete.
`send`	Allows TCP connections from the remote system to the local system to complete.
`trap`	Implementation of this flag is deferred to a future release.
`none`	No mode flags will be set.

If omitted, the `mode` attribute defaults to `send` and `receive`.

WAL Entries

The workstation access list (WAL) defines the individuals (user IDs) and groups (group IDs) granted access to the local UNICOS system from a specific remote node. The UNICOS security administrator can define the users and groups per node who are allowed to log in to or to access network services from each remote node in the network. The user IDs and group IDs are those of users and groups on the local UNICOS system. For example, whenever a user logs in to UNICOS, the login process references the WAL to validate the user’s right to access the UNICOS system from the workstation. The user ID and group ID are formatted with a period separating the `uid` and `gid`, as follows: `uid.gid`. A comma-separated list of allowed services follows the `uid.gid` pair; for example, `uid.gid = ftp, rexec, remsh.`.

The `name` attribute attaches an optional name to a WAL entry. It serves no function as input to the `spnet` command, but it can be useful for labeling the various security domains in a large network. The installation and configuration menu system requires this name, but a default name is generated automatically if it is missing when the `spnet` input file is imported. The name is used to associate WAL entries with network addresses because the creation of WAL entries is a separate step in the installation and configuration menu system.

The name is stored in kernel WAL entries and printed in the `list` output, but the kernel does not interpret the name. Only the first 8 characters are saved.

By default, all services are permitted to all users from a particular node if no entry exists in the WAL for that node. But when at least one entry exists for a node, a user from that node is granted remote access to UNICOS services only if there is a WAL entry for the user.

In the search for a WAL entry for a user, the applicable WAL entry applies if its address matches the user's remote node and it meets a stronger rule from the following list than any other entry (the rules are in order from strongest to weakest):

1. Its user ID and group ID match the user's user ID and group ID.
2. Its user ID matches the user's user ID and its group ID is the wildcard (`*`).
3. Its group ID matches the user's group ID and its user ID is `*`.
4. Its user ID and group ID are both `*` (this is the default entry).

The values for services controlled by WAL entries are as follows:

Service	**Description**
`all`	Allows all services.
`none`	Allows no service.
`ftp`	Allows `ftp`(1B) connection from the node to the Cray Research system.
`login`	Allows any kind of interactive access to UNICOS, including `remsh`(1B) interactive login access.
`lpd`	Allows `lpr`(1B) connection from the node.
`mail`	Allows mail connection from the node (deferred).
`nfs`	Allows NFS services to be accessed from the node (deferred).
`nqs`	Allows NQS transfers from the node.
`rexec`	Allows `rexec`(3C) connection from the node.
`rlogin`	Synonymous with `login`.
`rsh`	Allows `remsh`(1B) command access from the node, but not interactive access.
`telnet`	Synonymous with `login`.

IPSO Map Entries

To conserve space in the IP protocol, security levels and compartments are represented by numbers rather than by their ASCII equivalent. Internet Protocol Security Options (IPSO) maps are translation tables that allow the security administrator to control the numeric network representations and to adjust for differences in the implementation of labels among hosts on the network. Every incoming and outgoing packet from a UNICOS system to a particular host on the network has its label information translated through a designated IPSO map.

IPSO maps are numbered and optionally named. The map number, known as the `doi` number, is a 32-bit quantity that is included in every IP packet. A Domain of Interpretation (DOI) is a collection of hosts that share a common definition of label values and meanings. The UNICOS system validates the DOI number against the DOI value in the NAL entry for the remote host. When the `ipso` attribute in the NAL is set to `basic`, the DOI number is 0 implicitly. You can use the map domain name as a convenience within an `spnet` input file for associating NAL entries with maps. The installation and configuration menu system requires this name, but a default name is generated automatically when the `spnet` input file is imported if it is missing.

When you are creating an IPSO map, you must consider the following issues:

- The host-internal numeric level values range as follows: `syslow`, 0 through 16, `syshigh`.
- The host-internal numeric compartment values are each a power of 2 ranging from 2**0 to 2**62.
- The numeric network representations for levels range from 0 through 255, and for compartments, from 0 through 65534.
- Each local level or compartment value must correspond to only one network value.
- Each network level or compartment value must correspond to only one local value.
- Network administrators must ensure that network-wide mappings are nonpermuting; that is, a translation from host A to host B to host C to host A would result in the original label.
- DOI 0 supports four levels and no compartments.

Sample Input File

Following is a sample input file:

```
nal {
      ip net default {
            name = public;
            class = d;
            ipso = none;
            label = 0;
            mode = receive;
      }

      ip net 128.162.72 {
            name = pecannet;
            label = level4;
            ipso = none;
            class = c1;
      }

      station "NSC,V3,128,0" {
            name = station1;
            label = level4;
            authority = 0x80,0x10;
            ipso = none;
            class = c1;
      }

      cool {
            ipso = cipso;
            max label = level16, secadm, sysadm, sysops,
                  netadm, unicos, crayri, nqsdev, nqstst,
                  comp24, comp39, comp63;
            doi = yellow;
      }
}
```

```
        wal {
              ip net default {
                    *.* = none;
              }

              pecan-0net {
                    mgc.* = ftp, login, rsh;
                    *.group1 = all;
              }
        }
        map {
              cipso yellow = 42 {
                    levels {
                          Unclassified = 0;
                          Confidential = 1;
                          Secret = 2;
                          TopSecret = 3;
                    }
                    compartments {
                          usa = 1;
                          france = 5;
                          germany = 11;
                    }
              }
        }
```

Grammar

This subsection defines the grammar for the `spnet` command. The following conventions are used in this grammar:

`courier`	Indicates literal text, to be typed verbatim. Courier `UPPERCASE` denotes a grammar rule.
:	Indicates that a definition follows. Words for this symbol are "is defined as."
|	Indicates a succession of items from which to choose one item. The word for this symbol is "or."
ellipses (...)	Indicates 0 or more repetitions of the preceding item.
[ ]	Indicates that the enclosed item is optional.
italics	Indicates variable text or numbers, to be replaced with a value.
number..number	Indicates a range (for example, 0..16 indicates any number in the range 0 through 16).
!	Indicates a comment line. A comment can be placed on any line. Comments begin with a ! character and continue until the end of the line.
Name specification	The full specification of names is as follows: `[%a-zA-Z*][a-zA-Z0-9_.*-]*` This specification indicates one of the characters in the first set enclosed in brackets, plus any number of the characters in the second set.

Variable number specification

Variable numbers can be expressed in octal, decimal, or hexadecimal notation.

The following are reserved keywords in the `spnet` grammar:

```
A1             B2      cipso         ip      nal        {}
a1             b2      class         ipso    name       ;
auth-in        B3      compartments  label   net        ,
auth-out       b3      D             levels  none
authority      basic   d             map     receive
authority-in   C1      debug         max     send
authority-out  c1      doi           min     station
B1             C2      domain        mode    trap
b1             c2      host          modes   wal
```

Following are the simplified `spnet` grammar rules:

Grammar Rule **Definition**

```
FILE              :   SECTION ...

SECTION           :   NAL | WAL | MAP

NAL               :   nal { NAL_ENTRY ... }

WAL               :   wal { WAL_ENTRY ... }

MAP               :   map { MAP_ENTRY ... }

NAL_ENTRY         :   ADDRESS_LIST { NAL_ATTRIBUTES ... }

WAL_ENTRY         :   ADDRESS_LIST { WAL_ATTRIBUTES ... }

MAP_ENTRY         :   basic { [ MAP_LEVELS ] }
                  |   cipso [ [ domain ] [ domain_name = ] ] number
                              { [ MAP_LEVELS ] [ MAP_COMPARTMENTS ] }

ADDRESS_LIST      :   ADDRESS_VALUE [ , ADDRESS_VALUE ] ...

ADDRESS_VALUE     :   [ ip [ host ] ] IP_ADDRESS
                  |   [ ip [ net ] ] IP_ADDRESS [ ( NETMASK ) ]

IP_ADDRESS        :   host_or_network_name_or_dotted_IP_address

NETMASK           :   dotted_IP_address_mask

NAL_ATTRIBUTES    :   name = name ;
                  |   auth-in = AUTHORITY_LIST ;
```

```
                          |     authority-in = AUTHORITY_LIST ;
                          |     auth-out = AUTHORITY_LIST ;
                          |     authority-out = AUTHORITY_LIST ;
                          |     authority = AUTHORITY_LIST ;
                          |     class = d; | c1; | c2; | b1; | b2; | b3; | a1 ;
                          |     class = D; | C1; | C2; | B1; | B2; | B3; | A1 ;
                          |     domain = NAME; | number ;
                          |     doi = NAME; | number ;
                          |     ipso = none; | basic; | cipso ;
                          |     label = LABEL ;
                          |     min label = LABEL ;
                          |     max label = LABEL ;
                          |     mode = MODE [ [,] MODE ] ... ;
                          |     modes = MODE [ [,] MODE ] ... ;


AUTHORITY_LIST            :     AUTHORITY [ [,] AUTHORITY ] ...
AUTHORITY                 :     authority_number
                          |     #  authority_bit

LABEL                     :     LEVEL [ : COMPARTMENTS ]
                          |     LEVEL [ , COMPARTMENTS ]

LEVEL                     :     level_name
                          |     0..16

COMPARTMENTS              :     COMPARTMENT [ [,] COMPARTMENT ] ...

COMPARTMENT               :     compartment_name
                          |     compartment_mask
                          |     #  compartment_bit

MODE                      :     receive
                          |     send
                          |     trap

WAL_ATTRIBUTES            :     name = name ;
                          |     PERMENTRY [ PERMENTRY ] ...

PERMENTRY                 :     USER_LIST = SERVICES ;

USER_LIST                 :     USER.GROUP [ [,] USER.GROUP ] ...

USER                      :     user_name | uid | *
```

```
GROUP                   :     group_name | gid | *

SERVICES                :     SERVICE [ [,] SERVICE ] ...

SERVICE                 :     all
                        |     none
                        |     ftp
                        |     login
                        |     lpd
                        |     mail
                        |     nfs
                        |     nqs
                        |     rexec
                        |     rlogin
                        |     rsh
                        |     telnet

MAP_LEVELS              :     levels { MAP_LEVEL_ENTRY;
                                   [ MAP_LEVEL_ENTRY; ] ... }

MAP_LEVEL_ENTRY         :     local_level_name = remote_number
                        |     local_level_number = remote_number

MAP_COMPARTMENTS        :     compartments { MAP_COMPARTMENT_ENTRY;
                                   [ MAP_COMPARTMENT_ENTRY; ] ... }

MAP_COMPARTMENT_ENTRY   :     compartment_name = number
                        |     local_compartment_mask = remote_number
                        |     # local_compartment_bit = remote_number
```

NOTES

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

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

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

SEE ALSO

`ftpd`(8), `nfsd`(8) `rexecd`(8), `rlogind`(8), `rshd`(8), `telnetd`(8)

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

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

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

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

NAME

`spray` – Spray RPC packets

SYNOPSIS

`/etc/spray` [`-c` *count*] [`-d` *delay*] [`-i` *delay*] [`-l` *length*] *host*

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `spray` command sends a one-way stream of packets to *host* using Remote Procedure Call (RPC) requests and reports how many were received and the transfer rate. The *host* argument can be either a name or an Internet dot address.

The `spray` command accepts the following options:

`-c` *count*	Specifies the number of packets to send. The default value of *count* is the number of packets required to make the total stream size 100000 bytes.
`-d` *delay*	Specifies the number of microseconds to pause between sending each packet. The default is 0.
`-i` *delay*	Specifies that Internet control message protocol (ICMP) echo packets must be used, rather than RPC requests. Because ICMP automatically echos, this creates a two-way stream.
`-l` *length*	Specifies the number of bytes in the Ethernet packet that holds the RPC call message. Since the data is encoded using external Data Representation (XDR), which only deals with 32-bit quantities, not all values of *length* are possible, and `spray` rounds up to the nearest possible value. The default value of *length* is 86 bytes (the size of the RPC and user datagram protocol (UDP) headers).
host	Specifies name or Internet dot address to which packets are sent.

SEE ALSO

`ping`(8), `sprayd`(8)

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

NAME

`sprayd` – Performs RPC-based spray server function

SYNOPSIS

```
/etc/sprayd
```

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `sprayd` command is a Remote Procedure Call (RPC)-based server that records the packets sent by `spray`(8). The `sprayd` daemon is always registered as RPC program 100012. Usually, `inetd`(8) invokes the `sprayd` daemon.

FILES

`/etc/inetd.conf`	Contains configuration information and listens for incoming service requests
`/etc/rpc`	File that makes remote procedure calls

SEE ALSO

`inetd`(8), `spray`(8)

NAME

`spwcard` – Sets wildcard levels on system directories in a trusted environment

SYNOPSIS

`/etc/spwcard`

IMPLEMENTATION

All Cray Research systems

DESCRIPTION

The `spwcard` sets the wildcard security level on the appropriate system directories required for the UNICOS system to operate in multilevel mode. The directory names are defined within the `spwcard` program, and no arguments are required or allowed when executing the command.

`/dev/tty*` devices are reset to level 0 and null compartments. Usually, `spwcard` is executed by the `/etc/rc` (see `brc`(8)) script when the system is entering multiuser mode, but can be executed at any time by `root` to label system directories with the wildcard level.

NOTES

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

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

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

FILES

`/etc/spwcard`

SEE ALSO

General UNICOS System Administration, Cray Research publication SG–2301